Universidad Zaragoza
1542

Year : 2018/19

30249 - Software Engineering Laboratory

Syllabus Information

Academic Year:	2018/19
Subject:	30249 - Software Engineering Laboratory
Faculty / School:	110 -
Degree:	439 - Bachelor's Degree in Informatics Engineering
ECTS:	6.0
Year:	4
Semester:	Indeterminate
Subject Type:	
Module:	---

General information

Aims of the course

At the end of the course students will have developed a smart campus software application, from the requirements determination phase to the delivery of the finished product. This work will give them the opportunity to gain first contact with the domain of geographic information systems, as well as to learn and practice the technique of domain-driven design and some advanced patterns of software architecture, such as hexagonal architecture and event-driven architecture.

Context and importance of this course in the degree

In the first two years of the degree the student acquires skills and knowledge that will enable him/her to develop small computer applications, while in the third year the subjects of Software Engineering and Software Project enable him/her to tackle larger projects professionally. The Software Engineering Laboratory course deepens this training, while providing a practical dimension framed in the context of a specific problem area.

Recommendations to take this course

It is important to have taken the subjects of Software Engineering and Software Project before this course. Besides that it is convenient, although not essential, to have also studied Software Architectures and Software Project Management.

Learning goals

Competences

Upon passing the course, the student will be more competent to....

Successfully address the following cross-cutting issues:

CT1. Ability to conceive, design and develop engineering projects.
CT2. Ability to plan, budget, organize, direct and control tasks, people and resources.
CT4. Ability to solve problems and make decisions with initiative, creativity and critical thinking.
CT7. Ability to analyze and assess the social and environmental impact of technical solutions acting with ethics, professional responsibility and social commitment.
CT8. Ability to work in a multidisciplinary group and in a multilingual environment.

Successfully undertake the following tasks related to Computer Engineering in general:

CGC2. Ability to plan, design, deploy and manage projects, services and IT systems in all areas, leading their implementation and continuous improvement and assessing their economic and social impact.
CGC3. Ability to understand the importance of negotiation, effective work habits, leadership and communication skills in all software development environments.
CGC4. Ability to prepare the technical specifications of a computer installation that complies with current standards and regulations.
CGC8. Ability to analyze, design, build and maintain applications in a robust, secure and efficient manner, choosing the most appropriate paradigm and programming languages.

Successfully address the following Software Engineering related performance issues:

CEIS2. Ability to assess customer needs and specify software requirements to meet these needs, reconciling conflicting objectives by seeking acceptable compromises within the constraints of cost, time, existing systems already developed and the organisations themselves.
CEIS3. Ability to provide solutions to integration problems according to the available strategies, standards and technologies.
CEIS5. Ability to identify, assess and manage potential associated risks that may arise.
CEIS6. Ability to design appropriate solutions in one or more application domains using software engineering methods that integrate ethical, social, legal and economic aspects.

Learning goals

In order to pass this course, the student must demonstrate the following results...

It is able to propose different solutions to digitally preserve data and complete systems.
Learn about the activities involved in the process of building a component based system.
It is capable of applying domain engineering to identify, build, catalog and disseminate a set of software components that are applicable to existing and future software in a particular application domain.
Knows the characteristics and implications of an application domain when proposing a solution with software engineering methods.
Knows an infrastructure of processes and tools necessary to develop a software project, based on the best practices of software engineering available in a software factory business environment.
It puts into practice the knowledge acquired in the subjects of Software Engineering intensification in a specific project developed as a team: requirements, analysis, design, testing (verification and validation), project management.

Importance of learning goals

The reality of the industry shows that the development of software systems is almost always carried out on a software development infrastructure of a certain complexity and in a specific application domain. It is important that students face conditions as similar as possible to those they will encounter in their professional lives.

Assessment (1st and 2nd call)

Assessment tasks (description of tasks, marking system and assessment criteria)

The student must demonstrate that he or she has achieved the expected learning outcomes by taking an assessment test with two exercises:

Practical project (80%): the exercise consists of the delivery of results (technical report, source code and others) that reflect the work of the students in a "smart campus" software development project (learning outcomes 1-6).
Written test questions on concepts learned in theory and practice (20%): this exercise evaluates the knowledge acquired by each student in the theory, problems and practice sessions.

To pass the course, the total sum of both exercises must be at least 5 out of 10 points (it is not necessary to pass both exercises separately).

Methodology, learning tasks, syllabus and resources

Methodological overview

The learning process is based on:

1. Daily study and work
2. Learning concepts about a specific problem domain, and about the activities involved in developing a software project in this domain, during the lectures.
3. Applying these concepts to practical cases during problem-oriented interactive lectures
4. Laboratory assignments on a specialized problem domain (geographic information systems)
5. Teamwork on a project to develop a small software system following modern software engineering techniques

Learning tasks

1. Lectures in the classroom to develop the program
2. Problem-solving activities to put into practice the concepts and techniques in the program
3. Laboratory assignments to learn about the domain of geographic information systems
4. Team project: software development of a small software system

Syllabus

1. Introduction to geographic information systems.
2. Domain-driven design.
3. Advanced software architecture concepts: layered architectures, dependency inversion and hexagonal architecture.
4. Development of a team software project in the field of geographic information systems: smart campus application

Course planning and calendar

- Lectures (2 hours per week)
- Problems (1 hour per week)
- Laboratory assignments (5 sessions of 3 hours)

The students are expected to work:

- 35 hours in classroom activities (theory and problems)
- 15 hours in the laboratory assignments
- 105 hours of study and teamwork

Bibliography and recommended resources

[BB: Bibliografía básica / BC: Bibliografía complementaria]

- Zaragoza:
- [BB] Evans, Eric. Domain-driven design : tackling complexity in the heart of software / Eric Evans . Boston : Addison-Wesley, cop. 2004
- [BC] Vernon, Vaughn. Implementing Domain-Driven Design / Vaughn Vernon Addison Wesley, 2013.

Listado de URL

- Víctor Olaya. Sistemas de Información Geográfica (versión 1.0). Disponible bajo licencia Creative Commons Attribution [https://volaya.github.io/libro-sig/]
- Teruel:
- No hay relación bibliográfica para esta asignatura(Ver toda la bibliografía recomendada + enlace al catálogo)