SYLLABUS

1. Information regarding the programme

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

2. Information regarding the discipline

2.1 Name of the discipline	**Service Oriented Architecture**						
2.2 Course coordinator	**Lect. dr. Ioan Lazar**						
2.3 Seminar coordinator	**Lect. dr. Ioan Lazar**						
2.4. Year of study	**2**	2.5 Semester	**1**	2.6. Type of evaluation	**E**	2.7 Type of discipline	**Mandatory**

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

3.1 Hours per week	4	Of which: 3.2 course	2	3.3 seminar/laboratory	1+1
3.4 Total hours in the curriculum	56	Of which: 3.5 course	28	3.6 seminar/laboratory	28
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					8
Additional documentation (in libraries, on electronic platforms, field documentation)					7
Preparation for seminars/labs, homework, papers, portfolios and essays					8
Tutorship					2
Evaluations					8
Other activities:					

3.7 Total individual study hours	144
3.8 Total hours per semester	200
3.9 Number of ECTS credits	8

4. Prerequisites (if necessary)

4.1. curriculum	• Programming Fundamentals
4.2. competencies	• Good programming skills in at least one of the languages Java, C#

5. Conditions (if necessary)

5.1. for the course	• Course hall with projector
5.2. for the seminar /lab activities	• Laboratory with computers

6. Specific competencies acquired

Professional competencies	• C 4.3 Identify models and methods adequate to real life problem solving • C 2.1 Identify adequate software systems development methodologies • C 1.1 Proper description of programming paradigms and language specific mechanisms, and identification of semantical an syntactical differences
Transversal competencies	• CT1 Apply organized and efficient work rules and responsible attitude towards didactical and research field, in order to creatively use work potential; respect professional ethical principles • CT3 Use efficient methods and techniques for: learning, information search, research and development of capacities to adapt to the requirements of a dynamic society and to communicate in an international language

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

7.1 General objective of the discipline	Enhance the students understanding of service oriented concepts through a practical and pragmatic approach Provide the students with an environment in which they can explore the usage and usefulness of service oriented concepts in various business scenarios Induce a realistic and industry driven view of software design concepts such as design patterns and their inherent benefits
7.2 Specific objective of the discipline	Give students the ability to explore various object oriented programming languages Improve the students abilities to tackle business requirements Enhance the students understanding of business needs and business value Provide students with insights into the way of working towards achieving high quality software through skilled trainers from the IT industry

8. Content

8.1 Course	Teaching methods	Remarks
1. Servers exposing REST services [2h] 1.1 PD/Distributed Systems [2h] Distributed service design	Exposure: description, explanation, examples, discussion of case studies	

- Stateful versus stateless protocols and services - CRUD operations - Search operations References - FHIR specification, https://www.hl7.org/fhir/http.html - KOA framework, http://koajs.com/		
2. Server-side notifications [2h] 2.1 PD/Distributed Systems [1h] Distributed service design - Reactive (IO-triggered) and multithreaded designs - ReactiveX, http://reactivex.io/rxjs/ 2.2 PD/Distributed Systems [1h] Distributed message sending - Web Sockets - Web sockets API, https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API	Exposure: description, explanation, examples, discussion of case studies	
3. Securing client-server applications [2h] 3.1 IAS/Web Security [1h] Web security model - Browser security model including same-origin policy - Client-server trust boundaries - JSON Web Tokens, https://jwt.io/ - OAuth, https://oauth.net/2/ 3.2 IAS/Web Security [1h] Client-side security - Web tokens - Web user tracking	Exposure: description, explanation, examples, discussion of case studies	
4. Microservices [2h] 4.1 PD/Cloud Computing [2h] Cloud services - Software as a service - Security - Seneca framework, http://senecajs.org/	Exposure: description, explanation, examples, discussion of case studies	
5. Containers [2h] 5.1 PD/Cloud Computing [1.5h] Virtualization - Multiple virtual cloud servers - Deploy servicess on multiple servers - Migration of processes Docker - https://www.docker.com/ 5.2 PD/Cloud Computing, Familiarity [0.5h] Explain the advantages and disadvantages of using virtualized infrastructure.	Exposure: description, explanation, examples, discussion of case studies	
6. Command query responsibility segregation [2h] 6.1 No topic mapping [2h] - Separating the update and read operations - CQRS, https://martinfowler.com/bliki/CQRS.html	Exposure: description, explanation, examples, discussion of case studies	
7. Application architecture based on events [2h]	Exposure: description,	

7.1 No topic mapping [2h] - Domain event, event collaboration, event sourcing, aggreemment dispatcher, parallel model - Further patterns of EAA, https://martinfowler.com/eaaDev/	explanation, examples, discussion of case studies	
8. Integration patterns [2h] 8.1 No topic mapping [2h] - Messaging systems - Messaging channels - Enterprise integration patterns, http://www.enterpriseintegrationpatterns.com/	Exposure: description, explanation, examples, discussion of case studies	
9. Integration patterns [0h] 9.1 No topic mapping [0h] - Message construction - Message routing	Exposure: description, explanation, examples, discussion of case studies	
10. Advanced message queuing protocol [2h] 10.1 No topic mapping [2h] - Routing, topics, work queue, publish/subscribe, RPC - RabbitMQ, https://www.rabbitmq.com/getstarted.html	Exposure: description, explanation, examples, discussion of case studies	
11. Serverless architectures [2h] 11.1 No topic mapping [2h] - Backend as a service - Function as a service - https://martinfowler.com/articles/serverless.html	Exposure: description, explanation, examples, discussion of case studies	
12. IoT applications and services [2h] 12.1 No topic mapping [2h] - IoT devices, platforms, services	Exposure: description, explanation, examples, discussion of case studies	
8.2 Seminar / laboratory	Teaching methods	Remarks
1. Modern web apps [2h] 1.1 PD/Distributed Systems, Usage [1h] Implement a simple server: - exposing rest services (CRUD, search) - sending notifications 1.2 PL/Event-Driven and Reactive Programming, Usage [1h] Implement a client app: - using reactive handlers	Dialogue, debate, case studies, examples, proofs	
2. Modern web apps [2h] 2.1 IAS/Web Security, Usage [2h] Use client-side security capabilities in an application.	Dialogue, debate, case studies, examples, proofs	
3. Creating a system based on microservices [2h]	Dialogue, debate, case studies,	

3.1 PD/Distributed Systems, Familiarity [1h] Describe the scalability challenges associated with a service growing to accommodate many clients. 3.2 PD/Cloud Computing, Familiarity [0.5h] Explain strategies to synchronize a common view of shared data across a collection of devices. 3.3 PD/Cloud Computing, Usage [0.5h] Deploy an application that uses cloud infrastructure for computing and/or data resources.	examples, proofs	
4. Synchronizing servers [2h] 4.1 No learning outcome mapping, Familiarity [2h] Use integration patters to synchronize servers	Dialogue, debate, case studies, examples, proofs	
5. Services implemented using AMQP [0h] 5.1 No learning outcome mapping, Familiarity [0h] Use AMQP messaging brokers to implement services	Dialogue, debate, case studies, examples, proofs	
6. Systems based on serverless architectures [2h] 6.1 No learning outcome mapping, Familiarity [2h] Provide and consume services defined according to BaaS and FaaS	Dialogue, debate, case studies, examples, proofs	

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

- The course respects the IEEE and ACM Curriculla Recommendations for Computer Science studies;
- The course exists in the studying program of all major universities in Romania and abroad;
- The content of the course is considered the software companies as important for average programming skills.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.5 Seminar/lab activities	Implement a system with REST services, server side notifications, and data synchronization	Project grading	100%
10.6 Minimum performance standards			
➢ A minimum passing grade is defined by attaining at least 50% (5/10) points for the final project and each of the three lab assignments respectively. ➢ No more than 3 absences are allowed for the seminar/lab activities			

Date	Signature of course coordinator	Signature of seminar coordinator
20.04.18	**Lect. dr. Ioan Lazar**	**Lect. dr. Ioan Lazar**

Date of approval

Signature of the head of department

Prof. dr. Anca Andrei