Study programme(s): Information Technologies

Level: Bachelor

Course title: Architecture, Design and Patterns

Lecturer: Vladimir Kurbalija

Status: obligatory

ECTS: 7

Requirements: Object-oriented programming 1

Learning objectives

This course aims are to introduce students to a multitude of modelling techniques and designs to address the issue of software architecture in the context of object-oriented software development. Course covers all aspects of software design from architectural features (styles, models and views) to design models that could be described as "a common solution to common problems in a given context" on the lower level of abstraction.

Learning outcomes

Minimum: At the end of the course, it is expected that the successful student shows clear understanding of the impact of abstraction, modelling, architecture, and patterns in software product development and be able to critically discuss software architectures the key concepts, designs and patterns.

Desirable: At the end of the course, it is expected that the successful student is able to critically discuss the architectural alternatives and alternative designs, to generate a reasonable alternative for the problem and select between them, to identify an appropriate pattern for the problem and create it, and to apply practical skills in generating and developing software architecture and design based on functional requirements.

Syllabus

Theoretical instruction

Theoretical background of software architecture, analogy with architecture in general, the elements of software architecture, architectural styles (ABAS), architectural patterns (Event-based, Layered, Pipes & Filters, ...), architecture description languages, the interaction between requirements and architecture, master-plan vs. piecemeal growths, architecture analysis and evaluation (SAAM, Scenario-based evaluation), the architectural process and organization, model driven architecture, from the architecture to the model, re-usable architecture, design patterns, framework and tools.

Practical instruction

Case study analysis

Literature

1. Len Bass, Rick Kazman, Paul Clements, Software Architecture in Practice, Addison Wesley, second edition.
2. M. Shaw and D. Garlan, Software Architecture. Prentice Hall 1996
3. Erich Gamma, Richard Helm, Ralph Johnson, John M. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley.
4. H. Rumbaugh, M. Blaha, W. Premarlani, F. Eddy, W. Lorensen: Object-Oriented Modelling and Design, Prentice-
Hall
5. G. Booch: Object-Oriented Analysis and Design with Applications, Addison-Wesley, 1994 (2nd ed.)

Weekly teaching load				
Lectures: 3	Exercises: 0	Practical Exercises: 2	Student research:	Other:

Teaching methodology

During the lectures classical methodology is applied, through usage of beam-projector and slides projector. During the exercises the case studies and examples are analyzed by using the traditional methods of teaching and using the projector. Practical skills in architecture and modelling are developed with the introduction of the recommended tools. Students build their knowledge incrementally in each research topic. Knowledge is checked through the realization of projects that are presented during and at the end of the course.

Grading method (maximal number of points 100)

Pre-exam oblications	**points**	**Final exam**	**points**
Practical instruction	10	Oral exam	40
Seminar(s)	50		