

## SYLLABUS

### 1. Information regarding the programme

1.1 Higher education institution	<b>Babeş Bolyai University</b>
1.2 Faculty	<b>Faculty of Mathematics and Computer Science</b>
1.3 Department	<b>Department of Computer Science</b>
1.4 Field of study	<b>Computer Science</b>
1.5 Study cycle	<b>Master</b>
1.6 Study programme / Qualification	<b>Software engineering</b>

### 2. Information regarding the discipline

2.1 Name of the discipline	<b>Software design</b>						
2.2 Course coordinator	<b>Prof.PhD. Bazil Parv</b>						
2.3 Seminar coordinator	<b>Prof.PhD. Bazil Parv</b>						
2.4. Year of study	<b>1</b>	2.5 Semester	<b>2</b>	2.6. Type of evaluation	<b>E</b>	2.7 Type of discipline	<b>compulsory</b>

### 3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	<b>3</b>	Of which: 3.2 course	<b>2</b>	3.3 seminar/laboratory	<b>1</b>	
3.4 Total hours in the curriculum	<b>42</b>	Of which: 3.5 course	<b>28</b>	3.6 seminar/laboratory	<b>14</b>	
Time allotment:						hours
Learning using manual, course support, bibliography, course notes						<b>20</b>
Additional documentation (in libraries, on electronic platforms, field documentation)						<b>20</b>
Preparation for seminars/labs, homework, papers, portfolios and essays						<b>65</b>
Tutorship						<b>14</b>
Evaluations						<b>14</b>
Other activities: .....						-
3.7 Total individual study hours			<b>133</b>			
3.8 Total hours per semester			<b>175</b>			
3.9 Number of ECTS credits			<b>7</b>			

### 4. Prerequisites (if necessary)

4.1. curriculum	<ul style="list-style-type: none"> <li>• Fundamentals of programming</li> <li>• Object-oriented programming</li> <li>• Programming paradigms (optional)</li> </ul>
4.2. competencies	<ul style="list-style-type: none"> <li>• Average programming skills</li> </ul>

### 5. Conditions (if necessary)

5.1. for the course	<ul style="list-style-type: none"> <li>• Videoprojector, Internet access</li> </ul>
5.2. for the seminar /lab	<ul style="list-style-type: none"> <li>• Computers, Internet access, UML tool</li> </ul>

activities	
------------	--

## 6. Specific competencies acquired

<b>Professional competencies</b>	<ul style="list-style-type: none"> <li>• Understanding of the software design from the engineering perspective;</li> <li>• Understanding of the software design concepts and principles</li> <li>• Understanding of the software design process and its activities;</li> <li>• Proficient use of tools and languages specific to software systems development</li> <li>• Knowing the specifics of main architectural and design patterns and how to apply them to specific projects.</li> </ul>
<b>Transversal competencies</b>	<ul style="list-style-type: none"> <li>• Professional communication skills; concise and precise description, both oral and written, of professional results,</li> <li>• Independent and team work capabilities; able to fulfill different roles</li> <li>• Antepreneurial skills;</li> </ul>

## 7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	<ul style="list-style-type: none"> <li>• Know and understand fundamental concepts of software design.</li> <li>• Be able to apply the appropriate architectural and design patterns to different programming projects</li> </ul>
7.2 Specific objective of the discipline	<p>At the end of the course, students</p> <ul style="list-style-type: none"> <li>• know the main concepts and principles of software design</li> <li>• have a good understanding of the following terms: software architecture definition(s), architectural styles and models, architecture definition language(s); detailed design; design pattern, construction design;</li> <li>• learn the importance of architectural and detailed design and how to use tools for these tasks;</li> <li>• know several software system types ( taken from real-world applications) and the best recommended architectural styles and design patterns;</li> </ul>

## 8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction to Software Engineering Design. Motivation and general design concepts. Overview of the software engineering design	Exposure,description, explanation, debate and dialogue, discussion of case studies	
2. Software design fundamentals. UML Fundamentals.	Exposure,description, explanation	
3. UML structural modeling. UML behavioral modeling	Exposure,description, explanation, case studies	
4. Fundamentals of software architecture. Fundamentals of requirements engineering. Designing the software architecture	Exposure,description, explanation, case studies	

5. Overview and history of styles and patterns. Data-centered and data-flow systems	Exposure,description, explanation, case studies	
6. Distributed systems. Interactive and hierarchical systems	Exposure,description, explanation, case studies	
7. Overview of the detailed design. Structural and behavioral design of components	Exposure,description, explanation, discussion of case studies	
8. Creational design patterns in the detailed design. Abstract Factory. Factory Method	Exposure,description, explanation, discussion of case studies	
9. Creational design patterns in the detailed design. Builder. Prototype. Singleton.	Exposure,description, explanation, discussion of case studies	
10. Structural design patterns in the detailed design. Adapter. Composite. Facade	Exposure,description, explanation, discussion of case studies	
11. Behavioral design patterns in the detailed design. Iterator. Observer.	Exposure,description, explanation, discussion of case studies	
12. Construction design. Flow-, state-, and table-based construction design. Programming design language, styles, and quality evolution.	Exposure,description, explanation, discussion of case studies	
13. Software design management. Design management framework. Planning	Exposure,description, explanation, discussion of case studies	
14. Software design management. Implementation and termination. Final review	Exposure,description, explanation, discussion of case studies	

#### Bibliography

1. OTERO, C.E.: Software Engineering Design, CRC Press, 2012.  
site: <http://softwareengineeringdesign.com/Default.htm>
2. BASS, L., CLEMENTS, P., KAZMAN R.: Software Architecture in Practice, 2nd ed., Addison-Wesley, 2003
3. KRUCHTEN, PH.: Architectural Blueprints – The 4+1 View Model of Software Architecture, IEEE Software 12 (6), 1995, pp. 42-50.
4. SHAW, M.: The Coming-of-Age of Software Architecture Research, in Proc. of the 23rd ICSE, IEEE Comp. Soc. 2001, 656, [<http://www.cs.cmu.edu/afs/cs.cmu.edu/project/vit/ftp/pdf/shaw-keynote-rev.pdf>]
5. SHAW, M., GARLAN, D.: Software Architecture: Perspectives on an Emerging Discipline, Prentice-Hall, 1996.

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Administrivia	Conversation, debate, case studies	Seminar is organized as a total of 12 hours – 2 hours every other week
2. Establishing target application. First	Conversation, debate,	

