Grand Master

Ingeniería y Calidad del Software



tech universidad tecnológica



tech universidad tecnológica

Grand Master

Ingeniería y Calidad del 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-calidad-software**

Índice

01 Presentación — pág. 4

02 Objetivos — pág. 8

03 Competencias — pág. 16

04 Dirección del curso — pág. 20

05 Estructura y contenido — pág. 24

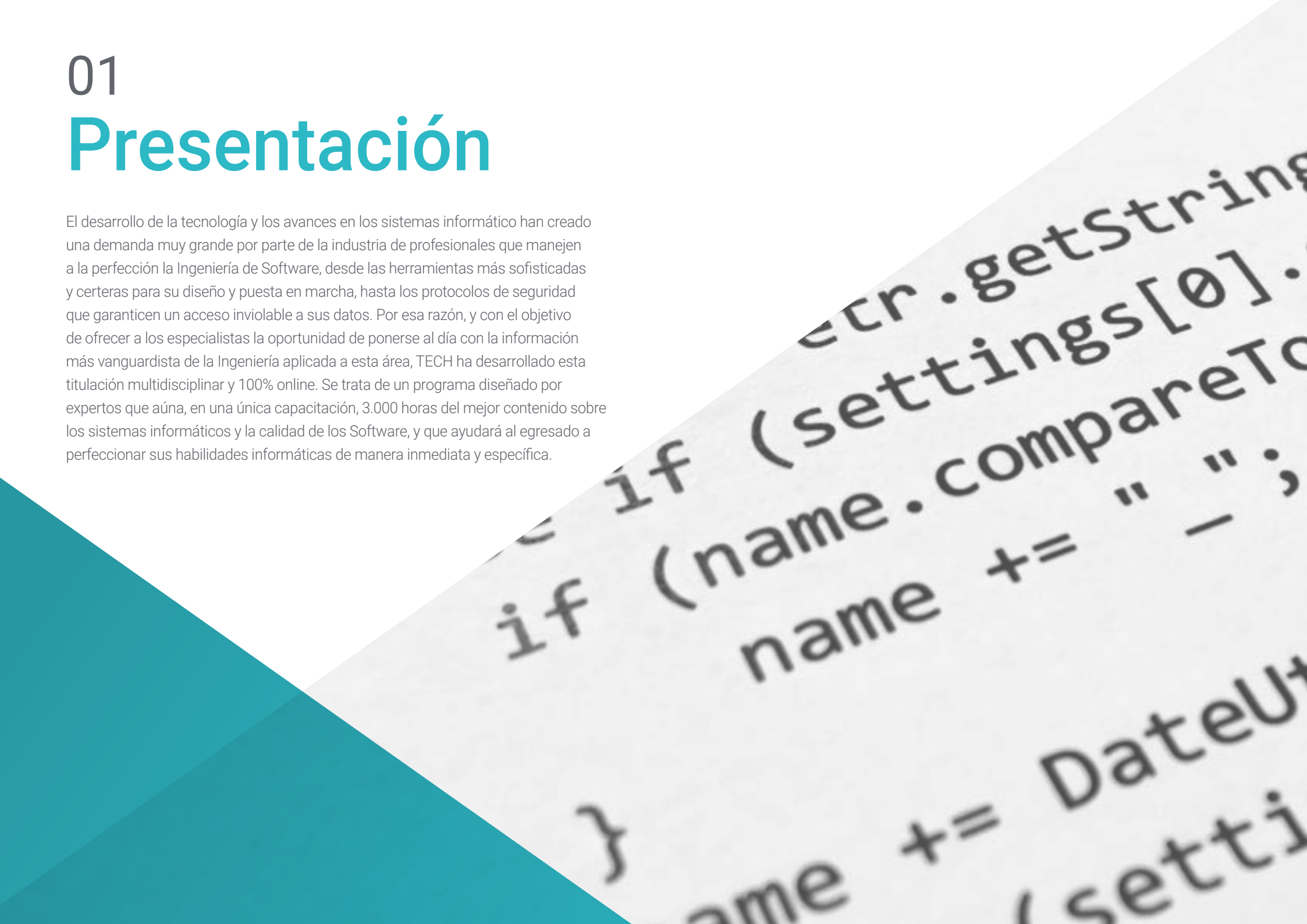
06 Metodología — pág. 46

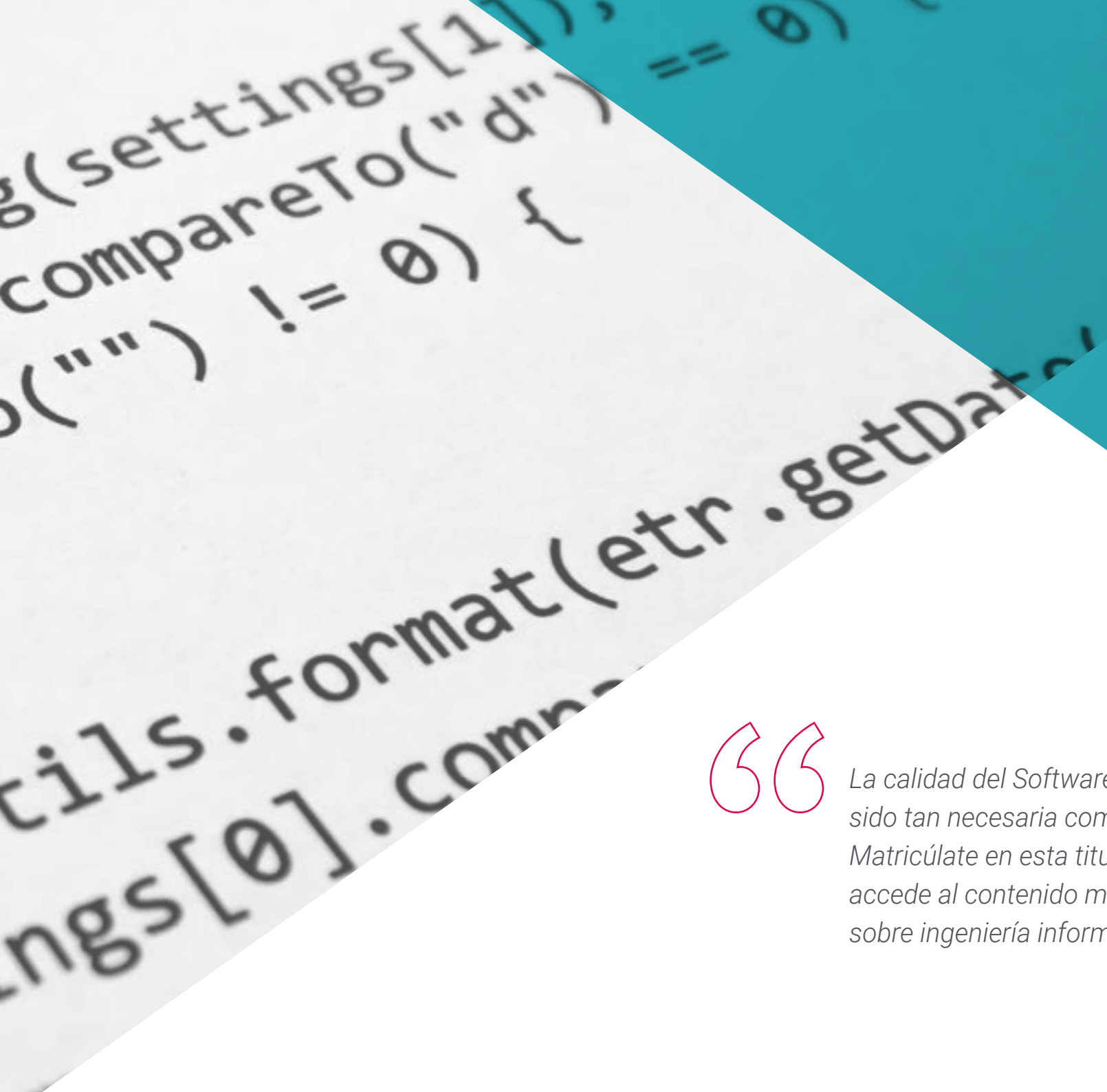
07 Titulación — pág. 54

01

Presentación

El desarrollo de la tecnología y los avances en los sistemas informático han creado una demanda muy grande por parte de la industria de profesionales que manejen a la perfección la Ingeniería de Software, desde las herramientas más sofisticadas y certeras para su diseño y puesta en marcha, hasta los protocolos de seguridad que garanticen un acceso inviolable a sus datos. Por esa razón, y con el objetivo de ofrecer a los especialistas la oportunidad de ponerse al día con la información más vanguardista de la Ingeniería aplicada a esta área, TECH ha desarrollado esta titulación multidisciplinar y 100% online. Se trata de un programa diseñado por expertos que aúna, en una única capacitación, 3.000 horas del mejor contenido sobre los sistemas informáticos y la calidad de los Software, y que ayudará al egresado a perfeccionar sus habilidades informáticas de manera inmediata y específica.





La calidad del Software nunca había sido tan necesaria como hasta ahora. Matricúlate en esta titulación online y accede al contenido más exhaustivo sobre ingeniería informática"

La ingeniería informática ha crecido en los últimos años de una manera exponencial debido a la evolución de la tecnología y las herramientas digitales, sobre todo en todo lo que se refiere a la web y a su usabilidad. Es por ello que en la actualidad el desarrollo de Software para funciones diversas está a la orden del día y el catálogo de programas es cada vez mayor. Sin embargo, esta cantidad no siempre es sinónimo de calidad, por lo que con frecuencia se encuentran aplicaciones que no cumplen su cometido, que devuelven errores o que vulneran gravemente la seguridad de las empresas. Por esa razón, la demanda de ingenieros informáticos especializados en esta área es cada vez mayor.

Es por ello que TECH ha decidido diseñar este Grand Master en Ingeniería y Calidad del Software, un programa multidisciplinar diseñado por expertos en el área y planteado de tal manera que el egresado podrá encontrar en él todas las herramientas necesarias para actualizar su conocimiento de manera exhaustiva y en base a las últimas novedades del sector. Se trata de una capacitación que aúna la teoría y la práctica en 20 módulos en los que se profundiza en la ingeniería de Software y en la calidad de proyectos de sistemas informáticos.

A lo largo de los 24 meses en los que se distribuye este programa 100% online, el ingeniero tendrá acceso al mejor temario que le permitirá mejorar sus habilidades en la normalización de las bases de datos y en el desacoplamiento entre componentes de un sistema, así como podrá ampliar sus conocimientos en materia de arquitecturas escalables, métricas de calidad y trabajo colaborativo.

Además, tendrá acceso a una moderna y vanguardista aula virtual en la que encontrará todas las herramientas que le permitirán sacarle el máximo rendimiento a esta titulación, incluidas cientos de horas de material adicional en diferentes formatos. Todo este contenido podrá ser descargado en cualquier dispositivo con conexión a internet, garantizando su consulta siempre que quiera y lo necesite.

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

- El desarrollo de casos prácticos presentados por expertos en ingeniería
- 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 diseño y conformación 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

Tendrás acceso a ejercicios de lenguaje HTML y a sus respuestas, de manera que podrás poner en práctica tus conocimientos y la teoría desarrollada a lo largo de la programación”

Gracias al módulo dedicado al DevOps contarás con los conocimientos más amplios y exhaustivos para hacer más rápido el ciclo de vida del desarrollo de Software y garantizar una entrega continua de alta calidad”

Gracias a esta titulación podrás poner en marcha tu propio proyecto de desarrollo de Software y aplicar en él las pruebas unitarias de estrés y resistencia más sofisticadas y novedosas para comprobar su calidad.

Ahonda en el Test Driven Development y obtén una visión amplia y especializada del diseño y el desarrollo de Software basado en pruebas.

Incluye, en su cuadro docente, a profesionales pertenecientes al ámbito de la Ingeniería 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 curso académico. Para ello, el profesional contará con la ayuda de un novedoso sistema de vídeo interactivo realizado por reconocidos expertos.



02

Objetivos

La ingeniería informática es un sector que está continuamente cambiando. Por eso TECH ha desarrollado esta titulación, no solo con el objetivo de poder aportar al especialista un conocimiento amplio y actualizado sobre su profesión, sino para que conozca al detalle las herramientas que le permitirán mantenerse al día tras la finalización de este Grand Master. Además, pondrá a su disposición el mejor material teórico, práctico y audiovisual con el fin de hacer de este programa una experiencia académica dinámica y altamente capacitante.





Si tu objetivo es convertirte en un especialista en Ingeniería y Calidad del Software, este Grand Master te aportará todo lo que necesitas para superar, con total garantía de éxito, tus expectativas profesionales"

Objetivos generales

- Desarrollar los criterios, tareas y metodologías avanzadas para comprender la relevancia de un trabajo orientado a la Calidad
- Analizar los factores clave en la Calidad de un proyecto de Software
- Desarrollar los aspectos normativos relevantes
- Implantar procesos de DevOps y de sistemas para el aseguramiento de la Calidad
- Reducir la deuda técnica de los proyectos con un enfoque de Calidad en lugar de un enfoque basado en la economía y los plazos cortos
- Dotar al alumno de conocimientos especializados para poder medir y cuantificar la Calidad de un proyecto de Software
- Defender las propuestas económicas de proyectos desde la base de la Calidad
- Adquirir nuevos conocimientos en Ingeniería de Software y Sistemas Informáticos
- Adquirir nuevas competencias en cuanto a nuevas tecnologías, últimas novedades en Software
- Tratar los datos generados en las actividades de la Ingeniería de Software y Sistemas Informáticos





Objetivos específicos

Módulo 1. Calidad del Software. Niveles de desarrollo TRL

- Desarrollar de forma clara y concisa los elementos que engloban la calidad del Software
- Aplicar los modelos y estándares en función de sistema, producto y proceso Software
- Profundizar en las normas ISO de Calidad aplicadas tanto de forma general como en partes específicas
- Aplicar las normas en función del ámbito del entorno (local, nacional e internacional)
- Examinar los niveles de madurez TRL y adaptarlos a las diferentes partes del proyecto Software a tratar
- Adquirir la capacidad de abstracción para aplicar uno o varios criterios de elementos y niveles de la Calidad del Software
- Distinguir los casos de aplicación de las normativas y niveles de madurez en un proyecto simulado de caso real

Módulo 2. Desarrollo de Proyectos Software. Documentación funcional y técnica

- Determinar la influencia de la gestión de proyecto en la calidad
- Desarrollar las diferentes fases de un proyecto
- Diferenciar los conceptos de calidad inherentes a la documentación funcional y técnica
- Analizar la fase de toma requisitos, la fase de análisis la gestión del equipo y la fase de construcción
- Establecer las diferentes metodologías de gestión de proyectos Software
- Generar criterio para decidir cuál es la metodología más adecuada en función del tipo de proyecto

Módulo 3. *Testing* de Software. Automatización de pruebas

- Establecer las diferencias entre calidad de producto, de proceso y calidad de uso
- Conocer la normativa ISO/IEC 15504
- Determinar los detalles de CMMI
- Aprender las claves de la integración continua, los repositorios y las repercusiones que tienen en un equipo de desarrollo de Software
- Establecer la relevancia de incorporar repositorios por proyectos de Software. Aprender a crearlos con TFS
- Analizar los diferentes tipos de pruebas fundamentales, como las pruebas de carga, unitarias, de *Stress* y de resistencia
- Asimilar la importancia de la escalabilidad del Software en diseño y desarrollo de sistemas de información

Módulo 4. Metodologías de Gestión de Proyectos de Software. Metodologías *Waterfall* frente a metodologías ágiles

- Determinar en qué consiste la metodología *Waterfall*
- Profundizar en la metodología *SCRUM*
- Establecer las diferencias entre *Waterfall* y *SCRUM*
- Concretar las diferencias entre las metodologías *Waterfall* y *SCRUM* y cómo lo ve el cliente
- Examinar el Panel Kanban
- Plantear un mismo proyecto con *WaterFall* y *SCRUM*
- Montar un proyecto híbrido

Módulo 5. TDD (*Test Driven Developement*). Diseño de Software guiado por las pruebas

- Conocer la aplicación práctica del TDD y sus posibilidades en la realización de pruebas de un proyecto Software en un futuro
- Completar casos de simulación real propuestos, como aprendizaje continuo de este concepto TDD
- Analizar, en los casos de simulación, hasta qué punto pueden acertar o fallar las pruebas, desde un punto de vista constructivo
- Determinar las alternativas a TDD, realizando un análisis comparativo entre ellas

Módulo 6. DevOps. Gestión de calidad del Software

- Analizar las deficiencias de un proceso tradicional
- Evaluar las posibles soluciones y elegir la más idónea
- Comprender las necesidades de negocio y sus impactos en la implantación
- Evaluar los costes de las mejoras a implementar
- Desarrollar un ciclo de vida de Software evolucionable, adaptado a las necesidades reales
- Anticipar posibles errores y evitarlos desde el proceso de diseño
- Fundamentar el uso de los distintos modelos de implantación

Módulo 7. DevOps e integración continua. Soluciones prácticas avanzadas en desarrollo de Software

- Identificar las etapas del ciclo de desarrollo y entrega de Software adaptados a los casos particulares
- Diseñar un proceso de entrega de Software mediante integración continua
- Construir e implementar integración y despliegue continuo basado en su diseño previo
- Establecer puntos de control de calidad automáticos en cada entrega de Software
- Mantener un proceso de entrega de Software automático y robusto
- Adaptar las necesidades futuras al proceso de integración y despliegue continuo
- Analizar y anticipar las vulnerabilidades de seguridad durante el proceso de entrega de Software y tras su entrega

Módulo 8. Diseño de bases de datos (BD). Normalización y rendimiento. Calidad del Software

- Valorar el uso del Modelo Entidad-Relación para el diseño previo de una base de datos
- Aplicar una entidad, un atributo, una clave, etc., para la mejor integridad de los datos
- Evaluar las dependencias, formas y reglas de la normalización de bases de datos
- Especializarse en el funcionamiento de un sistema de almacén de datos OLAP, elaborando y usando tanto la tabla de hechos como de la tabla de dimensiones
- Determinar los puntos clave para el rendimiento de la base de datos
- Completar casos de simulación real propuestos como aprendizaje continuo de diseño, normalización y rendimiento de la base de datos
- Establecer en los casos de simulación, las opciones a resolver en la creación de la base de datos desde un punto de vista constructivo

Módulo 9. Diseño de arquitecturas escalables. La arquitectura en el ciclo de vida del Software

- Desarrollar el concepto de arquitectura del Software y sus características
- Determinar los diferentes tipos de escalabilidad en la arquitectura del Software
- Analizar los diferentes niveles que pueden darse en una escalabilidad web
- Adquirir un conocimiento especializado sobre el concepto de ciclo de vida del Software, etapas y modelos
- Determinar el impacto de una arquitectura en el ciclo de vida de Software, con sus ventajas, limitaciones y herramientas de ayuda
- Completar casos de simulación real propuestos, como aprendizaje continuo de la arquitectura y ciclo de vida del Software
- Valorar, en los casos de simulación, hasta qué punto puede ser factible o innecesario el diseño de la arquitectura

Módulo 10. Criterios de Calidad ISO/IEC 9126. Métrica de Calidad del Software

- Desarrollar el concepto de criterios de Calidad y aspectos relevantes
- Examinar la norma ISO/IEC 9126, aspectos principales e indicadores
- Analizar las diferentes mediciones para que un proyecto Software cumpla las evaluaciones acordadas
- Examinar los atributos internos y externos a tratar en la calidad de un proyecto Software
- Distinguir las métricas en función del tipo de programación (estructurado, orientación a objetos, por capas, etc.)
- Completar casos de simulación real, como aprendizaje continuo de la medición de la calidad
- Ver en los casos de simulación hasta qué punto es factible o innecesario; es decir, desde un punto de vista constructivo de las autoras

Módulo 11. Metodologías, desarrollo y calidad en la Ingeniería de Software

- 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)
- Adquirir los conocimientos necesarios para la correcta aplicación de las metodologías ágiles en el desarrollo de Software, entre ellas SCRUM
- Conocer la metodología de desarrollo Lean para discriminar las actividades que no aportan valor en el proceso, en aras de obtener un Software de mayor calidad

Módulo 12. Gestión de proyectos de Software

- Conocer los conceptos fundamentales de la dirección de proyectos y el ciclo de vida de la gestión de proyectos
- Entender las distintas etapas de la gestión de proyectos como son el inicio, la planificación, la gestión de los *stakeholders* y el alcance
- 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
- Entender el funcionamiento de los procesos de aprovisionamiento, ejecución, monitorización, control y cierre de un proyecto
- Adquirir los conocimientos esenciales relacionados con la responsabilidad profesional derivada de la gestión de proyectos

Módulo 13. Plataformas de desarrollo del Software

- 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
- Conocer las técnicas necesarias para la depuración y pruebas de los desarrollos realizados
- 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
- Dominar los conceptos básicos, servicios y herramientas de la plataforma Google Clouds

Módulo 14. Computación en el cliente web

- Asimilar el proceso de creación de contenido web a través del lenguaje de marcado HTML
- Comprender los procedimientos y técnicas para mejor la apariencia de un documento escrito en HTML
- Conocer la evolución del lenguaje JavaScript
- 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
- Entender el uso de flujo basado en eventos y *Listeners,* así como el uso de *Toolkit* modernos y sistemas de alineamiento
- Conocer el concepto de usabilidad web, sus ventajas, principios, métodos y técnicas para hacer un sitio web usable por el usuario
- Establecer los conocimientos de la accesibilidad web, su importancia en las plataformas digitales actuales, metodologías, normas, estándares y determinar las escalas de conformidad

Módulo 15. Computación en servidor web

- Comprender los conceptos básicos, medios y avanzados del lenguaje PHP para la implementación de aplicaciones en el lado del servidor
- Adquirir los conocimientos necesarios para el modelamiento de los datos, sus relaciones, claves y normalizaciones
- Entender la construcción del modelo lógico de datos, la especificación de tablas, columnas, claves y dependencias, además, los conocimientos necesarios para el manejo físico de datos, tipos de ficheros, modos de acceso y organización de los mismos
- Aprender a integrar las aplicaciones desarrolladas en PHP con las bases de datos MariaDB y MySql
- Dominar el proceso de interacciones con el cliente, mediante el uso de: formularios, cookies y manejo de sesiones
- 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

Módulo 16. Gestión de la seguridad

- 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

Módulo 17. Seguridad en el Software

- Entender los problemas relacionados con la seguridad en el Software, sus vulnerabilidades y cómo se clasifican
- Conocer los principios de diseño, metodologías y estándares en la seguridad del Software
- Comprender la aplicación de la seguridad, en las diferentes fases del ciclo de vida del Software
- Adquirir los conocimientos necesarios para la codificación segura del ciclo de vida y sus técnicas de validación
- Asimilar las metodologías y procesos para garantizar la seguridad durante el desarrollo y la prestación de servicios en la nube
- Entender los fundamentos de la criptología y las diferentes técnicas de cifrado que existen en la actualidad

Módulos 18. Administración de servidores web

- Conocer el concepto, funcionamiento, arquitectura, recursos y contenidos de un servidor web
- Comprender el funcionamiento, estructura y manejo del protocolo HTTP
- Asimilar el concepto de arquitecturas distribuidas en múltiples servidores
- Dominar el funcionamiento de un servidor de aplicaciones y otro Proxy
- 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 necesarios para la instalación, administración y configuración y seguridad del servidor web de Microsoft Internet Information Services (IIS) así como, el servidor web gratuito Apache

Módulo 19. Auditoría de seguridad

- Adquirir los conocimientos requeridos para la correcta ejecución del proceso de auditoría y control interno informático
- Entender los procesos a realizar para la auditoría de seguridad en sistemas y redes
- Comprender las diferentes herramientas de apoyo, metodologías y el análisis posterior durante la auditoría de seguridad en internet y en los dispositivos móviles
- Aprender las propiedades y factores de influencia que condicionan los riegos empresariales y determinan la correcta implantación de una gestión de riesgo apropiada
- Conocer las medidas mitigadoras del riesgo, así como, las metodologías de implantación de un Sistema de Gestión de la Seguridad de la información y las normativas y estándares a utilizar
- Entender los procedimientos para la realización de la auditoría de seguridad, su trazabilidad y presentación de resultados

Módulo 20. Seguridad en aplicaciones online

- Adquirir los conocimientos necesarios para evaluar y detectar las vulnerabilidades de las aplicaciones online
- Entender las políticas y estándares de la seguridad a aplicar en las aplicaciones online
- Conocer los procedimientos a utilizar, durante el desarrollo de las aplicaciones web y su posterior validación a través de análisis y test de seguridad
- Aprender las medidas de seguridad para el despliego y producción de las aplicaciones web
- Comprender los conceptos, funciones y tecnologías a aplicar en la seguridad de los servicios web, así como los test de seguridad y medidas protectoras
- Asimilar los procedimientos de realización del *hacking* ético, análisis de malware y forense
- Conocer las medidas mitigadoras y de contención de incidentes sobre servicios web
- Incorporar técnicas de buenas prácticas para el desarrollo e implementación de aplicaciones online

03

Competencias

Manejar a la perfección las herramientas de diseño, planificación, gestión y desarrollo de Software es una tarea muy compleja, entre otras cosas, por la cantidad de procesos que engloba. Sin embargo, el curso de este Grand Master aportará al egresado toda la información que necesita para perfeccionar sus competencias en el control de los instrumentos de esta área. De esta manera podrá cumplir con total garantía de éxito las tareas y culminará sus proyectos obteniendo los resultados más prometedores y de calidad del sector de la ingeniería informática.





Especializarte en esta área te permitirá desarrollar competencias de liderazgo específicas para dirigir proyectos de gestión de Software, habilidad muy valorada por las empresas dedicadas a la ingeniería informática"

Competencias generales

- Reducir la deuda técnica de los proyectos con un enfoque de calidad en lugar de un enfoque basado en la economía y los plazos cortos
- Medir y cuantificar la calidad de un proyecto Software
- Realizar el *Test-DrivenDevelopment* (TDD) de forma correcta, para que elevar los estándares de calidad del Software
- Justificar la presupuestación de proyectos orientados a la calidad
- Desarrollar las normas, modelos y estándares de la calidad
- Examina las diferentes evaluaciones de madurez en la tecnología
- Reducir el riesgo y asegurar el mantenimiento y control de las versiones posteriores
- Dominar las fases en las que se descompone un proyecto
- Diseñar, gestionar e implementar proyectos de Ingeniería Software y de sistemas informáticos

Podrás conocer al detalle las principales bases de datos y acceder a simulaciones de proyectos reales para su diseño aplicado a empresas de diferentes sectores"



Competencias específicas

- Evaluar un sistema Software en cuanto al grado de avance en el proceso del proyecto
- Abordar estos puntos de fiabilidad, métrica y garantía en los proyectos Software de manera correcta y estratégica
- Abordar el proceso de decisión de la metodología a utilizar en el proyecto
- Dominar los aspectos normativos imprescindibles para la creación de Software
- Desarrollar el *Testing* de forma automática
- Establecer una comunicación adecuada con el cliente, entendiendo la forma en la que percibe el proyecto según la metodología aplicada
- Elaborar la lista de requisitos de las pruebas
- Realizar la abstracción, división en pruebas más unitarias y eliminar lo que no aplique a la buena realización de las pruebas del proyecto Software a realizar
- Actualizar la lista de requisitos de las pruebas de forma mesurada y correcta
- Adaptar la cultura DevOps a las necesidades de negocio
- Desarrollar las últimas prácticas y herramientas en la integración y despliegue continuo
- Refactorizar y afrontar la gestión y coordinación de los datos
- 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 graficas 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 auditoria de seguridad en internet y en los dispositivos móviles
- Entender las políticas y estándares de la seguridad a aplicar en las aplicaciones online

04

Dirección del curso

La dirección y la docencia de este Grand Master en Ingeniería y Calidad del Software corre a cargo de un equipo de expertos en ingeniería con muchos años de experiencia en la gestión y en el desarrollo de proyectos técnicos y especializados. Su bagaje profesional aporta a esta titulación un plus de calidad que se verá reflejado en una mejor contextualización del contenido por parte del egresado, así como la implementación a la experiencia académica de casos prácticos reales y simulados, pero siempre dirigidos a ofrecer un programa 100% online dinámico, vanguardista y basado en la realidad inmediata del sector.





El equipo de ingenieros encargado de la docencia de este Grand Master estará a tu disposición para guiarte y ayudarte a resolver cualquier cuestión o duda que te surja sobre el temario o sobre la profesión"

Dirección



D. Molina Molina, Jerónimo

- IA Engineer & Software Architect. NASSAT - Internet Satélite en Movimiento
- Consultor Sr. Hexa Ingenieros. Introductor de la Inteligencia Artificial (ML y CV)
- Experto en Soluciones Basadas en Inteligencia Artificial en los campos de *Computer Vision,* ML/DL y NLP
- Actualmente investigando posibilidades de aplicación de *Transformers* y de *Reinforcement Learning* en proyecto de investigación personal
- Experto Universitario en Creación y Desarrollo de Empresas. Bancaixa-FUNDEUN Alicante
- Ingeniero en Informática. Universidad de Alicante
- Máster en Inteligencia Artificial. Universidad Católica de Ávila
- MBA-Executive. Foro Europeo Campus Empresarial

Profesores

D. Martínez Calvo, Francisco Javier

- Arquitecto - Analista orgánico y funcional
- Consultor técnico - Informático
- Desarrollo y Soporte a Proyecto Médico Europeo, FNMT, PPG e integración de PCL en HexaIngenieros
- Formador Visual Studio, SqlServer, CCNA (routers y switch Cisco), Programación web PHP y .NET en varios centros (Salesianos, Maforem, Dreamsoft)
- Ingeniero Técnico Industrial, especialización en Electricidad, Electrónica Industrial
- Máster Cibernos en .NET. MCAD
- Máster Eidos en Programación Avanzada. Nivel Experto
- Máster WEB. Certificaciones Dreamweaver, Fireworks, Flash y ActionScript, versiones MX

D. Tenrero Morán, Marcos

- DevOps Engineer-Allot Communications
- Application Lifecycle Management & DevOps-Meta4 Spain. Cegid
- Ingeniero automatización QA-Meta4 Spain. Cegid
- Graduado en Ingeniería de Computadores por la Universidad Rey Juan Carlos
- Desarrollo de aplicaciones profesionales para Android-Universidad Galileo, Guatemala
- Desarrollo de Servicios en la nube (nodeJs, JavaScript, HTML5) - UPM
- Integración Continua con Jenkins-Meta4. Cegid
- Desarrollo Web con Angular-CLI (4), Ionic y nodeJS. Meta4.Universidad Rey Juan Carlos

Dra. Acebes Tamargo, Patricia

- Departamento de Operaciones, trabaja con Elasticsearch y Kivana. Sirt
- Investigadora Línea Human Factor y AI Aplications. CTIC Centro Tecnológico
- Investigadora Línea Unidad Negocio. CTIC Centro Tecnológico
- Departamento Salud Digital y Envejecimiento Activo. CTIC Centro Tecnológico
- Departamento Data Science. CTIC Centro Tecnológico
- Doctoranda en Ingeniería informática
- Licenciada en Economía Universidad de Oviedo
- Cursando Máster en Análisis de Datos UCJC
- Cursando Máster en inteligencia Artificial (ia). UNED
- Cursando Matemáticas e ingeniería tic UNED
- Máster en Blockchain, Smart Contracts y Criptomonedas. Universidad de Alcalá
- Postgrado en Ingeniería de Blockchain. EADA
- Máster Universitario en Economía instrumentos Análisis Económico
- Máster Fiscalidad Colegio de Economistas

Dña. Rodríguez Míguez, Cándida

- CoFounder y City Leader de la red Galicia AI (asociación Spain AI)
- Streamer académico en YouTube
- Proyecto SISAP para SERGAS, funcionalidad web auto vacunación COVID Cita Internet. INDRA Producción S.L.
- OSAL Aixiña. Colaboración en remoto TFM
- Docente Sesión introductoria sobre Inteligencia Artificial. WordPress Galicia
- Ingeniero Informático Especializado en Software. ESEI Ourense. Universidad de Vigo
- Máster Universitario en Ingeniería Informática, especialidad Desarrollo Grandes Sistemas de Software. ESEI Ourense. Universidad de Vigo
- Ciclo Superior en Gestión Comercial y Marketing. Centro Privado FP Novacaixagalicia Ourense
- Ciclo Superior en Administración de Sistemas Informáticos. Centro Privado FP Novacaixagalicia Ourense

D. Pi Morell, Oriol

- Product Owner de Hosting y correo. CDMON
- Analista Funcional y Software Engineer en diferentes organizaciones como Fihoca, Atmira, CapGemini
- Docente de diferentes Cursos como BPM en CapGemini, ORACLE Forms CapGemini, de Procesos de negocio Atmira
- Licenciado en Ingeniería técnica de Informática de Gestión por la Universidad Autónoma
- Máster en Inteligencia Artificial
- Máster en Dirección y Administración de empresas. MBA
- Máster en Dirección de Sistemas de Información Experiencia Docente
- Postgrado, Postgrado Patrones de diseño. Universitat Oberta de Catalunya

05

Estructura y contenido

Para el desarrollo de esta titulación TECH ha basado la estructura de su contenido en tres pilares fundamentales: la información más actualizada y completa del sector de la Ingeniería informática especializada en el área del Software, las recomendaciones del equipo docente, y la innovadora metodología pedagógica del *Relearning.* Además, en su compromiso por ofrecer un programa adaptado y personalizado a las necesidades académicas de cada egresado, no solo en el diseño de horarios, sino también en el nivel de profundización, el alumno encontrará en el aula virtual cientos de horas de material adicional con el que podrá ahondar en los aspectos que considere más relevantes para su práctica profesional.





Gracias a este programa podrás diseñar arquitecturas escalables verticales, horizontales y combinadas, basadas en las técnicas y protocolos informáticos más avanzados, completos y actualizados"

Módulo 1. Calidad del Software. Niveles de desarrollo TRL

1.1. Elementos que influyen en la Calidad de Software (I). La deuda técnica
- 1.1.1. La deuda técnica. Causas y consecuencias
- 1.1.2. Calidad del Software. Principios generales
- 1.1.3. Software sin principios y con principios de calidad
  - 1.1.3.1. Consecuencias
  - 1.1.3.2. Necesidad de aplicación de principios de calidad en el Software
- 1.1.4. Calidad del Software. Tipología
- 1.1.5. Software de Calidad. Rasgos específicos

1.2. Elementos que influyen en la Calidad de Software (II). Costes asociados
- 1.2.1. Calidad del Software. Elementos influyentes
- 1.2.2. Calidad del Software. Ideas erróneas
- 1.2.3. Calidad del Software. Costes asociados

1.3. Modelos de Calidad del Software (I). Gestión del conocimiento
- 1.3.1. Modelos de Calidad generales
  - 1.3.1.1. Gestión de la Calidad total
  - 1.3.1.2. Modelo Europeo de Excelencia Empresarial (EFQM)
  - 1.3.1.3. Modelo Seis-sigma
- 1.3.2. Modelos de la gestión del conocimiento
  - 1.3.2.1. Modelo Dyba
  - 1.3.2.2. Modelo Seks
- 1.3.3. Factoría de experiencia y paradigma QIP
- 1.3.4. Modelos de Calidad en el uso (25010)

1.4. Modelos de Calidad del Software (III). Calidad en datos, procesos y modelos SEI
- 1.4.1. Modelo de Calidad de datos
- 1.4.2. Modelado del proceso Software
- 1.4.3. *Software & Systems Process Engineering Metamodel Specification* (SPEM)
- 1.4.4. Modelos del SEI
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. Normas ISO de Calidad del Software (I). Análisis de los estándares
- 1.5.1. Normas ISO 9000
  - 1.5.1.1. Normas ISO 9000
  - 1.5.1.2. Familia ISO de normas de Calidad (9000)
- 1.5.2. Otras normas ISO relacionadas con Calidad
- 1.5.3. Normas de Modelado de Calidad (ISO 2501)
- 1.5.4. Normas de Medida de la Calidad (ISO 2502n)

1.6. Normas ISO de Calidad del Software (II). Requisitos y evaluación
- 1.6.1. Normas sobre requisitos de Calidad (2503n)
- 1.6.2. Normas sobre evaluación de la Calidad (2504n)
- 1.6.3. ISO/IEC 24744:2007

1.7. Niveles de desarrollo TRL (I). Niveles el 1 al 4
- 1.7.1. Niveles TRL
- 1.7.2. Nivel 1: principios básicos
- 1.7.3. Nivel 2: concepto y/o aplicación
- 1.7.4. Nivel 3: función crítica analítica
- 1.7.5. Nivel 4: validación de componente en entorno de laboratorio

1.8. Niveles de desarrollo TRL (II). Niveles del 5 al 9
- 1.8.1. Nivel 5: validación de componente en entorno relevante
- 1.8.2. Nivel 6: modelo sistema/subsistema
- 1.8.3. Nivel 7: demostración en entorno real
- 1.8.4. Nivel 8: sistema completo y certificado
- 1.8.5. Nivel 9: éxito en el entorno real

1.9. Niveles de Desarrollo TRL. Usos
- 1.9.1. Ejemplo de empresa con entorno de laboratorio
- 1.9.2. Ejemplo de empresa I+D+i
- 1.9.3. Ejemplo de empresa de I+D+i industrial
- 1.9.4. Ejemplo de empresa mixta laboratorio-ingeniería

1.10. Calidad del Software. Detalles clave
- 1.10.1. Detalles metodológicos
- 1.10.2. Detalles técnicos
- 1.10.3. Detalles en la gestión de proyectos Software
  - 1.10.3.1. Calidad de los sistemas informáticos
  - 1.10.3.2. Calidad del producto Software
  - 1.10.3.3. Calidad del proceso Software

Módulo 2. Desarrollo de proyectos Software. Documentación funcional y técnica

2.1. Gestión de proyectos
- 2.1.1. Gestión de proyectos en la Calidad del Software
- 2.1.2. Gestión de proyectos. Ventajas
- 2.1.3. Gestión de proyectos. Tipología

2.2. Metodología en la gestión del proyecto
- 2.2.1. Metodología en la gestión de proyectos
- 2.2.2. Metodologías de proyectos. Tipología
- 2.2.3. Metodologías en la gestión de proyectos. Aplicación

2.3. Fase de Identificación de requisitos
- 2.3.1. Identificación de los requisitos de un proyecto
- 2.3.2. Gestión de las reuniones de un proyecto
- 2.3.3. Documentación a aportar

2.4. Modelo
- 2.4.1. Fase inicial
- 2.4.2. Fase de análisis
- 2.4.3. Fase de construcción
- 2.4.4. Fase de pruebas
- 2.4.5. Entrega

2.5. Modelo de datos a utilizar
- 2.5.1. Determinación del nuevo modelo de datos
- 2.5.2. Identificación del plan de migración de datos
- 2.5.3. Juego de datos

2.6. Repercusiones en otros proyectos
- 2.6.1. Repercusión de un proyecto. Ejemplos

2.7. *MUST* del Proyecto
- 2.7.1. *MUST* de Proyecto
- 2.7.2. Identificación de los *MUST* del proyecto
- 2.7.3. Identificación de los puntos de ejecución para la entrega de un proyecto

2.8. El equipo para la Construcción del Proyecto
- 2.8.1. Roles a intervenir según el proyecto
- 2.8.2. Contacto con RRHH para contratación
- 2.8.3. Entregables y calendario del proyecto

2.9. Aspectos técnicos de un proyecto Software
- 2.9.1. Arquitecto del proyecto. Aspectos técnicos
- 2.9.2. Líderes técnicos
- 2.9.3. Construcción del proyecto Software
- 2.9.4. Evaluación de la Calidad del código, Sonar

2.10. Entregables del proyecto
- 2.10.1. Análisis funcional
- 2.10.2. Modelo de datos
- 2.10.3. Diagrama de estados
- 2.10.4. Documentación técnica

Módulo 3. *Testing* de Software. Automatización de pruebas

3.1. Modelos de calidad del Software
- 3.1.1. Calidad de producto
- 3.1.2. Calidad de proceso
- 3.1.3. Calidad de uso

3.2. Calidad de proceso
- 3.2.1. Calidad de proceso
- 3.2.2. Modelos de madurez
- 3.2.3. Normativa ISO 15504
  - 3.2.3.1. Propósitos
  - 3.2.3.2. Contexto
  - 3.2.3.3. Etapas

3.3. Normativa ISO/IEC 15504
- 3.3.1. Categorías de proceso
- 3.3.2. Proceso de desarrollo. Ejemplo
- 3.3.3. Fragmento de perfil
- 3.3.4. Etapas

3.4. CMMI (*Capability Maturity Model Integration*)
- 3.4.1. Integración de Modelos de madurez de capacidades
- 3.4.2. Modelos y áreas. Tipología
- 3.4.3. Áreas de proceso
- 3.4.4. Niveles de capacidad
- 3.4.5. Administración de procesos
- 3.4.6. Administración de proyectos

3.5. Gestión de cambios y repositorios
- 3.5.1. Gestión de cambios en Software
  - 3.5.1.1. Ítem de configuración. Integración Continua
  - 3.5.1.2. Líneas
  - 3.5.1.3. Flujogramas
  - 3.5.1.4. *Branches*
- 3.5.2. Repositorio
  - 3.5.2.1. Control de versiones
  - 3.5.2.2. Equipo de trabajo y uso del repositorio
  - 3.5.2.3. Integración continua en el repositorio

3.6. *Team Foundation Server* (TFS)
- 3.6.1. Instalación y configuración
- 3.6.2. Creación de un proyecto de equipo
- 3.6.3. Incorporación de contenido al control de código fuente
- 3.6.4. *TFS on Cloud*

3.7. *Testing*
- 3.7.1. Motivación para la realización de pruebas
- 3.7.2. Pruebas de verificación
- 3.7.3. Pruebas beta
- 3.7.4. Implementación y mantenimiento

3.8. Pruebas de carga
- 3.8.1. *Load testing*
- 3.8.2. Pruebas con *LoadView*
- 3.8.3. Pruebas con K6 *Cloud*
- 3.8.4. Pruebas con *Loader*

3.9. Pruebas unitarias de estrés y de resistencia
- 3.9.1. Motivación de las pruebas unitarias
- 3.9.2. Herramientas para *Unit Testing*
- 3.9.3. Motivación de las pruebas de estrés
- 3.9.4. Pruebas usando *StressTesting*
- 3.9.5. Motivación para las pruebas de resistencia
- 3.9.6. Pruebas usando *LoadRunner*

3.10. La Escalabilidad. Diseño de Software escalable
- 3.10.1. La escalabilidad y la arquitectura del Software
- 3.10.2. La independencia entre capas
- 3.10.3. El acoplamiento entre capas. Patrones de arquitectura

Módulo 4. Metodologías de gestión de proyectos Software. Metodologías *Waterfall* frente a Metodologías Agiles

4.1. Metodología *Waterfall*
- 4.1.1. Metodología *Waterfall*
- 4.1.2. Metodología *Waterfall*. Influencia en la Calidad del Software
- 4.1.3. Metodología *Waterfall*. Ejemplos

4.2. Metodología Agile
- 4.2.1. Metodología Agile
- 4.2.2. Metodología Agile. Influencia en la Calidad del Software
- 4.2.3. Metodología Agile. Ejemplos

4.3. Metodología SCRUM
- 4.3.1. Metodología SCRUM
- 4.3.2. Manifiesto SCRUM
- 4.3.3. Aplicación de SCRUM

4.4. Panel Kanban
- 4.4.1. Método Kanban
- 4.4.2. Panel Kanban
- 4.4.3. Panel Kanban. Ejemplo de aplicación

4.5. Gestión de Proyecto en *Waterfall*
- 4.5.1. Fases en un proyecto
- 4.5.2. Visión en un proyecto *Waterfall*
- 4.5.3. Entregables a tener en cuenta

4.6. Gestión de proyecto en SCRUM
- 4.6.1. Fases en un proyecto SCRUM
- 4.6.2. Visión en un proyecto SCRUM
- 4.6.3. Entregables a considerar

4.7. Waterfall vs. SCRUM. Comparativa
- 4.7.1. Planteamiento de un proyecto piloto
- 4.7.2. Proyecto aplicando *Waterfall*. Ejemplo
- 4.7.3. Proyecto aplicando SCRUM. Ejemplo

4.8. Visión del Cliente
- 4.8.1. Documentos en un Waterfall
- 4.8.2. Documentos en un SCRUM
- 4.8.3. Comparativa

4.9. Estructura de Kanban
- 4.9.1. Historias de usuario
- 4.9.2. *Backlog*
- 4.9.3. Análisis de Kanban

4.10. Proyectos híbridos
- 4.10.1. Construcción del proyecto
- 4.10.2. Gestión proyecto
- 4.10.3. Entregables a considerar

Módulo 5. TDD (*Test Driven Developement*). Diseño de Software guiado por las pruebas

5.1. TDD. *Test Driven Development*
- 5.1.1. TDD. *Test Driven Development*
- 5.1.2. TDD. Influencia del TDD en la Calidad
- 5.1.3. Diseño y Desarrollo basado en pruebas. Ejemplos

5.2. Ciclo de TDD
- 5.2.1. Elección de un requisito
- 5.2.2. Realización de pruebas. Tipologías
  - 5.2.2.1. Pruebas unitarias
  - 5.2.2.2. Pruebas de integración
  - 5.2.2.3. Pruebas *End To End*
- 5.2.3. Verificación de la prueba. Fallos
- 5.2.4. Creación de la implementación
- 5.2.5. Ejecución de las pruebas automatizadas
- 5.2.6. Eliminación de la duplicación
- 5.2.7. Actualización de la lista de requisitos
- 5.2.8. Repetición del ciclo TDD
- 5.2.9. Ciclo TDD. Ejemplo teórico-práctico

5.3. Estrategias de Implementación de TDD
- 5.3.1. Implementación falsa
- 5.3.2. Implementación triangular
- 5.3.3. Implementación obvia

5.4. TDD. Uso. Ventajas e inconvenientes
- 5.4.1. Ventajas de uso
- 5.4.2. Limitaciones de uso
- 5.4.3. Balance de Calidad en la implementación

5.5. TDD. Buenas prácticas
- 5.5.1. Reglas TDD
- 5.5.2. Regla 1: tener un test previo que falle antes de codificar en producción
- 5.5.3. Regla 2: no escribir más de un test unitario
- 5.5.4. Regla 3: no escribir más código de lo necesario
- 5.5.5. Errores y anti patrones a evitar en una TDD

5.6. Simulación de proyecto real para usar TDD (I)
- 5.6.1. Descripción general del proyecto (Empresa A)
- 5.6.2. Aplicación de la TDD
- 5.6.3. Ejercicios propuestos
- 5.6.4. Ejercicios. *Feedback*

5.7. Simulación de proyecto real para usar TDD (II)
- 5.7.1. Descripción general del proyecto (Empresa B)
- 5.7.2. Aplicación de la TDD
- 5.7.3. Ejercicios propuestos
- 5.7.4. Ejercicios. *Feedback*

5.8. Simulación de proyecto real para usar TDD (III)
- 5.8.1. Descripción general del proyecto (Empresa C)
- 5.8.2. Aplicación de la TDD
- 5.8.3. Ejercicios propuestos
- 5.8.4. Ejercicios. *Feedback*

5.9. Alternativas a TDD. *Test Driven Development*
- 5.9.1. TCR (*Test Commit Revert*)
- 5.9.2. BDD (*Behavior Driven Development*)
- 5.9.3. ATDD (*Acceptance Test Driven Development*)
- 5.9.4. TDD. Comparativa teórica

5.10. TDD TCR, BDD y ATDD. Comparación práctica
- 5.10.1. Definición del problema
- 5.10.2. Resolución con TCR
- 5.10.3. Resolución con BDD
- 5.10.4. Resolución con ATDD

Módulo 6. DevOps. Gestión de Calidad del Software

6.1. DevOps. Gestión de Calidad del Software
- 6.1.1. DevOps
- 6.1.2. DevOps y Calidad del Software
- 6.1.3. DevOps. Beneficios de la Cultura DevOps

6.2. DevOps. Relación con Agile
- 6.2.1. Entrega acelerada
- 6.2.2. Calidad
- 6.2.3. Reducción de costes

6.3. Puesta en marcha de DevOps
- 6.3.1. Identificación de problemas
- 6.3.2. Implantación en una compañía
- 6.3.3. Métricas de implantación

6.4. Ciclo de entrega de Software
- 6.4.1. Métodos de diseño
- 6.4.2. Convenios
- 6.4.3. Hoja de ruta

6.5. Desarrollo de código libre de errores
- 6.5.1. Código mantenible
- 6.5.2. Patrones de desarrollo
- 6.5.3. *Testing* de código
- 6.5.4. Desarrollo de Software a nivel de código. Buenas prácticas

6.6. Automatización
- 6.6.1. Automatización. Tipos de pruebas
- 6.6.2. Coste de la automatización y mantenimiento
- 6.6.3. Automatización. Mitigando errores

6.7. Despliegues
- 6.7.1. Valoración de objetivos
- 6.7.2. Diseño de un proceso automático y adaptado
- 6.7.3. Retroalimentación y capacidad de respuesta

6.8. Gestión de incidentes
- 6.8.1. Preparación para incidentes
- 6.8.2. Análisis y resolución del incidente
- 6.8.3. ¿Cómo evitar futuros errores?

6.9. Automatización de despliegues
- 6.9.1. Preparación para despliegues automáticos
- 6.9.2. Evaluación de la salud del proceso automático
- 6.9.3. Métricas y capacidad de vuelta atrás

6.10. Buenas prácticas. Evolución de DevOps
- 6.10.1. Guía de buenas prácticas aplicando DevOps
- 6.10.2. DevOps. Metodología para el equipo
- 6.10.3. Evitando nichos

Módulo 7. DevOps e integración continua. Soluciones prácticas avanzadas en desarrollo de Software

7.1. Flujo de la Entrega de Software
- 7.1.1. Identificación de actores y artefactos
- 7.1.2. Diseño del flujo de entrega de Software
- 7.1.3. Flujo de entrega de Software. Requisitos entre etapas

7.2. Automatización de procesos
- 7.2.1. Integración continua
- 7.2.2. Despliegue continuo
- 7.2.3. Configuración de entornos y gestión de secretos

7.3. *Pipelines* declarativos
- 7.3.1. Diferencias entre *pipelines* tradicionales, como código y declarativos
- 7.3.2. *Pipelines* declarativos
- 7.3.3. *Pipelines* declarativos en Jenkins
- 7.3.4. Comparación de proveedores de integración continua

7.4. Puertas de Calidad y retroalimentación enriquecida
- 7.4.1. Puertas de Calidad
- 7.4.2. Estándares de Calidad con puertas de Calidad. Mantenimiento
- 7.4.3. Requisitos de negocio en las solicitudes de integración

7.5. Gestión de artefactos
- 7.5.1. Artefactos y ciclo de vida
- 7.5.2. Sistemas de almacenamiento y gestión de artefactos
- 7.5.3. Seguridad en la gestión de artefactos

7.6. Despliegue continuo
- 7.6.1. Despliegue continuo como contenedores
- 7.6.2. Despliegue continuo con PaaS

7.7. Mejora del tiempo de ejecución del *Pipeline*: análisis estático y *Git Hooks*
- 7.7.1. Análisis estático
- 7.7.2. Reglas de estilo del código
- 7.7.3. *Git Hooks* y *Tests* unitarios
- 7.7.4. El impacto de la infraestructura

7.8. Vulnerabilidades en contenedores
- 7.8.1. Vulnerabilidades en contenedores
- 7.8.2. Escaneo de imágenes
- 7.8.3. Informes periódicos y alertas

Módulo 8. Diseño de Bases de Datos (BD). Normalización y rendimiento. Calidad del Software

8.1. Diseño de bases de datos
- 8.1.1. Bases de Datos. Tipología
- 8.1.2. Bases de datos usados actualmente
  - 8.1.2.1. Relacionales
  - 8.1.2.2. Clave-Valor
  - 8.1.2.3. Basadas en grafos
- 8.1.3. La Calidad del dato

8.2. Diseño del Modelo Entidad-Relación (I)
- 8.2.1. Modelo de Entidad-Relación. Calidad y documentación
- 8.2.2. Entidades
  - 8.2.2.1. Entidad fuerte
  - 8.2.2.2. Entidad débil
- 8.2.3. Atributos

8.2.4. Conjunto de relaciones
8.2.4.1. 1 a 1
8.2.4.2. 1 a muchos
8.2.4.3. Muchos a 1
8.2.4.4. Muchos a muchos
8.2.5. Claves
8.2.5.1. Clave primaria
8.2.5.2. Clave foránea
8.2.5.3. Clave primaria entidad débil
8.2.6. Restricciones
8.2.7. Cardinalidad
8.2.8. Herencia
8.2.9. Agregación
8.3. Modelo Entidad-Relación (II). Herramientas
8.3.1. Modelo Entidad-Relación. Herramientas
8.3.2. Modelo Entidad-Relación. Ejemplo práctico
8.3.3. Modelo Entidad-Relación factible
8.3.3.1. Muestra visual
8.3.3.2. Muestra en representación de tablas
8.4. Normalización de la Base de Datos (BD) (I). Consideraciones en Calidad del Software
8.4.1. Normalización de la BD y Calidad
8.4.2. Dependencias
8.4.2.1. Dependencia funcional
8.4.2.2. Propiedades de la dependencia funcional
8.4.2.3. Propiedades deducidas
8.4.3. Claves
8.5. Normalización de la Base de Datos (BD) (II). Formas Normales y reglas del Codd
8.5.1. Formas normales
8.5.1.1. Primera Forma Normal (1FN)
8.5.1.2. Segunda Forma Normal (2FN)
8.5.1.3. Tercera Forma Normal (3FN)
8.5.1.4. Forma normal de Boyce-Codd (FNBC)
8.5.1.5. Cuarta Forma Normal (4FN)
8.5.1.6. Quinta Forma Normal (5FN)
8.5.2. Reglas de Codd
8.5.2.1. Regla 1: información
8.5.2.2. Regla 2: acceso garantizado
8.5.2.3. Regla 3: tratamiento sistemático de los valores nulos
8.5.2.4. Regla 4: descripción de la base de datos
8.5.2.5. Regla 5: sub-lenguaje integral
8.5.2.6. Regla 6: actualización de vistas
8.5.2.7. Regla 7: insertar y actualizar
8.5.2.8. Regla 8: independencia física
8.5.2.9. Regla 9: independencia lógica
8.5.2.10. Regla 10: independencia de la integridad
8.5.2.10.1. Reglas de integridad
8.5.2.11. Regla 11: distribución
8.5.2.12. Regla 12: no-subversión
8.5.3. Ejemplo práctico
8.6. Almacén de datos/Sistema OLAP
8.6.1. Almacén de datos
8.6.2. Tabla de hechos
8.6.3. Tabla de dimensiones
8.6.4. Creación del sistema OLAP. Herramientas
8.7. Rendimiento de la Base de Datos (BD)
8.7.1. Optimización de índices
8.7.2. Optimización de consultas
8.7.3. Particionado de tablas
8.8. Simulación de proyecto real para diseño BD (I)
8.8.1. Descripción general del proyecto (Empresa A)
8.8.2. Aplicación del diseño de Bases de Datos
8.8.3. Ejercicios propuestos

8.8.4. Ejercicios propuestos. Feedback
8.9. Simulación de proyecto real para diseño BD (II)
8.9.1. Descripción general del proyecto (Empresa B)
8.9.2. Aplicación del diseño de bases de datos
8.9.3. Ejercicios propuestos
8.9.4. Ejercicios propuestos. Feedback
8.10. Relevancia de la Optimización de BBDD en la Calidad del Software
8.10.1. Optimización del diseño
8.10.2. Optimización del código de consultas
8.10.3. Optimización del código de procedimientos almacenados
8.10.4. Influencia de los *Triggers* en la Calidad del Software. Recomendaciones de uso

Módulo 9. Diseño de arquitecturas escalables. La arquitectura en el ciclo de vida del Software

9.1. Diseño de arquitecturas escalables (I)
9.1.1. Arquitecturas Escalables
9.1.2. Principios de una arquitectura escalable
9.1.2.1. Confiable
9.1.2.2. Escalable
9.1.2.3. Mantenible
9.1.3. Tipos de escalabilidad
9.1.3.1. Vertical
9.1.3.2. Horizontal
9.1.3.3. Combinado
9.2. Arquitecturas DDD (*Domain-Driven Design*)
9.2.1. El Modelo DDD. Orientación al dominio
9.2.2. Capas, reparto de responsabilidad y patrones de diseño
9.2.3. Desacoplamiento como base de la Calidad
9.3. Diseño de arquitecturas escalables (II). Beneficios, limitaciones y estrategias de diseño
9.3.1. Arquitectura escalable. Beneficios
9.3.2. Arquitectura escalable. Limitaciones
9.3.3. Estrategias para el desarrollo de arquitecturas escalables (T abla descriptiva)
9.4. Ciclo de vida del Software (I). Etapas
9.4.1. Ciclo de vida del Software
9.4.1.1. Etapa de planificación
9.4.1.2. Etapa de análisis
9.4.1.3. Etapa de diseño
9.4.1.4. Etapa de implementación
9.4.1.5. Etapa de pruebas
9.4.1.6. Etapa de Instalación/Despliegue
9.4.1.7. Etapa de uso y mantenimiento
9.5. Modelos de ciclos de vida del Software
9.5.1. Modelo en cascada
9.5.2. Modelo repetitivo
9.5.3. Modelo en espiral
9.5.4. Modelo Big Bang
9.6. Ciclo de vida del Software (II). Automatización
9.6.1. Ciclos de vida de desarrollo de Software. Soluciones
9.6.1.1. Integración y desarrollo continuos (CI/CD)
9.6.1.2. Metodologías *Agile*
9.6.1.3. DevOps / Operaciones de producción
9.6.2. Tendencias futuras
9.6.3. Ejemplos prácticos
9.7. Arquitectura Software en el ciclo de vida del Software
9.7.1. Beneficios
9.7.2. Limitaciones
9.7.3. Herramientas
9.8. Simulación de proyecto real para diseño de arquitectura Software (I)
9.8.1. Descripción general del proyecto (Empresa A)
9.8.2. Aplicación del Diseño de arquitectura del Software
9.8.3. Ejercicios propuestos
9.8.4. Ejercicios propuestos. Feedback

9.9. Simulación de proyecto real para para diseño de arquitectura Software (II)
- 9.9.1. Descripción general del proyecto (Empresa B)
- 9.9.2. Aplicación del diseño de arquitectura del Software
- 9.9.3. Ejercicios propuestos
- 9.9.4. Ejercicios propuestos. Feedback

9.10. Simulación de proyecto real para para diseño de arquitectura Software (III)
- 9.10.1. Descripción general del proyecto (Empresa C)
- 9.10.2. Aplicación del diseño de arquitectura del Software
- 9.10.3. Ejercicios propuestos
- 9.10.4. Ejercicios propuestos. Feedback

Módulo 10. Criterios de Calidad ISO, IEC 9126. Métrica de Calidad del Software

10.1. Criterios de calidad. Norma ISO, IEC 9126
- 10.1.1. Criterio de calidad
- 10.1.2. Calidad del Software. Justificación. Norma ISO, IEC 9126
- 10.1.3. La Medición de la calidad del Software como indicador clave

10.2. Criterios de la calidad del Software. Características
- 10.2.1. Fiabilidad
- 10.2.2. Funcionalidad
- 10.2.3. Eficiencia
- 10.2.4. Usabilidad
- 10.2.5. Mantenibilidad
- 10.2.6. Portabilidad

10.3. Norma ISO, IEC 9126 (I). Presentación
- 10.3.1. Descripción de la Norma ISO, IEC 9126
- 10.3.2. Funcionalidad
- 10.3.3. Fiabilidad
- 10.3.4. Usabilidad
- 10.3.5. Mantenibilidad
- 10.3.6. Portabilidad
- 10.3.7. Calidad en uso
- 10.3.8. Métricas de Calidad del Software
- 10.3.9. Métricas de Calidad en ISO 9126

10.4. Norma ISO, IEC 9126 (II). Modelos McCall y Boehm
- 10.4.1. Modelo McCall: factores de Calidad
- 10.4.2. Modelo Boehm
- 10.4.3. Nivel intermedio. Características

10.5. Métrica de Calidad del Software (I). Elementos
- 10.5.1. Medida
- 10.5.2. Métrica
- 10.5.3. Indicador
  - 10.5.3.1. Tipos de indicadores
- 10.5.4. Medidas y modelos
- 10.5.5. Alcance de las métricas del Software
- 10.5.6. Clasificación de las métricas del Software

10.6. Medición de calidad del Software (II). Práctica de la medición
- 10.6.1. Recogida de datos métricos
- 10.6.2. Medición de atributos internos del producto
- 10.6.3. Medición de atributos externos del producto
- 10.6.4. Medición de recursos
- 10.6.5. Métricas para sistemas orientados a objetos

10.7. Diseño de un indicador único de calidad del Software
- 10.7.1. Indicador único como calificador global
- 10.7.2. Desarrollo del indicador, justificación y aplicación
- 10.7.3. Ejemplo de aplicación. Necesidad de conocer el detalle

10.8. Simulación de proyecto real para medición de calidad (I)
- 10.8.1. Descripción general del proyecto (empresa A)
- 10.8.2. Aplicación de la medición de calidad
- 10.8.3. Ejercicios propuestos
- 10.8.4. Ejercicios propuestos. *Feedback*

10.9. Simulación de proyecto real para medición de calidad (II)
- 10.9.1. Descripción general del proyecto (empresa B)
- 10.9.2. Aplicación de la medición de calidad
- 10.9.3. Ejercicios propuestos
- 10.9.4. Ejercicios propuestos. *Feedback*

10.10. Simulación de proyecto real para medición de calidad (III)
- 10.10.1. Descripción general del proyecto (empresa C)
- 10.10.2. Aplicación de la medición de calidad
- 10.10.3. Ejercicios propuestos
- 10.10.4. Ejercicios propuestos. *Feedback*

Módulo 11. Metodologías, desarrollo y Calidad en la Ingeniería de Software

11.1. Desarrollo de Software basado en modelos
- 11.1.1. La necesidad de
- 11.1.2. Modelado de objetos
- 11.1.3. UML
- 11.1.4. Herramientas CASE

11.2. Modelado de aplicaciones y patrones de diseño con UML
- 11.2.1. Modelado avanzado de requisitos
- 11.2.2. Modelado estático avanzado
- 11.2.3. Modelado dinámico avanzado
- 11.2.4. Modelado de componentes
- 11.2.5. Introducción a los patrones de diseño con UML
- 11.2.6. *Adapter*
- 11.2.7. *Factory*
- 11.2.8. Singleton
- 11.2.9. *Strategy*
- 11.2.10. *Composite*
- 11.2.11. Facade
- 11.2.12. *Observer*

11.3. Ingeniería dirigida por modelos
- 11.3.1. Introducción
- 11.3.2. Metamodelado de sistemas
- 11.3.3. MDA
- 11.3.4. DSL
- 11.3.5. Refinamientos de modelos con OCL
- 11.3.6. Transformaciones de modelos

11.4. Ontologías en la Ingeniería de Software
- 11.4.1. Introducción
- 11.4.2. Ingeniería de la ontología
- 11.4.3. Aplicación de las ontologías en la ingeniería de Software

Módulo 12. Gestión de proyectos de Software

12.1. La gestión de los *Stakeholders* y del alcance
- 12.1.1. Identificar a los interesados
- 12.1.2. Desarrollar el plan para la gestión de los interesados
- 12.1.3. Gestionar el compromiso de los interesados
- 12.1.4. Controlar el compromiso de los interesados
- 12.1.5. El objetivo del proyecto
- 12.1.6. La gestión del alcance y su plan
- 12.1.7. Recopilar los requisitos
- 12.1.8. Definir el enunciado del alcance
- 12.1.9. Crear la WBS (EDT)
- 12.1.10. Verificar y controlar el alcance

12.2. El desarrollo del cronograma
- 12.2.1. La gestión del tiempo y su plan
- 12.2.2. Definir las actividades
- 12.2.3. Establecimiento de la secuencia de las actividades
- 12.2.4. Estimación de recursos de las actividades
- 12.2.5. Estimación de la duración de las actividades
- 12.2.6. Desarrollo del cronograma y cálculo del camino crítico
- 12.2.7. Control del cronograma

12.3. El desarrollo del presupuesto y la respuesta a los riesgos
- 12.3.1. Estimar los costes
- 12.3.2. Desarrollar el presupuesto y la curva S
- 12.3.3. Control de costes y método del valor ganado
- 12.3.4. Los conceptos de riesgo
- 12.3.5. Cómo hacer un análisis de riesgos
- 12.3.6. El desarrollo del plan de respuesta

12.4. La comunicación y los recursos humanos
- 12.4.1. Planificar la gestión de las comunicaciones
- 12.4.2. Análisis de requisitos de comunicaciones
- 12.4.3. Tecnología de las comunicaciones
- 12.4.4. Modelos de comunicación
- 12.4.5. Métodos de comunicación
- 12.4.6. Plan de gestión de las comunicaciones
- 12.4.7. Gestionar las comunicaciones
- 12.4.8. La gestión de los recursos humanos
- 12.4.9. Principales actores y sus roles en los proyectos
- 12.4.10. Tipos de organizaciones
- 12.4.11. Organización del proyecto
- 12.4.12. El equipo de trabajo

12.5. El aprovisionamiento
- 12.5.1. El proceso de adquisiciones
- 12.5.2. Planificación
- 12.5.3. Búsqueda de suministradores y solicitud de ofertas
- 12.5.4. Adjudicación del contrato
- 12.5.5. Administración del contrato
- 12.5.6. Los contratos
- 12.5.7. Tipos de contratos
- 12.5.8. Negociación del contrato

12.6. Ejecución, monitorización y control y cierre
- 12.6.1. Los grupos de procesos
- 12.6.2. La ejecución del proyecto
- 12.6.3. La monitorización y control del proyecto
- 12.6.4. El cierre del proyecto

12.7. Responsabilidad profesional
- 12.7.1. Responsabilidad profesional
- 12.7.2. Características de la responsabilidad social y profesional
- 12.7.3. Código deontológico del líder de proyectos
- 12.7.4. Responsabilidad vs. PMP®
- 12.7.5. Ejemplos de responsabilidad
- 12.7.6. Beneficios de la profesionalización

Módulo 13. Plataformas de desarrollo del Software

13.1. Introducción al desarrollo de aplicaciones
- 13.1.1. Aplicaciones de escritorio
- 13.1.2. Lenguaje de programación
- 13.1.3. Entornos de desarrollo integrado
- 13.1.4. Aplicaciones web
- 13.1.5. Aplicaciones móviles
- 13.1.6. Aplicaciones en la nube

13.2. Desarrollo de aplicaciones e interfaz gráfica en Java
- 13.2.1. Entornos de desarrollo integrados para Java
- 13.2.2. Principales IDE para Java
- 13.2.3. Introducción a la plataforma de desarrollo Eclipse
- 13.2.4. Introducción a la plataforma de desarrollo NetBeans
- 13.2.5. Modelo Vista Controlador para las interfaces gráficas de usuario
- 13.2.6. Diseñar una interfaz gráfica en Eclipse
- 13.2.7. Diseñar una interfaz gráfica en NetBeans

13.3. Depuración y pruebas en Java
- 13.3.1. Pruebas y depuración de programas en Java
- 13.3.2. Depuración en Eclipse
- 13.3.3. Depuración en NetBeans

13.4. Desarrollo de aplicaciones e interfaz gráfica en .NET
- 13.4.1. Net Framework
- 13.4.2. Componentes de la plataforma de desarrollo .NET
- 13.4.3. Visual Studio .NET
- 13.4.4. Herramientas de .NET para GUI
- 13.4.5. La GUI con Windows Presentation Foundation
- 13.4.6. Depurar y compilar una aplicación de WPF

13.5. Programación para redes .NET
- 13.5.1. Introducción a la programación para redes en .NET
- 13.5.2. Peticiones y respuestas en .NET
- 13.5.3. Uso de protocolos de aplicación en .NET
- 13.5.4. Seguridad en la programación para redes en .NET

13.6. Entornos de desarrollo de aplicaciones móviles
- 13.6.1. Aplicaciones móviles
- 13.6.2. Aplicaciones móviles Android
- 13.6.3. Pasos para el desarrollo en Android
- 13.6.4. El IDE Android Studio

13.7. Desarrollo de aplicaciones en el entorno Android Studio
- 13.7.1. Instalar e iniciar Android Studio
- 13.7.2. Ejecución de una aplicación Android
- 13.7.3. Desarrollo de la interfaz gráfica en Android Studio
- 13.7.4. Iniciando actividades en Android Studio

13.8. Depuración y publicación de aplicaciones Android
- 13.8.1. Depuración de una aplicación en Android Studio
- 13.8.2. Memorizar aplicaciones en Android Studio
- 13.8.3. Publicación de una aplicación en Google Play

13.9. Desarrollo de aplicaciones para la nube
- 13.9.1. *Cloud computing*
- 13.9.2. Niveles de *Cloud:* SaaS, PaaS, IaaS
- 13.9.3. Principales plataformas de desarrollo en la nube
- 13.9.4. Referencias bibliográficas

13.10. Introducción a Google Cloud Platform
- 13.10.1. Conceptos básicos de Google Cloud Platform
- 13.10.2. Servicios de Google Cloud Platform
- 13.10.3. Herramientas de Google Cloud Platform

Módulo 14. Computación en el cliente web

14.1. Introducción a HTML
- 14.1.1. Estructura de un documento
- 14.1.2. Color
- 14.1.3. Texto
- 14.1.4. Enlaces de hipertexto
- 14.1.5. Imágenes
- 14.1.6. Listas
- 14.1.7. Tablas
- 14.1.8. Marcos (*frames*)
- 14.1.9. Formularios
- 14.1.10. Elementos específicos para tecnologías móviles
- 14.1.11. Elementos en desuso

14.2. Hojas de estilo web (CSS)
- 14.2.1. Elementos y estructura de una hoja de estilos
  - 14.2.1.1. Creación de hojas de estilo
  - 14.2.1.2. Aplicación de estilos. Selectores
  - 14.2.1.3. Herencia de estilos y aplicación en cascada
  - 14.2.1.4. Formateado de páginas mediante estilos
  - 14.2.1.5. Estructura de páginas mediante estilos. El modelo de cajas
- 14.2.2. Diseño de estilos para diferentes dispositivos
- 14.2.3. Tipos de hojas de estilos: estáticas y dinámicas. Las pseudo-clases
- 14.2.4. Buenas prácticas en el uso de hojas de estilo

14.3. Introducción e historia de JavaScript
- 14.3.1. Introducción
- 14.3.2. Historia de JavaScript
- 14.3.3. Entorno de desarrollo que vamos a usar

14.4. Nociones básicas de programación web
- 14.4.1. Sintaxis básica de JavaScript
- 14.4.2. Tipos de datos primitivos y operadores
- 14.4.3. Variables y ámbitos
- 14.4.4. Cadenas de texto y *Template Literals*
- 14.4.5. Números y booleanos
- 14.4.6. Comparaciones

14.5. Estructuras complejas en JavaScript
- 14.5.1. Vectores o *arrays* y objetos
- 14.5.2. Conjuntos
- 14.5.3. Mapas
- 14.5.4. Disyuntivas
- 14.5.5. Bucles

14.6. Funciones y objetos
- 14.6.1. Definición e invocación de funciones
- 14.6.2. Argumentos
- 14.6.3. Funciones flecha
- 14.6.4. Funciones de retrollamada o *Callback*
- 14.6.5. Funciones de orden superior
- 14.6.6. Objetos literales
- 14.6.7. El objeto *This*
- 14.6.8. Objetos como espacios de nombres: el objeto *Math* y el objeto *Date*

14.7. El Modelo de Objetos del Documento (DOM)
- 14.7.1. ¿Qué es el DOM?
- 14.7.2. Un poco de historia
- 14.7.3. Navegación y obtención de elementos
- 14.7.4. Un DOM virtual con JSDOM
- 14.7.5. Selectores de consulta o *Query Selectors*
- 14.7.6. Navegación mediante propiedades
- 14.7.7. Asignación de atributos a los elementos
- 14.7.8. Creación y modificación de nodos
- 14.7.9. Actualización del estilo de los elementos del DOM

14.8. Desarrollo web moderno
- 14.8.1. Flujo basado en eventos y *Listeners*
- 14.8.2. *Toolkits* web modernos y sistemas de alineamiento
- 14.8.3. Modo estricto de JavaScript
- 14.8.4. Algo más sobre funciones
- 14.8.5. Promesas y funciones asíncronas
- 14.8.6. *Closures*
- 14.8.7. Programación funcional
- 14.8.8. POO en JavaScript

14.9. Usabilidad web
- 14.9.1. Introducción a la usabilidad
- 14.9.2. Definición de usabilidad
- 14.9.3. Importancia del diseño web centrado en el usuario
- 14.9.4. Diferencias entre accesibilidad y usabilidad
- 14.9.5. Ventajas y problemas en la combinación de accesibilidad y usabilidad
- 14.9.6. Ventajas y dificultades en la implantación de sitios web usables
- 14.9.7. Métodos de usabilidad
- 14.9.8. Análisis de requerimiento de usuario
- 14.9.9. Principios del diseño conceptual. Creación de prototipos orientados al usuario
- 14.9.10. Pautas para la creación de sitios web usables
  - 14.9.10.1. Pautas de usabilidad de Jakob Nielsen
  - 14.9.10.2. Pautas de usabilidad de Bruce Tognazzini
- 14.9.11. Evaluación de la usabilidad

14.10. Accesibilidad web
- 14.10.1. Introducción
- 14.10.2. Definición de accesibilidad web
- 14.10.3. Tipos de discapacidades
  - 14.10.3.1. Discapacidades temporales o permanentes
  - 14.10.3.2. Discapacidades visuales
  - 14.10.3.3. Discapacidades auditivas
  - 14.10.3.4. Discapacidades motrices
  - 14.10.3.5. Discapacidad neurológicas o cognitivas
  - 14.10.3.6. Dificultades derivadas del envejecimiento
  - 14.10.3.7. Limitaciones derivadas del entorno
  - 14.10.3.8. Barreras que impiden el acceso a la web
- 14.10.4. Ayudas técnicas y productos de apoyo para superar las barreras
  - 14.10.4.1. Ayudas para personas ciegas
  - 14.10.4.2. Ayudas para persona con baja visión
  - 14.10.4.3. Ayudas para personas con daltonismo
  - 14.10.4.4. Ayudas para personas con discapacidad auditiva
  - 14.10.4.5. Ayudas para personas con discapacidad motriz
  - 14.10.4.6. Ayudas para personas con discapacidad cognitiva y neurológica

14.10.5. Ventajas y dificultades en la implantación de la accesibilidad web
14.10.6. Normativa y estándares sobre accesibilidad web
14.10.7. Organismos regulatorios de la accesibilidad web
14.10.8. Comparativa de normas y estándares
14.10.9. Guías para el cumplimiento de normativas y estándares
14.10.9.1. Descripción de las pautas principales (imágenes, enlaces videos, etc.)
14.10.9.2. Pautas para una navegación accesible
14.10.9.2.1. Perceptibilidad
14.10.9.2.2. Operatividad
14.10.9.2.3. Comprensibilidad
14.10.9.2.4. Robustez
14.10.10. Descripción del proceso de la conformidad en accesibilidad web
14.10.11. Niveles de conformidad
14.10.12. Criterios de conformidad
14.10.13. Requisitos de conformidad
14.10.14. Metodología de evaluación de la accesibilidad en sitios web

Módulo 15. Computación en servidor web

15.1. Introducción a la programación en el servidor: PHP
15.1.1. Conceptos básicos de programación en el servidor
15.1.2. Sintaxis básica de PHP
15.1.3. Generación de contenido HTML con PHP
15.1.4. Entornos de desarrollo y pruebas: XAMPP
15.2. PHP avanzado
15.2.1. Estructuras de control con PHP
15.2.2. Funciones en PHP
15.2.3. Manejo de *Arrays* en PHP
15.2.4. Manejo de cadenas con PHP
15.2.5. Orientación a objetos en PHP
15.3. Modelos de datos
15.3.1. Concepto de dato. Ciclo de vida de los datos
15.3.2. Tipos de datos
15.3.2.1. Básicos
15.3.2.2. Registros
15.3.2.3. Dinámicos
15.4. El modelo relacional
15.4.1. Descripción
15.4.2. Entidades y tipos de entidades
15.4.3. Elementos de datos. Atributos
15.4.4. Relaciones: tipos, subtipos, cardinalidad
15.4.5. Claves. Tipos de claves
15.4.6. Normalización. Formas normales
15.5. Construcción del modelo lógico de datos
15.5.1. Especificación de tablas
15.5.2. Definición de columnas
15.5.3. Especificación de claves
15.5.4. Conversión a formas normales. Dependencias
15.6. El modelo físico de datos. Ficheros de datos
15.6.1. Descripción de los ficheros de datos
15.6.2. Tipos de ficheros
15.6.3. Modos de acceso
15.6.4. Organización de ficheros
15.7. Acceso a bases de datos desde PHP
15.7.1. Introducción a MariaDB
15.7.2. Trabajar con una base de datos MariaDB: el lenguaje SQL
15.7.3. Acceder a la base de datos MariaDB desde PHP
15.7.4. Introducción a MySql
15.7.5. Trabajar con una base de datos MySql: el lenguaje SQL
15.7.6. Acceder a la base de datos MySql desde PHP
15.8. Interacción con el cliente desde PHP
15.8.1. Formularios PHP
15.8.2. Cookies
15.8.3. Manejo de sesiones
15.9. Arquitectura de aplicaciones web
15.9.1. El patrón Modelo-Vista-Controlador
15.9.2. Controlador
15.9.3. Modelo
15.9.4. Vista

15.10. Introducción a los servicios web
- 15.10.1. Introducción a XML
- 15.10.2. Arquitecturas orientas a servicios (SOA): servicios web
- 15.10.3. Creación de servicios web SOAP y REST
- 15.10.4. El protocolo SOAP
- 15.10.5. El protocolo REST

Módulo 16. Gestión de la seguridad

16.1. La seguridad de la información
- 16.1.1. Introducción
- 16.1.2. La seguridad de la información implica la confidencialidad, integridad y disponibilidad
- 16.1.3. La seguridad es un asunto económico
- 16.1.4. La seguridad es un proceso
- 16.1.5. La clasificación de la información
- 16.1.6. La seguridad en la información implica la gestión de los riesgos
- 16.1.7. La seguridad se articula con controles de seguridad
- 16.1.8. La seguridad es tanto física como lógica
- 16.1.9. La seguridad implica a las personas

16.2. El profesional de la seguridad de la información
- 16.2.1. Introducción
- 16.2.2. La seguridad de la información como profesión
- 16.2.3. Las certificaciones ISC2
- 16.2.4. El estándar ISO 27001
- 16.2.5. Buenas prácticas de seguridad en la gestión de servicios TI
- 16.2.6. Modelos de madurez para la seguridad de la información
- 16.2.7. Otras certificaciones, estándares y recursos profesionales

16.3. Control de accesos
- 16.3.1. Introducción
- 16.3.2. Requisitos del control de accesos
- 16.3.3. Mecanismos de autenticación
- 16.3.4. Métodos de autorización
- 16.3.5. Contabilidad y auditoría de accesos
- 16.3.6. Tecnologías triple A

16.4. Programas, procesos y políticas de seguridad de la información
- 16.4.1. Introducción
- 16.4.2. Programas de gestión de la seguridad
- 16.4.3. La gestión de riesgos
- 16.4.4. Diseño de políticas de seguridad

16.5. Planes de continuidad de negocio
- 16.5.1. Introducción a los PCN
- 16.5.2. Fase I y II
- 16.5.3. Fase III y IV
- 16.5.4. Mantenimiento del PCN

16.6. Procedimientos para a correcta protección de la empresa
- 16.6.1. Redes DMZ
- 16.6.2. Sistemas de detección de intrusos
- 16.6.3. Listas de control de accesos
- 16.6.4. Aprender del atacante: *Honeypot*

16.7. Arquitectura de seguridad. Prevención
- 16.7.1. Visión general. Actividades y modelo de capas
- 16.7.2. Defensa perimetral (firewalls, WAFs, IPS, etc.)
- 16.7.3. Defensa del punto final (equipos, servidores y servicios)

16.8. Arquitectura de seguridad. Detección
- 16.8.1. Visión general detección y supervisión
- 16.8.2. Logs, ruptura de tráfico cifrado, grabación y *Siems*
- 16.8.3. Alertas e inteligencia

16.9. Arquitectura de seguridad. Reacción
- 16.9.1. Reacción. Productos, servicios y recursos
- 16.9.2. Gestión de incidentes
- 16.9.3. CERTS y CSIRTs

16.10. Arquitectura de seguridad. Recuperación
- 16.10.1. Resiliencia, conceptos, requerimientos de negocio y normativa
- 16.10.2. Soluciones IT de resiliencia
- 16.10.3. Gestión y gobierno de las crisis

Módulo 17. Seguridad en el Software

17.1. Problemas de la seguridad en el Software
- 17.1.1. Introducción al problema de la seguridad en el Software
- 17.1.2. Vulnerabilidades y su clasificación
- 17.1.3. Propiedades Software seguro
- 17.1.4. Referencias

17.2. Principios de diseño seguridad del Software
- 17.2.1. Introducción
- 17.2.2. Principios de diseño seguridad del Software
- 17.2.3. Tipos de S-SDLC
- 17.2.4. Seguridad del Software en las fases del S-SDLC
- 17.2.5. Metodologías y estándares
- 17.2.6. Referencias

17.3. Seguridad en el ciclo de vida del Software en las fases de requisitos y diseño
- 17.3.1. Introducción
- 17.3.2. Modelado de ataques
- 17.3.3. Casos de abuso
- 17.3.4. Ingeniería de requisitos de seguridad
- 17.3.5. Análisis de riesgo. Arquitectónico
- 17.3.6. Patrones de diseño
- 17.3.7. Referencias

17.4. Seguridad en el ciclo de vida del Software en las fases de codificación, pruebas y operación
- 17.4.1. Introducción
- 17.4.2. Pruebas de seguridad basadas en riesgo
- 17.4.3. Revisión de código
- 17.4.4. Test de penetración
- 17.4.5. Operaciones de seguridad
- 17.4.6. Revisión externa
- 17.4.7. Referencias

17.5. Codificación segura aplicaciones I
- 17.5.1. Introducción
- 17.5.2. Prácticas de codificación segura
- 17.5.3. Manipulación y validación de entradas
- 17.5.4. Desbordamiento de memoria
- 17.5.5. Referencias

17.6. Codificación segura aplicaciones II
- 17.6.1. Introducción
- 17.6.2. *Integers Overflows*, errores de truncado y problemas con conversiones de tipo entre números enteros
- 17.6.3. Errores y excepciones
- 17.6.4. Privacidad y confidencialidad
- 17.6.5. Programas privilegiados
- 17.6.6. Referencias

17.7. Seguridad en el desarrollo y en la nube
- 17.7.1. Seguridad en el desarrollo; metodología y práctica
- 17.7.2. Modelos PaaS, IaaS,CaaS y SaaS
- 17.7.3. Seguridad en la nube y para servicios en la nube

17.8. Cifrado
- 17.8.1. Fundamentos de la criptología
- 17.8.2. Cifrado simétrico y asimétrico
- 17.8.3. Cifrado en reposo y en tránsito

17.9. Automatización y orquestación de seguridad (SOAR)
- 17.9.1. Complejidad del tratamiento manual: necesidad de automatizar las tareas
- 17.9.2. Productos y servicios
- 17.9.3. Arquitectura SOAR

17.10. Seguridad en el teletrabajo
- 17.10.1. Necesidad y escenarios
- 17.10.2. Productos y servicios
- 17.10.3. Seguridad en el teletrabajo

Módulo 18. Administración de servidores web

18.1. Introducción a servidores web
- 18.1.1. ¿Qué es un servidor web?
- 18.1.2. Arquitectura y funcionamiento de un servidor web
- 18.1.3. Recursos y contenidos en un servidor web
- 18.1.4. Servidores de aplicaciones
- 18.1.5. Servidores Proxy
- 18.1.6. Principales servidores web del mercado
- 18.1.7. Estadística de uso servidores web
- 18.1.8. Seguridad en servidores web
- 18.1.9. Balanceo de carga en servidores web
- 18.1.10. Referencias

18.2. Manejo del protocolo HTTP
- 18.2.1. Funcionamiento y estructura
- 18.2.2. Descripción de peticiones o *request methods*
- 18.2.3. Códigos de estado
- 18.2.4. Cabeceras
- 18.2.5. Codificación del contenido. Páginas de códigos

  Realización de peticiones HTTP en Internet mediante un proxy, *Livehttpheaders* o método similar, analizando el protocolo utilizado

18.3. Descripción de arquitecturas distribuidas en múltiples servidores
- 18.3.1. Modelo de 3 capas
- 18.3.2. Tolerancia a fallos
- 18.3.3. Reparto de carga
- 18.3.4. Almacenes de estado de sesión.
- 18.3.5. Almacenes de caché

18.4. Internet Information Services (IIS)
- 18.4.1. ¿Qué es IIS?
- 18.4.2. Historia y evolución de IIS
- 18.4.3. Principales ventajas y características de IIS7 y posteriores
- 18.4.4. Arquitectura IIS7 y posteriores

18.5. Instalación, administración y configuración de IIS
- 18.5.1. Preámbulo
- 18.5.2. Instalación de Internet Information Services (IIS)
- 18.5.3. Herramientas de administración de IIS
- 18.5.4. Creación, configuración y administración de sitios web
- 18.5.5. Instalación y manejo de extensiones en IIS

18.6. Seguridad avanzada en IIS
- 18.6.1. Preámbulo
- 18.6.2. Autentificación, autorización, y control de acceso en IIS
- 18.6.3. Configuración de un sitio web seguro en IIS con SSL
- 18.6.4. Políticas de seguridad implementada en IIS 18.x

18.7. Introducción a Apache
- 18.7.1. ¿Qué es Apache?
- 18.7.2. Principales ventajas de Apache
- 18.7.3. Características principales de Apache
- 18.7.4. Arquitectura

18.8. Instalación y configuración de Apache
- 18.8.1. Instalación inicial de Apache
- 18.8.2. Configuración de Apache

18.9. Instalación y configuración de los diferentes módulos en Apache
- 18.9.1. Instalación de módulos en Apache
- 18.9.2. Tipos de módulos
- 18.9.3. Configuración segura de Apache

18.10. Seguridad avanzada
- 18.10.1. Autenticación, autorización y control de acceso
- 18.10.2. Métodos de autenticación
- 18.10.3. Configuración segura de Apache con SSL

Módulo 19. Auditoría de seguridad

19.1. Introducción a los sistemas de información y su auditoría
- 19.1.1. Introducción a los sistemas de información y el rol de la auditoría informática
- 19.1.2. Definiciones de auditoría informática y de control interno informático
- 19.1.3. Funciones y objetivos de la auditoría informática
- 19.1.4. Diferencias entre control interno y auditoría informática

19.2. Controles internos de los sistemas de información
- 19.2.1. Organigrama funcional de un centro de proceso de datos
- 19.2.2. Clasificación de los controles de los sistemas de información
- 19.2.3. La regla de oro

19.3. El proceso y las fases de la auditoría de sistemas de información
- 19.3.1. Evaluación de riesgos (EDR) y otras metodologías de auditoría informática
- 19.3.2. Ejecución de una auditoría de Sistemas de Información. Fases de auditoría
- 19.3.3. Habilidades fundamentales del auditor de Sistemas de Información

19.4. Auditoría técnica de seguridad en sistemas y redes
- 19.4.1. Auditorías técnicas de seguridad. Test de intrusión. Conceptos previos
- 19.4.2. Auditorías de seguridad en sistemas. Herramientas de apoyo
- 19.4.3. Auditorías de seguridad en redes. Herramientas de apoyo

19.5. Auditoría técnica de seguridad en internet y dispositivos móviles
- 19.5.1. Auditoría de seguridad en Internet. Herramientas de apoyo
- 19.5.2. Auditoría de seguridad en dispositivos móviles. Herramientas de apoyo
- 19.5.3. Anexo 1. Estructura de informe ejecutivo e informe técnico
- 19.5.4. Anexo 2. Inventario de herramientas
- 19.5.5. Anexo 3. Metodologías

19.6. Sistema de gestión de seguridad de la información
- 19.6.1. Seguridad de los SI: propiedades y factores de influencia
- 19.6.2. Riesgos empresariales y gestión de riesgos: implantación de controles
- 19.6.3. SG de la Seguridad de la Información (SGSI): concepto y factores críticos para el éxito
- 19.6.4. SGSI-Modelo PDCA
- 19.6.5. SGSI ISO-IEC 27001: contexto de la organización
- 19.6.6. Contexto de la organización
- 19.6.7. Liderazgo
- 19.6.8. Planificación
- 19.6.9. Soporte
- 19.6.10. Operación
- 19.6.11. Evaluación del desempeño
- 19.6.12. Mejora
- 19.6.13. Anexo a ISO 27001/ISO-IEC 27002: objetivos y controles
- 19.6.14. Auditoría del SGSI

19.7. Realización de la auditoría
- 19.7.1. Procedimientos
- 19.7.2. Técnicas

19.8. Trazabilidad
- 19.8.1. Metodologías
- 19.8.2. Análisis

19.9. Custodia
- 19.9.1. Técnicas
- 19.9.2. Resultados

19.10. Reportes y presentación de pruebas
- 19.10.1. Tipos de reportes
- 19.10.2. Análisis de los datos
- 19.10.3. Presentación de pruebas

Módulo 20. Seguridad en aplicaciones online

20.1. Vulnerabilidades y problemas de seguridad en las aplicaciones online
- 20.1.1. Introducción a la seguridad en las aplicaciones online
- 20.1.2. Vulnerabilidades de seguridad en el diseño de las aplicaciones web
- 20.1.3. Vulnerabilidades de seguridad en la implementación de las aplicaciones web
- 20.1.4. Vulnerabilidades de seguridad en el despliegue de las aplicaciones web
- 20.1.5. Listas oficiales de vulnerabilidades de seguridad

20.2. Políticas y estándares para la seguridad de las aplicaciones online
- 20.2.1. Pilares para la seguridad de las aplicaciones online
- 20.2.2. Política de seguridad
- 20.2.3. Sistema de gestión de seguridad de la información
- 20.2.4. Ciclo de vida de desarrollo seguro de Software
- 20.2.5. Estándares para la seguridad de las aplicaciones

20.3. Seguridad en el diseño de las aplicaciones web
- 20.3.1. Introducción a la seguridad de las aplicaciones web
- 20.3.2. Seguridad en el diseño de las aplicaciones web

20.4. Test de la seguridad y protección online de las aplicaciones web
- 20.4.1. Análisis y test de la seguridad de las aplicaciones web
- 20.4.2. Seguridad en el despliegue y producción de las aplicaciones web

20.5. Seguridad de los servicios web
- 20.5.1. Introducción a la seguridad de los servicios web
- 20.5.2. Funciones y tecnologías de la seguridad de los servicios web

20.6. Test de la seguridad y protección online de los servicios web
- 20.6.1. Evaluación de la seguridad de los servicios web
- 20.6.2. Protección online. *Firewalls* y *Gateways* XML





20.7. *Hacking* ético, malware y *Forensic*
- 20.7.1. *Hacking* ético
- 20.7.2. Análisis de malware
- 20.7.3. Análisis forense

20.8. Buenas prácticas para garantizar la seguridad en las aplicaciones
- 20.8.1. Manual de buenas prácticas en el desarrollo de las aplicaciones online
- 20.8.2. Manual de buenas prácticas en la implementación de las aplicaciones online

20.9. Errores comunes que perjudican la seguridad de las aplicaciones
- 20.9.1. Errores comunes en el desarrollo
- 20.9.2. Errores comunes en el hospedaje
- 20.9.3. Errores comunes en la producción

Accediendo a este Grand Master no solo estarás dando un paso decisivo para ampliar tus conocimientos sobre la Ingeniería informática especializada en Software, sino hacia una proyección profesional próspera y llena de éxitos”

06

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"

Estudio de Caso para contextualizar todo el contenido

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"



Accederás a un sistema de aprendizaje basado en la reiteración, con una enseñanza natural y progresiva a lo largo de todo el temario.



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

El presente programa de TECH es una enseñanza intensiva, creada desde 0, que propone los retos y decisiones más exigentes en este campo, ya sea en el ámbito nacional o internacional. Gracias a esta metodología se impulsa el crecimiento personal y profesional, dando un paso decisivo para conseguir el éxito. El método del caso, técnica que sienta las bases de este contenido, garantiza que se sigue la realidad económica, social y profesional más vigente.

> *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

TECH aúna de forma eficaz la metodología del Estudio de Caso con un sistema de aprendizaje 100% online basado en la reiteración, que combina elementos didácticos diferentes en cada lección.

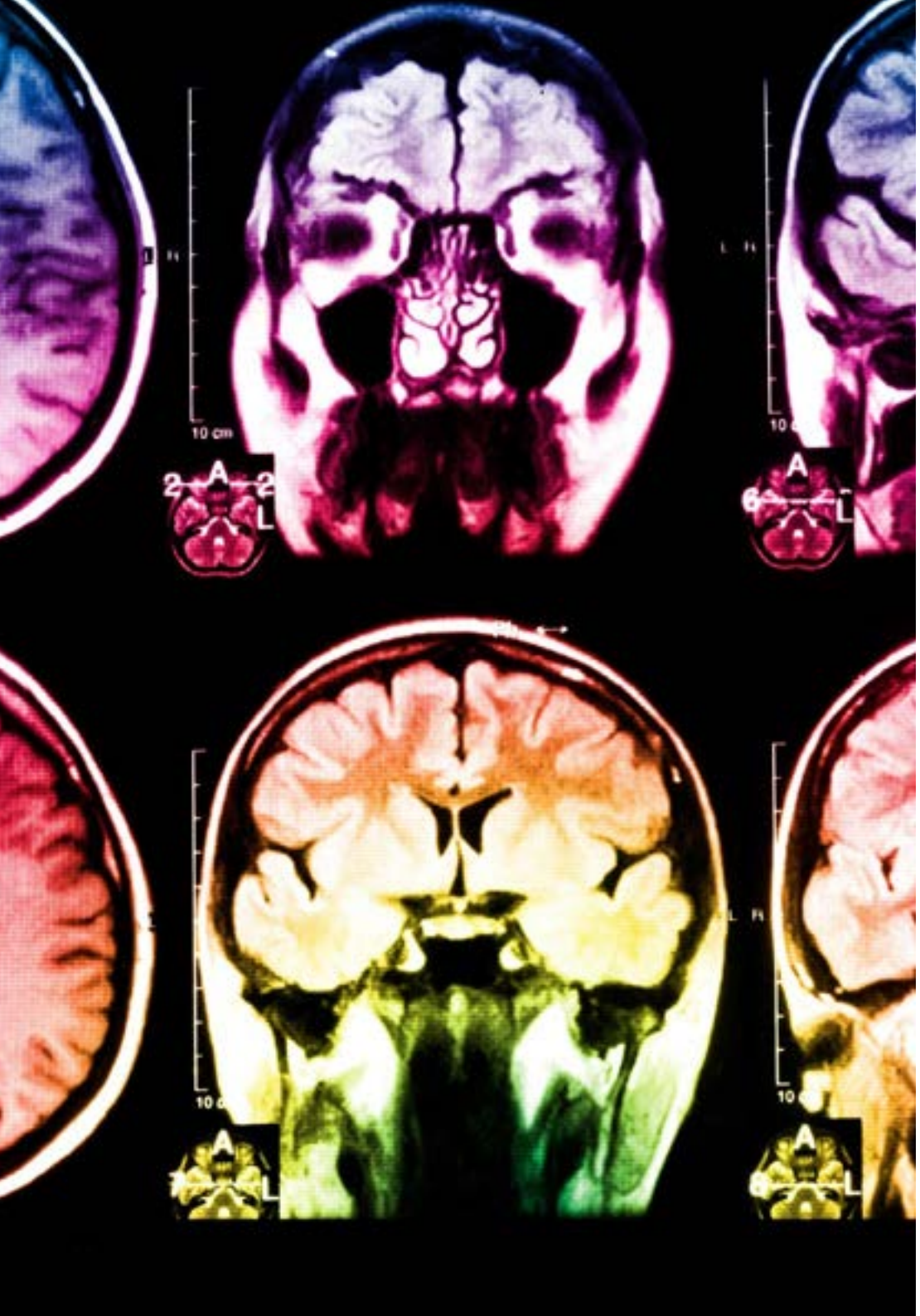
Potenciamos el Estudio de Caso con el mejor método de enseñanza 100% online: el Relearning.

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.

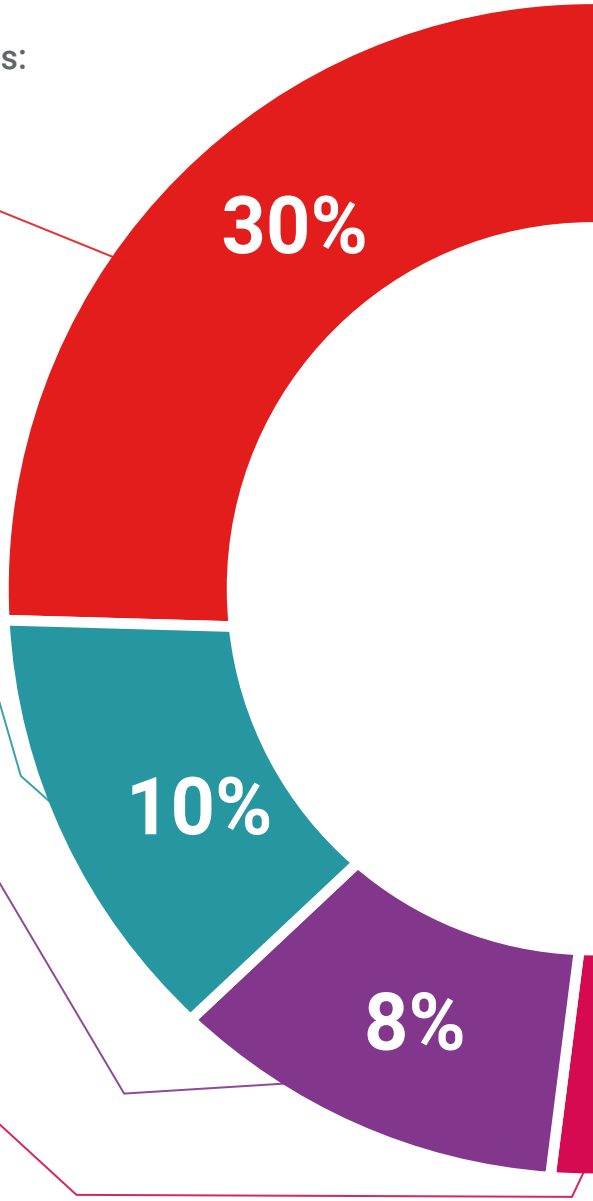
El denominado Learning from an Expert afianza el conocimiento y el recuerdo, y genera seguridad en las futuras decisiones difíciles.

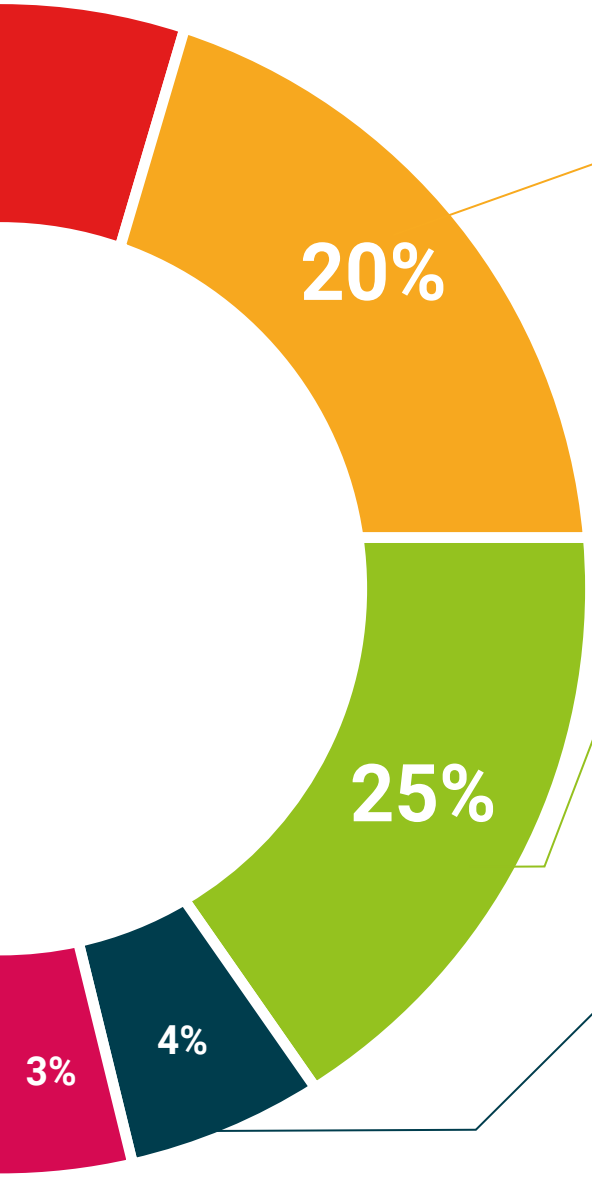
Prácticas de habilidades y competencias

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 casos de estudio elegidos expresamente para esta titulación. 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.

07

Titulación

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





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

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

Tras la superación de la evaluación, el alumno recibirá por correo postal* con acuse de recibo su correspondiente título de **Grand Máster** emitido por **TECH Universidad Tecnológica.**

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

Título: **Grand Master en Ingeniería y Calidad del 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 y Calidad del Software

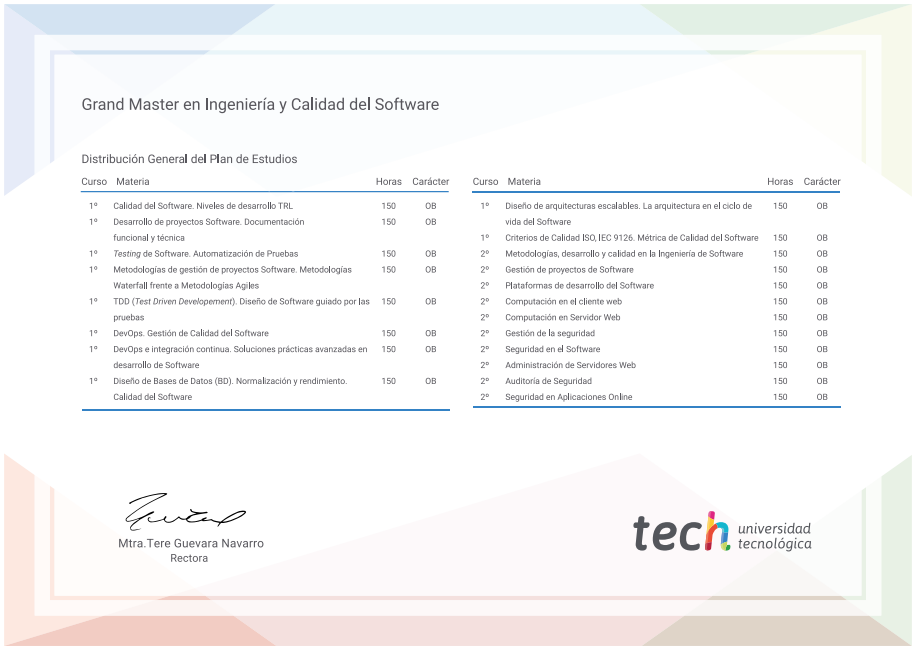
Se trata de un título propio de esta Universidad con una duración de 3.000 horas, con fecha de inicio dd/mm/aaaa y fecha de finalización dd/mm/aaaa.

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 y Calidad del Software

Distribución General del Plan de Estudios

Curso	Materia	Horas	Carácter
1º	Calidad del Software. Niveles de desarrollo TRL	150	OB
1º	Desarrollo de proyectos Software. Documentación funcional y técnica	150	OB
1º	*Testing* de Software. Automatización de Pruebas	150	OB
1º	Metodologías de gestión de proyectos Software. Metodologías Waterfall frente a Metodologías Agiles	150	OB
1º	TDD (*Test Driven Developement*). Diseño de Software guiado por las pruebas	150	OB
1º	DevOps. Gestión de Calidad del Software	150	OB
1º	DevOps e integración continua. Soluciones prácticas avanzadas en desarrollo de Software	150	OB
1º	Diseño de Bases de Datos (BD). Normalización y rendimiento. Calidad del Software	150	OB

Curso	Materia	Horas	Carácter
1º	Diseño de arquitecturas escalables. La arquitectura en el ciclo de vida del Software	150	OB
1º	Criterios de Calidad ISO, IEC 9126. Métrica de Calidad del Software	150	OB
2º	Metodologías, desarrollo y calidad en la Ingeniería de Software	150	OB
2º	Gestión de proyectos de Software	150	OB
2º	Plataformas de desarrollo del Software	150	OB
2º	Computación en el cliente web	150	OB
2º	Computación en Servidor Web	150	OB
2º	Gestión de la seguridad	150	OB
2º	Seguridad en el Software	150	OB
2º	Administración de Servidores Web	150	OB
2º	Auditoría de Seguridad	150	OB
2º	Seguridad en Aplicaciones Online	150	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 y Calidad del Software

Modalidad: **Online**

Duración: **2 años**

Titulación: **TECH Universidad Tecnológica**

Horas lectivas: **3.000 h.**

Grand Master

Ingeniería y Calidad del Software



tech universidad tecnológica