高级硕士
软件质量与工程

tech 科学技术大学

tech 科学技术大学

高级硕士
软件质量与工程

- 模式:**在线**
- 时长: **2年**
- 学位: **TECH 科技大学**
- 课程表:**自由安排时间**
- 考试模式:**在线**

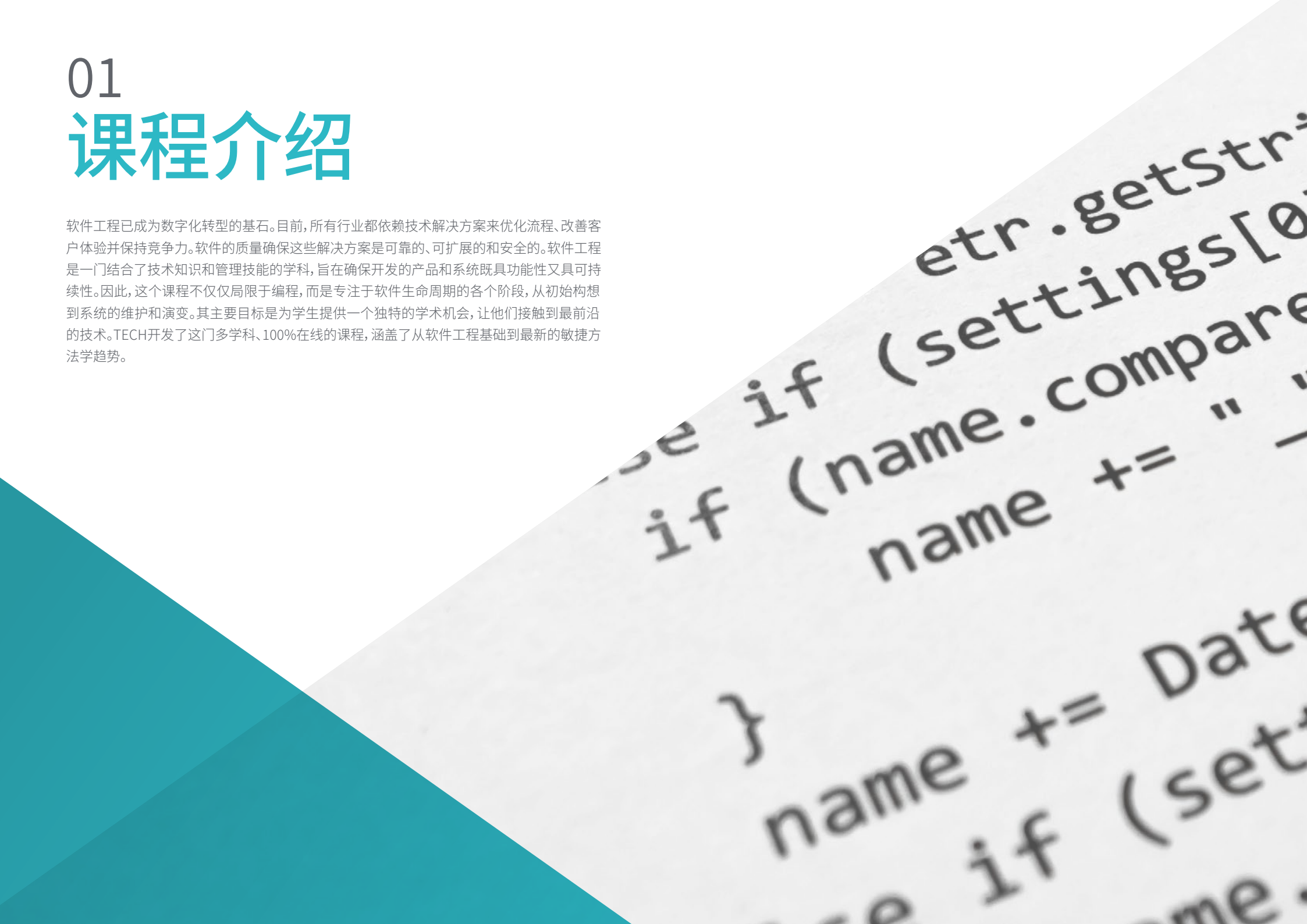
网页链接: www.techtitute.com/cn/information-technology/advanced-master-degree/advanced-master-degree-software-engineering-quality

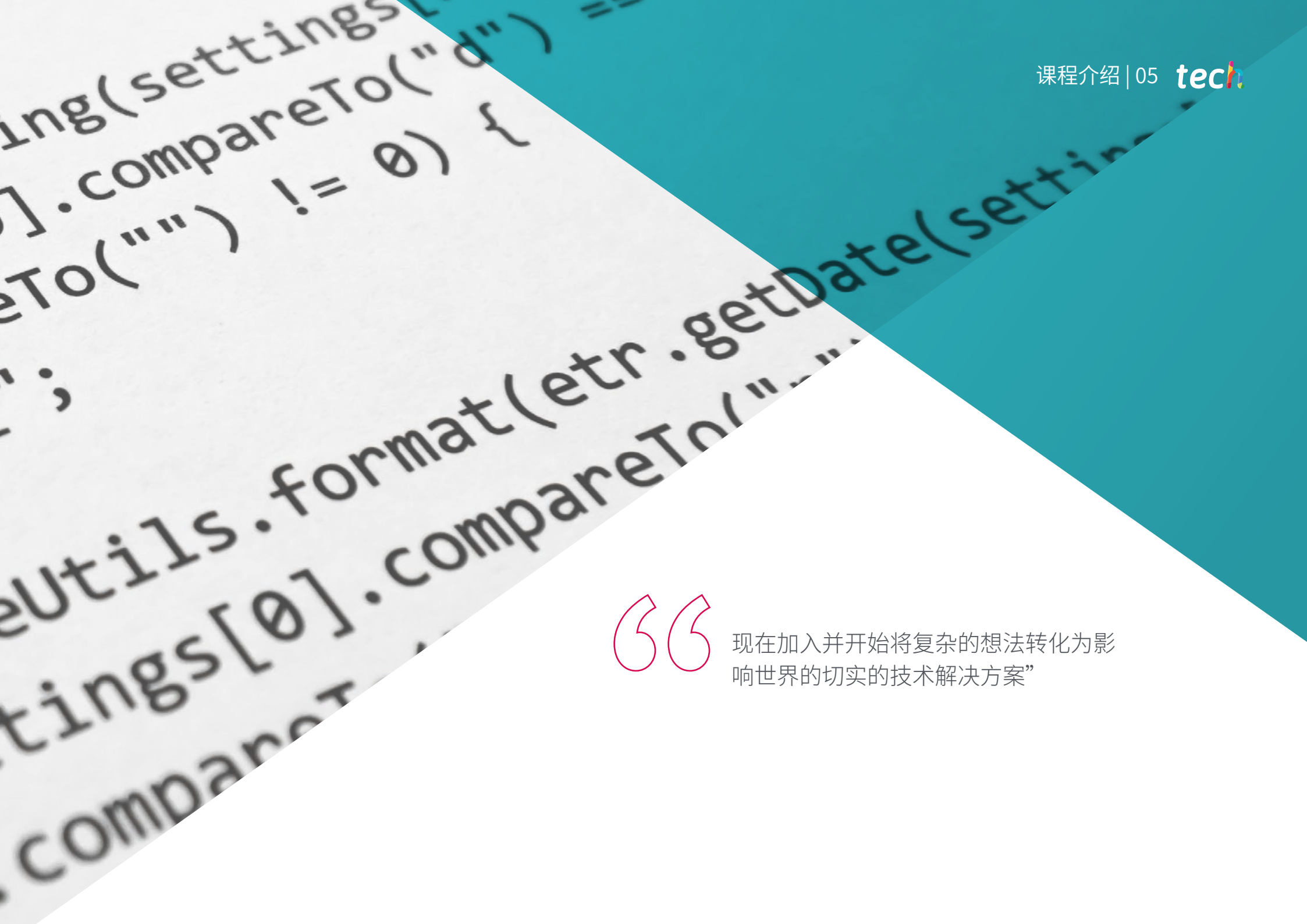
目录

01

课程介绍

软件工程已成为数字化转型的基石。目前，所有行业都依赖技术解决方案来优化流程、改善客户体验并保持竞争力。软件的质量确保这些解决方案是可靠的、可扩展的和安全的。软件工程是一门结合了技术知识和管理技能的学科，旨在确保开发的产品和系统既具功能性又具可持续性。因此，这个课程不仅仅局限于编程，而是专注于软件生命周期的各个阶段，从初始构想到系统的维护和演变。其主要目标是为学生提供一个独特的学术机会，让他们接触到最前沿的技术。TECH开发了这门多学科、100%在线的课程，涵盖了从软件工程基础到最新的敏捷方法学趋势。





现在加入并开始将复杂的想法转化为影响世界的切实的技术解决方案”

软件质量确保系统不仅满足功能要求，而且直观、安全和长期可持续。这一方面对于金融、医疗或交通等关键领域尤其重要，因为这些领域的故障可能导致严重后果。此外，通过优先考虑质量，公司可以确保能够快速适应不断的技术进步并有效响应不断增长的市场需求。

通过使用敏捷开发，DevOps和实施国际质量标准等方法，软件工程保证在更短的时间内交付产品。此外，随着人工智能，cloud computing 和网络安全等新兴技术的融合，成本控制和最大限度减少关键错误的质量水平得到了提高。在此背景下，TECH设计的课程旨在培养软件设计、开发、管理和质量保证方面的高素质专业人才。为了获得必要的技能，高级硕士的教学大纲包括有关技术项目管理和战略管理的最新概念。这种方法对于已经担任重要职务并希望更新知识的工程师以及首次渴望领导该领域团队和项目的工程师来说都具有附加价值。

该课程的主要优势之一是它将 100% 在线，无需出门或调整特定的时间表。此外，学生还可以采用适合自己学习进度的Relearning学习方法。这种灵活的方法非常有用，因为它允许学生有效地组织他们的日常义务无论是专业还是家庭相关的，从而实现全面发展。

此项**软件质量与工程高级硕士** 是市场上最完整和有质量的方案。主要特点是：

- 由计算机专家提出的实际案例的发展
- 内容图文并茂,示意性强,实用性强,为那些视专业实践至关重要的学科提供了科学和实用的信息
- 进行自我评估以改善学习的实践练习
- 特别强调软件质量与工程方面的创新方法
- 理论知识,专家预论,争议主题讨论论坛和个人反思工作
- 可以通过任何连接互联网的固定或便携设备访问课程内容

> 通过 TECH，你不仅可以学习开发软件，还可以创建能够改变人们和公司生活的系统”

通过当前学术领域最具创新性的教学方法掌握最先进的工程技能和工具”

通过 100% 在线学习来提高您的专业期望，同时不会干扰您的个人和家庭责任。

成为一名专业的工程领导者，随时准备在世界各地学习。

教学人员包括来自计算机领域的专业人员，以及来自主要协会和著名大学的公认专家。他们将自己的工作经验带到课程中。

通过采用最新的教育技术制作的多媒体内容，专业人士将能够进行情境化学习，即通过模拟环境进行沉浸式培训，以应对真实情况。

这个课程的设计重点是基于问题的学习，通过这种方式，学生必须尝试解决整个学术课程中提出的不同专业实践情况。为此，职业人士将得到由著名专家开发的创新互动视频系统的协助。

02

为什么在TECH学习?

TECH 是世界上最大的数字大学。拥有超过14,000个大学课程的令人印象深刻的目录，涵盖11种语言，我们以就业率99%的领先地位跻身行业前列。此外，超过6,000名享有国际声誉的顶尖教授团队。

在世界上最大的数字大学学习并确保您的职业成功。未来始于TECH”

福布斯评选的全球最佳在线大学

著名的商业和金融杂志福布斯将泰晤士河科技大学评为《世界上最好的在线大学》。他们在数字版最近的一篇文章中提到了这一点，并在文中重复了这所学校的成功故事，«这要归功于它提供的学术课程，精选的师资队伍以及旨在培养未来专业人员的创新学习方法»

最好的国际教学团队

TECH 的教学人员由 6,000 多名具有最高国际声望的教授组成。教授、研究人员和跨国公司高层管理人员，其中包括：Isaiah Covington，波士顿凯尔特人队的表现教练；Magda Romanska，哈佛MetaLAB的首席研究员；Ignacio Wistumba，MD安德森癌症中心转化分子病理学部门的主席；以及D.W Pine，TIME杂志的创意总监等。

世界上最大的数字化大学

TECH 是世界上最大的数字大学。我们是最大的教育机构，拥有最好，最广泛的数字课程目录，100%在线且涵盖绝大多数知识领域。我们提供世界上最多的自主学位、官方研究生学位和本科学位。总共有超过 14,000 个大学学位，涵盖十种不同的语言，使我们成为世界上最大的教育机构。

大学里最全面的学习计划

TECH 提供大学中最全面的课程，其主题涵盖基本概念以及特定科学领域的主要科学进步。这些课程也不断更新，以确保学生拥有最先进的学术技能和最需要的专业技能。通过这种方式，大学学位为毕业生在职业成功道路上提供了显著的优势。

独特的学习方法

TECH 是第一所在所有学位中采用Relearning的大学。这是最好的在线学习方法，获得著名教育机构提供的国际教学质量认证。并且，这一颠覆性的学术模式与“案例教学法”相辅相成，构成了独特的在线教学策略。还提供创新的教料，包括详细的视频，信息图表和交互式摘要。

NBA 官方在线大学

TECH是NBA的官方在线大学。由于与主要篮球联盟达成协议，该校为学生提供独家大学课程，以及专注于联盟业务和体育产业其他领域的各种教料。每个课程都有独特设计的课程设置，并邀请了杰出的演讲嘉宾：这些职业运动员具有卓越的运动经历，将分享他们在相关主题上的经验。

就业率领先者

TECH 已成功成为就业能力领先的大学。99%的学生在完成大学课程后不到一年时间，就能在所学专业领域找到工作。同样多的人也成功地立即提升了自己的职业生涯。这一切都归功于一种学习方法，该方法的有效性基于掌握专业发展所必需的实践技能。

Google Partner Premier

这家北美科技巨头已向 TECH 授予Google Partner Premier徽章。该奖项仅授予全球 3% 的公司，凸显了该大学为学生提供的有效，灵活和定制的体验。这一认可不仅认可了 TECH 数字基础设施的最高严谨性，性能和投资，而且还使该大学成为世界上最前沿的科技公司之一。

被学生评价为最佳大学

学生们将TECH定位为世界上在主要评价平台上评价最高的大学，突出其获得的4.9分（满分5分）的高评价，这一评分来自于超过1,000条评论。这些成绩巩固了TECH作为国际标杆大学机构的地位，反映了其教育模式的卓越性和积极影响。

03

教学大纲

软件质量与工程高级硕士课程旨在为软件工程的所有关键领域提供全面和高级的专业化。第一个模块侧重于基础知识，涵盖从软件设计和需求管理到技术架构和敏捷方法的所有内容。随着课程的进展，学生将深入研究更专业的领域，例如测试自动化，持续集成和质量保证。此外，课程还包括技术项目管理等科目，参与者可学习如何领导多学科团队。

这个高级硕士将帮助您成为软件质量与工程领域中与众不同的专家”

模块 1. 软件的质量TRL发展水平

1.1. 影响软件质量的要素 (I) 技术债务
- 1.1.1. 技术债务原因和后果
- 1.1.2. 软件的质量总体原则
- 1.1.3. 无原则和有原则的高质量软件
  - 1.1.3.1. 后果
  - 1.1.3.2. 在软件中应用质量原则的必要性
- 1.1.4. 软件的质量分类
- 1.1.5. 优质软件具体特点

1.2. 影响软件质量的要素 (II) 相关的费用
- 1.2.1. 软件的质量影响的因素
- 1.2.2. 软件的质量错误的想法
- 1.2.3. 软件的质量相关的费用

1.3. 软件质量模型 (I) :知识管理
- 1.3.1. 一般质量模型
  - 1.3.1.1. 全面质量管理
  - 1.3.1.2. 欧洲商业卓越模式 (EFQM)
  - 1.3.1.3. 六西格玛模型
- 1.3.2. 知识管理模型
  - 1.3.2.1. 迪巴模型
  - 1.3.2.2. Seks模型
- 1.3.3. 经验和 QIP 范式
- 1.3.4. 使用的质量模型 (25010)

1.4. 软件质量模型 (III) :数据、流程和 SEI 模型的质量
- 1.4.1. 数据质量模型
- 1.4.2. 软件过程建模
- 1.4.3. Software & Systems Process Engineering Metamodel Specification (SPEM)
- 1.4.4. SEI 模型
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. ISO 软件质量标准 (I)标准分析
- 1.5.1. ISO 9000 标准
  - 1.5.1.1. ISO 9000 标准
  - 1.5.1.2. ISO 质量标准系列 (9000)
- 1.5.2. 其他与质量相关的 ISO 标准
- 1.5.3. 质量建模标准 (ISO 2501)
- 1.5.4. 质量测量标准 (ISO 2502n)

1.6. ISO 软件质量标准 (II)要求和评估
- 1.6.1. 质量要求标准 (2503n)
- 1.6.2. 质量评估规范 (2504n)
- 1.6.3. ISO/IEC 24744.2007

1.7. TRL发展水平 (I) 1 至 4 级
- 1.7.1. TRL 水平
- 1.7.2. 第一级:基础原则
- 1.7.3. 第二级:概念和/或应用
- 1.7.4. 第三级:分析关键功能
- 1.7.5. 第四级:实验室环境的组件验证

1.8. TRL发展水平 (II) 5 至 9 级
- 1.8.1. 第五级:相关环境的组件验证
- 1.8.2. 第六级:系统/子系统模型
- 1.8.3. 第七级:真实环境演示
- 1.8.4. 第八级:完整且经过认证的系统
- 1.8.5. 第九级:真实环境中的成功

1.9. TRL发展水平用途
- 1.9.1. 具有实验环境的公司示例
- 1.9.2. I+D+i公司的例子
- 1.9.3. 工业I+D+i 公司的例子
- 1.9.4. 联合实验室工程公司的例子

1.10. 软件的质量关键细节
- 1.10.1. 方法的细节
- 1.10.2. 技术细节
- 1.10.3. 项目管理软件的详细信息
  - 1.10.3.1. 计算机系统的质量
  - 1.10.3.2. 软件产品质量
  - 1.10.3.3. 软件过程质量

模块 2. 软件项目的开发功能和技术文档

2.1. 项目管理
- 2.1.1. 软件质量项目管理
- 2.1.2. 项目管理优势
- 2.1.3. 项目管理分类

2.2. 项目管理的方法
- 2.2.1. 项目管理的方法
- 2.2.2. 项目的方法分类
- 2.2.3. 项目管理的方法用处

2.3. 需求识别阶段
- 2.3.1. 项目需求的识别
- 2.3.2. 管理项目的会议
- 2.3.3. 要提供的文件

2.4. 模型
- 2.4.1. 初始阶段
- 2.4.2. 分析阶段
- 2.4.3. 建设阶段
- 2.4.4. 测试阶段
- 2.4.5. 交付

2.5. 使用的数据模型
- 2.5.1. 新数据模型的确定
- 2.5.2. 确定数据迁移计划
- 2.5.3. 数据集

2.6. 对其他项目的影响
- 2.6.1. 一个项目的影响实例

2.7. 项目的MUST
- 2.7.1. 项目的MUST
- 2.7.2. 项目MUST的识别
- 2.7.3. 确定项目交付的执行点

2.8. 项目建设团队
- 2.8.1. 根据项目的干预角色
- 2.8.2. 与人力资源部联系以进行招聘
- 2.8.3. 可交付成果和项目进度表

2.9. 软件项目的技术方面
- 2.9.1. 项目架构师技术方面
- 2.9.2. 技术负责人
- 2.9.3. 软件项目的建构
- 2.9.4. 代码质量评估Sonar

2.10. 项目交付
- 2.10.1. 功能分析
- 2.10.2. 数据模型
- 2.10.3. 状态图
- 2.10.4. 技术文档

模块 3. 软件测试测试自动化

3.1. 软件质量模型
- 3.1.1. 产品质量
- 3.1.2. 过程质量
- 3.1.3. 使用质量

3.2. 过程质量
- 3.2.1. 过程质量
- 3.2.2. 成熟度模型
- 3.2.3. ISO 15504 标准
  - 3.2.3.1.目的
  - 3.2.3.2.背景
  - 3.2.3.3.阶段

3.3. ISO/IEC 15504 标准
- 3.3.1. 过程类别
- 3.3.2. 开发过程。例子
- 3.3.3. 个人资料片段
- 3.3.4. 阶段

3.4. CMMI (Capability Maturity Model Integration)
- 3.4.1. CMMI能力成熟度模型的集成
- 3.4.2. 模型和区域分类
- 3.4.3. 工艺领域
- 3.4.4. 能力级别
- 3.4.5. 流程管理
- 3.4.6. 项目管理

3.5. 变更管理和存储库
- 3.5.1. 软件变更管理
  - 3.5.1.1. 配置项持续整合
  - 3.5.1.2. 线条
  - 3.5.1.3. 流程图
  - 3.5.1.4. 分支
- 3.5.2. 存储库
  - 3.5.2.1.版本控制
  - 3.5.2.2.工作组和存储库的使用
  - 3.5.2.3.存储库的持续集成

3.6. Team Foundation Server (TFS)
- 3.6.1. 安装和配置
- 3.6.2. 创建团队项目
- 3.6.3. 将内容添加到控制源代码
- 3.6.4. 云上 TFS

3.7. 测试
- 3.7.1. 测试的动机
- 3.7.2. 验证测试
- 3.7.3. 贝塔测试
- 3.7.4. 实施与维护

3.8. 负载测试
- 3.8.1. 负载测试
- 3.8.2. 使用 LoadView进行测试
- 3.8.3. 使用K6 Cloud 进行测试
- 3.8.4. 用Loader测试

3.9. 单位压力和耐力测试
- 3.9.1. 单元测试的动机
- 3.9.2. unit testing的工具
- 3.9.3. 压力测试的动机
- 3.9.4. 使用压力测试进行测试
- 3.9.5. 耐力测试的动机
- 3.9.6. 使用LoadRunner进行测试

3.10. 可扩展性可扩展的软件设计
- 3.10.1. 软件的扩展性和架构
- 3.10.2. 层与层之间的独立性
- 3.10.3. 层之间的耦合 建构模式

模块 4. 架构模式瀑布模型方法与敏捷方法对比

4.1. 瀑布式方法
- 4.1.1. 瀑布式方法
- 4.1.2. 瀑布式方法对软件质量的影响
- 4.1.3. 瀑布式方法实例

4.2. 敏捷方法论
- 4.2.1. 敏捷方法论
- 4.2.2. 敏捷方法对软件质量的影响
- 4.2.3. 敏捷方法实例

4.3. SCRUM方法论
- 4.3.1. SCRUM方法论
- 4.3.2. SCRUM 宣言
- 4.3.3. Scrum 应用程序

4.4. Kanban版面
- 4.4.1. Kanban方法
- 4.4.2. Kanban版面
- 4.4.3. Kanban版面应用示例

4.5. 瀑布项目的管理
- 4.5.1. 项目的阶段
- 4.5.2. 瀑布项目的愿景
- 4.5.3. 需要考虑的可交付成果

4.6. SCRUM 的项目管理
- 4.6.1. SCRUM 项目的阶段
- 4.6.2. SCRUM 项目的愿景
- 4.6.3. 需要考虑的可交付成果

4.7. 瀑布比SCRUM。比较
- 4.7.1. 试点项目提案
- 4.7.2. 应用瀑布的项目例子
- 4.7.3. 应用SCRUM的项目例子

4.8. 客户的看法
- 4.8.1. 瀑布的文件
- 4.8.2. SCRUM的文件
- 4.8.3. 比较

4.9. Kanban结构
- 4.9.1. 用户故事
- 4.9.2. Backlog
- 4.9.3. Kanban分析

4.10. 混合项目
- 4.10.1. 项目建设
- 4.10.2. 项目管理
- 4.10.3. 需要考虑的可交付成果

模块 5. TDD (Test Driven Developement).测试驱动的软件设计

5.1. TDD. Test Driven Development
- 5.1.1. TDD. Test Driven Development
- 5.1.2. TDD.TDD 对质量的影响
- 5.1.3. 基于测试的设计和开发。实例

5.2. TDD周期
- 5.2.1. 选择需求
- 5.2.2. 测试。类型
  - 5.2.2.1.单一测试
  - 5.2.2.2.集成测试
  - 5.2.2.3.End To End测试
- 5.2.3. 测试验证故障
- 5.2.4. 创建实施
- 5.2.5. 运行自动化测试
- 5.2.6. 消除复制
- 5.2.7. 需求清单更新
- 5.2.8. 重复循环 TDD
- 5.2.9. TDD周期理论-实践示例

5.3. TDD 实施策略
- 5.3.1. 虚假实施
- 5.3.2. 三角实施
- 5.3.3. 显而易见的实施

5.4. TDD.用途优势和劣势
- 5.4.1. 使用的优势
- 5.4.2. 使用的限制
- 5.4.3. 实施的质量平衡

5.5. TDD.最佳实践
- 5.5.1. TDD 规则
- 5.5.2. 规则 1：在生产中编码之前进行失败的预测试
- 5.5.3. 规则 2：不要编写多个单元测试
- 5.5.4. 规则 3：不要编写不必要的代码
- 5.5.5. TDD 中要避免的错误和反模式

5.6. 使用TDD模拟真实项目 (一)
- 5.6.1. 项目概况 (A公司)
- 5.6.2. TDD 的应用
- 5.6.3. 拟议的练习
- 5.6.4. 练习反馈信息

5.7. 使用TDD模拟真实项目 (II)
- 5.7.1. 项目概况 (B公司)
- 5.7.2. TDD 的应用
- 5.7.3. 拟议的练习
- 5.7.4. 练习反馈信息

5.8. 使用TDD模拟真实项目 (III)
- 5.8.1. 项目概况 (C公司)
- 5.8.2. TDD 的应用
- 5.8.3. 拟议的练习
- 5.8.4. 练习反馈信息

5.9. TDD 的替代方案测试驱动开发
- 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.理论比较

5.10. TDD TCR, BDD 和 ATDD.实际比较
- 5.10.1. 定义问题
- 5.10.2. TCR 分辨率
- 5.10.3. BDD 分辨率
- 5.10.4. ATDD 分辨率

模块 6. DevOps.软件质量管理

6.1. DevOps.软件质量管理
- 6.1.1. DevOps
- 6.1.2. DevOps 和软件质量
- 6.1.3. DevOps.DevOps 文化的好处

6.2. DevOps.与敏捷的关系
- 6.2.1. 加急交付
- 6.2.2. 质量
- 6.2.3. 降低成本

6.3. DevOps 调试
- 6.3.1. 识别问题
- 6.3.2. 在公司实施
- 6.3.3. 实施指标

6.4. 软件交付周期
- 6.4.1. 设计方法
- 6.4.2. 合约
- 6.4.3. 路线图

6.5. 无错误代码开发
- 6.5.1. 可维护的代码
- 6.5.2. 发展模式
- 6.5.3. 代码测试
- 6.5.4. 代码级别的软件开发最佳实践

6.6. 自动化
- 6.6.1. 自动化证据的类型
- 6.6.2. 自动化和维护成本
- 6.6.3. 自动化减轻错误

6.7. 部署
- 6.7.1. 目标评估
- 6.7.2. 设计一个自动和适应的过程
- 6.7.3. 反馈和响应

6.8. 计算机安全
- 6.8.1. 对事件发生的防范
- 6.8.2. 事件分析与解决
- 6.8.3. 如何避免未来的错误

6.9. 自动化部署
- 6.9.1. 准备自动部署
- 6.9.2. 自动过程健康评估
- 6.9.3. 指标和回滚能力

6.10. 好的做法DevOps 的演进
- 6.10.1. 应用 DevOps 的良好实践指南
- 6.10.2. DevOps.团队的方法
- 6.10.3. 避开利基

模块 7. DevOps 和持续集成软件开发中先进的实用解决方案

7.1. 软件交付流程
- 7.1.1. 识别参与者和工件
- 7.1.2. 软件交付流程设计
- 7.1.3. 软件交付流程阶段之间的要求

7.2. 过程自动化
- 7.2.1. 持续整合
- 7.2.2. 持续部署
- 7.2.3. 设置环境和管理机密

7.3. 声明式管道
- 7.3.1. 传统管道之间的差异，例如代码和声明式的
- 7.3.2. 声明式管道
- 7.3.3. Jenkins 的声明式管道
- 7.3.4. 持续集成供应商的比较

7.4. 质量门和丰富的反馈
- 7.4.1. 质量门
- 7.4.2. 质量门的质量标准。维护
- 7.4.3. 集成请求的业务需求

7.5. 工件管理
- 7.5.1. 工件和生命周期
- 7.5.2. 工件存储和管理系统
- 7.5.3. 工件管理安全

7.6. 持续部署
- 7.6.1. 作为容器持续部署
- 7.6.2. 使用 PaaS 进行持续部署

7.7. 管道运行时改进：静态分析和Git Hooks
- 7.7.1. 静态分析
- 7.7.2. 代码风格的规则
- 7.7.3. Git Hooks 和 单元测试
- 7.7.4. 基础设施的影响

7.8. 容器漏洞
- 7.8.1. 容器漏洞
- 7.8.2. 图像扫描
- 7.8.3. 定期报告和警报

模块 8. 数据库设计(BD)。规范和性能软件质量

8.1. 数据库设计
- 8.1.1. 数据库分类
- 8.1.2. 当前使用的数据库
  - 8.1.2.1. 关系
  - 8.1.2.2. 核心价值
  - 8.1.2.3. 基于图表
- 8.1.3. 数据质量

8.2. 实体关系模型的设计（一）
- 8.2.1. 实体-关系的模型质量和文档
- 8.2.2. 实体机构
  - 8.2.2.1. 强大的实体
  - 8.2.2.2. 弱的实体
- 8.2.3. 属性
- 8.2.4. 关系集
  - 8.2.4.1. 一对一
  - 8.2.4.2. 一对多
  - 8.2.4.3. 多对一
  - 8.2.4.4. 多对多
- 8.2.5. 关键
  - 8.2.5.1. 主键
  - 8.2.5.2. 外键
  - 8.2.5.3. 弱实体主键
- 8.2.6. 限制
- 8.2.7. 基数
- 8.2.8. 继承
- 8.2.9. 聚合

8.3. 实体关系模型（II）工具
- 8.3.1. 实体关系模型工具
- 8.3.2. 实体关系模型实际例子
- 8.3.3. 可行的实体关系模型
  - 8.3.3.1. 视觉展示
  - 8.3.3.2. 表格的示例

8.4. 数据库规范化 (BD) (I)软件质量注意事项
- 8.4.1. 数据库标准和质量
- 8.4.2. 依赖
  - 8.4.2.1. 功能依赖
  - 8.4.2.2. 功能依赖属性
  - 8.4.2.3. 推断属性
- 8.4.3. 关键

8.5. 数据库规范化 (BD) (II)范式和 Codd 规则
- 8.5.1. 范式
  - 8.5.1.1. 第一范式 (1NF)
  - 8.5.1.2. 第二范式 (2NF)
  - 8.5.1.3. 第三范式 (3NF)
  - 8.5.1.4. Boyce-Codd 范式 (FNBC)
  - 8.5.1.5. 第四范式 (4NF)
  - 8.5.1.6. 第五范式 (5NF)
- 8.5.2. 科德规则
  - 8.5.2.1. 规则 1：信息
  - 8.5.2.2. 规则 2：保证访问
  - 8.5.2.3. 规则 3：空值的系统处理
  - 8.5.2.4. 规则 4：数据库描述
  - 8.5.2.5. 规则 5：综合子语言
  - 8.5.2.6. 规则 6：更新视图
  - 8.5.2.7. 规则 7：插入和更新
  - 8.5.2.8. 规则 8：身体独立
  - 8.5.2.9. 规则 9：逻辑独立
  - 8.5.2.10 .规则 10：完整性的独立性
    - 8.5.2.10.1.完整性规则
  - 8.5.2.11. 规则 11：分配
  - 8.5.2.12. 规则 12：不得颠覆
- 8.5.3. 实际例子

8.6. 数据仓库/OLAP系统
- 8.6.1. 数据存储
- 8.6.2. 事实表
- 8.6.3. 尺寸表
- 8.6.4. 创建OLAP系统工具

8.7. 数据库性能 (BD)
- 8.7.1. 索引优化
- 8.7.2. 查询优化
- 8.7.3. 表分区

8.8. DB设计的真实项目模拟 (一)
- 8.8.1. 项目概况 (A公司)
- 8.8.2. 数据库设计应用
- 8.8.3. 拟议的练习
- 8.8.4. 建议练习反馈信息

8.9. DB设计的真实项目模拟 (二)
- 8.9.1. 项目概况 (B公司)
- 8.9.2. 数据库设计应用
- 8.9.3. 拟议的练习
- 8.9.4. 拟议的练习反馈信息

8.10. 数据库优化与软件质量的相关性
- 8.10.1. 设计优化
- 8.10.2. 查询代码优化
- 8.10.3. 存储过程代码优化
- 8.10.4. Triggers对软件质量的影响使用建议

模块 9. 可扩展架构的设计软件生命周期的架构

9.1. 可扩展架构设计 (一)
- 9.1.1. 可扩展架构
- 9.1.2. 可扩展架构的原则
  - 9.1.2.1. 值得信赖
  - 9.1.2.2. 可扩展性
  - 9.1.2.3. 可维护
- 9.1.3. 可扩展类型
  - 9.1.3.1. 垂直的
  - 9.1.3.2. 水平的
  - 9.1.3.3. 合并的

9.2. DDD 架构 (Domain-Driven Design)
- 9.2.1. DDD 模型领域驱动设计
- 9.2.2. 层、职责分担和设计模式
- 9.2.3. 脱钩是质量的基础

9.3. 可扩展架构的设计 (II) 好处、限制和设计策略
- 9.3.1. 可扩展的架构益处
- 9.3.2. 可扩展的架构局限性
- 9.3.3. 可扩展架构的开发策略 (描述表)

9.4. 软件生命周期 (一) 阶段
- 9.4.1. 软件的生命周期
  - 9.4.1.1. 策划阶段
  - 9.4.1.2. 分析阶段
  - 9.4.1.3. 设计阶段
  - 9.4.1.4. 实施阶段
  - 9.4.1.5. 测试阶段
  - 9.4.1.6. 安装/部署阶段
  - 9.4.1.7. 使用与维护阶段

9.5. 软件生命周期模型
- 9.5.1. 级联模型
- 9.5.2. 重复模型
- 9.5.3. 螺旋模型
- 9.5.4. 大爆炸模型

9.6. 软件生命周期 (二)。自动化
- 9.6.1. 软件开发生命周期解决方案
  - 9.6.1.1. 持续集成和开发 (CI/CD)
  - 9.6.1.2. 敏捷方法
  - 9.6.1.3. DevOps / 生产运营
- 9.6.2. 未来的趋势
- 9.6.3. 实际案例

9.7. 软件生命周期的软件架构
- 9.7.1. 益处
- 9.7.2. 局限性
- 9.7.3. 工具

9.8. 软件架构设计的真实项目模拟 (一)
- 9.8.1. 项目概况 (A公司)
- 9.8.2. 软件架构设计应用
- 9.8.3. 拟议的练习
- 9.8.4. 建议练习反馈信息

9.9. 软件架构设计的真实项目模拟 (II)
- 9.9.1. 项目概况 (B公司)
- 9.9.2. 软件架构设计应用
- 9.9.3. 拟议的练习
- 9.9.4. 建议练习反馈

9.10. 软件架构设计的真实项目模拟 (III)
- 9.10.1. 项目概况 (C公司)
- 9.10.2. 软件架构设计应用
- 9.10.3. 拟议的练习
- 9.10.4. 拟议的练习反馈信息

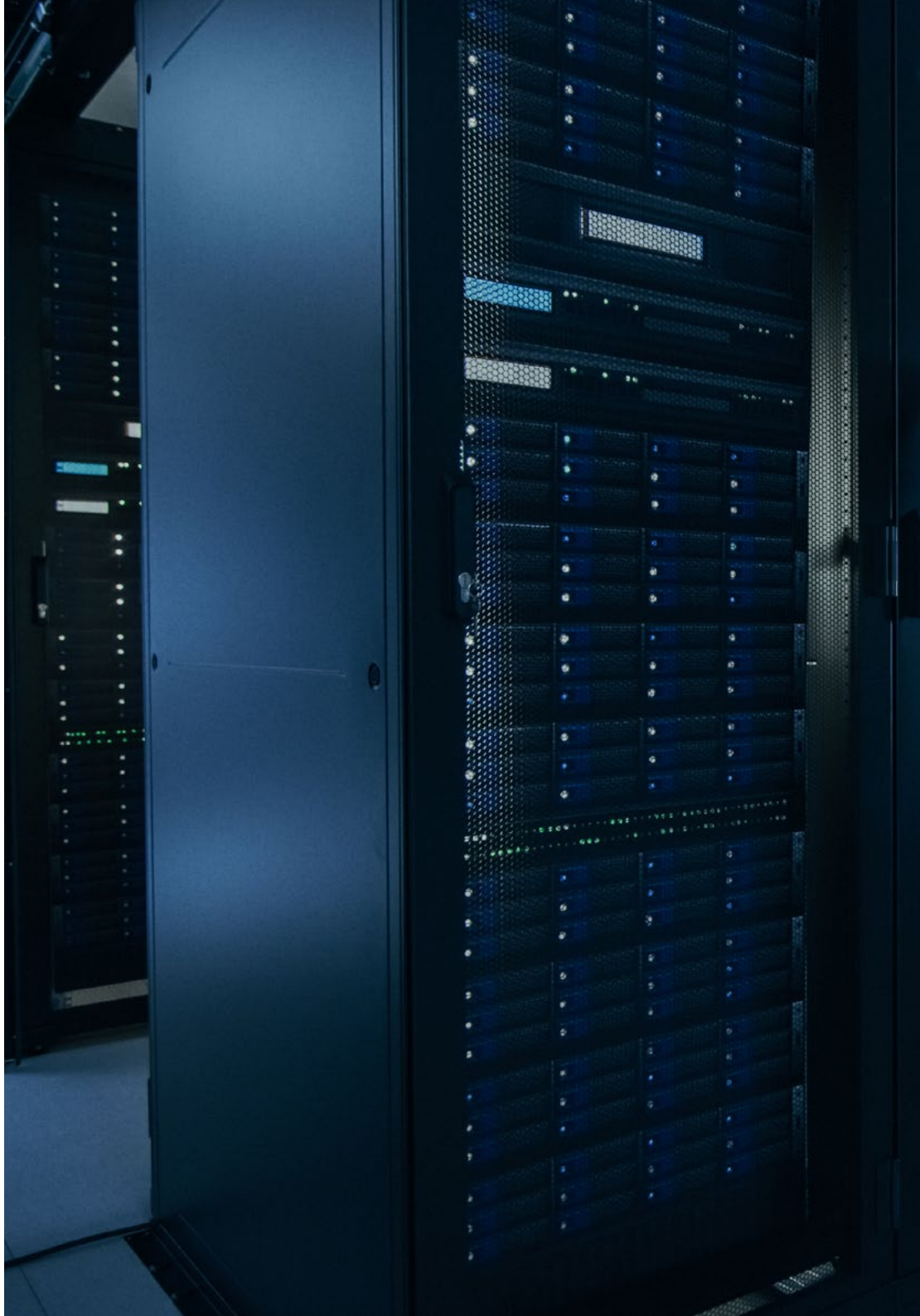
模块 10. ISO 质量标准, IEC 9126。软件质量指标

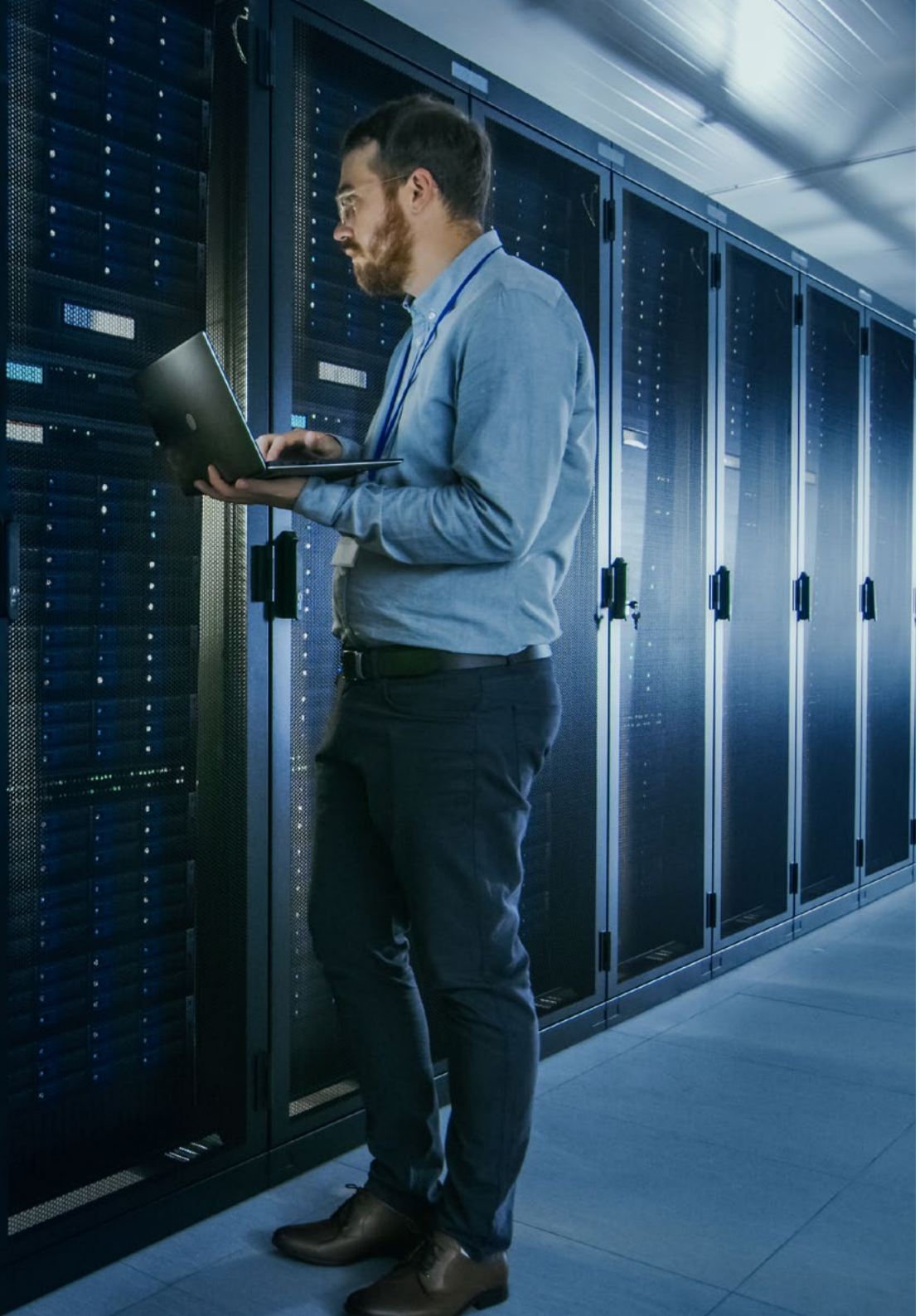
10.1. 质量标准ISO 标准, IEC 9126
- 10.1.1. 质量标准
- 10.1.2. 软件的质量理论依据ISO 标准, IEC 9126
- 10.1.3. 软件质量测量作为关键指标

10.2. 软件质量标准特点
- 10.2.1. 可靠性
- 10.2.2. 功能性
- 10.2.3. 效率
- 10.2.4. 可用性
- 10.2.5. 可维护性
- 10.2.6. 可移植性

10.3. ISO 标准, IEC 9126 (I)介绍
- 10.3.1. ISO 标准说明, IEC 9126
- 10.3.2. 功能性
- 10.3.3. 可靠性
- 10.3.4. 可用性
- 10.3.5. 可维护性

10.3.6. 可移植性
10.3.7. 使用质量
10.3.8. 软件质量指标
10.3.9. ISO 9126 的质量指标
10.4. ISO 标准，IEC 9126 (II)McCall 和 Boehm 模型
10.4.1. McCall模型. 质量因素
10.4.2. Boehm模型
10.4.3. 中级水平特点
10.5. 软件质量度量 (I)元素
10.5.1. 测量
10.5.2. 度量几何
10.5.3. 指标
10.5.3.1.指标类型
10.5.4. 测量和模型
10.5.5. 软件度量范围
10.5.6. 软件指标分类
10.6. 软件质量测量（二）测量实践
10.6.1. 指标数据收集
10.6.2. 内部产品属性的测量
10.6.3. 产品外部属性的测量
10.6.4. 资源测量
10.6.5. 面向对象系统的度量
10.7. 单一软件质量指标的设计
10.7.1. 单一指标作为全局限定
10.7.2. 指标的制定、理由和应用
10.7.3. 应用示例需要了解详情
10.8. 质量测量的真实项目模拟（一）
10.8.1. 项目概况（A公司）
10.8.2. 质量测量的应用
10.8.3. 拟议的练习
10.8.4. 建议练习反馈信息





10.9. 质量测量的真实项目模拟(二)
- 10.9.1. 项目概况(B公司)
- 10.9.2. 质量测量的应用
- 10.9.3. 拟议的练习
- 10.9.4. 建议练习反馈信息

10.10. 质量测量的真实项目模拟(三)
- 10.10.1. 项目概况(C公司)
- 10.10.2. 质量测量的应用
- 10.10.3. 拟议的练习
- 10.10.4. 建议练习反馈信息

模块 11. 软件工程的方法、开发和质量

11.1. 基于模型的软件开发
- 11.1.1. 需求
- 11.1.2. 物件建模
- 11.1.3. UML
- 11.1.4. CASE工具

11.2. 使用 UML 的应用程序建模和设计模式
- 11.2.1. 高级需求建模
- 11.2.2. 高级静态建模
- 11.2.3. 高级动态建模
- 11.2.4. 组件建模
- 11.2.5. 使用 UML 简介设计模式
- 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. 模型驱动工程
- 11.3.1. 简介
- 11.3.2. 系统元建模
- 11.3.3. MDA
- 11.3.4. DSL
- 11.3.5. 使用 OCL 改进模型
- 11.3.6. 模型转换

11.4. 软件工程中的本体
- 11.4.1. 简介
- 11.4.2. 本体工程
- 11.4.3. 本体在软件工程的应用

模块 12. 软件项目管理

12.1. 利益相关者和范围管理
- 12.1.1. 辨识利益相关者
- 12.1.2. 制定利益相关者的管理计划
- 12.1.3. 管理利益相关者的参与
- 12.1.4. 控制利益相关者的参与
- 12.1.5. 项目目标
- 12.1.6. 范围管理和计划
- 12.1.7. 收集需求
- 12.1.8. 定义范围说明
- 12.1.9. 创建WBS (EDT)
- 12.1.10. 检查和控制范围

12.2. 时间表的制定
- 12.2.1. 时间管理和计划
- 12.2.2. 定义活动
- 12.2.3. 确定活动顺序
- 12.2.4. 活动资源估算
- 12.2.5. 活动估计持续时间
- 12.2.6. 进度制定和关键路径计算
- 12.2.7. 进度控制

12.3. 预算制定和风险应对
- 12.3.1. 估算成本
- 12.3.2. 制定预算和 S 曲线
- 12.3.3. 成本控制与挣值法
- 12.3.4. 风险概念
- 12.3.5. 如何进行风险分析?
- 12.3.6. 制定应对计划

12.4. 沟通和人力资源
- 12.4.1. 规划沟通管理
- 12.4.2. 通信需求分析
- 12.4.3. 通讯技术
- 12.4.4. 沟通模式
- 12.4.5. 沟通方式
- 12.4.6. 通讯管理计划
- 12.4.7. 管理通讯
- 12.4.8. 人力资源管理
- 12.4.9. 主要参与者及在项目中的角色
- 12.4.10 组织的类型
- 12.4.11. 项目组织
- 12.4.12. 工作团队

12.5. 供应
- 12.5.1. 采购流程
- 12.5.2. 规划
- 12.5.3. 搜索供应商并请求报价
- 12.5.4. 合同的授予
- 12.5.5. 合同的管理
- 12.5.6. 合同
- 12.5.7. 合同的类型
- 12.5.8. 合同的谈判

12.6. 执行、监控和控制以及关闭
- 12.6.1. 进程组
- 12.6.2. 项目实施
- 12.6.3. 项目监控
- 12.6.4. 项目的关闭

12.7. 职业责任
- 12.7.1. 职业责任
- 12.7.2. 社会和职业责任的特征
- 12.7.3. 项目负责人的道德规范
- 12.7.4. 责任和PMP®
- 12.7.5. 责任的例子
- 12.7.6. 专业化的好处

模块 13. 软件开发平台

13.1. 应用程序开发简介
- 13.1.1. 桌面应用程序
- 13.1.2. 编程语言
- 13.1.3. 集成开发环境
- 13.1.4. 网络应用程序
- 13.1.5. 移动应用程序
- 13.1.6. 云应用程序

13.2. Java的应用程序开发和图形界面
- 13.2.1. Java 的集成开发环境
- 13.2.2. Java 的顶级 IDE
- 13.2.3. Eclipse开发平台简介
- 13.2.4. NetBeans 开发平台简介
- 13.2.5. 用于图形用户界面的模型视图控制器
- 13.2.6. 在 Eclipse设计图形界面
- 13.2.7. 在 NetBeans设计图形界面

13.3. Java的调试和测试
- 13.3.1. 用Java测试和调试程序
- 13.3.2. 在Eclipse 调试
- 13.3.3. 在NetBeans 调试

13.4. .NET 的应用程序开发和图形界面
- 13.4.1. 网络框架
- 13.4.2. .NET 开发平台组件
- 13.4.3. Visual Studio .NET
- 13.4.4. 用于 GUI 的 .NET 工具
- 13.4.5. 带有Windows Presentation Foundation 的 GUI
- 13.4.6. 调试和构建 WPF 应用程序

13.5. .NET 网络编程
- 13.5.1. .NET 网络编程简介
- 13.5.2. .NET 中的请求和响应
- 13.5.3. .NET 使用的应用程序协议
- 13.5.4. .NET 网络的编程安全性

13.6. 移动应用程序开发环境
- 13.6.1. 移动应用程序
- 13.6.2. 安卓手机应用程序
- 13.6.3. 安卓开发步骤
- 13.6.4. 安卓Studio IDE

13.7. 安卓Studio 环境的应用开发
- 13.7.1. 安装并启动安卓Studio
- 13.7.2. 运行安卓应用程序
- 13.7.3. 安卓Studio 图形界面的开发
- 13.7.4. 在安卓Studio 启动活动

13.8. 调试和发布安卓应用程序
- 13.8.1. 安卓Studio的调试应用
- 13.8.2. 在安卓Studio中记忆应用程序
- 13.8.3. 在 Google Play上发布应用

13.9. 云应用开发
- 13.9.1. Cloud computing
- 13.9.2. Cloud的层次：SaaS, PaaS, IaaS
- 13.9.3. 云开发主要平台
- 13.9.4. 参考书目

13.10. 谷歌 云平台简介
- 13.10.1. 谷歌云平台的基础
- 13.10.2. 谷歌云平台的服务
- 13.10.3. 谷歌云平台的工具

模块 14. Web 客户端的计算

14.1. HTML 简介
- 14.1.1. 文件的结构
- 14.1.2. 颜色
- 14.1.3. 文本
- 14.1.4. 超文本链接
- 14.1.5. 图片
- 14.1.6. 列表
- 14.1.7. 表格
- 14.1.8. 框架(Frames)
- 14.1.9. 表格
- 10.1.14. 移动技术的特定元素
- 11.1.14. 不推荐使用的项目

14.2. 网页样式表 (CSS)
- 14.2.1. 样式表的元素和结构
  - 14.2.1.1. 创建样式表
  - 14.2.1.2. 样式的应用选择器
  - 14.2.1.3. 样式继承和级联
  - 14.2.1.4. 使用样式的页面格式
  - 14.2.1.5. 通过样式的页面结构盒子模型
- 14.2.2. 不同设备的设计风格
- 14.2.3. 样式表的类型：静态和动态伪类
- 14.2.4. 使用样式表的良好实践

14.3. JavaScript的简介和历史
- 14.3.1. 简介
- 14.3.2. JavaScript的历史
- 14.3.3. 将要使用的开发环境

14.4. 网络编程的基本概念
- 14.4.1. 基本 JavaScript 语法
- 14.4.2. 原始数据类型和运算
- 14.4.3. 变量和范围
- 14.4.4. 字符串和模板文字
- 14.4.5. 数字和布尔值
- 14.4.6. 比较

14.5. JavaScript的复杂结构
- 14.5.1. 向量或数组和物件
- 14.5.2. 套组
- 14.5.3. 地图
- 14.5.4. 困境
- 14.5.5. 循环

14.6. 功能和物件
- 14.6.1. 函数定义和调用
- 14.6.2. 论据
- 14.6.3. 箭头函数
- 14.6.4. 回调功能或callback函数
- 14.6.5. 高阶函数
- 14.6.6. 文字物件
- 14.6.7. This对象
- 14.6.8. 作为命名空间的物件：Math物件和Date物件

14.7. 文档物件模型 (DOM)
- 14.7.1. 什么是DOM?
- 14.7.2. 历史简介
- 14.7.3. 浏览和获取项目
- 14.7.4. 有 JSDOM 的虚拟 DOM
- 14.7.5. 查询选择器或Query Selectors
- 14.7.6. 通过属性导航
- 14.7.7. 以元素分配属性
- 14.7.8. 节点创建和修改
- 14.7.9. 更新 DOM 元素的样式

14.8. 现代网络开发
- 14.8.1. 基于事件和listeners的流量
- 14.8.2. 现代网络Toolkits和对齐系统
- 14.8.3. JavaScript 严格模式
- 14.8.4. 更多关于函数
- 14.8.5. 承诺和异步函数
- 14.8.6. Closures
- 14.8.7. 函数式编程
- 14.8.8. JavaScript的 POO

14.9. 网站可用性
- 14.9.1. 可用性简介
- 14.9.2. 可用性的定义
- 14.9.3. 以用户为中心的网页设计的重要性
- 14.9.4. 可访问性和可用性之间的差异
- 14.9.5. 可访问性和可用性相结合的优点和问题
- 14.9.6. 发展可用网站的优势和难点
- 14.9.7. 可用性方法
- 14.9.8. 用户需求分析
- 14.9.9. 概念设计原则面向用户的原型设计
- 14.9.10. 创建可用网站的指南
  - 14.9.10.1. Jakob Nielsen可用性指南
  - 14.9.10.2. Bruce Tognazzini可用性指南
- 14.9.11. 可用性评估

14.10. 网站可访问性
- 14.10.1. 简介
- 14.10.2. 网络可访问性的定义
- 14.10.3. 残疾类型
  - 14.10.3.1. 暂时或永久性残疾
  - 14.10.3.2. 视力障碍
  - 14.10.3.3. 听力障碍
  - 14.10.3.4. 运动障碍
  - 14.10.3.5. 神经或认知障碍
  - 14.10.3.6. 老化带来的困难
  - 14.10.3.7. 来自环境的限制
  - 14.10.3.8. 阻止访问网络的障碍
- 14.10.4. 克服障碍的技术辅助和支持产品
  - 14.10.4.1. 盲人援助
  - 14.10.4.2. 帮助视力低下的人
  - 14.10.4.3. 帮助有色盲的人
  - 14.10.4.4. 对听力障碍者的援助
  - 14.10.4.5. 对运动障碍者的援助
  - 14.10.4.6. 为有认知和神经障碍的人提供援助
- 14.10.5. 网页无障碍实施的优势和难点
- 14.10.6. 网页无障碍法规和标准
- 14.10.7. 网络无障碍监管机构
- 14.10.8. 规范与标准的比较
- 14.10.9. 遵守法规和标准的指南
  - 14.10.9.1. 主要指南的描述（图片、视频链接等）
  - 14.10.9.2. 无障碍导航指南
    - 14.10.9.2.1. 感知能力
    - 14.10.9.2.2. 可操作性
    - 14.10.9.2.3. 可理解性
    - 14.10.9.2.4. 坚固性
- 14.10.10. Web 可访问性合规流程的描述
- 14.10.11. 合规水平
- 14.10.12. 合规标准
- 14.10.13. 合规要求
- 14.10.14. 网站可访问性评估方法

模块 15. Web 服务器的计算

15.1. 服务端编程简介：PHP
- 15.1.1. 服务器编程基础
- 15.1.2. 基本 PHP 语法
- 15.1.3. 使用 PHP 生成 HTML 内容
- 15.1.4. 开发和测试环境：XAMPP

15.2. 高级 PHP
- 15.2.1. 使用 PHP 控制结构
- 15.2.2. PHP中的功能
- 15.2.3. PHP 中Arrays
- 15.2.4. 用 PHP 处理字符串
- 15.2.5. PHP 中的面向物件

15.3. 数据模型
- 15.3.1. 数据概念数据生命周期
- 15.3.2. 数据类型
  - 15.3.2.1. 基本
  - 15.3.2.2. 记录
  - 15.3.2.3. 动态

15.4. 关系模型
- 15.4.1. 描述
- 15.4.2. 实体和实体类型
- 15.4.3. 数据元素属性
- 15.4.4. 关系：类型、子类型、基数
- 15.4.5. 密码密码类型
- 15.4.6. 标准化范式

15.5. 逻辑数据模型的构建
- 15.5.1. 表规格
- 15.5.2. 列定义
- 15.5.3. 主要规格
- 15.5.4. 转换为正常形式依赖

15.6. 物理数据模型数据文件
- 15.6.1. 数据文件说明
- 15.6.2. 文件类型
- 15.6.3. 访问模式
- 15.6.4. 文件组织

15.7. 从 PHP 访问数据库
- 15.7.1. MariaDB 简介
- 15.7.2. 使用 MariaDB 数据库：SQL 语言
- 15.7.3. 从 PHP 访问 MariaDB 数据库
- 15.7.4. MySQL 简介
- 15.7.5. 使用 MySql 数据库：SQL 语言
- 15.7.6. 从 PHP 访问 MySql 数据库

15.8. 通过 PHP 与客户端交互
- 15.8.1. PHP 表单
- 15.8.2. Cookies
- 15.8.3. 会话处理

15.9. 网络应用架构
- 15.9.1. 模型视图控制器模式
- 15.9.2. 控制
- 15.9.3. 模型
- 15.9.4. 查看

15.10. 网络服务简介
- 15.10.1. XML 简介
- 15.10.2. 面向服务的架构 (SOA)：Web 服务
- 15.10.3. 创建 SOAP 和 REST Web 服务
- 15.10.4. SOAP 协议
- 15.10.5. REST 协议

模块 16. 安全管理

16.1. 信息安全
- 16.1.1. 简介
- 16.1.2. 信息安全意味着机密性、完整性和可用性
- 16.1.3. 安全是一个经济问题
- 16.1.4. 安全是一个过程
- 16.1.5. 信息分类
- 16.1.6. 信息安全涉及风险管理
- 16.1.7. 安全性与安全控制相结合
- 16.1.8. 安全性既是物理也是逻辑的
- 16.1.9. 安全涉及到的人

16.2. 信息安全专业人士
- 16.2.1. 简介
- 16.2.2. 信息安全作为一种职业
- 16.2.3. 认证 (ISC)2
- 16.2.4. ISO 27001 标准
- 16.2.5. IT 服务管理中的良好安全实践
- 16.2.6. 信息安全成熟度模型
- 16.2.7. 其他认证、标准和专业资源

16.3. 访问控制
- 16.3.1. 简介
- 16.3.2. 访问控制要求
- 16.3.3. 认证机制
- 16.3.4. 授权方式
- 16.3.5. 访问会计和审计
- 16.3.6. 三A技术

16.4. 信息安全计划、流程和政策
- 16.4.1. 简介
- 16.4.2. 安全管理程序
- 16.4.3. 风险管理
- 16.4.4. 设计安全策略

16.5. 业务连续性计划
- 16.5.1. PCN 简介
- 16.5.2. 第一阶段和第二阶段
- 16.5.3. 第三阶段和第四阶段
- 16.5.4. PCN的维护

16.6. 正确保护公司的流程
- 16.6.1. DMZ网络
- 16.6.2. 入侵检测系统
- 16.6.3. 访问控制列表
- 16.6.4. 向攻击者学习：Honeypot

16.7. 安全架构预防
- 16.7.1. 概述。活动和层模型
- 16.7.2. 周界防御（防火墙、WAFsIPS 等）
- 16.7.3. 端点防御（计算机、服务器和服务）

16.8. 安全架构探测
- 16.8.1. 概览检测和监控
- 16.8.2. 日志、加密流量突破、记录和 Siems
- 16.8.3. 警报和情报

16.9. 安全架构反应
- 16.9.1. 反应产品、服务和资源
- 16.9.2. 计算机安全
- 16.9.3. CERTS 和 CSIRTs

16.10. 安全架构恢复
- 16.10.1. 弹性、概念、业务要求和法规
- 16.10.2. 弹性 IT 解决方案
- 16.10.3. 危机管理和治理

模块 17. 软件安全

17.1. 软件的安全问题
- 17.1.1. 软件安全问题的简介
- 17.1.2. 漏洞及分类
- 17.1.3. 保护软件属性
- 17.1.4. 参考文献

17.2. 软件安全设计原则
- 17.2.1. 简介
- 17.2.2. 软件安全设计原则
- 17.2.3. S-SDLC的类型
- 17.2.4. S-SDLC 阶段的软件安全
- 17.2.5. 方法和标准
- 17.2.6. 参考文献

17.3. 需求和设计阶段软件生命周期的安全性
- 17.3.1. 简介
- 17.3.2. 攻击建模
- 17.3.3. 滥用的案件
- 17.3.4. 安全需求工程
- 17.3.5. 风险分析建筑
- 17.3.6. 设计模式
- 17.3.7. 参考文献

17.4. 软件生命周期中的编码、测试和运行阶段的安全性
- 17.4.1. 简介
- 17.4.2. 基于风险的安全测试
- 17.4.3. 代码审查
- 17.4.4. 渗透测试
- 17.4.5. 安全行动
- 17.4.6. 外部审查
- 17.4.7. 参考文献

17.5. 安全编码应用程序 I
- 17.5.1. 简介
- 17.5.2. 安全编码实践
- 17.5.3. 输入操作和验证
- 17.5.4. 内存溢出
- 17.5.5. 参考文献

17.6. 安全编码应用程序 II
- 17.6.1. 简介
- 17.6.2. 整数溢出、截断错误和整数之间的类型转换问题
- 17.6.3. 错误和异常
- 17.6.4. 隐私和保密
- 17.6.5. 特权程序
- 17.6.6. 参考文献

17.7. 开发和云的安全性
- 17.7.1. 发展保障；方法与实践
- 17.7.2. PaaS、IaaS、CaaS 和 SaaS 模型
- 17.7.3. 云和云服务的安全性

17.8. 加密
- 17.8.1. 密码学基础
- 17.8.2. 对称和非对称加密
- 17.8.3. 静态和传输中的加密

17.9. 安全自动化和编排 (SOAR)
- 17.9.1. 人工处理的复杂性；需要自动化任务
- 17.9.2. 产品和服务
- 17.9.3. SOAR-架构

17.10. 远程办公的安全性
- 17.10.1. 需求和场景
- 17.10.2. 产品和服务
- 17.10.3. 远程办公安全

模块 18. 网络服务器管理

18.1. 网络服务器简介
- 18.1.1. 什么是网络服务器?
- 18.1.2. Web 服务器的架构和操作
- 18.1.3. Web 服务器上的资源和内容
- 18.1.4. 应用服务器
- 18.1.5. 代理服务器
- 18.1.6. 市场上的主要网络服务器
- 18.1.7. Web 服务器使用情况统计
- 18.1.8. Web 服务器的安全性
- 18.1.9. Web 服务器的负载平衡
- 18.1.10. 参考文献

18.2. 掌握HTTP协议
- 18.2.1. 操作和结构。
- 18.2.2. 请求或请求方法的描述
- 18.2.3. 状态码
- 18.2.4. 标题
- 18.2.5. 内容编码代码页
- 18.2.6. 通过代理、Livehttpheaders或类似方法在互联网上发出 HTTP 请求，分析使用的协议

18.3. 分布在多个服务器中的架构描述
- 18.3.1. 3层模型
- 18.3.2. 容错
- 18.3.3. 负载分担
- 18.3.4. 会话状态存储
- 18.3.5. 缓存存储

18.4. 互联网信息服务 (IIS)
- 18.4.1. 什么是 IIS?
- 18.4.2. IIS的历史和演变
- 18.4.3. IIS7 及更高版的主要优点和特性
- 18.4.4. IIS7 及以后的架构

18.5. IIS的安装、管理和配置
- 18.5.1. 前言
- 18.5.2. 安装互联网信息服务 (IIS)
- 18.5.3. IIS 管理工具
- 18.5.4. 网站的创建、配置和管理
- 18.5.5. 在 IIS 安装和扩展

18.6. IIS 中的高级安全性
- 18.6.1. 前言
- 18.6.2. IIS 中的身份验证、授权和访问控制
- 18.6.3. 使用 SSL 在 IIS 的配置安全网站
- 18.6.4. IIS 8.x 实现的安全策略

18.7. Apache简介
- 18.7.1. 什么是Apache?
- 18.7.2. Apache的主要优点
- 18.7.3. Apache的主要特点
- 18.7.4. 建筑学

18.8. Apache安装和配置
- 18.8.1. Apache的初始安装
- 18.8.2. Apache配置

18.9. Apache不同模块的安装和配置
- 18.9.1. Apache的安装模块
- 18.9.2. 模块类型
- 18.9.3. 安全的Apache配置

18.10. 高级安全
- 18.10.1. 身份验证、授权和访问控制
- 18.10.2. 身份验证方法
- 18.10.3. 使用 SSL 保护Apache配置

模块 19. 安全审计

19.1. 信息系统及审计简介
- 19.1.1. 信息系统简介和计算机审计的作用
- 19.1.2. 计算机审计和计算机内部控制的定义
- 19.1.3. 计算机审计的职能和目标
- 19.1.4. 内部控制与计算机审计的区别

19.2. 信息系统的内部控制
- 19.2.1. 数据处理中心功能组织图
- 19.2.2. 信息系统控制的分类
- 19.2.3. 黄金法则

19.3. 信息系统审计的过程和阶段
- 19.3.1. 风险评估 (EDR) 和其他计算机审计方法
- 19.3.2. 执行信息系统审计审计的阶段
- 19.3.3. 信息系统审计员的基本技能

19.4. 系统和网络中的技术安全审计
- 19.4.1. 技术安全审计渗透测试之前的概念
- 19.4.2. 系统安全审计支持工具
- 19.4.3. 网络安全审计支持工具

19.5. 互联网和移动设备的技术安全审计
- 19.5.1. 互联网安全审计支持工具
- 19.5.2. 移动设备的安全审计支持工具
- 19.5.3. 附录1执行报告和技术报告的结构
- 19.5.4. 附录2工具清单
- 19.5.5. 附录3方法

19.6. 信息安全管理体系
- 19.6.1. IS 安全性：属性和影响因素
- 19.6.2. 业务风险和风险管理：控制的实施
- 19.6.3. 信息安全 SG (ISMS)：成功的概念和关键因素
- 19.6.4. ISMS-PDCA模型
- 19.6.5. ISMS ISO-IEC 27001：组织环境
- 19.6.6. 组织背景
- 19.6.7. 领导力
- 19.6.8. 规划
- 19.6.9. 支援
- 19.6.10. 运作
- 19.6.11. 绩效评估绩效计划
- 19.6.12. 变好
- 19.6.13. ISO 27001/ISO-IEC 27002 附录：目标和控制
- 19.6.14. SGSI 审核

19.7. 执行审计
- 19.7.1. 过程
- 19.7.2. 技术

19.8. 追溯性
- 19.8.1. 方法
- 19.8.2. 分析

19.9. 保管
- 19.9.1. 技术
- 19.9.2. 结果

19.10. 报告和提供证据
- 19.10.1. 报告类型
- 19.10.2. 数据分析
- 19.10.3. 出示证据

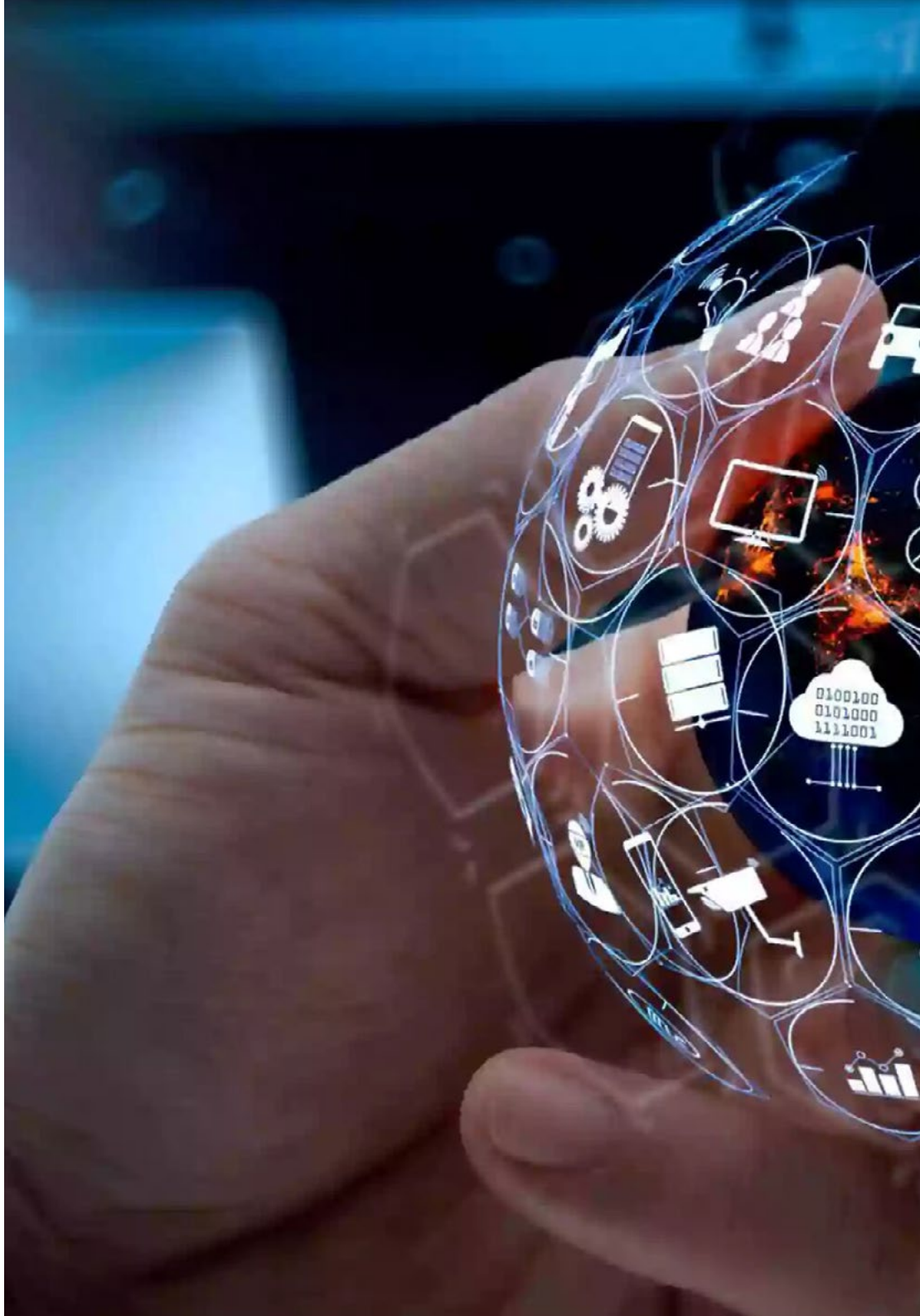
模块 20. 在线应用程序的安全

20.1. 在线应用程序的漏洞和安全问题
- 20.1.1. 在线应用程序安全性简介
- 20.1.2. Web 应用程序设计的安全漏洞
- 20.1.3. Web 应用程序实现的安全漏洞
- 20.1.4. Web 应用程序部署的安全漏洞
- 20.1.5. 安全漏洞的官方列表

20.2. 在线应用程序安全的政策和标准
- 20.2.1. 在线应用程序安全的支柱
- 20.2.2. 安全策略
- 20.2.3. 信息安全管理体系
- 20.2.4. 安全的软件开发生命周期
- 20.2.5. 应用安全标准

20.3. Web 应用程序设计的安全性
- 20.3.1. Web 应用程序安全性简介
- 20.3.2. Web 应用程序设计的安全性

20.4. Web 应用程序的安全性和在线保护测试
- 20.4.1. Web应用程序安全性分析与测试
- 20.4.2. Web 应用程序的部署和生产的安全性



20.5. 网络服务安全
- 20.5.1. Web 服务安全性简介
- 20.5.2. Web 服务安全特性和技术

20.6. Web 服务的安全性和在线保护测试
- 20.6.1. 网络服务安全评估
- 20.6.2. 在线保护防火墙和 XML 网关

20.7. 道德黑客攻击、恶意软件和取证
- 20.7.1. 道德黑客
- 20.7.2. 恶意软件分析
- 20.7.3. 取证分析

20.8. 确保应用程序安全的良好做法
- 20.8.1. 开发在线应用程序的良好做法手册
- 20.8.2. 实施在线申请的良好做法手册

20.9. 危害应用程序安全的常见错误
- 20.9.1. 开发中的常见错误
- 20.9.2. 常见的托管错误
- 20.9.3. 生产中的常见错误

> 完整的教学大纲将帮助您掌握大数据领域并成为一名成功的商业战略架构师”

04

教学目标

软件质量与工程高级硕士旨在培养高质量软件系统设计、开发和管理方面的高素质专业人才。该课程特别注重技术项目的质量和战略管理，同时也提高了适应该领域快速发展的能力。这确保学生不仅做好准备应对未来的挑战，而且能够引领软件领域的创新。

利用软件工程的艺术将想法转化为有效的技术解决方案”

总体目标

- 培养设计、开发和维护复杂且可扩展的软件系统的高级技能，应用软件工程中的最佳实践和方法
- 对学生进行软件质量保证培训，为他们提供工具和技术，以确保技术解决方案的可靠性、安全性和性能
- 提升技术项目管理的领导力，培养多学科团队管理、动态环境中战略规划和决策的技能
- 通过专门研究新工具、新技术和新趋势，提高适应快速技术进步的能力，使我们始终处于软件工程的前沿
- 培养整个软件生命周期的质量管理技能，从最初的规划到系统的维护和持续改进
- 加强沟通和团队合作技能，这对于与不同利益相关者有效合作、管理期望和确保技术项目的成功至关重要

具体目标

模块 1. 软件的质量TRL发展水平
- 了解不同级别的技术成熟度及其与软件质量的关系
- 评估 TRL 每个阶段的软件开发情况以及它对产品最终质量的影响

模块 2. 软件项目的开发功能和技术文档
- 培养在软件项目中创建清晰、详细的功能和技术文档的技能
- 分析准确文档对项目管理和软件质量的重要性

模块 3. 软件测试自动化测试
- 培养设计和执行软件应用程序自动化测试的技能
- 使用测试自动化工具实施高效的测试解决方案

模块 4. 架构模式瀑布模型方法与敏捷方法对比
- 分析软件项目管理中瀑布方法和敏捷方法之间的差异
- 根据项目类型评估每种方法的优点和局限性

模块 5. TDD (Test Driven Developement).测试驱动的软件设计
- 在编写生产代码之前培养编写单元测试的技能
- 通过在开发过程中实施 TDD 来提高软件质量

模块 6. DevOps.软件质量管理
- 探索 DevOps 理念及其对软件质量持续改进的影响
- 学习整合开发和运营实践，实现更敏捷，更高效的软件生命周期

模块 7. DevOps 和持续集成软件开发中先进的实用解决方案
- 深入研究 DevOps 框架内的高级持续集成技术
- 实施实用的持续集成解决方案，实现软件开发和部署过程的自动化

模块 8. 数据库设计(BD)。规范和性能软件质量
- 分析数据库设计原则，包括规范化和性能优化
- 了解正确的数据库设计如何有助于软件质量

模块 9. 可扩展架构的设计软件生命周期的架构
- 深入研究可扩展架构的设计原则及其对软件质量和性能的影响
- 评估可扩展软件应用程序的不同架构模式

模块 10. ISO 质量标准，IEC 9126。软件质量指标
- 了解这些标准的软件质量标准以及如何应用它们
- 实施质量指标以不断评估和改进软件应用程序

模块 11. 软件工程的方法、开发和质量
- 深入研究软件工程中最广泛使用的方法及其与质量的关系
- 制定一套将软件项目中的开发、测试和质量相结合的综合方法

模块 12. 软件项目管理
- 培养软件项目管理技能，从规划到执行
- 管理与软件开发项目相关的资源、时间和风险

模块 13. 软件开发平台
- 探索不同的软件开发平台及其功能
- 根据开发平台的功能、灵活性以及与不同项目的兼容性对其进行评估

模块 14. Web 客户端的计算
- 分析 Web 应用程序开发中客户端计算的执行方式
- 开发利用客户端计算来改善交互和性能的应用程序

模块 15. Web 服务器的计算
- 探索用于 Web 服务器计算的技术和技术
- 了解服务器上的数据处理，业务逻辑和用户管理

模块 16. 安全管理
- 评估应用程序安全风险并实施预防措施
- 在软件生命周期的所有阶段实施安全控制

模块 17. 软件安全
- 探索软件开发安全性的最佳实践
- 分析软件中最常见的漏洞并学习如何缓解这些漏洞

模块 18. 网络服务器管理
- 了解 Web 服务器在应用程序开发和部署中的作用
- 培养网络服务器管理和维护的技能

模块 19. 安全审计
- 通过审计和渗透测试评估系统安全性
- 实施持续审计流程以提高软件安全性

模块 20. 在线应用程序的安全
- 实施解决方案以保护在线应用程序免受外部和内部威胁
- 建立安全和审计政策，确保在线应用程序的完整性

提高你的技能并成为创造尖端技术解决方案的领导者”

05

职业前景

毕业生将有能力承担高质量软件开发、实施和管理方面的领导角色。凭借您在软件架构、在线应用程序安全、技术项目管理和敏捷方法等关键领域的高级专业知识，您将能够领导开发团队。此外，您的培训将使您能够担任软件项目的管理职位，而您的创新和领导多学科团队的能力将使您有能力应对数字环境的挑战并为任何组织的成功做出重大贡献。

TECH 让你有机会在最令人兴奋的领域实现你的梦想，将想法转化为能够改善人们生活的有形产品”

毕业生简介

软件质量与工程高级硕士毕业生的培养目标是培养能够领导和管理高影响力技术项目的高素质专业人才。确保软件开发所有阶段的质量、安全性和效率，您将掌握敏捷和传统的方法。此外，您将能够设计和开发可扩展、高效和安全的软件系统，应用国际质量标准和先进方法如 DevOps 和持续集成。

> 成为世界上最大的数字大学的保证公司成功的专家。

- **软件和系统安全：**有能力实施先进的安全实践，包括在线应用程序中的数据保护和漏洞管理
- **软件质量保证.** 能够应用国际标准（ISO、IEC 9126）和自动化测试工具来确保软件的可靠性和性能
- **可扩展架构的开发：**能够设计和构建能够发展并适应市场需求且不影响质量或安全性的软件系统
- **持续集成和 DevOps：**能够实施和管理持续集成流程，确保高效、不间断地交付新的软件功能

完成高级硕士课程后，您将能够在以下职位上运用您的知识和技能：

1. **首席技术官（CTO）：**负责公司的技术战略方向，领导开发团队并监督创新技术解决方案的实施
2. **软件质量经理：**负责监督和确保软件流程和产品符合既定的质量标准，领导持续改进计划和软件测试
3. **软件架构师：**负责复杂软件系统结构和架构的首席设计师，确保其可扩展、安全且高效
4. **软件项目负责人：**负责软件项目的规划、执行和交付，管理多学科团队，确保项目按时、按预算、按照适当的质量标准完成
5. **计算机安全专家：**负责保护应用程序、基础设施和数据免受网络威胁，实施有效的安全策略和政策
6. **软件安全审计员：**进行全面审计，识别应用程序和系统中的漏洞，提出改进和解决方案，确保软件安全

> 如果你想在数字世界中有所作为，请选择这条道路，它将使你成为创建优质软件的专家”

06

学习方法

TECH 是世界上第一所将案例研究方法与 Relearning一种基于指导性重复的100% 在线学习系统相结合的大学。

这种颠覆性的教学策略旨在为专业人员提供机会，以强化和严格的方式更新知识和发展技能。这种学习模式将学生置于学习过程的中心，让他们发挥主导作用，适应他们的需求，摒弃传统方法。

我们的课程使你准备好在不确定的环境中面对新的挑战并获得事业上的成功"

学生:所有TECH课程的首要任务

在 TECH 的学习方法中,学生是绝对的主角。

每个课程的教学工具的选择都考虑到了时间,可用性和学术严谨性的要求,这些要求如今不仅是学生的要求也是市场上最具竞争力的职位的要求。

通过TECH的异步教育模式,学生可以选择分配学习的时间,决定如何建立自己的日常生活以及所有这一切,而这一切都可以在他们选择的电子设备上舒适地进行。学生不需要参加现场课程,而他们很多时候都不能参加。您将在适合您的时候进行学习活动。您始终可以决定何时何地学习。

> 在TECH,你不会有线下课程(那些你永远不能参加)”

国际上最全面的学习计划

TECH的特点是提供大学环境中最完整的学术大纲。这种全面性是通过创建教学大纲来实现的，教学大纲不仅包括基本知识，还包括每个领域的最新创新。

通过不断更新，这些课程使学生能够跟上市场变化并获得雇主最看重的技能。通过这种方式，那些在TECH完成学业的人可以获得全面的准备，为他们的职业发展提供显着的竞争优势。

更重要的是，他们可以通过任何设备，个人电脑，平板电脑或智能手机来完成的。

> TECH模型是异步的，因此将您随时随地使用PC，平板电脑或智能手机学习，学习时间不限”

案例研究或案例方法

案例法一直是世界上最好的院系最广泛使用的学习系统。该课程于1912年开发，目的是让法学专业学生不仅能在理论内容的基础上学习法律，还能向他们展示复杂的现实生活情境。因此，他们可以做出决策并就如何解决问题做出明智的价值判断。1924年被确立为哈佛大学的一种标准教学方法。

在这种教学模式下，学生自己可以通过耶鲁大学或斯坦福大学等其他知名机构使用的边做边学或设计思维等策略来建立自己的专业能力。

这种以行动为导向的方法将应用于学生在TECH进行的整个学术大纲。这样你将面临多种真实情况，必须整合知识，调查，论证和捍卫你的想法和决定。这一切的前提是回答他在日常工作中面对复杂的特定事件时如何定位自己的问题。

学习方法

在TECH，案例研究通过最好的100%在线教学方法得到加强：Relearning。

这种方法打破了传统的教学技术，将学生置于等式的中心，为他们提供不同格式的最佳内容。通过这种方式，您可以回顾和重申每个主题的关键概念并学习将它们应用到实际环境中。

沿着这些思路，根据多项科学研究，重复是最好的学习方式。因此，TECH在同一课程中以不同的方式重复每个关键概念8到16 次，目的是确保在学习过程中充分巩固知识。

Relearning 将使你的学习事半功倍，让你更多地参与到专业学习中，培养批判精神，捍卫论点，对比观点：这是通往成功的直接等式。

100%在线虚拟校园，拥有最好的教学材料

为了有效地应用其方法论，TECH 专注于为毕业生提供不同格式的教材：文本，互动视频，插图和知识图谱等。这些课程均由合格的教师设计，他们的工作重点是通过模拟将真实案例与复杂情况的解决结合起来，研究应用于每个职业生涯的背景并通过音频，演示，动画，图像等基于重复的学习。

神经科学领域的最新科学证据表明，在开始新的学习之前考虑访问内容的地点和背景非常重要。能够以个性化的方式调整这些变量可以帮助人们记住知识并将其存储在海马体中，以长期保留它。这是一种称为神经认知情境依赖电子学习的模型，有意识地应用于该大学学位。

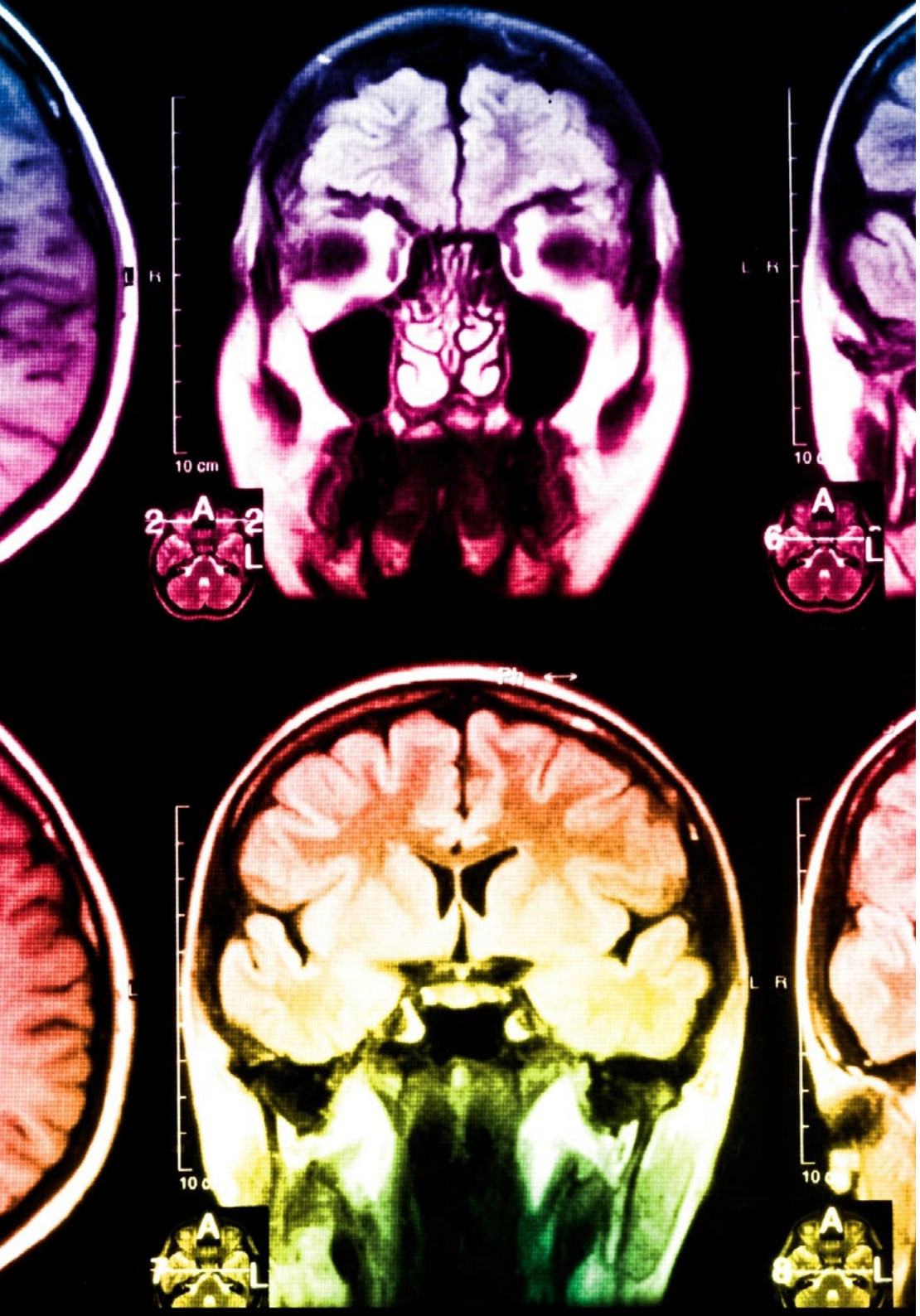
另一方面，也是为了尽可能促进指导者与被指导者之间的联系，提供了多种实时和延 迟交流的可能性（内部信息，论坛，电话服务，与技术秘书处的电子邮件联系，聊天和视频会议）。

同样，这个非常完整的虚拟校园将TECH学生根据个人时间或工作任务安排学习时间。通过这种方式，您将根据您加速的专业更新，对学术内容及其教学工具进行全局控制。

> 该课程的在线学习模式将您安排您的时间和学习进度，使其适应您的日程安排”

这个方法的有效性由四个关键成果来证明：

1. 遵循这种方法的学生不仅实现了对概念的吸收，而且还通过练习评估真实情况和应用知识来发展自己的心理能力。
2. 学习扎根于实践技能使学生能够更好地融入现实世界。
3. 由于使用了现实中出现的情况，思想和概念的学习变得更加容易和有效。
4. 感受到努力的成效对学生是一种重要的激励，这会转化为对学习更大的兴趣并增加学习时间。



最受学生重视的大学方法

这种创新学术模式的成果可以从TECH毕业生的整体满意度中看出。

学生们对教学质量，教材质量，课程结构及其目标的评价非常好。难怪该机构成为根据global score评分被学生评为最受欢迎的大学，获得了5分中的4.9分 。

由于TECH掌握着最新的技术和教学前沿，因此可以从任何具有互联网连接的设备（计算机，平板电脑，智能手机）访问学习内容。

你可以利用模拟学习环境和观察学习法（即向专家学习）的优势进行学习。

因此，在这门课程中，将提供精心准备的最好的教育材料：

学习材料

所有的教学内容都是由教授这门课程的专家专门为这门课程创作的，因此，教学的发展是具体的。

这些内容之后被应用于视听格式，这将创造我们的在线工作方式，采用最新的技术，使我们能够保证给你提供的每一件作品都有高质量。

技能和能力的实践

你将开展活动以发展每个学科领域的具体能力和技能。在我们所处的全球化框架内我们提供实践和氛围帮你获得成为专家所需的技能和能力。

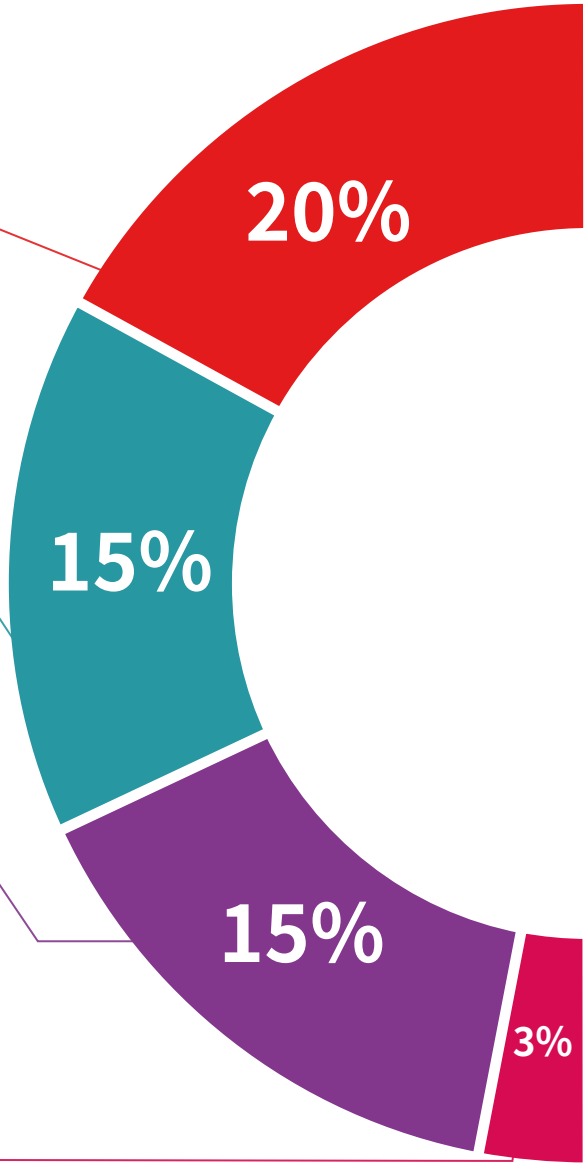
互动式总结

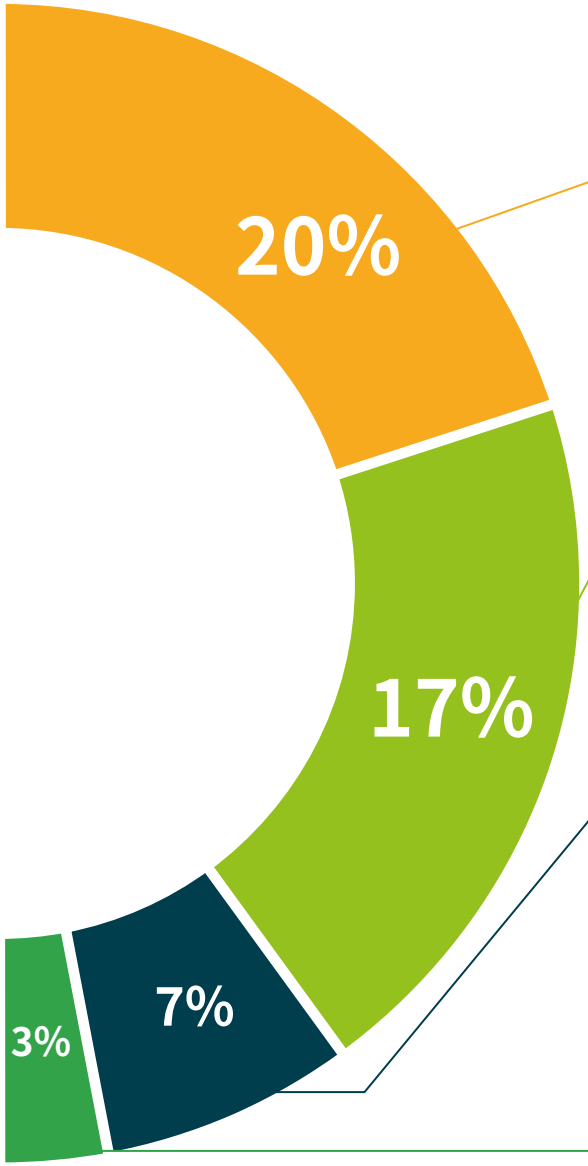
我们以有吸引力和动态的方式将内容呈现在多媒体中，包括音频，视频，图像，图表和概念图，以巩固知识。

这一用于展示多媒体内容的独特教育系统被微软公司评为 "欧洲成功案例"。

延伸阅读

最新文章，共识文件，国际指南...在我们的虚拟图书馆中，您将可以访问完成培训所需的一切。





案例研究

您将完成一系列有关该主题的最佳案例研究。由国际上最优秀的专家介绍，分析和指导案例。

Testing & Retesting

在整个课程中，我们会定期评估和重新评估你的知识。我们在米勒金字塔的4个层次中的3个层次上这样做。

大师班

科学证据表明第三方专家观察的效果显著。

向专家学习可以增强知识和记忆力、，并为我们今后做出艰难的决定建立信心。

快速行动指南

TECH以工作表或快速行动指南的形式提供课程中最相关的内容。一种帮助学生在学习中进步的综合,实用和有效的方法。

07

教学人员

这个软件质量与工程高级硕士是由一支在技术和专业项目管理和开发拥有多年经验的工程专家团队进行的。他们的专业背景为这个学位提供了有质量的教学。学生情境化的学习内容，将真实和模拟的实际案例带到学术经验中。此外，这个硕士提供 100% 的课程。

加入最抢手的软件质量与工程专家职业，
并利用这个机会成为他们中的一员”

国际客座董事

Darren Pulsipher es un **是一位经验丰富的软件架构师，**在软件和固件开发方面拥有杰出的**国际业绩的创新者**。事实上，他拥有高度发达的**沟通，项目管理**和**商业技能**，这使他能够领导全球范围内的重要举措。

在其职业生涯中，他还曾担任过许多重要职位，包括英特尔公司公共部门解决方案的**首席架构师，**在那里他**为**公共部门的客户，**合作伙伴**和**用户推广**现代业务，流程和**技术**。他还创立了**Yoly Inc.**并担任**首席执行官**，致力于利用**大数据和Web 2.0技术**开发 **基于**软件即服务（**SaaS**）**的社交媒体**聚合**和**诊断**工具**。

此外，他还曾在其他公司工作过，**例如担任戴尔科技集团**高级**工程总监，**领导大数据云业务部门，**带领美国和中国的团队**管理大型项目**以及**业务**部门**的重组和成功整合。他还曾担任 XanGo 的**首席信息官，负责管理**帮助**台支持，**生产**支持 和 解决方案开**发**等**项目。

他在众多专业领域中脱颖而出：**Edge to Cloud，网络安全，生成人工智能，软件开发，网络技术，云原生开发**和**容器生态系统**。他通过自己制作和展示的每周**播客**和**时事通讯** “Embracing Digital Transformation”分享知识，帮助组织通过利用人员，**流程**和**技术成功**实现**数字化**转**型**。

Pulsipher, Darren 先生

- 英特尔公共部门首席解决方案架构师，美国加利福尼亚州
- 加州“Embracing Digital Transformation”节目主持人兼制作人
- 阿肯色州 Yoly Inc. 创始人兼首席执行官
- 戴尔科技公司阿肯色州工程高级总监
- 犹他州XanGo首席信息官
- 加州 Cadence Design Systems 高级架构师
- 加州朗讯科技公司高级项目流程经理
- 加利福尼亚州 Cemax-Icon 软件工程师
- 加拿大 ISG Technologies 软件工程师
- 菲尼克斯大学技术管理 MBA
- 杨百翰大学计算机科学与电子工程学士学位

> 通过TECH你将能够与世界上最优秀的专业人士一起学习”

管理

Molina Molina, Jerónimo 先生

- Helphone 人工智能负责人
- NASSAT 的 AI 工程师兼软件架构师，移动互联网卫星
- Hexa Engineer 高级顾问
- 人工智能（ML 和 CV）介绍人
- 计算机视觉、ML/DL 和 NLP 领域人工智能解决方案
- Bancaixa 和 Fundeun 企业创建与发展大学专家
- 阿利坎特大学计算机工程师
- 阿维拉天主教大学的人工智能硕士
- 欧洲商业校园论坛 MBA 高级管理人员

教师

Rodríguez Míguez, Cándida 女士

- Getronics 初级应用程序开发人员
- 加利西亚人工智能网络联合创始人兼城市负责人
- Indra 初级软件工程师
- EDISA 的 Web 开发人员
- 毕业于维哥大学计算机工程专业
- 维哥大学计算机工程硕士学位

Pi Morell, Oriol 先生

- Fihoca 功能分析师
- CDmon 托管和电子邮件产品负责人
- Atmira 和 CapGemini 的功能分析师和软件工程师
- Capgemini 讲师，组建 Capgemini 和 Atmira
- 巴塞罗那自治大学计算机管理技术工程学位。
- 阿维拉天主教大学的人工智能硕士
- IMF Smart Education 工商管理硕士学位
- IMF Smart Education 信息系统管理硕士学位
- 加泰罗尼亚大学（UOC）设计模式研究生课程

Martínez, Francisco Javier 先生

- 电气电子专业工业技术工程师
- HEXA Engineers 的软件技术员
- Everis 高级 .Net 开发人员/.Net 解决方案架构师
- LaLiga 的软件分析师/架构师
- BBVA 的 Microsoft 现场工程师
- 自由职业 IT 技术顾问
- 在多个中心（Salesiano s，Maforem，Dreamsoft）担任 VisualStudio，SqlServer、CCNA（思科路由器和交换机）、PHP Web 编程和 .Net 培训师
- 工业技术工程师锥体电气、工业电子专业
- Cybernos .NET，MCAD硕士
- Eidos 高级编程硕士，专家级
- 拥有Dreamweave r、Fireworks、Flash 和 ActionScript 认证、MX 版本的 Web 管理员

Tenrero Morán, Marcos 先生

- Allot Communications 的 DevOps 工程师
- Cegid Meta4 应用程序生命周期管理经理
- Cegid Meta4 的 QA 自动化工程师
- 伽利略大学专业安卓应用程序开发硕士学位。危地马拉
- 马德里理工大学云服务开发、Node.Js、JavaScript、HTML5 硕士学位
- 使用 Angular-CLI (4)、Ionic 和 Node.Js、Meta4 进行 Web 开发胡安卡洛斯国王大学
- 毕业于雷伊胡安卡洛斯大学计算机工程专业

Acebes Tamargo, Patricia 女士

- 大数据咨询公司
- 运营部，在 Sirt 使用 Elasticsearch 和 Kivana 进行合作
- TIC 技术中心在线研究员人为因素和人工智能应用
- CTIC 技术中心在线研究员业务部
- CTIC 技术中心数字健康与积极老龄化系
- CTIC 技术中心数据科学系
- 瓦伦西亚理工大学人工智能计算机科学博士学位
- 毕业于奥维耶多大学经济系
- UCJC 数据分析硕士
- UNED 人工智能研究硕士学位
- 阿尔卡拉大学区块链，智能合约和加密货币硕士学位
- EADA 区块链工程研究生
- 奥维耶多大学经济学、计量经济学、经济分析硕士学位
- 经济学院税务硕士

08

学位

软件质量与工程高级硕士除了保证接受最严格和最新的培训，还可以获得由TECH 科技大学颁发的高级硕士学位证书。

顺利完成该课程后你将获得大学学位证书
无需出门或办理其他手续"

这个**Nombre del programa高级硕士**包含了市场上最完整和最新的课程。

评估通过后，学生将通过邮寄收到**TECH科技大学**颁发的相应的**高级硕士**学位。

学位由**TECH科技大学**颁发，证明在高级硕士学位中所获得的资质，并满足工作交流，竞争性考试和职业评估委员会的要求。

学位：**软件质量与工程高级硕士**

模式:**在线**

时长: **2年**


tech 科学技术大学

授予

文凭

_____先生/女士, 身份证号码_____

顺利通过并获得认证的课程是

高级硕士

软件质量与工程

这是一个由本大学授予的学位，相当于3,000个小时，
开始日期是 dd/mm/aaaa,结束日期是dd/mm/aaaa。

截至2018年6月28日，TECH是一所被公共教育部认可的私立高等教育机构。

2020年六月17日

Tere Guevara Navarro女士
校长

Unique TECH Code: AFWORD23S techtitute.com/certificates


软件质量与工程高级硕士

教学大纲概况

学期	学科	小时	性质
1°	软件的质量TRL发展水平	150	必修
1°	软件项目的开发功能和技术文档	150	必修
1°	软件测试自动化测试	150	必修
1°	架构模式瀑布模型方法与敏捷方法对比	150	必修
1°	TDD (Test Driven Developement),测试驱动的软件设计	150	必修
1°	DevOps.软件质量管理	150	必修
1°	DevOps 和持续集成软件开发中先进的实用解决方案	150	必修
1°	数据库设计(BD)。规范和性能软件质量	150	必修
1°	可扩展架构的设计软件生命周期的架构	150	必修
1°	ISO 质量标准, IEC 9126。软件质量指标	150	必修
2°	软件工程的方法、开发和质量	150	必修
2°	软件项目管理	150	必修
2°	软件开发平台	150	必修
2°	Web 客户端的计算	150	必修
2°	Web 服务器的计算	150	必修
2°	安全管理	150	必修
2°	软件安全	150	必修
2°	网络服务器管理	150	必修
2°	安全审计	150	必修
2°	在线应用程序的安全	150	必修

Tere Guevara Navarro女士
校长

tech 科学技术大学

*海牙加注。如果学生要求为他们的纸质资格证书提供海牙加注，TECH EDUCATION将采取必要的措施来获得，但需要额外的费用。

tech 科学技术大学

高级硕士
软件质量与工程

- 模式:**在线**
- 时长: **2年**
- 学位: **TECH 科技大学**
- 课程表:**自由安排时间**
- 考试模式:**在线**

高级硕士
软件质量与工程

tech 科学技术大学