FIŞA DISCIPLINEI[1]

1. Date despre program

1.1 Instituţia de învăţământ superior	Universitatea Politehnică Timișoara
1.2 Facultatea[2] / Departamentul[3]	Facultatea de Inginerie Hunedoara / Inginerie Electrică și Informatică Industrială
1.3 Catedra	—
1.4 Domeniul de studii (denumire/cod[4])	INGINERIE ELECTRICĂ / 90
1.5 Ciclul de studii	Licenta
1.6 Programul de studii (denumire/cod/calificarea)	INGINERIE ELECTRICĂ ȘI CALCULATOARE / 60 / Inginer

2. Date despre disciplină

2.1 Denumirea disciplinei/Categoria formativă[5]			Inginerie software / DS				
2.2 Titularul activităţilor de curs			Ș. L. Dr. Iordan Anca-Elena				
2.3 Titularul activităţilor aplicative[6]			Ș. L. Dr. Ing. Abrudean Cristian				
2.4 Anul de studii[7]	IV	**2.5** Semestrul	II	**2.6** Tipul de evaluare	E	**2.7** Regimul disciplinei[8]	DO

3. Timp total estimat - ore pe semestru: activități didactice directe (asistate integral sau asistate parțial) și activități de pregătire individuală (neasistate) [9]

3.1 Număr de ore asistate integral/săptămână	3 , format din:	**3.2** ore curs	2	**3.3** ore seminar/laborator/proiect	1
3.1* Număr total de ore asistate integral/sem.	42 , format din:	**3.2*** ore curs	28	**3.3*** ore seminar/laborator/proiect	14
3.4 Număr de ore asistate parțial/săptămână	, format din:	**3.5** ore practică		**3.6** ore elaborare proiect de diplomă	
3.4* Număr total de ore asistate parțial/ semestru	, format din:	**3.5*** ore practică		**3.6*** ore elaborare proiect de diplomă	
3.7 Număr de ore activități neasistate/ săptămână	4,14 , format din:	ore documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren			1
		ore studiu individual după manual, suport de curs, bibliografie și notițe			2
		ore pregătire seminarii/laboratoare, elaborare teme de casă și referate, portofolii și eseuri			1,14
3.7* Număr total de ore activități neasistate/ semestru	58 , format din:	ore documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren			14
		ore studiu individual după manual, suport de curs, bibliografie și notițe			28
		ore pregătire seminarii/laboratoare, elaborare teme de casă și referate, portofolii și eseuri			16
3.8 Total ore/săptămână [10]	7,14				
3.8* Total ore/semestru	100				
3.9 Număr de credite	4				

4. **Precondiţii** (acolo unde este cazul)

4.1 de curriculum	• Programarea calculatoarelor și limbaje de programare 1, Programarea calculatoarelor și limbaje de programare 2, Programare orientată pe obiecte, Baze de date
4.2 de competenţe	•

[1] Formularul corespunde Fișei Disciplinei promovată prin OMECTS 5703/18.12.2011 și cerințelor Standardelor specifice ARACIS valabile începând cu 01.10.2017.
[2] Se înscrie numele facultății care gestionează programul de studiu căruia îi aparține disciplina.
[3] Se înscrie numele departamentului căruia i-a fost încredințată susținerea disciplinei și de care aparține titularul cursului.
[4] Se înscrie codul prevăzut în HG nr.140/16.03.2017 sau în HG similare actualizate anual.
[5] Disciplina se încadrează potrivit planului de învățământ în una dintre următoarele categorii formative: disciplină fundamentală (DF), disciplină de domeniu (DD), discipină de specialitate (DS) sau disciplina complementară (DC).
[6] Prin activități aplicative se înțeleg activitățile de: seminar (S) / laborator (L) / proiect (P) / practică (Pr).
[7] Anul de studii în care este prevăzută disciplina în planul de învățământ.
[8] Disciplina poate avea unul din următoarele regimuri: disciplină impusă (DI), disciplină opțională (DO) sau disciplină facultativă (Df).
[9] Numărul de ore de la rubricile 3.1*, 3.2*,…,3.8* se obțin prin înmulțirea cu 14 (săptămâni) a numărului de ore din rubricile 3.1, 3.2,…, 3.8. Informațiile din rubricile 3.1, 3.4 și 3.7 sunt chei de verificare folosite de ARACIS sub forma: (3.1)+(3.4) ≥ 28 ore/săpt. și (3.8) ≤ 40 ore/săpt.
[10] Numărul total de ore / săptămână se obține prin însumarea numărului de ore de la punctele 3.1, 3.4 și 3.7.

5. Condiţii (acolo unde este cazul)

5.1 de desfăşurare a cursului	• Sală de curs echipată cu videoproiector și conexiune la Internet. • Studenții nu se vor prezenta la prelegeri cu telefoanele mobile deschise.
5.2 de desfăşurare a activităţilor practice	• Sală de laborator echipată cu cu videoproiector și computere. • Studenții nu se vor prezenta la activitățile practice cu telefoanele mobile deschise.

6. Competenţe la formarea cărora contribuie disciplina

Competenţe specifice	C 3. C 3.1. Identificarea modelelor standard ale componentelor electrice şi electronice ce definesc funcţionarea sistemelor electrice modulare şi a metodelor de control software; C 3.2. Interpretarea datelor numerice obţinute în urma simulării şi testării modulelor electrice, electronice şi informatice; C 3.3. Utilizarea instrumentelor informatice pentru integrarea modulelor în sisteme electrice; C 3.4. Evaluarea performanţelor şi limitărilor obţinute pentru fiecare modul electric, electronic, informatic, precum şi a sistemului electric în ansamblu; C 3.5. Elaborarea de proiecte profesionale pe baza modelării, simulării şi testării modulelor sistemelor electrice. C 6. C 6.1. Descrierea structurii sistemelor informatice şi a modalităţii de accesare distribuită a resurselor; C 6.2. Identificarea şi interpretarea corectă a erorilor semnalate în sistem; C 6.3. Instalarea, configurarea şi întreţinerea aplicaţiilor software specifice ingineriei electrice; C 6.4. Monitorizarea funcţionării corecte a sistemului specific şi identificarea anomaliilor de funcţionare a aplicaţiilor software; C 6.5. Proiectarea sistemelor informatice aferente aplicaţiilor specifice ingineriei electrice. •
Competențele profesionale în care se înscriu competențele specifice	• C 3. Modelarea, simularea şi testarea asistată de calculator a modulelor electrice, electronice şi informatice ale sistemelor electrice. • C 6. Configurarea, realizarea, testarea, exploatarea şi întreţinerea sistemelor informatice specifice domeniului ingineriei electrice.
Competențele transversale în care se înscriu competențele specifice	•

7. Obiectivele disciplinei (asociate competențelor de la punctul 6)

7.1 Obiectivul general al disciplinei	• Scopul acestei discipline îl reprezintă însuşirea principalelor concepte, metode şi tehnici ale ingineriei software, cu accent pe reutilizarea codului sursă atât din punct de vedere al programării orientate pe obiect, cât şi din punct de vedere al programării generice în limbajele de programare C++ și C#.
7.2 Obiectivele specifice	Obiectivele specifice ale acestei discipline sunt: • Cunoaşterea etapelor ciclului de viaţă al unui sistem soft; • Înţelegerea conceptelor legate de modelarea softului; • Familiarizarea cu limbajul de modelare UML; • Abilitatea de a utiliza instrumente CASE.

8. Conţinuturi[11]

8.1 Curs	Număr de ore	Metode de predare[12]
1. Fazele dezvoltării unui sistem soft 1.1. Ciclul de viaţă al softului. Ciclul de viaţă V 1.2. Faza de analiză 1.3. Faza de proiectare 1.4. Faza de implementare 1.5. Faza de testare	2	Studenții au acces la curs în format electronic. Se vor utiliza atât prezentări interactive cât și tradiționale. Se vor folosi: problematizarea, studiu de caz, conversația.
2. Limbajul de modelare unificat UML 2.1. Caracteristicile şi avantajele UML 2.2. Tipuri de diagrame 2.3. Instrumente CASE: IBM Rational Rhapsody, Microsoft Visio, ArgoUML, StarUML 2.4. Diagrame specifice fazei de analiză 2.5. Diagrame specifice fazei de proiectare 2.6. Diagrame specifice fazei de implementare	4	
3. Analiza sistemelor soft 3.1. Colectarea şi analiza cerinţelor 3.2. Instrumente utilizate în specificarea cerinţelor. IBM Rational DOORS 3.3. Metode de analiză orientate pe obiecte	4	
4. Proiectarea sistemelor soft în C++ și C# 4.1. Caracteristicile unui proiectării orientate pe obiecte 4.2. Metode de proiectare 4.3. Principii de proiectare 4.3.1. Principiul deschis-închis 4.3.2. Principiul substituţiei 4.3.3. Principiul inversiunii dependenţelor 4.3.4. Principiul responsabilităţii unice 4.3.5. Principiul segregării interfeţelor 4.4. Şabloane arhitecturale 4.5. Şabloane de proiectare 4.5.1. Şabloane creaţionale 4.5.2. Şabloane structurale 4.5.3. Şabloane comportamentale	10	
5. Verificarea şi validarea sistemelor soft 5.1. Testarea programelor 5.1.1. Criterii de alegere a datelor de test 5.1.2. Testarea white-box 5.1.3. Testarea black-box 5.2. Instrumente de testare: QA-C++, csUnit, NUnit, xUnit.net 5.3. Verificarea modelelor, Verificarea specificaţiei şi a corectitudinii algoritmilor 5.4. Testarea modulelor şi a sistemului final 5.5. Verificarea documentaţiei 5.6. Validarea programelor 5.7. Certificarea produselor soft	6	
6. Managementul unui proiect software 6.1. Managementul software 6.2. Managementul configuraţiei 6.3. Managementul echipei 6.4. Instrumente integrate de manegement al proiectelor soft: JIRA, MKS Integrity 6.5. Estimarea costului unui sistem soft	2	

[11] Se detaliază toate activitățile didactice prevăzute prin planul de învățământ (tematicile prelegerilor și ale seminariilor, lista lucrărilor de laborator, conținuturile etapelor de elaborare a proiectelor, tematica fiecărui stagiu de practică). Titlurile lucrărilor de laborator care se efectuează pe standuri vor fi însoțite de notația „(*)".

[12] Prezentarea metodelor de predare va include și folosirea noilor tehnologii (e-mail, pagină personalizată de web, resurse în format electronic etc.).

Bibliografie[13]

1. Dorin Bocu, Răzvan Bocu, Provocări și metode de abordare în managementul proiectelor IT, Editura Albastră, Cluj-Napoca, 2013
2. Dorin Bocu, Răzvan Bocu, Modelare obiect orientată cu UML, Editura Albastră, Cluj-Napoca, 2006
3. Mircea Cezar Preda, Ana-Maria Mirea, Doina Lavinia Preda, Constantin Teodorescu-Mihai, Introducere în programarea orientată-obiect. Concepte fundamentale din perspectiva ingineriei software, Editura Polirom, Iași, 2010
4. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Șabloane de proiectare. Elemente de software reutilizabil orientat pe obiect, Editura Teora, 2002
5. Alan Ezust, Paul Ezust, Introduction to Design Patterns in C++ with Qt, Prentice Hall, 2011
6. Steven John Metsker, Design Patterns in C#, Addison-Wesley Professional, 2004
7. Pankaj Jalote, A Concise Introduction to Software Engineering, Springer-Verlag Londra, 2008

8.2 Activităţi aplicative[14]	Număr de ore	Metode de predare
1. Elaborarea diagramelor specifice fazei de analiză: diagrama cazurilor de utilizare şi diagrame de activităţi utilizând ArgoUML IBM Rational Rhapsody şi Visual Studio	2	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
2. Elaborarea diagramelor specifice fazei de proiectare: diagrama de clase, diagrama de pachete, diagrame de secvenţă, diagrame de colaborare şi diagrame de stare utilizând ArgoUML, IBM Rational Rhapsody şi Visual Studio	2	
3. Elaborarea diagramelor specifice fazei de implementare: diagrama de componente utilizând ArgoUML, IBM Rational Rhapsody şi Visual Studio	2	
4. Analiza, proiectarea şi implementarea de aplicaţii în C++ şi C# ce utilizează şabloane creaţionale	2	
5. Analiza, proiectarea şi implementarea de aplicaţii în C++ şi C# ce utilizează şabloane structurale	2	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
6. Analiza, proiectarea şi implementarea de aplicaţii în C++ şi C# ce utilizează şabloane comportamentale	2	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
7. Testare folosind instrumentele: QA-C++, csUnit, NUnit, xUnit.net	2	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.

Bibliografie[15]

1. Anca Daniela Ioniță, Modelarea UML în ingineria sistemelor de programare, Editura All, București, 2003
2. Tony Bevi, C# Design Pattern Essentials, Ability First Limited, 2012
3. Roy Osherove, The Art of Unit Testing with examples in C#, Manning Publications, 2013
4. Bender James, McWherter Jeff, Professional Test Driven Development with C#, Wrox, 2014
5. Jeff Langr, Modern C++ Programming with Test-Driven Development, 2013
6. Samek Miro, Practical UML Statecharts in C/C++, Newnes, 2013

9. Coroborarea conţinuturilor disciplinei cu aşteptările reprezentanţilor comunităţii epistemice, asociaţiilor profesionale şi angajatori reprezentativi din domeniul aferent programului

- Disciplina vine în întâmpinarea așteptărilor angajatorilor reprezentativi din domeniul aferent programului prin conținutul orelor de curs și laborator.

10. Evaluare

[13] Cel puțin un un titlu trebuie să aparțină colectivului disciplinei iar cel puțin un titlu trebuie să se refere la o lucrare de referință pentru disciplină, de circulație națională și internațională, existentă în biblioteca UPT.
[14] Tipurile de activități aplicative sunt cele precizate în nota de subsol 5. Dacă disciplina conține mai multe tipuri de activități aplicative atunci ele se trec consecutiv în liniile tabelului de mai jos. Tipul activității se va înscrie într-o linie distinctă sub forma: „Seminar:", „Laborator:", „Proiect:" și/sau „Practică:".
[15] Cel puțin un titlu trebuie să aparțină colectivului disciplinei.

Tip activitate	**10.1** Criterii de evaluare[16]	**10.2** Metode de evaluare	**10.3** Pondere din nota finală
10.4 Curs	Cunoștințe teoretice.	Oral - subiecte teoretice și aplicații	0,66
10.5 Activităţi aplicative	**S:**		
	L: Abilităţi în analiza, proiectarea și implementarea aplicațiilor de laborator	Oral – aplicații utilizând calculatorul	0,34
	P[17]**:**		
	Pr:		
10.6 Standard minim de performanţă (se prezintă cunoștințele minim necesare pentru promovarea disciplinei și modul în care se verifică stăpânirea lor[18])			
• La finalul cursului, respectiv al laboratorului, studenţii trebuie să aibă cunoştinţe teoretice şi abilităţi de cercetare, strict necesare viitorilor specialişti dovedind competenţe în analiza, proiectarea, implementarea și testarea sistemelor informatice dezvoltate în limbajele de programare C++ și C#.			

Data completării	**Titular de curs (semnătura)**	**Titular activităţi aplicative (semnătura)**
04.09.2017	……	……
Director de departament (semnătura)	**Data avizării în Consiliul Facultăţii**[19]	**Decan (semnătura)**
……	06.09.2017	……

[16] Fișele disciplinelor trebuie să conțină procedura de evaluare a disciplinei cu precizarea criteriilor, a metodelor și a formelor de evaluare, precum și cu precizarea ponderilor atribuite acestora în nota finală. Criteriile de evaluare se formulează în mod distinct pentru fiecare activitate prevăzută în planul de învățământ (curs, seminar, laborator, proiect). Ele se vor referi și la formele de verificare pe parcurs (teme de casă, referate ș.a.)

[17] În cazul când proiectul nu este o disciplină distinctă, în această rubrică se va preciza și modul în care rezultatul evaluării proiectului condiționează admiterea studentului la evaluarea finală din cadrul disciplinei.

[18] Nu se va explica cum se acorda nota de promovare.

[19] Avizarea este precedată de discutarea punctului de vedere al board-ului de care aparține programul de studii cu privire la fișa disciplinei.