Professional Master's Degree
Advanced Software Engineering

tech technological university

tech technological university

Professional Master's Degree
Advanced Software Engineering

- Modality: **online**
- Duration: **12 months**
- Certificate: **TECH Technological University**
- Dedication: **16h/week**
- Schedule: **at your own pace**
- Exams: **online**

Website: www.techtitute.com/in/information-technology/professional-master-degree/master-advanced-software-engineering

Index

01 Introduction p. 4

02 Objectives p. 8

03 Skills p. 14

04 Structure and Content p. 18

05 Methodology p. 32

06 Certificate p. 40

01

Introduction

Software development would not be possible without software engineering. Thanks to the advances that have been made in technology, today it is possible to find increasingly complex and specific systems and structures, designed on the basis of programming specifications in order to meet the needs and demands of the market. It is a methodical and orderly process that requires specialized knowledge of computer science and its tools, something that the student will acquire through this comprehensive program. Through a multidisciplinary experience, you will be immersed in the key IT processes and systems integration and acquire the necessary skills to carry out projects at the highest level based on stakeholders requirements. All of this in just 12 months of 100% online learning.

If you are looking for a program that will guide you to start your own software project from scratch, this Professional Master's Degree is perfect for you. What are you waiting for to enroll?"

For more than 6 decades, Software Engineering has been at the forefront of the technological revolution through the development of increasingly complex and specialized programs and applications. It is an area that has served as a support for many others to advance towards progress and whose application is extrapolated to practically all existing specialties: medicine, agriculture, teaching, administration, industry, etc. No matter which way you look at it, even the simplest computer process, such as sending an e-mail or using an instant messaging service, something that is frankly an everyday occurrence nowadays, has required exhaustive design and programming to achieve its purpose: to satisfy the needs of human beings.

The wide range of opportunities that arise from this science, in addition to its numerous applications, make it one of the most in-demand fields labor market, not only to create new projects, but also to supervise, maintain and update existing ones. For this reason, and in keeping with TECH's commitment to offer all its graduates the possibility of specializing in this field, the university has decided to launch this comprehensive program in Advanced Software Engineering.

It is an educational program that includes 1,500 hours of the best theoretical-practical and additional content, covering the entire field, from its origin to the design, creation and management of innovative and modern information systems. Throughout this 12-month, the IT specialist will be able to delve into the intricacies of this specialty: its technical and structural requirements, the keys to creating secure architectures, the integration of ICT-based services, the management of stakeholders and their scope, the development of a project from the outset to its launch, and much more!.

All this 100% online, thanks to which the graduate will be able to access the program of this Professional Master's Degree whenever and wherever they want, without face-to-face classes or restricted schedules. In addition, you can access the Virtual Campus from any device with an Internet connection, whether it is a PC, tablet or cell phone. It is, therefore, a unique opportunity to specialize in Software Engineering through a program adapted to your educational needs and to the most demanding requirements of today's IT industry.

This **Professional Master's Degree in Advanced Software Engineering** contains the most complete and up-to-date program on the market. Its most notable features are:

- Case studies presented by experts in Computing Engineering
- The graphic, schematic, and practical contents with which they are created, provide practical information on the disciplines that are essential for professional practice
- Practical exercises where self-assessment can be used to improve learning
- Its special emphasis on innovative methodologies
- Theoretical lessons, questions to the expert, debate forums on controversial topics, and individual reflection assignments
- Content that is accessible from any fixed or portable device with an Internet connection

A program that delves into the basics of Software Engineering: from its origins, to the computing processes that are carried out today"

You will have access to a practical guide that covers the principles of Software Engineering, from the initial process through to construction and deployment"

A program designed to enable you to apply the most innovative strategies in requirements modeling from the very beginning.

Would you like to acquire the necessary skills to design complex and alternative architectures through data flow? With this Professional Master's Degree, you will achieve it in less than 12 months.

The program's teaching staff includes professionals from the sector who contribute their work experience to this program, as well as renowned specialists from leading societies and prestigious universities.

The multimedia content, developed with the latest educational technology, will provide the professional with situated and contextual learning, i.e., a simulated environment that will provide immersive knowledge programmed to learn in real situations.

This program is designed around Problem-Based Learning, whereby the professional must try to solve the different professional practice situations that arise throughout the program. For this purpose, the student will be assisted by an innovative interactive video system created by renowned and experienced experts.

02

Objectives

Software Engineering has become one of the main tools in today's technological development. Without the work carried out by millions of IT professionals, it would not have been possible to reach the advanced level of digital innovation we know today. For this reason, the objective of this Professional Master's Degree is to provide graduates interested in this field with all the information they need to know in detail and develop their own programs and applications in a successful and efficient manner, based on the highest quality and rigorousness.

If your objectives with this Professional Master's Degree include mastering Scrum and other agile methodology techniques, TECH will provide you with everything you need to ensure that you are able to master these techniques"

General Objectives

- Scientific and technological education, as well as preparation for the professional practice of Software Engineering, all with a transversal and versatile academic experience adapted to new technology and innovations in this field
- Obtain wide knowledge in the field of software engineering, but also in the field of computation and computer structure, including the mathematical, statistical and physical basis essential in engineering

Work intensively on software testing through TDD, ATDD and BDD, so that you will be able to build computer structures of the highest quality"

Specific Objectives

Module 1. Software Engineering

- Lay the foundations of software engineering and modeling, learning the main processes and concepts
- Understand the software process and the different models for its development including agile technologies
- Know the main standards related to software quality and project management

Module 2. Advanced Software Engineering

- Know in depth the different agile methodologies used in software engineering
- Learn to develop using scrum, extreme programming and reuse-based software development techniques
- Understand the concepts and processes of software design, learning also about architecture design and about component-level and pattern-based design
- Introduce the concept of DevOps and its main practices
- Learn how to test software, with methodologies such as Test-Driven Development, Acceptance Test-Driven Development, Behavior-Driven Development, BDD and Cucumber
- Understand the different patterns of system architectures and software design, as well as the architecture of cloud applications

Module 3. Requirements Engineering

- Understand requirements engineering, their development, elaboration, negotiation and validation
- Learn the modeling of requirements and the different elements such as scenarios, information, analysis classes, flow, behavior and patterns
- Understand the importance of requirements engineering in the software development process
- Learn how to perform requirements analysis, as well as how to properly document them
- Have an in-depth knowledge of the requirements sources and requirements elicitation techniques, as they are an essential part of the process
- Understand requirements validation and negotiation processes, as well as requirements modeling and management
- Acquire the necessary knowledge for the management of critical systems and the formal specification of requirements

Module 4. Software Engineering Processes

- Deepen the improvement of the software development process and software quality using ISO/IEC standards
- Understand and apply prototyping as an essential part of the development process
- Know the software engineering framework and the ISO/IEC 12207 standard
- Learn the characteristics of the unified software development process and planning in the context of agile software development
- Learn the different styles of distributed software design and service-oriented software architectures
- Learn the essential concepts in graphical user interface design
- Understand the basics of web application development

Module 5. Quality and Information Systems Auditing

- Delve into the strategies and techniques of software testing, software quality factors and different metrics used
- Acquire the essential knowledge of IT security management systems
- Introduce the concepts of intellectual property in information management systems
- Prepare students in the creation of business continuity and disaster recovery plans
- Learn how to plan the management of the security and to handle the principal mechanisms for the protection of assets information
- Learn about the different types of audits and the process carried out during the IT audit

Module 6. Integration Systems

- Acquire the essential concepts related to information systems in the enterprise, as well as identify the opportunities and needs of information systems in the enterprise
- Learn the basics of Business Intelligence, its strategies and implementation, as well as the present and future of BI
- Understand the functioning of systems for integrated enterprise resource management
- Understand digital transformation, from the point of view of business innovation, financial and production management, marketing and human resources management

Module 7. Software Reuse

- Know the general overview of the software reuse strategy
- Learn the different patterns related to software reuse, both design, creation, structural and behavioral
- Learn about the concept of framework, as well as to the main types such as those for graphical user interface design, web application development and object persistence management in databases
- Understand how the widely used Model View Controller (MVC) pattern currently works

Module 8. Information Technology Services

- Train in ICT investment decision making and information systems planning
- Know the control objectives for information and related technologies (COBIT)
- Learn how the Information Technology Infrastructure Library (ITIL) works, strategies, service design, transitions and operations
- Delve into the service management system, knowing the basic principles of UNE-ISO/IEC 20000-1, the structure of the ISO/IEC 20000 series of standards and the requirements of the Service Management System (SMS)
- Understand the functioning of information systems and technologies, their components, classifications, architectures and forms of system integration
- Learn the ISO/IEC 12207 standard, the analysis, design, implementation and acceptance of information systems

Module 9. Information Systems Security

- Learning schedule development for time management, budget development and risk response
- Analyze the nature of network attacks and the different types of security architectures
- Understand the various techniques of system protection and secure code development
- Know the essential components of botnets and spam, as well as malware and malicious code
- Lay the foundations for forensic analysis in the world of software and computer audits
- Obtain a global perspective on security, cryptography and classical cryptanalysis
- Understand the fundamentals of symmetric cryptography and asymmetric cryptography, as well as their main algorithms

Module 10. Project Management

- Understand how quality management works in projects, including planning, assurance, control, statistical concepts and available tools
- Understand the functioning of the processes of procurement, execution, monitoring, control and closure of a project
- Acquire the essential knowledge related to the professional responsibility derived from project management
- Know the fundamental concepts of project management and the project management life cycle
- Understand the different stages of project management such as initiation, planning, stakeholder management and scoping

03

Skills

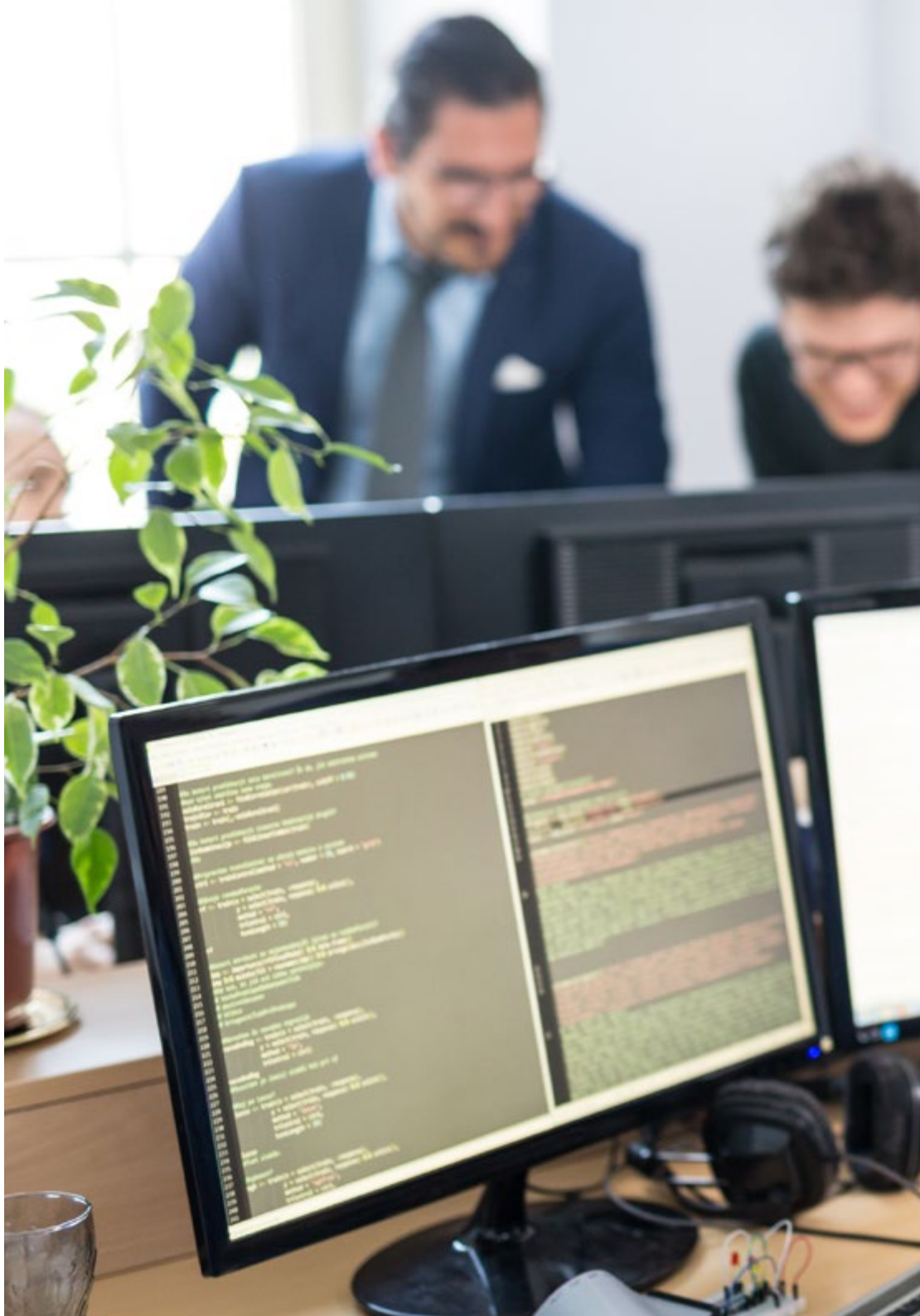
Among the most significant characteristics of all the programs offered by TECH is that they allow the graduate to improve their skills throughout the program. This is possible thanks to the acquisition of specialized knowledge on the subject, in this case on Software Engineering. However, the key to achieving this lies in the resolution of use cases based on situations that represent the current context of the IT industry, which allows the student to put their skills into practice, apply the strategies outlined in the syllabus and expand their skills in the design, analysis, management and launch of applications and programs.

The best program on the current educational market to implement the most innovative and effective prototyping techniques in the software environment into your IT practice"

General Skills

- Respond to the current needs of the Advanced Software Engineering area
- Master the different working systems in Advanced Software Engineering
- Describe and take advantage of free software and open knowledge available on the net

Thanks to the knowledge of software testing strategies and techniques you will be able to work on improving other people's projects to make them more intuitive and specialized"



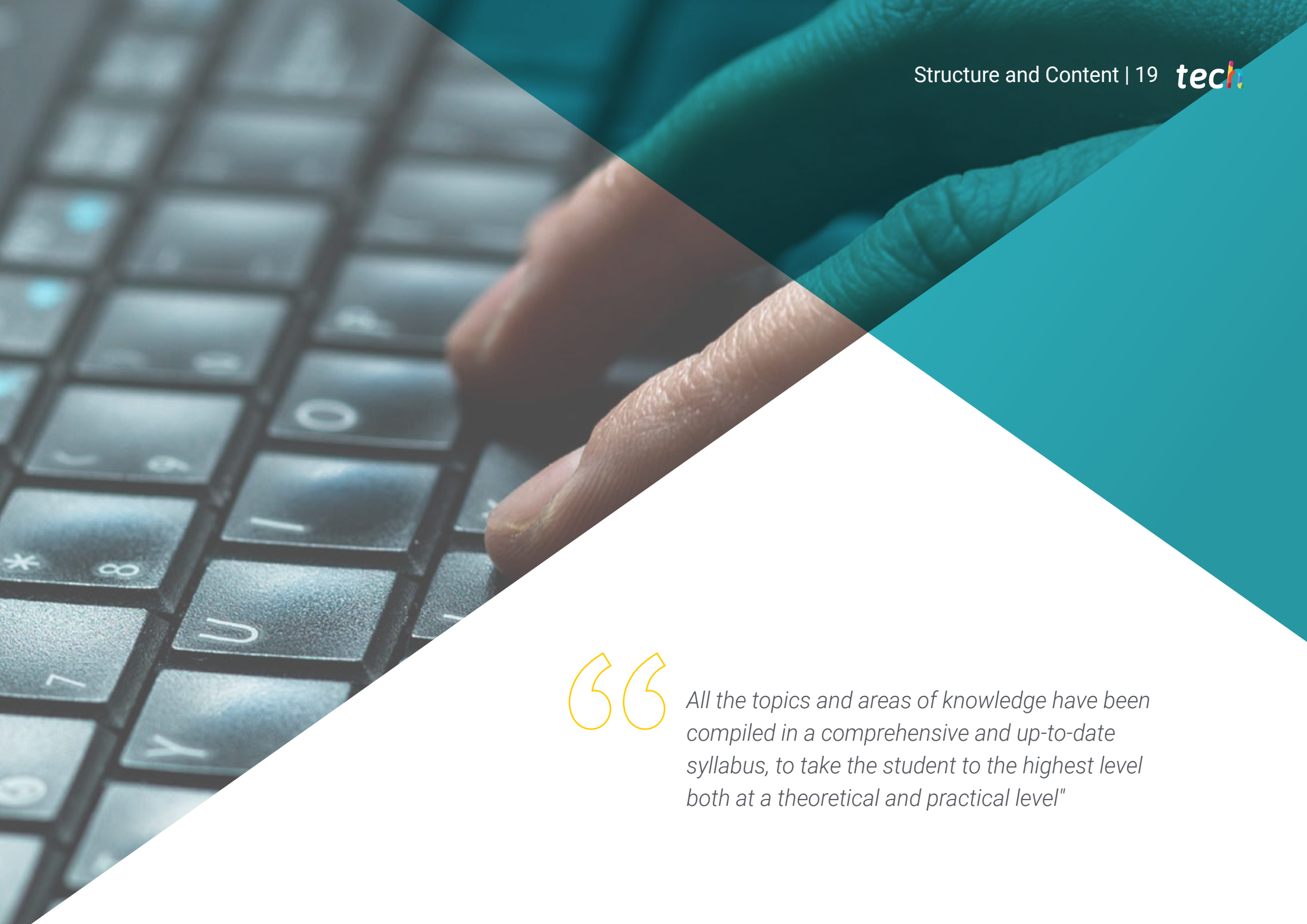
Specific Skills

- Develop in-depth knowledge of all facets of human-computer interaction and how they involve computer developments
- Be proficient in the use of databases
- Develop different types of network applications
- Work as a software engineer
- Control the use of advanced databases
- Perform advanced programming
- Know how to reuse software
- Create interfaces and network applications

04

Structure and Content

The syllabus has been designed on the basis of educational efficiency, carefully selecting the contents to offer a complete program, which includes all the fields of study that are essential to achieve real knowledge of the subject With the latest updates and aspects of the sector.



All the topics and areas of knowledge have been compiled in a comprehensive and up-to-date syllabus, to take the student to the highest level both at a theoretical and practical level"

Module 1. Software Engineering

1.1. Introduction to Software Engineering and Modeling
- 1.1.1. The Nature of Software
- 1.1.2. The Unique Nature of Webapps
- 1.1.3. Software Engineering
- 1.1.4. The Software Process
- 1.1.5. Software Engineering Practice
- 1.1.6. Software Myths
- 1.1.7. How It All Begins
- 1.1.8. Object-Oriented Concepts
- 1.1.9. Introduction to UML

1.2. The Software Process
- 1.2.1. A General Process Model
- 1.2.2. Prescriptive Process Models
- 1.2.3. Specialized Process Models
- 1.2.4. The Unified Process
- 1.2.5. Personal and Team Process Models
- 1.2.6. What is Agility?
- 1.2.7. What Is an Agile Process?
- 1.2.8. Scrum
- 1.2.9. Agile Process Toolkit

1.3. Principles that Guide Software Engineering
- 1.3.1. Principles that Guide the Process
- 1.3.2. Principles that Guide the Practice
- 1.3.3. Communication Principles
- 1.3.4. Planning Principles
- 1.3.5. Modeling Principles
- 1.3.6. Construction Principles
- 1.3.7. Deployment Principles

1.4. Understanding the Requirements
- 1.4.1. Requirements Engineering
- 1.4.2. Establish the Principles
- 1.4.3. Inquiry of Requirements
- 1.4.4. Development of Cases Studies
- 1.4.5. Elaboration of the Requirements Model
- 1.4.6. Negotiation of Requirements
- 1.4.7. Validation of Requirements

1.5. Requirements Modeling I: Scenarios, Information and Analysis Classes
- 1.5.1. Analysis of Requirements
- 1.5.2. Scenario-Based Modeling
- 1.5.3. UML Models that provide the Case Study
- 1.5.4. Data Modeling Concepts
- 1.5.5. Class-Based Modeling
- 1.5.6. Class Diagrams

1.6. Requirements Modeling II: Flow, Behavior and Patterns
- 1.6.1. Requirements that Shape Strategies
- 1.6.2. Flow-Oriented Modeling
- 1.6.3. Status Diagrams
- 1.6.4. Creation of a Behavioral Model
- 1.6.5. Sequence Diagrams
- 1.6.6. Communication Diagrams
- 1.6.7. Patterns for Requirements Modeling

1.7. Design Concepts
- 1.7.1. Design in Software Engineering
- 1.7.2. The Design Process
- 1.7.3. Design Concepts
- 1.7.4. Object-Oriented Design Concepts
- 1.7.5. Model of the Design

1.8. Designing the Architecture:
- 1.8.1. Software Architecture
- 1.8.2. Architectural Genres
- 1.8.3. Architectural Styles
- 1.8.4. Architectural Design
- 1.8.5. Evolution of Alternative Designs for Architecture
- 1.8.6. Mapping the Architecture Using Data Flow

1.9. Component-Level and Pattern-Based Design
- 1.9.1. What Is a Component?
- 1.9.2. Class-Based Component Design
- 1.9.3. Creation of the Design at the Component Level
- 1.9.4. Design of Traditional Components
- 1.9.5. Component-Based Development
- 1.9.6. Design Patterns
- 1.9.7. Pattern-Based Software Design
- 1.9.8. Architectural Patterns
- 1.9.9. Design Patterns at the Component Level
- 1.9.10. User Interface Design Patterns

1.10. Software Quality and Project Management
- 1.10.1. Quality
- 1.10.2. Software Quality
- 1.10.3. The Software Quality Dilemma
- 1.10.4. Achieving Software Quality
- 1.10.5. Software Quality Assurance
- 1.10.6. The Administrative Spectrum
- 1.10.7. The Staff
- 1.10.8. The Product
- 1.10.9. The Process
- 1.10.10. The Project
- 1.10.11. Principles and Practices

Module 2. Advanced Software Engineering

2.1. Introduction to Agile Methodologies
- 2.1.1. Process Models and Methodologies
- 2.1.2. Agility and Agile Processes
- 2.1.3. Agile Manifesto
- 2.1.4. Some Agile Methodologies
- 2.1.5. Agile vs. Traditional

2.2. Scrum
- 2.2.1. Origins and Philosophy of Scrum
- 2.2.2. Scrum Values
- 2.2.3. Scrum Process Flow
- 2.2.4. Scrum Roles
- 2.2.5. Scrum Artifacts
- 2.2.6. Scrum Events
- 2.2.7. User Stories
- 2.2.8. Scrum Extensions
- 2.2.9. Agile Estimates
- 2.2.10. Scrum Scaling

2.3. Extreme Programming
- 2.3.1. Justification and Overview of XP
- 2.3.2. The XP Life Cycle
- 2.3.3. The Five Core Values
- 2.3.4. The Twelve Basic Practices in XP
- 2.3.5. Roles of Participants
- 2.3.6. XP Industrial
- 2.3.7. Critical Assessment of XP

2.4. Software Development Based on Reusability
- 2.4.1. Software Reuse
- 2.4.2. Code Reuse Levels
- 2.4.3. Specific Reuse Techniques
- 2.4.4. Component-Based Development
- 2.4.5. Benefits and Problems of Reuse
- 2.4.6. Reuse Planning

2.5. System Architecture and Software Design Patterns
- 2.5.1. Architectural Design
- 2.5.2. General Architectural Patterns
- 2.5.3. Fault Tolerant Architectures
- 2.5.4. Distributed Systems Architectures
- 2.5.5. Design Patterns
- 2.5.6. Gamma Patterns
- 2.5.7. Interaction Design Patterns

2.6. Cloud Application Architecture
- 2.6.1. Cloud Computing Fundamentals
- 2.6.2. Cloud Application Quality
- 2.6.3. Architectural Styles
- 2.6.4. Design Patterns

2.7. Software Testing: TDD, ATDD and BDD
- 2.7.1. Software Verification and Validation
- 2.7.2. Software Testing
- 2.7.3. Test-Driven Development (TDD)
- 2.7.4. Acceptance Test-Driven Development (ATDD)
- 2.7.5. Test-Driven Development (BDD)
- 2.7.6. BDD and Cucumber

2.8. Software Process Improvement
- 2.8.1. Software Process Improvement
- 2.8.2. The Process Improvement Approach
- 2.8.3. Maturity Models
- 2.8.4. The CMMI Model
- 2.8.5. CMMI V2.0
- 2.8.6. CMMI and Agile

2.9. Software Product Quality: SQuaRE
- 2.9.1. Software Quality
- 2.9.2. Software Product Quality Models
- 2.9.3. ISO/IEC 25000 Family
- 2.9.4. ISO/IEC 25010: Quality Model and Quality Characteristics
- 2.9.5. ISO/IEC 25012: The Quality of the Data
- 2.9.6. ISO/IEC 25020 Software Quality Measurement
- 2.9.7. ISO/IEC 13.013, 13.013 and 13.013: Software and Data Quality Metrics
- 2.9.8. ISO/IEC 25040 Software Assessment
- 2.9.9. Accreditation Process

2.10. Introduction to DevOps
- 2.10.1. DevOps Concept
- 2.10.2. Core Practices

Module 3. Requirements Engineering

3.1. Introduction to Requirements Engineering
- 3.1.1. The Importance of Requirements
- 3.1.2. Concept of Requirement
- 3.1.3. Dimensions of Requirements
- 3.1.4. Levels and Types of Requirements
- 3.1.5. Requirements Characteristics
- 3.1.6. Requirements Engineering
- 3.1.7. The Requirements Engineering Process
- 3.1.8. Frameworks for Requirements Engineering
- 3.1.9. Best Practices in Requirements Engineering
- 3.1.10. The Business Analyst

3.2. Sources of Requirements
- 3.2.1. The Requirements Network
- 3.2.2. The Stakeholders
- 3.2.3. Business Requirements
- 3.2.4. Vision and Scope Document

3.3. Requirements Elicitation Techniques
- 3.3.1. Requirements Elicitation
- 3.3.2. Problems in Requirements Elicitation
- 3.3.3. Contexts of Discovery
- 3.3.4. Interviews
- 3.3.5. Observation and Learning
- 3.3.6. Ethnography
- 3.3.7. Workshops
- 3.3.8. Focus Groups
- 3.3.9. Questionnaires
- 3.3.10. Brainstorming and Creative Techniques
- 3.3.11. Group Media
- 3.3.12. Analysis of System Interfaces
- 3.3.13. Document Analysis and "Archeology"
- 3.3.14. Case Studies and Scenarios
- 3.3.15. Prototypes
- 3.3.16. Reverse Engineering
- 3.3.17. Reuse of Requirements
- 3.3.18. Good Elicitation Practices

3.4. User Requirements
- 3.4.1. Person
- 3.4.2. Case Studies and User Stories
- 3.4.3. Scenarios
- 3.4.5. Types of Scenarios
- 3.4.6. How to Discover Scenarios

3.5. Prototyping Techniques
- 3.5.1. Prototyping
- 3.5.2. Prototypes According to their Scope
- 3.5.3. Prototypes According to their Seasonality
- 3.5.4. The Fidelity of a Prototype
- 3.5.5. User Interface Prototypes
- 3.5.6. Prototype Evaluation

3.6. Requirements Analysis
- 3.6.1. Requirements Analysis
- 3.6.2. Best Practices in Requirements Analysis
- 3.6.3. The Data Dictionary
- 3.6.4. Requirements Prioritization

3.7. Requirements Documentation
- 3.7.1. The Software Requirements Specification Document
- 3.7.2. Structure and Contents of an SRS
- 3.7.3. Natural Language Documentation
- 3.7.4. EARS: Easy Approach to Requirements Syntax
- 3.7.5. Non-Functional Requirements
- 3.7.6. Attributes and Templates in Table Form
- 3.7.7. Good Specifications Practices

3.8. Validation and Negotiation of Requirements
- 3.8.1. Requirements Validation
- 3.8.2. Requirements Validation Techniques
- 3.8.3. Requirements Negotiation

3.9. Requirements Modeling and Management
- 3.9.1. Requirements Modeling
- 3.9.2. The User Perspective
- 3.9.3. The Data Perspective
- 3.9.4. The Functional or Flow-Oriented Perspective
- 3.9.5. The Behavioral Perspective
- 3.9.6. Requirements Volatility
- 3.9.7. Requirements Management Process
- 3.9.8. Tools for Requirements Management
- 3.9.9. Best Practices in Requirements Management

3.10. Critical Systems and Formal Specification
- 3.10.1. Critical Systems
- 3.10.2. Risk-Driven Specification
- 3.10.3. Formal Specification

Module 4. Software Engineering Processes

4.1. Software Engineering Framework
- 4.1.1. Software Features
- 4.1.2. The Main Processes in Software Engineering
- 4.1.3. Software Development Process Models
- 4.1.4. Standard Reference Framework for the Software Development Process: The ISO/IEC 12207 Standard

4.2. Unified Software Development Process
- 4.2.1. Unified Process
- 4.2.2. Dimensions of the Unified Process
- 4.2.3. Case Studies-Driven Development Process
- 4.2.4. Fundamental Workflows of Unified Processes

4.3. Planning in the Context of Agile Software Development
- 4.3.1. Characteristics of Agile Software Development
- 4.3.2. Different Planning Time Horizons in Agile Development
- 4.3.3. Scrum Agile Development Framework and Planning Time Horizons
- 4.3.4. User Stories as a Planning and Estimating Unit
- 4.3.5. Common Techniques for Deriving an Estimate
- 4.3.6. Scales for Interpreting Estimates
- 4.3.7. Planning Poker
- 4.3.8. Common Scheduling Types: Delivery Scheduling and Iteration Scheduling

4.4. Distributed Software Design Styles and Service-Oriented Software Architectures
- 4.4.1. Communication Models in Distributed Software Systems
- 4.4.2. Middleware
- 4.4.3. Architecture Patterns for Distributed Systems
- 4.4.4. General Software Service Design Process
- 4.4.5. Design Aspects of Software Services
- 4.4.6. Composition of Services
- 4.4.7. Web Services Architecture
- 4.4.8. Infrastructure and SOA Components

4.5. Introduction to Model Driven Software Development
- 4.5.1. The Model Concept
- 4.5.2. Model-Driven Software Development
- 4.5.3. MDA Model-Driven Development Framework
- 4.5.4. Elements of a Transformation Model

4.6. Graphical User Interface Design
- 4.6.1. Principles of User Interface Design
- 4.6.2. Architectural Design Patterns for Interactive Systems: Model View Controller (MVC)
- 4.6.3. User Experience (UX)
- 4.6.4. User-Centered Design
- 4.6.5. Graphical User Interface Analysis and Design Process
- 4.6.6. Usability of User Interfaces
- 4.6.7. Accessibility in User Interfaces

4.7. Web Application Design
- 4.7.1. Characteristics of Web Applications
- 4.7.2. Web Application User Interface
- 4.7.3. Navigation Design
- 4.7.4. Basic Interaction Protocol for Web Applications
- 4.7.5. Architecture Styles for Web Applications

4.8. Software Testing Strategies and Techniques and Software Quality Factors
- 4.8.1. Testing Strategies
- 4.8.2. Test Case Designs
- 4.8.3. Value for Money
- 4.8.4. Quality Models
- 4.8.5. ISO/IEC 25000 Family of Standards (SQuaRE)
- 4.8.6. Product Quality Model (ISO 2501n)
- 4.8.7. Data Quality Models (ISO 2501n)
- 4.8.8. Software Quality Management

4.9. Introduction to Software Engineering Metrics
- 4.9.1. Basic Concepts: Measurements, Metrics and Indicators
- 4.9.2. Types of Metrics in Software Engineering
- 4.9.3. The Measurement Process
- 4.9.4. ISO 25024: External and Quality Metrics in Use
- 4.9.5. Object-Oriented Metrics

4.10. Software Maintenance and Re-Engineering
- 4.10.1. Maintenance Process
- 4.10.2. Standard Maintenance Process Framework. ISO/EIEC 14764
- 4.10.3. Software Re-Engineering Process Model
- 4.10.4. Inverse Engineering

Module 5. Quality and Information Systems Auditing

5.1. Introduction to Information Security Management Systems
- 5.1.1. Fundamental Principles of ISMSs
- 5.1.2. ISMS Golden Rules
- 5.1.3. Role of IT Audit in ISMSs

5.2. Safety Management Planning
- 5.2.1. Concepts Related to Safety Management
- 5.2.2. Classification of Information: Objectives, Concepts and Roles
- 5.2.3. Implementation of Security Policies: Security Policies, Standards and Procedures
- 5.2.4. Risk Management: Information Assets Risk Principles and Analysis

5.3. Main Mechanisms for the Protection of Information Assets I
- 5.3.1. Summary of the Main Cryptographic Tools for the Protection of the CIA Triad
- 5.3.2. Consideration of Privacy, Anonymity and Adequate Management of User Traceability Requirements

5.4. Main Mechanisms for the Protection of Information Assets II
- 5.4.1. Communications Security: Protocols, Devices and Security Architectures
- 5.4.2. Operating System Security

5.5. ISMS Internal Controls
- 5.5.1. ISMS Controls Taxonomy: Administrative, Logical and Physical Controls
- 5.5.2. Classification of Controls According to How Threats Are Addressed: Controls for Threat Prevention, Detection and Correction
- 5.5.3. Implementation of Internal Control Systems in ISMSs

5.6. Types of Audits
- 5.6.1. Difference between Audit and Internal Control
- 5.6.2. Internal vs. External Audit
- 5.6.3. Audit Classification according to the Objective and Type of Analysis

5.7. Screenwriter and Screenplay: Subject Matter and Object Protected by Intellectual Property
- 5.7.1. Introduction to Penetration Testing and Forensic Analysis
- 5.7.2. Definition and Relevance of Fingerprinting and Footprinting Concepts

5.8. Vulnerability Scanning and Network Traffic Monitoring
- 5.8.1. Tools for Vulnerability Analysis in Systems
- 5.8.2. Main Vulnerabilities in the Context of Web Applications
- 5.8.3. Analysis of Communications Protocols

5.9. The IT Audit Process
- 5.9.1. Life Cycle Concept in Systems Development
- 5.9.2. Activity and Process Monitoring: Collection and Treatment of Evidence
- 5.9.3. IT Audit Methodology
- 5.9.4. IT Audit Process
- 5.9.5. Identification of the Main Crimes and Misdemeanors in the Context of Information Technologies
- 5.9.6. Computer Crime Investigation: Introduction to Forensic Analysis and its relation to Computer Auditing

5.10. Business Continuity and Disaster Recovery Plans
- 5.10.1. Definition of Business Continuity Plan and the Business Interruption Concept
- 5.10.2. NIST Recommendation on Business Continuity Plans
- 5.10.3. Disaster Recovery Plan
- 5.10.4. Disaster Recovery Plan Process

Module 6. Integration Systems

6.1. Introduction to Information Systems in the Company
- 6.1.1. The Role of Information Systems
- 6.1.2. What Is an Information System?
- 6.1.3. Dimensions of Information Systems
- 6.1.4. Business Processes and Information Systems
- 6.1.5. The IS/IT Department

6.2. Opportunities and Needs of Corporate Information Systems
- 6.2.1. Organizations and Information Systems
- 6.2.2. Features of Organizations
- 6.2.3. Impact of Corporate Information Systems
- 6.2.4. Information Systems to Achieve a Competitive Advantage
- 6.2.5. Use of Systems in Corporate Administration and Management

6.3. Basic Concepts of Information Systems and Technologies
- 6.3.1. Data, Information and Knowledge
- 6.3.2. Technology and Information Systems
- 6.3.3. Technology Components
- 6.3.4. Classification and Types of Information Systems
- 6.3.5. Service and Business Process Based Architectures
- 6.3.6. Forms of Systems Integration

6.4. Systems for the Integrated Management of Company Resources
- 6.4.1. Business Needs
- 6.4.2. An Integrated Corporate Information System
- 6.4.3. Acquisition vs. Development
- 6.4.4. ERP Implementation
- 6.4.5. Implications for Management
- 6.4.6. Leading ERP Vendors

6.5. Supply Chain and Customer Relationship Management Information Systems
- 6.5.1. Definition of Supply Chain
- 6.5.2. Effective Supply Chain Management
- 6.5.3. The Role of Information Systems
- 6.5.4. Supply Chain Management Solutions
- 6.5.5. Customer Relationship Management
- 6.5.6. The Role of Information Systems
- 6.5.7. CRM System Implementation
- 6.5.8. Critical Success Factors in CRM Implementation
- 6.5.9. CRM, e-CRM and Other Trends

6.6. ICT Investment Decision-Making and Information Systems Planning
- 6.6.1. Criteria for ICT Investment Decisions
- 6.6.2. Linking the Project to the Management and Business Plan
- 6.6.3. Management Implications
- 6.6.4. Business Process Re-Design
- 6.6.5. Management's Decision on Implementation Methodologies
- 6.6.6. Need for Information Systems Planning
- 6.6.7. Objectives, Participants and Moments
- 6.6.8. Structure and Development of the Systems Planning
- 6.6.9. Monitoring and Updating

6.7. Security Considerations in the Use of ICT
- 6.7.1. Risk Analysis
- 6.7.2. Information Systems Security
- 6.7.3. Practical Advice

6.8. Feasibility of ICT Project Implementation and Financial Aspects in Information Systems Projects
- 6.8.1. Description and Objectives
- 6.8.2. EVS Participants
- 6.8.3. Techniques and Procedures
- 6.8.4. Cost Structure
- 6.8.5. Financial Projection
- 6.8.6. Budgets

6.9. Intelligence Management
- 6.9.1. What Is Business Intelligence?
- 6.9.2. BI Implementation Strategy
- 6.9.3. Present and Future in BI

6.10. ISO/IEC 12207
- 6.10.1. What is "ISO/IEC 12207"?
- 6.10.2. Information Systems Analysis
- 6.10.3. Information System Design
- 6.10.4. Implementation and Acceptance of the Information System

Module 7. Software Reuse

7.1. General Overview of Software Reuse
- 7.1.1. What Is Software Reuse?
- 7.1.2. Advantages and Disadvantages of Software Reuse
- 7.1.3. Main Software Reuse Techniques

7.2. Introduction to Design Patterns
- 7.2.1. What Is a Design Pattern?
- 7.2.2. Catalog of the Main Design Patterns
- 7.2.3. How to Use Patterns to Solve Design Problems
- 7.2.4. How to Select the Best Design Pattern

7.3. Creation Patterns I
- 7.3.1. Creation Patterns
- 7.3.2. Abstract Factory Pattern
- 7.3.3. Example of Abstract Factory Pattern Implementation
- 7.3.4. Builder Pattern
- 7.3.5. Builder Implementation Example
- 7.3.6. Abstract Factory Pattern vs. Builder

7.4. Creation Patterns II
- 7.4.1. Factory Method Pattern
- 7.4.2. Factory Method vs. Abstract Factory
- 7.4.3. Singleton Pattern

7.5. Structural Patterns I
- 7.5.1. Structural Patterns
- 7.5.2. Adapter Pattern
- 7.5.3. Bridge Pattern

7.6. Structural Patterns II
- 7.6.1. Composite Pattern
- 7.6.2. Decorator Pattern

7.7. Structural Patterns III
- 7.7.1. Facade Pattern
- 7.7.2. Proxy Pattern

7.8. Behavioral Patterns I
- 7.8.1. Concept of Behavioral Patterns
- 7.8.2. Behavior Pattern: Chain of Responsibility
- 7.8.3. Behavior Pattern Order

7.9. Behavioral Patterns II
- 7.9.1. Interpreter Pattern
- 7.9.2. Iterator Pattern
- 7.9.3. Observer Pattern
- 7.9.4. Strategy Pattern

7.10. Frameworks
- 7.10.1. Concept of Framework
- 7.10.2. Development Using Frameworks
- 7.10.3. Model View Controller Pattern
- 7.10.4. Framework for Graphical User Interface Design
- 7.10.5. Frameworks for Web Application Development
- 7.10.6. Frameworks for Managing Object Persistence in Databases

Module 8. Information Technology Services

8.1. Digital Transformation I
- 8.1.1. Business Innovation
- 8.1.2. Production Management
- 8.1.3. Financial Management

8.2. Digital Transformation II
- 8.2.1. Marketing
- 8.2.2. HR Management
- 8.2.3. The Integrated Information System

8.3. Case Study
- 8.3.1. Company Presentation
- 8.3.2. Methodologies to Analyze the Acquisition of IT
- 8.3.3. Determining the Costs, Benefits and Risks
- 8.3.4. Economic Evaluation of Investment

8.4. ICT Governance and Management
- 8.4.1. Definition of IT and Information Systems Governance
- 8.4.2. Difference between IT Systems Governance and Management
- 8.4.3. Framework for IT Systems Governance and Management
- 8.4.4. Regulations and IT Systems Governance and Management

8.5. ICT Corporate Governance
- 8.5.1. What Is Good Corporate Governance?
- 8.5.2. ICT Governance Background
- 8.5.3. The ISO/IEC 38500:2008 Standard
- 8.5.4. Implementation of Good ICT Governance
- 8.5.5. ICT Governance and Best Practices
- 8.5.6. Corporate Governance:Summary and Trends

8.6. Control Objectives for Information and Related Technologies (COBIT)
- 8.6.1. Application Framework
- 8.6.2. Domain: Planning and Organization
- 8.6.3. Domain: Acquisition and Implementation
- 8.6.4. Domain: Delivery and Support
- 8.6.5. Domain: Supervision and Evaluation
- 8.6.6. Application of the COBIT Guide

8.7. The Information Technology Infrastructure Library (ITIL)
- 8.7.1. Introduction to ITIL
- 8.7.2. Service Strategies
- 8.7.3. Service Design
- 8.7.4. Transition between Services
- 8.7.5. Service Operation
- 8.7.6. Improving the Service

8.8. The Service Management System
- 8.8.1. Basic Principles of UNE-ISO/IEC 20000-1
- 8.8.2. The Structure of the ISO/IEC 20000 Regulations
- 8.8.3. Service Management System (SMS) Requirements
- 8.8.4. Design and Transition of New or Modified Services
- 8.8.5. Service Provision Processes
- 8.8.6. Groups of Processes

8.9. The Software Asset Management System
- 8.9.1. Justification of Needs
- 8.9.2. Background
- 8.9.3. Presentation of the 19770 Standard
- 8.9.4. Management Implementation

8.10. Business Continuity Management
- 8.10.1. Business Continuity Plan
- 8.10.2. Implementation of a BCP

Module 9. Information Systems Security

9.1. A global Perspective on Security, Cryptography and Classical Cryptanalysis
- 9.1.1. Computer Security: Historical Perspective
- 9.1.2. What Does "Security" Mean, Exactly?
- 9.1.3. History of Cryptography
- 9.1.4. Substitution Ciphers
- 9.1.5. Case Study: The Enigma Machine

9.2. Symmetric Cryptography
- 9.2.1. Introduction and Basic Terminology
- 9.2.2. Symmetric Encryption
- 9.2.3. Modes of Operation
- 9.2.4. DES
- 9.2.5. The New Advanced Encryption Standard (AES)
- 9.2.6. Encryption in Flow
- 9.2.7. Cryptanalysis

9.3. Asymmetric Cryptography
- 9.3.1. Origins of Public Key Cryptography
- 9.3.2. Basic Concepts and Operation
- 9.3.3. The RSA Algorithm
- 9.3.4. Digital Certificates
- 9.3.5. Key Storage and Management

9.4. Network Attacks
- 9.4.1. Network Threats and Attacks
- 9.4.2. Enumeration
- 9.4.3. Traffic Interception: Sniffers
- 9.4.4. Denial of Service Attacks
- 9.4.5. ARP Cache Poisoning Attacks

9.5. Security Architectures
- 9.5.1. Traditional Security Architectures
- 9.5.2. Secure Socket Layer: SSL
- 9.5.3. SSH Protocol
- 9.5.4. Virtual Private Networks (VPNs)
- 9.5.5. External Storage Unit Protection Mechanisms
- 9.5.6. Hardware Protection Mechanisms

9.6. System Protection Techniques and Secure Code Development
- 9.6.1. Operational Safety
- 9.6.2. Resources and Controls
- 9.6.3. Monitoring
- 9.6.4. Intrusion Detection Systems
- 9.6.5. Host IDS
- 9.6.6. Network IDS
- 9.6.7. Signature-Based IDS
- 9.6.8. Lure Systems
- 9.6.9. Basic Security Principles in Code Development
- 9.6.10. Failure Management
- 9.6.11. Public Enemy Number 1: Buffer Overflows
- 9.6.12. Cryptographic Botches

9.7. Botnets and Spam
- 9.7.1. Origin of the Problem
- 9.7.2. Spam Process
- 9.7.3. Sending Spam
- 9.7.4. Refinement of Mailing Lists
- 9.7.5. Protection Techniques
- 9.7.6. Anti-Spam Service offered by Third Parties
- 9.7.7. Study Cases
- 9.7.8. Exotic Spam

9.8. Web Auditing and Attacks
- 9.8.1. Information Gathering
- 9.8.2. Attack Techniques
- 9.8.3. Tools

9.9. Malware and Malicious Code
- 9.9.1. What Is Malware?
- 9.9.2. Types of Malware
- 9.9.3. Virus
- 9.9.4. Cryptovirus
- 9.9.5. Worms
- 9.9.6. Adware
- 9.9.7. Spyware
- 9.9.8. Hoaxes
- 9.9.9. Phishing
- 9.9.10. Trojans
- 9.9.11. The Malware Economy
- 9.9.12. Possible Solutions

9.10. Forensic Analysis
- 9.10.1. Evidence Collection
- 9.10.2. Evidence Analysis
- 9.10.3. Anti-Forensic Techniques
- 9.10.4. Case Study

Module 10. Project Management

10.1. Fundamental Concepts of Project Management and the Project Management Life Cycle
- 10.1.1. What Is a Project?
- 10.1.2. Common Methodology
- 10.1.3. What Is Project Management?
- 10.1.4. What Is a Project Plan?
- 10.1.5. Benefits
- 10.1.6. Project Life Cycle
- 10.1.7. Process Groups or Project Management Life Cycle
- 10.1.8. The Relationship between Process Groups and Knowledge Areas
- 10.1.9. Relationships between Product and Project Life Cycle

10.2. Start-Up and Planning
- 10.2.1. From the Idea to the Project
- 10.2.2. Development of the Project Record
- 10.2.3. Project Kick-Off Meeting
- 10.2.4. Tasks, Knowledge and Skills in the Start-Up Process
- 10.2.5. The Project Plan
- 10.2.6. Development of the Basic Plan: Steps
- 10.2.7. Tasks, Knowledge and Skills in the Planning Process

10.3. Stakeholders and Outreach Management
- 10.3.1. Identify Stakeholders
- 10.3.2. Develop Plan for Stakeholder Management
- 10.3.3. Manage Stakeholder Engagement
- 10.3.4. Control Stakeholder Engagement
- 10.3.5. The Objective of the Project
- 10.3.6. Scope and Plan Management
- 10.3.7. Gathering Requirements
- 10.3.8. Define the Scope Statement
- 10.3.9. Create the WBS
- 10.3.10. Verify and Control the Scope

10.4. Schedule Development
- 10.4.1. Time and Plan Management
- 10.4.2. Define Activities
- 10.4.3. Establishment of the Sequence of Activities
- 10.4.4. Estimated Resources for Activities
- 10.4.5. Estimated Duration of Activities
- 10.4.6. Schedule Development and Critical Path Calculation
- 10.4.7. Schedule Control

10.5. Budget Development and Risk Response
- 10.5.1. Estimate Costs
- 10.5.2. Develop Budget and S-Curve
- 10.5.3. Cost Control and Earned Value Method
- 10.5.4. Risk Concepts
- 10.5.5. How to Perform a Risk Analysis
- 10.5.6. Response Plan Development

10.6. Quality Management
- 10.6.1. Quality Planning
- 10.6.2. Assuring Quality
- 10.6.3. Quality Control
- 10.6.4. Basic Statistical Concepts
- 10.6.5. Quality Management Tools

10.7. Communication and Human Resources
- 10.7.1. Planning Communications Management
- 10.7.2. Communications Requirements Analysis
- 10.7.3. Communication Technology
- 10.7.4. Communication Models
- 10.7.5. Communication Methods
- 10.7.6. Communications Management Plan
- 10.7.7. Communications Management
- 10.7.8. Human Resources Management
- 10.7.9. Main Stakeholders and their Roles in the Projects
- 10.7.10. Types of Organization
- 10.7.11. Project Organization
- 10.7.12. The Work Equipment

10.8. Procurement
- 10.8.1. The Procurement Process
- 10.8.2. Planning
- 10.8.3. Search for Suppliers and Request for Quotations
- 10.8.4. Contract Allocation
- 10.8.5. Contract Administration
- 10.8.6. Contracts
- 10.8.7. Types of Contracts
- 10.8.8. Contract Negotiation

10.9. Execution, Monitoring and Control and Closure
- 10.9.1. Process Groups
- 10.9.2. Project Execution
- 10.9.3. Project Monitoring and Control
- 10.9.4. Project Closure

10.10. Professional Responsibility
- 10.10.1. Professional Responsibility
- 10.10.2. Characteristics of Social and Professional Responsibility
- 10.10.4. Liability vs. PMP®
- 10.10.5. Examples of Liability
- 10.10.6. Benefits of Professionalization

05

Methodology

This academic program offers students a different way of learning. Our methodology uses a cyclical learning approach: **Relearning.**

This teaching system is used, for example, in the most prestigious medical schools in the world, and major publications such as the **New England Journal of Medicine** have considered it to be one of the most effective.

Discover Relearning, a system that abandons conventional linear learning, to take you through cyclical teaching systems: a way of learning that has proven to be extremely effective, especially in subjects that require memorization"

Case Study to contextualize all content

Our program offers a revolutionary approach to developing skills and knowledge. Our goal is to strengthen skills in a changing, competitive, and highly demanding environment.

At TECH, you will experience a learning methodology that is shaking the foundations of traditional universities around the world"

You will have access to a learning system based on repetition, with natural and progressive teaching throughout the entire syllabus.

The student will learn to solve complex situations in real business environments through collaborative activities and real cases.

A learning method that is different and innovative

This TECH program is an intensive educational program, created from scratch, which presents the most demanding challenges and decisions in this field, both nationally and internationally. This methodology promotes personal and professional growth, representing a significant step towards success. The case method, a technique that lays the foundation for this content, ensures that the most current economic, social and professional reality is taken into account.

> *Our program prepares you to face new challenges in uncertain environments and achieve success in your career"*

The case method has been the most widely used learning system among the world's leading Information Technology schools for as long as they have existed. The case method was developed in 1912 so that law students would not only learn the law based on theoretical content. It consisted of presenting students with real-life, complex situations for them to make informed decisions and value judgments on how to resolve them. In 1924, Harvard adopted it as a standard teaching method.

What should a professional do in a given situation? This is the question that you are presented with in the case method, an action-oriented learning method. Throughout the course, students will be presented with multiple real cases. They will have to combine all their knowledge and research, and argue and defend their ideas and decisions.

Relearning Methodology

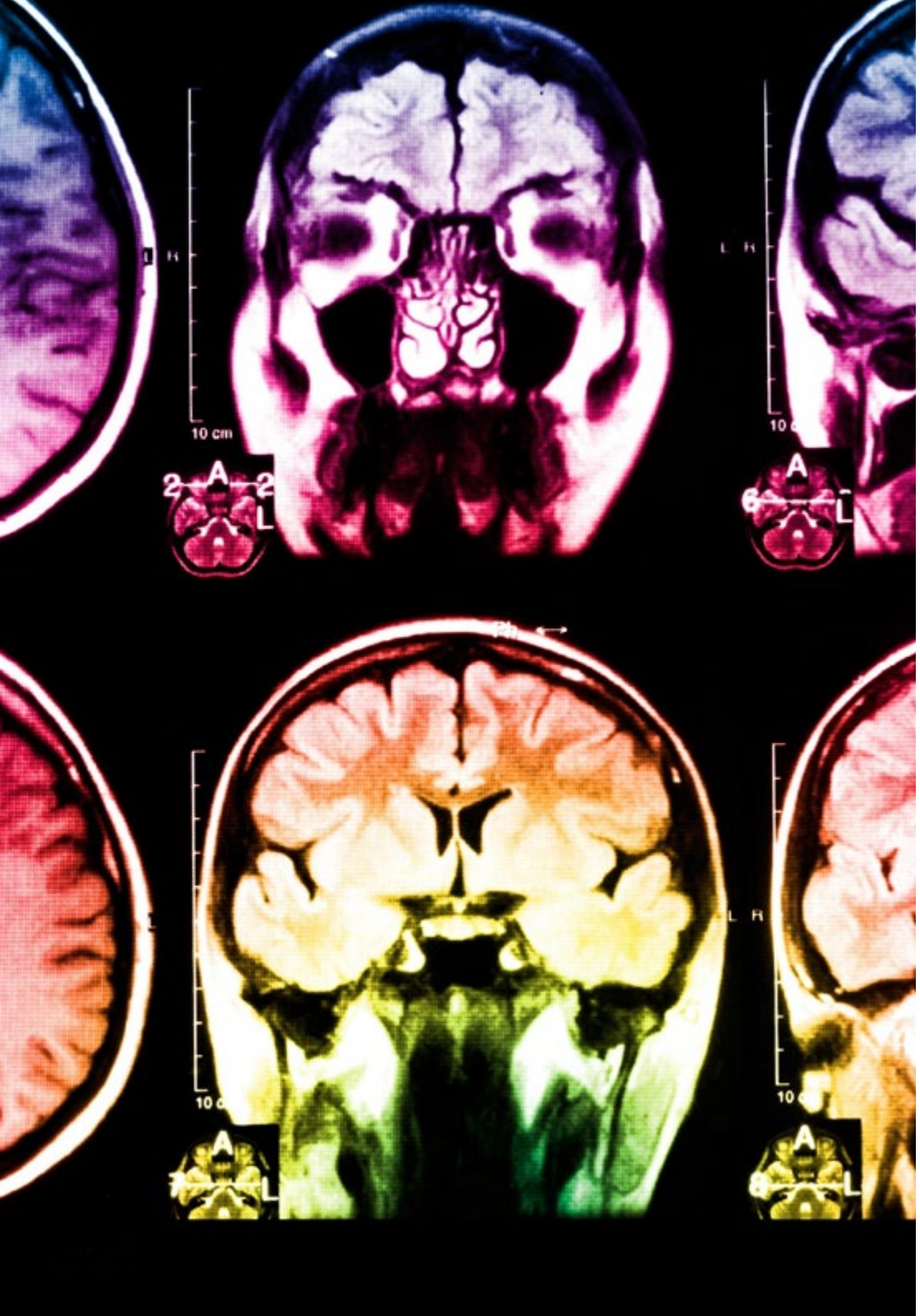
TECH effectively combines the Case Study methodology with a 100% online learning system based on repetition, which combines different teaching elements in each lesson.

We enhance the Case Study with the best 100% online teaching method: Relearning.

In 2019, we obtained the best learning results of all online universities in the world.

At TECH you will learn using a cutting-edge methodology designed to train the executives of the future. This method, at the forefront of international teaching, is called Relearning.

Our university is the only one in the world authorized to employ this successful method. In 2019, we managed to improve our students' overall satisfaction levels (teaching quality, quality of materials, course structure, objectives...) based on the best online university indicators.



In our program, learning is not a linear process, but rather a spiral (learn, unlearn, forget, and re-learn). Therefore, we combine each of these elements concentrically. This methodology has trained more than 650,000 university graduates with unprecedented success in fields as diverse as biochemistry, genetics, surgery, international law, management skills, sports science, philosophy, law, engineering, journalism, history, and financial markets and instruments. All this in a highly demanding environment, where the students have a strong socio-economic profile and an average age of 43.5 years.

Relearning will allow you to learn with less effort and better performance, involving you more in your training, developing a critical mindset, defending arguments, and contrasting opinions: a direct equation for success.

From the latest scientific evidence in the field of neuroscience, not only do we know how to organize information, ideas, images and memories, but we know that the place and context where we have learned something is fundamental for us to be able to remember it and store it in the hippocampus, to retain it in our long-term memory.

In this way, and in what is called neurocognitive context-dependent e-learning, the different elements in our program are connected to the context where the individual carries out their professional activity.

This program offers the best educational material, prepared with professionals in mind:

Study Material

All teaching material is produced by the specialists who teach the course, specifically for the course, so that the teaching content is highly specific and precise.

These contents are then applied to the audiovisual format, to create the TECH online working method. All this, with the latest techniques that offer high quality pieces in each and every one of the materials that are made available to the student.

Classes

There is scientific evidence suggesting that observing third-party experts can be useful.

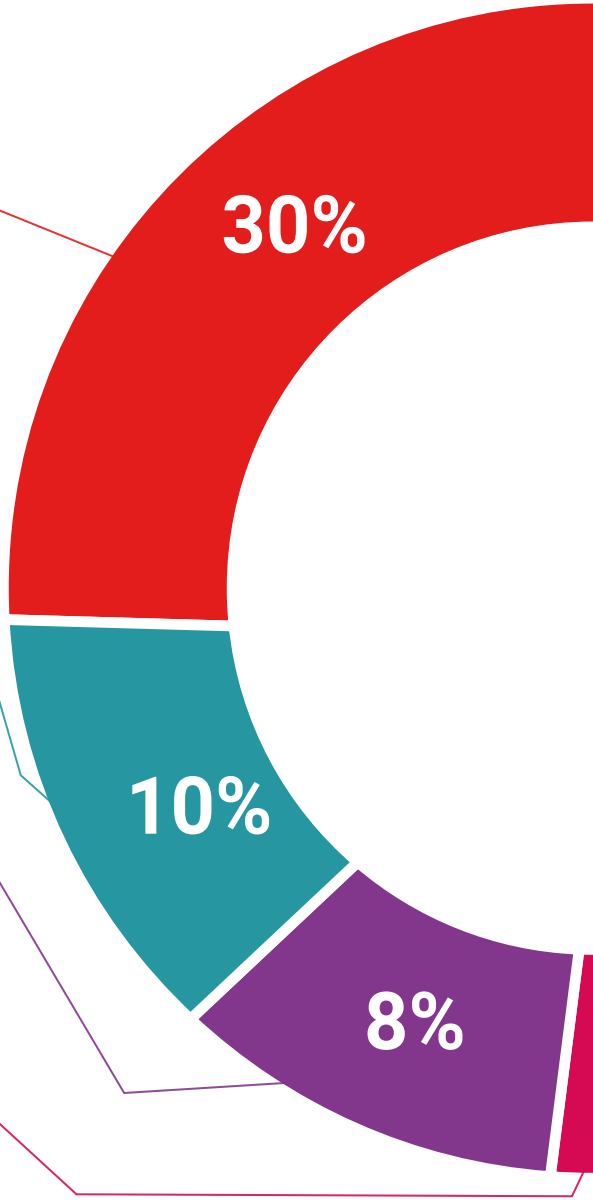
Learning from an Expert strengthens knowledge and memory, and generates confidence in future difficult decisions.

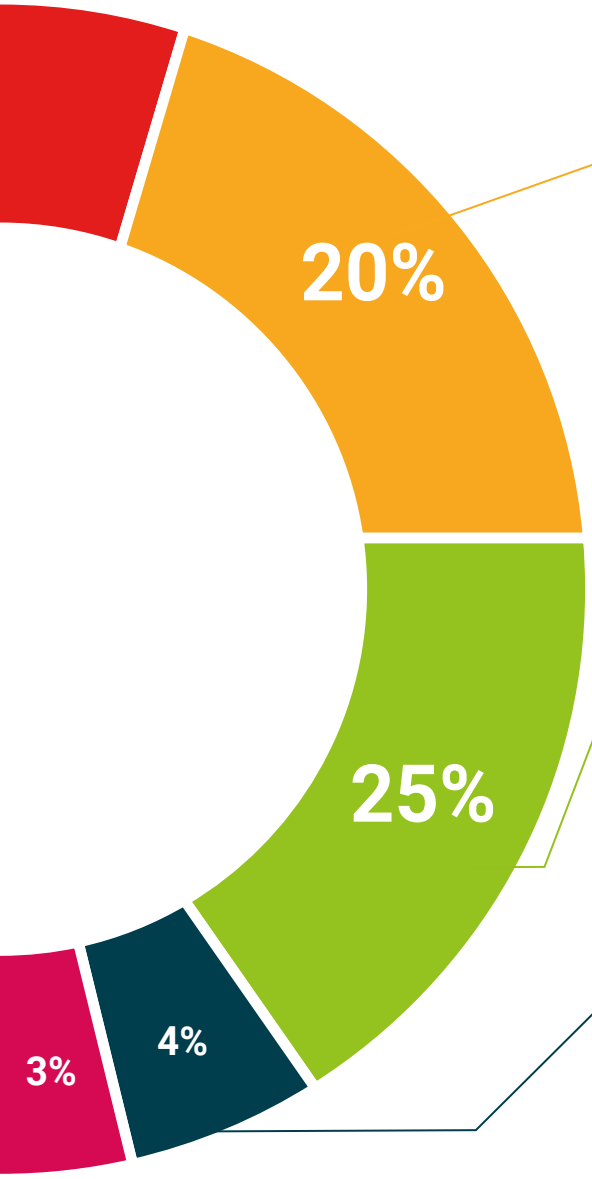
Practising Skills and Abilities

They will carry out activities to develop specific skills and abilities in each subject area. Exercises and activities to acquire and develop the skills and abilities that a specialist needs to develop in the context of the globalization that we are experiencing.

Additional Reading

Recent articles, consensus documents and international guidelines, among others. In TECH's virtual library, students will have access to everything they need to complete their course.





Case Studies

Students will complete a selection of the best case studies chosen specifically for this program. Cases that are presented, analyzed, and supervised by the best specialists in the world.

Interactive Summaries

The TECH team presents the contents attractively and dynamically in multimedia lessons that include audio, videos, images, diagrams, and concept maps in order to reinforce knowledge.

This exclusive educational system for presenting multimedia content was awarded by Microsoft as a "European Success Story".

Testing & Retesting

We periodically evaluate and re-evaluate students' knowledge throughout the program, through assessment and self-assessment activities and exercises, so that they can see how they are achieving their goals.

06

Certificate

The Professional Master's Degree in Advanced Software Engineering guarantees students, in addition to the most rigorous and up-to-date education, access to a Professional Master's Degree issued by TECH Technological University.

Successfully complete this program and receive your university qualification without having to travel or fill out laborious paperwork"

This **Professional Master's Degree in Advanced Software Engineering** contains the most complete and up-to-date program on the market.

After the student has passed the assessments, they will receive their corresponding Professional Master's Degree diploma issued by **TECH Technological University** via tracked delivery*.

The certificate issued by **TECH Technological University** will reflect the qualification obtained in the Professional Master's Degree, and meets the requirements commonly demanded by labor exchanges, competitive examinations and professional career evaluation committees.

Title: **Professional Master's Degree in Advanced Software Engineering**
Official Nº of Hours: **1,500 h.**

tech technological university

Awards the following

CERTIFICATE

to

Mr./Ms. _____, with identification number _____.
For having successfully passed and accredited the following program

PROFESSIONAL MASTER'S DEGREE
in
Advanced Software Engineering

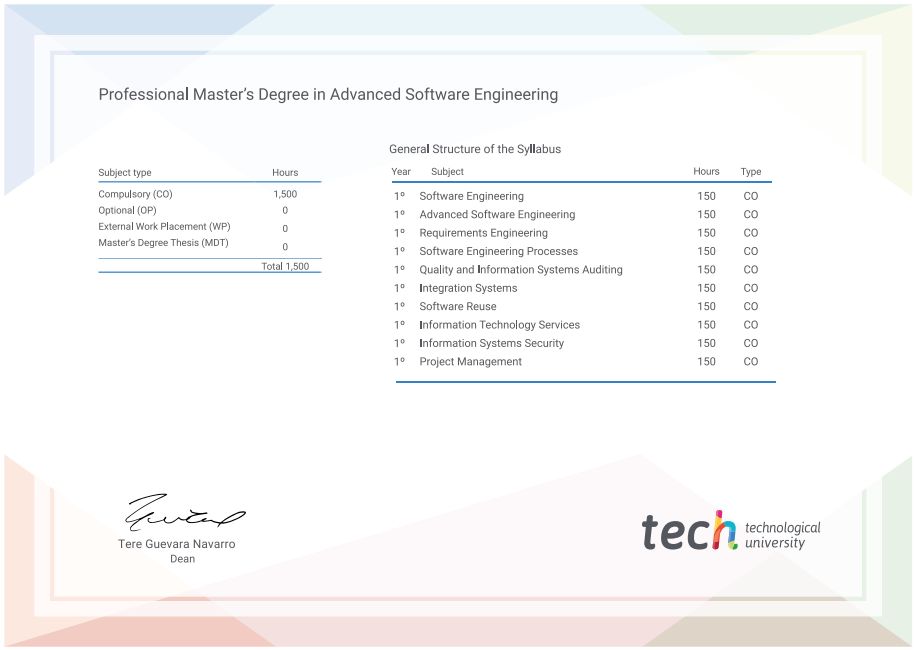
This is a qualification awarded by this University, equivalent to 1,500 hours, with a start date of dd/mm/yyyy and an end date of dd/mm/yyyy.

TECH is a Private Institution of Higher Education recognized by the Ministry of Public Education as of June 28, 2018.

June 17, 2020

Tere Guevara Navarro
Dean

Unique TECH Code: AFWORD23S techtitute.com/certificates


Professional Master's Degree in Advanced Software Engineering

Subject type	Hours
Compulsory (CO)	1,500
Optional (OP)	0
External Work Placement (WP)	0
Master's Degree Thesis (MDT)	0
	Total 1,500

General Structure of the Syllabus

Year	Subject	Hours	Type
1º	Software Engineering	150	CO
1º	Advanced Software Engineering	150	CO
1º	Requirements Engineering	150	CO
1º	Software Engineering Processes	150	CO
1º	Quality and Information Systems Auditing	150	CO
1º	Integration Systems	150	CO
1º	Software Reuse	150	CO
1º	Information Technology Services	150	CO
1º	Information Systems Security	150	CO
1º	Project Management	150	CO

Tere Guevara Navarro
Dean

tech technological university

*Apostille Convention. In the event that the student wishes to have their paper certificate issued with an apostille, TECH EDUCATION will make the necessary arrangements to obtain it, at an additional cost.

tech technological university

Professional Master's Degree
Advanced Software Engineering

- Modality: **online**
- Duration: **12 months**
- Certificate: **TECH Technological University**
- Dedication: **16h/week**
- Schedule: **at your own pace**
- Exams: **online**

Professional Master's Degree

Advanced Software Engineering

tech technological university