FIȘA DISCIPLINEI

Proiectarea Sistemelor Software

Anul universitar 2025 - 2026

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea Babeș-Bolyai din Cluj-Napoca
1.2. Facultatea	Facultatea de Matematică și Informatică
1.3. Departamentul	Departamentul de Informatică
1.4. Domeniul de studii	Informatică
1.5. Ciclul de studii	Master
1.6. Programul de studii / Calificarea	Inginerie Software
1.7. Forma de învățământ	Cu frecvență

2. Date despre disciplină

2.1. Denumirea disciplinei	Proiectarea Sistemelor Software					Codul disciplinei	**MME8065**
2.2. Titularul activităților de curs	Conf. Univ. Dr. Molnar Arthur-Jozsef						
2.3. Titularul activităților de seminar	Conf. Univ. Dr. Molnar Arthur-Jozsef						
2.4. Anul de studiu	**1**	2.5. Semestrul	**2**	2.6. Tipul de evaluare	**E**	2.7. Regimul disciplinei	**Opțional**

3. Timpul total estimat (ore pe semestru al activităților didactice)

3.1. Număr de ore pe săptămână	**3**	din care: 3.2. curs	**2**	3.3. seminar/ laborator/proiect	**1**
3.4. Total ore din planul de învățământ	**42**	din care: 3.5. curs	**28**	3.6 seminar/laborator/proiect	**14**
Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)					**ore**
Studiul după manual, suport de curs, bibliografie și notițe (AI)					30
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					40
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					40
Tutoriat (consiliere profesională)					10
Examinări					10
Alte activități					3
3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)				**133**	
3.8. Total ore pe semestru				**175**	
3.9. Numărul de credite				**7**	

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	-
4.2. de competențe	Competențe de programare în cel puțin un limbaj de programare care suportă paradigma orientată obiect; cunoștințe și competențe legate de fazele importante ale ciclului de dezvoltare software.

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sală de curs cu video-proiector și acces Internet.
5.2. de desfășurare a seminarului/ laboratorului	Sală de curs cu video-proiector și acces Internet.

6.1. Competențele specifice acumulate[1]

Competențe profesionale/esențiale	• Analiza, proiectarea și implementarea de sisteme software • Folosirea metodologiilor și instrumentelor specifice limbajelor de programare și ingineriei programării
Competențe transversale	• Abilități de muncă în echipă, cu preluarea diferitelor roluri de execuție și conducere pentru realizarea unor proiecte • Abilități de comunicare profesională: descrierea clară, concisă, verbală și în scris, a rezultatelor profesionale, abilități de negociere

6.2. Rezultatele învățării

Cunoștințe	• Absolventul/a are cunoștințe necesare pentru a concepe, modela și proiecta aplicații software complexe • Absolventul/a posedă cunoștințe fundamentale de modelare prin care analizează probleme din viața reală, le transpune în cerințe concrete și elaborează un model software corespunzător
Aptitudini	• Absolventul/a are abilitatea de a urmări întregul ciclu de viață al dezvoltării unui sistem software • Absolventul/a este capabil să folosească limbajul de specialitate și terminologia specifică ingineriei software, astfel încât să poată comunica și interacționa cu membrii unor echipe de lucru
Responsabilități și autonomie	• Absolventul/a are abilități de a realiza demersul de educare și pregătire pe diverse teme legate de dezvoltarea software • Absolventul/a este în măsură să aplice cunoștințe avansate de sisteme informatice, plecând de la studierea la un nivel ridicat de abstractizare a diferitelor sisteme, fiind capabil/ă să ofere soluții de implementare pentru aplicații la sisteme informatice complexe, integrate

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	Dobândirea de cunoștințe și competențe privind proiectarea, dezvoltarea și mentenanța sistemelor software moderne de mare complexitate.
7.2 Obiectivele specifice	• Cunoașterea și înțelegerea conceptelor fundamentale ale proiectării sistemelor software. • Cunoașterea tipurilor importante de sisteme software și a stilurilor arhitecturale și a șabloanelor de proiectare utilizate pentru dezvoltarea acestora. • Cunoașterea șabloanelor arhitecturale și de proiectare necesare pentru implementarea unei game diverse de sisteme informatice și competențele necesare pentru a le aplica.

[1] Se poate opta pentru competențe sau pentru rezultatele învățării, respectiv pentru ambele. În cazul în care se alege o singură variantă, se va șterge tabelul aferent celeilalte opțiuni, iar opțiunea păstrată va fi numerotată cu 6.

8. Conținuturi

<table>
<tr><th>8.1 Curs</th><th>Metode de predare</th></tr>
<tr><td>Introducere
Ciclul de dezvoltare software și procesul software; exemple practice – provocările de scalabilitate pentru o platformă care trece de la 1 la 1 milion de utilizatori.</td><td rowspan="9">• Expunere interactivă
• Explicație
• Conversație
• Exemple
• Demonstrație didactică</td></tr>
<tr><td>Provocări în domeniul dezvoltării software
Volatilitatea cerințelor, procesul, tehnologia, practici etice și profesionale, administrarea influențelor de design. Exemple practice de la companii și platforme importante.</td></tr>
<tr><td>Ciclul de dezvoltare software
Cerințe, arhitectura software, proiectarea detaliată, proiectarea de construcție, proiectarea interfeței om-calculator, documentarea și administrarea documentației de proiectare.</td></tr>
<tr><td>Șabloane și stiluri de arhitectură software
Arhitectura stratificată, client-server, peer-to-peer, MVC, broker, blackboard, master-slave, arhitecturi orientate pe servicii, microservicii, blockchain și contracte inteligente.</td></tr>
<tr><td>Stabilirea arhitecturii sistemului</td></tr>
<tr><td>Stabilirea stivei tehnologice</td></tr>
<tr><td>Prezentarea studiilor de caz
Arhitecturi ale unor sisteme de scară largă, precum cele utilizate pentru streaming media, social media, administrarea documentelor sau platforme de e-mail.</td></tr>
<tr><td>Proiectarea de construcție și în detaliu
Principiile SOLID, principii pentru proiectarea componentelor, șabloane de proiectare.</td></tr>
<tr><td>Securitatea software
Discuție și prezentări ale unor vulnerabilități hardware și software în sisteme software existente, introducere în analiza de risc pentru securitate și protecția datelor cu caracter personal, modelarea amenințărilor, bazele de date MITRE ATT&CK și Common Vulnerabilities and Exposures.</td></tr>
<tr><td>Calitatea și mentenanța software
Standarde și unelte pentru asigurarea calității software, anti-șabloane, code smells, datoria tehnică și refactorizarea.</td></tr>
<tr><td colspan="2">Bibliografie
1. E. Gamma, R. Helm, R.Johnson, J. Vlissides – Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley, 1995.
2. Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra - Head First Design Patterns, O'Reilly Media, 2004.
3. William J. Brown, Raphael C. Malveau, Hays W. "Skip" McCormick, Thomas J. Mowbray - AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, Wiley, 1998.
4. Hohpe Gregor, Woolf Bobby - Enterprise Integration Patterns, Addison-Wesley, 2003 (some resources at https://www.enterpriseintegrationpatterns.com/).
5. Martin Fowler - Refactoring. Improving the Design of Existing Code. Addison-Wesley, 1999.
6. Carlos Otero - Software Engineering Design - Theory and Practice, CRC Press, Taylor & Francis Group, 2012.
7. Alex Xu – System Design Interview – An Insider's Guide (Volumes I and II), ByteByteGo, 2020.
8. Martin Kleppmann - Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems, O'Reilly Media, 2017.
9. Robert C. Martin - Clean Architecture: A Craftsman's Guide to Software Structure and Design, Pearson, 2017.</td></tr>
<tr><th>8.2 Seminar / laborator</th><th>Metode de predare</th></tr>
<tr><td>Introducere; prezentarea metodei de evaluare și a proiectelor de semestru.</td><td rowspan="7">• Expunere interactivă
• Explicație
• Conversație
• Exemple
• Demonstrație didactică</td></tr>
<tr><td>Discuție inițială legată de proiectul de seminar și documentația de arhitectură.</td></tr>
<tr><td>Lucru pe proiectul de seminar și documentația de arhitectură. Prezentări tehnice susținute de studenți.</td></tr>
<tr><td>Prezentare legată de procesul de proiectare software în cazul aplicațiilor existente. Prezentări tehnice susținute de studenți.</td></tr>
<tr><td>Evaluarea primei faze a proiectului de seminar.</td></tr>
<tr><td>Lucru pe proiectul de seminar și documentația de arhitectură. Prezentări tehnice susținute de studenți.</td></tr>
<tr><td>Evaluarea fazei finale a proiectului de seminar.</td></tr>
<tr><td colspan="2">Bibliografie</td></tr>
</table>

1. E. Gamma, R. Helm, R.Johnson, J. Vlissides – ***Design Patterns: Elements of Reusable Object-Oriented Software***, Addison Wesley, 1995.
2. Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra - ***Head First Design Patterns***, O'Reilly Media, 2004.
3. William J. Brown, Raphael C. Malveau, Hays W. "Skip" McCormick, Thomas J. Mowbray - ***AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis***, Wiley, 1998.
4. Hohpe Gregor, Woolf Bobby - ***Enterprise Integration Patterns***, Addison-Wesley, 2003 (some resources at https://www.enterpriseintegrationpatterns.com/).
5. Martin Fowler - ***Refactoring. Improving the Design of Existing Code***. Addison-Wesley, 1999.
6. Carlos Otero - ***Software Engineering Design - Theory and Practice***, CRC Press, Taylor & Francis Group, 2012.
7. Alex Xu – ***System Design Interview – An Insider's Guide*** (Volumes I and II), ByteByteGo, 2020.
8. Martin Kleppmann - ***Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems***, O'Reilly Media, 2017.
9. Robert C. Martin - ***Clean Architecture: A Craftsman's Guide to Software Structure and Design***, Pearson, 2017.

9. Coroborarea conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatori reprezentativi din domeniul aferent programului

- Cursul respectă curriculumul IEEE și ACM pentru domeniul Informatică.
- Cursul există în programele de studiu ale universităților importante din România și din străinătate.

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	Prezentare în timpul cursului sau seminarului.	Calitatea prezentării și a exemplelor.	**20%**
	Crearea documentului de arhitectură pentru o aplicație software de mare complexitate.	Calitatea și nivelul de detaliu al documentației.	**40%**
	Examen scris în sesiunea de examene sau de restanțe.	Calitatea răspunsurilor furnizate.	**10%**
10.5 Seminar/laborator	Analiza arhitecturii și a evoluției unei aplicații complexe cu sursă deschisă.	Calitatea și nivelul de detaliu al documentației.	**30%**
10.6 Standard minim de performanță			
• Studenții vor respecta standardele de integritate academică. • O medie cel puțin egală cu 5.00 trebuie obținută prin susținerea prezentărilor în echipă, a proiectelor de seminar și a examenului scris.			

11. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)

Nu se aplică.

Data completării: 30.04.2025

Semnătura titularului de curs

Conf. Univ. Dr. Molnar Arthur-Jozsef

Semnătura titularului de seminar

Conf. Univ. Dr. Molnar Arthur-Jozsef

Data avizării în departament:
...

Semnătura directorului de departament

Conf.dr. Adrian STERCA