.SWitCH();

June 2021| switch@portotechhub.com

Resource IT Conversion training

isep Instituto Superior de Engenharia do Porto

Porto. TechHub

Goal

- **To make a graduate with or without a STEM background to be able to:**

- **Develop high-quality standalone and web-based applications and services**
  - Understand and apply an iterative and incremental/agile development process
  - Analyze the requirements of a problem presented by a client and, as a team, specify a solution
  - Design the solution using appropriate architectural and design patterns
  - Implement the solution in Java/JavaScript using Test Driven Development (TDD) and Continuous Integration (CI)

- **Know and apply the essential technologies and best practices to be productive in an OO (Java) or web development team using CI**

The SWitCH approach

- **The SWitCH program aims to fulfil this objective:**
- A careful selection of the candidates
- 1 school year (32 weeks of classes)
- An innovative pedagogical approach based on project-based learning
- A barebones approach, focusing on the minimum content necessary for the students to be successful in the typical professional environment
- A semi-professional work environment based on teamwork, peer-learning and Scrum
- Focused and intensive: 35 hours/week in the classroom

Pedagogical approach

- **We call it CDIO-IL and it is a combination of:**

- Disciplinary teaching (33%)
  - Lectures and regular lab classes

- Project Based Learning (67%)
  - Project development in a team of 9-10 elements
  - Scrum software development process (2-weeks sprints)
  - Project is supplied by a real software company (product owner)
    - Global backlog of user stories and sprints' backlogs adapted to the learning process' needs

Pedagogical approach

- **How can it be so fast and effective?**
- By the careful application of selected active learning pedagogical patterns and project-based learning

- Learn by example
  - Providing solutions upfront help the student scaffolding knowledge

- Learn by doing
  - All work is to be produced in the context of the project, including in regular classes
  - The immediate application of new knowledge in a real application helps the students cement the knowledge and learning more effectively

Pedagogical approach

- **How can it be so fast and effective?**
- Teamwork and Peer-learning
  - Provides a first support network and helps the student consolidate learning
- Feedback
  - The teacher's main role is to provide quality feedback, promoting rework and improvement
- Focus on quality of work done
  - Grades reflect quality (below standards, met the standards, noteworthy)
  - Deliveries that don't meet the requirements and minimum quality standards are not accepted

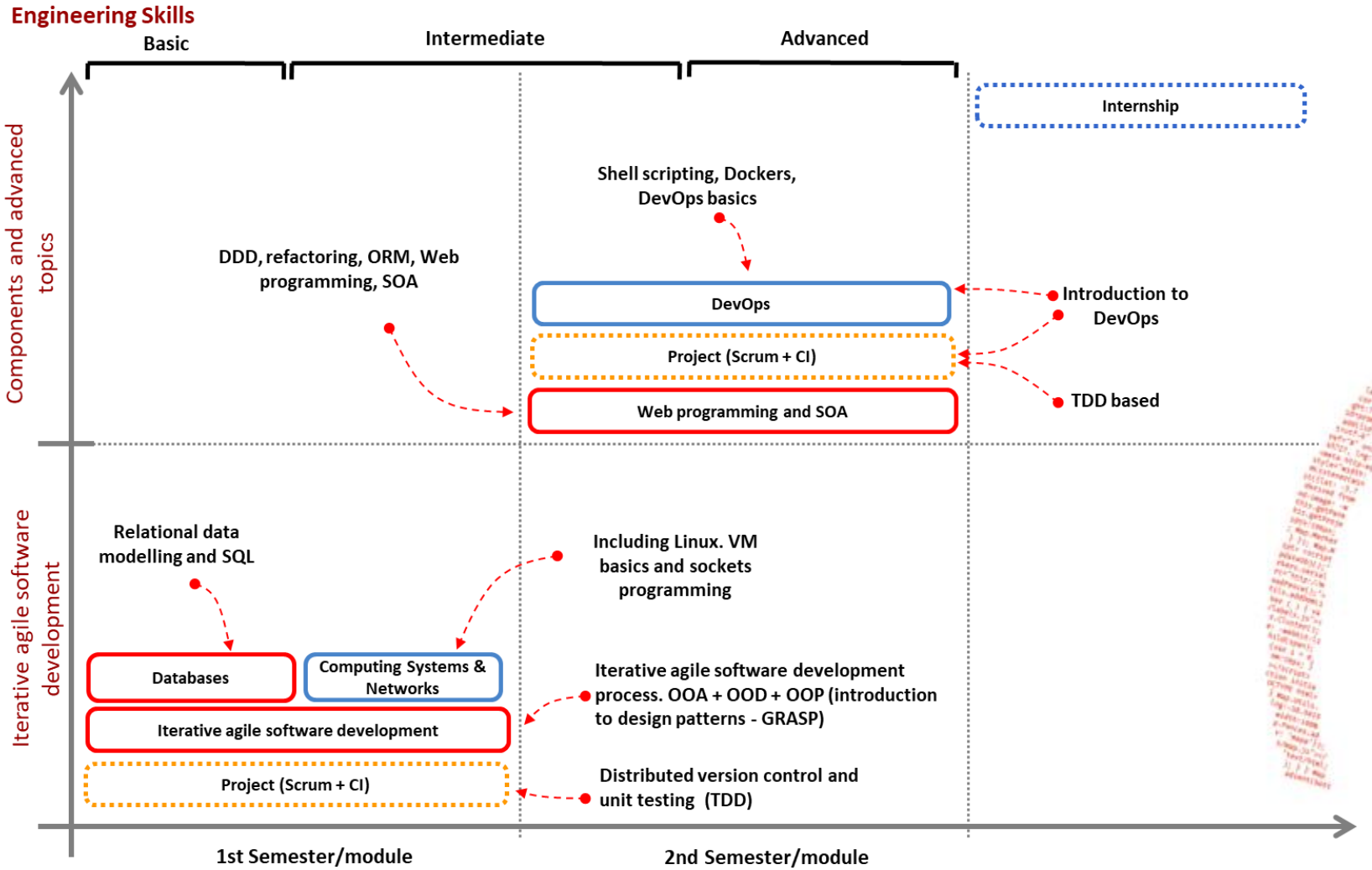
Program Structure

- **Two 16-week semesters**
- **Up to 30 students/class, 9-10 elements/group**
- **35 hours/week workload**
  - 15 hours/week of regular classes
  - 20 hours/week of autonomous work in a scrum team
- **3 simultaneous courses each semester**
  - 1 core software engineering and programming course
  - 1 technical course covering key supporting technologies and competencies
  - 1 project course to develop a product for a client

Program Structure

- **1st semester**
  - Iterative agile software development
  - Computing systems and networks
  - Databases
  - Project I
- **2nd semester**
  - Web programming and Service-Oriented Architecture (SOA)
  - DevOps
  - Project II

Program Structure



Timeline

2021/22 Calendar

June 09 to August 19	August 23 to September 03	September 06 and 07	September 08 to 22	September 23 and 24	October 06
Applications	Screening + Recruitment & Selection	Appointment of interviews	Interviews	Candidates Selection	Start of classes

Course Information

Iterative agile software development

- **Main goals**
  - Understand the need for the existence of a software development process (SDP) and the dimensions and stakeholders of the SDP
  - Apply an iterative and incremental agile SDP, using appropriate artefacts and notations for describing the problem, the analysis, the design and the solution
  - Apply methodologies of OO analysis and design, including simple OO design principles and patterns, namely GRASP, SOLID and GoF
  - Apply methodologies and tools for converting design to implementation in Java
  - Apply object-relational mapping techniques and tools

Iterative agile software development

- **Technologies**
  - UML
  - Software design patterns
  - Git version control system
  - Java
  - JUnit

Databases

- **Main goals**
  - Understand the fundamentals of database management systems (DBMS), with particular emphasis on relational databases
  - Apply relational data modelling principles to design a database for a project, taking into account functional requirements and constraints
  - Understand relational algebra and use SQL to manipulate information in a relational database

Databases

- **Technologies**
  - SQL
  - Oracle DBMS

Computing systems and networks

- **Main goals**
  - Understand the functional organization of a conventional computer
  - Use information representation and logical operations
  - Use shell script programming to solve small scale problems
  - Understand how the TCP/IP stack and the most relevant associated protocols work
  - Be able to design network application protocols, develop and implement network applications in Java

Computing systems and networks

- **Technologies**
  - Linux
  - TCP/IP
  - HTTP
  - Sockets
  - Java

Project I

- **Main goals**
  - Work in a team, in the context of a software development project, applying TDD and Scrum
  - Use the software development and teamwork tools and approaches that are appropriate for each moment and context
  - Apply methodologies of OO analysis and design, including simple OO design principles and patterns
  - Apply the appropriate methodologies and tools to implement and test the design in Java
  - Understand essential documentation and presentation best practices and apply them to communicate the results of the software project

Project I

- **Technologies**
  - UML
  - Scrum
  - Git + Jenkins + SonarQube
  - Java
  - JUnit
  - Oracle DBMS
  - SQL

Web programming & SOA

- **Main goals**
  - Analyze the requirements and concepts of a web-based Distributed and Decentralized Software Systems (DDSS)
  - Design DDSS using industry standard design and architecture styles and patterns, especially Domain-Driven Design (DDD)
  - Implement DDSS according to design models using a Java and JavaScript stack
  - Apply an iterative and incremental agile Software Development Process, using a test-driven approach

Web programming & SOA

- **Technologies**
  - DDD
  - DDSS software design and architectural patterns
  - JPA ORM
  - Java
  - Javascript/TypeScript
  - Web Services (REST)
  - React.JS
  - Etc.

DevOps

- **Main goals**
  - Understand the fundamentals of computer systems administration and security
  - Be able to install and manage Linux systems
  - Use virtual machines locally and in virtualization infrastructures
  - Use appropriate DevOps/scripting tools to automate the deployment of applications and containers
  - Use the different components of a continuous delivery pipeline

DevOps

- **Technologies**
  - Windows, Linux
  - VirtualBox, VMware
  - Chef
  - Ansible
  - Docker
  - Etc.

Project II

- **Main goals**
  - Work with a team, in the context of a software development project, applying Scrum and continuous integration (CI)
  - Apply industry standard methodologies and architectural patterns for DDSS analysis and design
  - Apply DevOps methodologies and tools to implement, test and deploy the application
  - Understand essential documentation and presentation best practices and apply them to communicate the results of the software project

Project II

- **Technologies**
  - UML
  - Scrum
  - Git + Jenkins + SonarQube (or an equivalent stack)
  - Docker containers
  - Java, JavaScript/TypeScript
  - Database (Oracle or MySQL)
  - Etc.

.SWitCH();

YOU CAN DO IT!

ASSOCIATED COMPANIES