

# Harta Informaticii: Un Ghid Practic pentru Navigarea în Informatică

Alessia - Diana Marin, Alexandra - Sofia Csizmadia,  
Alexandru Turculeț, Bianca - Mădălina Bădescu,  
Bogdan - Petru Mitoiu, Flavius-Andrei Neamu,  
Georgiana Andreea Cront, Isabella - Mariana Nicorescu,  
Maria Cinăzan, Mihai Robert Șerban, Raul Daniel Macovei,  
Ştefana Ivanik, Vlad Crăciunescu

Departamentul de Informatică, Facultatea de Matematică și Informatică,  
Universitatea de Vest din Timișoara, Timișoara, România

May 2025

## Abstract

Această lucrare propune o hartă a informaticii realizată din două componente: o aplicație dezvoltată în Python și lucrarea scrisă care o completează. Lucrarea este organizată în jurul domeniilor principale, fiecare domeniu fiind explorat din mai multe perspective: de la activități specifice la relațiile cu alte domenii, probleme actuale și în curs de rezolvare, personalități de marcă, dar și forumuri științifice de relevanță.

Harta oferă o modalitate de orientare în vastul domeniu al informaticii, atât pentru persoanele care doresc să activeze în domeniu cât și pentru persoanele curioase. Dorim să menționăm că acest proiect a fost realizat și se va utiliza doar în scop educativ.

## Contents

<b>1</b>	<b>Introducere</b>	<b>19</b>
1.1	Motivație . . . . .	19
<b>2</b>	<b>Algoritmi și Structuri de Date</b>	<b>20</b>
2.1	Teoria Algoritmilor . . . . .	21

2.1.1	Activitățile principale . . . . .	21
2.1.2	Relațiile cu celelalte subdomenii . . . . .	24
2.1.3	Probleme importante și probleme deschise . . . . .	25
2.1.4	Persoane importante . . . . .	26
2.1.5	Forumuri importante . . . . .	26
2.1.6	Dimensiunea locală și dimensiunea globală . . . . .	26
2.2	Algoritmi Fundamentali . . . . .	27
2.2.1	Activitățile principale . . . . .	27
2.2.2	Relațiile cu celelalte subdomenii . . . . .	29
2.2.3	Probleme importante și probleme deschise . . . . .	29
2.2.4	Persoane importante . . . . .	30
2.2.5	Forumuri importante . . . . .	30
2.2.6	Dimensiunea globală . . . . .	30
2.3	Structuri de Date Clasice . . . . .	30
2.3.1	Activitățile principale . . . . .	30
2.3.2	Relațiile cu celelalte subdomenii . . . . .	32
2.3.3	Probleme importante și probleme deschise . . . . .	32
2.3.4	Persoane importante . . . . .	33
2.3.5	Forumuri importante . . . . .	33
2.3.6	Dimensiunea locală și dimensiunea globală . . . . .	33
2.4	Structuri de Date Avansate . . . . .	33
2.4.1	Activitățile principale . . . . .	33
2.4.2	Relațiile cu celelalte subdomenii . . . . .	35
2.4.3	Probleme importante și probleme deschise . . . . .	35
2.4.4	Persoane importante . . . . .	35
2.4.5	Forumuri importante . . . . .	36
2.4.6	Dimensiunea globală . . . . .	36
2.5	Algoritmi pe Grafuri . . . . .	36
2.5.1	Activitățile principale . . . . .	36
2.5.2	Relațiile cu celelalte subdomenii . . . . .	38
2.5.3	Probleme importante și probleme deschise . . . . .	38
2.5.4	Persoane importante . . . . .	38
2.5.5	Forumuri importante . . . . .	38
2.5.6	Dimensiunea locală și dimensiunea globală . . . . .	39
2.6	Algoritmi Geometrici . . . . .	39
2.6.1	Activitățile principale . . . . .	39
2.6.2	Relațiile cu celelalte subdomenii . . . . .	41
2.6.3	Probleme importante și probleme deschise . . . . .	41
2.6.4	Persoane importante . . . . .	41
2.6.5	Forumuri importante . . . . .	41

2.6.6	Dimensiunea locală și dimensiunea globală . . . . .	41
2.7	Algoritmi Probabilistici și Randomizați . . . . .	42
2.7.1	Activitățile principale . . . . .	42
2.7.2	Relațiile cu celelalte subdomenii . . . . .	44
2.7.3	Probleme importante și probleme deschise . . . . .	44
2.7.4	Persoane importante . . . . .	44
2.7.5	Forumuri importante . . . . .	44
2.7.6	Dimensiunea locală și dimensiunea globală . . . . .	44
2.8	Algoritmi Paraleli și Distribuiți . . . . .	45
2.8.1	Activitățile principale . . . . .	45
2.8.2	Relațiile cu celelalte subdomenii . . . . .	46
2.8.3	Probleme importante și probleme deschise . . . . .	47
2.8.4	Persoane importante . . . . .	47
2.8.5	Forumuri importante . . . . .	47
2.8.6	Dimensiunea locală și dimensiunea globală . . . . .	47
<b>3</b>	<b>Limbaje de programare</b>	<b>48</b>
3.1	Programare imperativă . . . . .	48
3.1.1	Activitățile principale . . . . .	48
3.1.2	Relațiile cu celelalte subdomenii . . . . .	50
3.1.3	Probleme Importante și Probleme Deschise . . . . .	50
3.1.4	Persoane Importante . . . . .	51
3.1.5	Forum-uri Importante . . . . .	51
3.1.6	Dimensiunea locală și dimensiunea globală . . . . .	51
3.2	Programare orientată pe obiect . . . . .	51
3.2.1	Activitățile principale . . . . .	51
3.2.2	Probleme Importante și Probleme Deschise . . . . .	53
3.2.3	Persoane Importante . . . . .	53
3.2.4	Forum-uri Importante . . . . .	53
3.2.5	Dimensiunea locală și dimensiunea globală . . . . .	53
3.3	Programare funcțională . . . . .	54
3.3.1	Activitățile principale . . . . .	54
3.3.2	Probleme Importante și Probleme Deschise . . . . .	55
3.3.3	Persoane Importante . . . . .	56
3.3.4	Forum-uri Importante . . . . .	56
3.3.5	Dimensiunea locală și dimensiunea globală . . . . .	56
3.4	Programare logică . . . . .	56
3.4.1	Activitățile principale . . . . .	56
3.4.2	Relațiile cu celelalte subdomenii . . . . .	58
3.4.3	Probleme Importante și Probleme Deschise . . . . .	58
3.4.4	Persoane Importante . . . . .	58

3.4.5	Forum-uri Importante . . . . .	58
3.4.6	Dimensiunea locală și dimensiunea globală . . . . .	58
3.5	Programare declarativă . . . . .	59
3.5.1	Activitățile principale . . . . .	59
3.5.2	Relațiile cu celealte subdomenii . . . . .	60
3.5.3	Probleme Importante și Probleme Deschise . . . . .	61
3.5.4	Persoane Importante . . . . .	61
3.5.5	Forum-uri Importante . . . . .	61
3.5.6	Dimensiunea locală și dimensiunea globală . . . . .	61
3.6	Programare reactivă și event-based . . . . .	61
3.6.1	Activitățile principale . . . . .	62
3.6.2	Probleme Importante și Probleme Deschise . . . . .	63
3.6.3	Persoane Importante . . . . .	63
3.6.4	Forum-uri Importante . . . . .	63
3.6.5	Dimensiunea locală și dimensiunea globală . . . . .	64
3.7	Programare concurentă și paralelă . . . . .	64
3.7.1	Activitățile principale . . . . .	64
3.7.2	Relațiile cu celealte subdomenii . . . . .	66
3.7.3	Probleme Importante și Probleme Deschise . . . . .	66
3.7.4	Persoane Importante . . . . .	66
3.7.5	Forum-uri Importante . . . . .	66
3.7.6	Dimensiunea locală și dimensiunea globală . . . . .	66
3.8	Metaprogramming and Reflective programming . . . . .	67
3.8.1	Activitățile principale . . . . .	67
3.8.2	Relațiile cu celealte subdomenii . . . . .	69
3.8.3	Probleme Importante și Probleme Deschise . . . . .	69
3.8.4	Persoane Importante . . . . .	69
3.8.5	Forum-uri Importante . . . . .	69
3.8.6	Dimensiunea locală și dimensiunea globală . . . . .	69
<b>4</b>	<b>Arhitectură</b>	<b>70</b>
4.1	Arhitectura hardware . . . . .	70
4.1.1	Activitățile principale . . . . .	70
4.1.2	Relațiile cu celealte subdomenii . . . . .	71
4.1.3	Probleme Importante și Probleme Deschise . . . . .	71
4.1.4	Persoane Importante . . . . .	71
4.1.5	Forumuri Importante . . . . .	72
4.1.6	Dimensiunea locală și dimensiunea globală . . . . .	72
4.2	Arhitectura software . . . . .	72
4.2.1	Activitățile principale . . . . .	72
4.2.2	Relațiile cu celealte subdomenii . . . . .	73

4.2.3	Probleme Importante și Probleme Deschise . . . . .	73
4.2.4	Persoane Importante . . . . .	73
4.2.5	Forumuri Importante . . . . .	74
4.2.6	Dimensiunea locală și dimensiunea globală . . . . .	74
4.3	Arhitectura sistemelor distribuite . . . . .	74
4.3.1	Activitățile principale . . . . .	74
4.3.2	Relațiile cu celealte subdomenii . . . . .	75
4.3.3	Probleme Importante și Probleme Deschise . . . . .	75
4.3.4	Persoane Importante . . . . .	75
4.3.5	Forumuri Importante . . . . .	76
4.3.6	Dimensiunea locală și dimensiunea globală . . . . .	76
4.4	Arhitectura retelelor și comunicațiilor . . . . .	76
4.4.1	Activitățile principale . . . . .	76
4.4.2	Relațiile cu celealte subdomenii . . . . .	77
4.4.3	Probleme Importante și Probleme Deschise . . . . .	77
4.4.4	Persoane Importante . . . . .	77
4.4.5	Forumuri Importante . . . . .	77
4.4.6	Dimensiunea locală și dimensiunea globală . . . . .	77
4.5	Arhitectura Bazei de Date . . . . .	78
4.5.1	Activitățile principale . . . . .	78
4.5.2	Relațiile cu celealte subdomenii . . . . .	79
4.5.3	Probleme Importante și Probleme Deschise . . . . .	79
4.5.4	Persoane Importante . . . . .	79
4.5.5	Forumuri Importante . . . . .	79
4.5.6	Dimensiunea locală și dimensiunea globală . . . . .	79
4.6	Arhitectura Cloud si DevOps . . . . .	80
4.6.1	Activitățile principale . . . . .	80
4.6.2	Relațiile cu celealte subdomenii . . . . .	81
4.6.3	Probleme Importante și Probleme Deschise . . . . .	81
4.6.4	Persoane Importante . . . . .	81
4.6.5	Forumuri Importante . . . . .	81
4.6.6	Dimensiunea locală și dimensiunea globală . . . . .	81
4.7	Arhitectura securității informaticice . . . . .	82
4.7.1	Activitățile principale . . . . .	82
4.7.2	Relațiile cu celealte subdomenii . . . . .	83
4.7.3	Probleme Importante și Probleme Deschise . . . . .	83
4.7.4	Persoane Importante . . . . .	83
4.7.5	Forumuri Importante . . . . .	83
4.7.6	Dimensiunea locală și dimensiunea globală . . . . .	83

<b>5 Sisteme de operare și rețelistică</b>	<b>84</b>
5.1 Teoria planificării proceselor . . . . .	84
5.1.1 Activități principale . . . . .	84
5.1.2 Relațiile cu celealte subdomenii . . . . .	85
5.1.3 Probleme Importante și Probleme Deschise . . . . .	85
5.1.4 Persoane Importante . . . . .	86
5.1.5 Forumuri Importante . . . . .	86
5.1.6 Dimensiunea locală și dimensiunea globală . . . . .	86
5.2 Sisteme de fișiere . . . . .	86
5.2.1 Activitățile principale . . . . .	86
5.2.2 Relațiile cu celealte subdomenii . . . . .	87
5.2.3 Probleme Importante și Probleme Deschise . . . . .	88
5.2.4 Persoane Importante . . . . .	88
5.2.5 Forumuri Importante . . . . .	88
5.2.6 Dimensiunea locală și dimensiunea globală . . . . .	88
5.3 Gestiunea memoriei . . . . .	88
5.3.1 Activitățile principale . . . . .	88
5.3.2 Relațiile cu celealte subdomenii . . . . .	89
5.3.3 Probleme Importante și Probleme Deschise . . . . .	90
5.3.4 Persoane Importante . . . . .	90
5.3.5 Forumuri Importante . . . . .	90
5.3.6 Dimensiunea locală și dimensiunea globală . . . . .	90
5.4 Rețele de calculatoare . . . . .	90
5.4.1 Activitățile principale . . . . .	91
5.4.2 Relațiile cu celealte subdomenii . . . . .	91
5.4.3 Probleme Importante și Probleme Deschise . . . . .	92
5.4.4 Persoane Importante . . . . .	92
5.4.5 Forumuri Importante . . . . .	92
5.4.6 Dimensiunea locală și dimensiunea globală . . . . .	92
5.5 Programare concurrentă . . . . .	92
5.5.1 Activitățile principale . . . . .	93
5.5.2 Relațiile cu celealte subdomenii . . . . .	94
5.5.3 Probleme Importante și Probleme Deschise . . . . .	94
5.5.4 Persoane Importante . . . . .	94
5.5.5 Forumuri Importante . . . . .	95
5.5.6 Dimensiunea locală și dimensiunea globală . . . . .	95
5.6 Analiza Comportamentelor Programelor . . . . .	95
5.6.1 Activitățile principale . . . . .	95
5.6.2 Relațiile cu celealte subdomenii . . . . .	96
5.6.3 Probleme Importante și Probleme Deschise . . . . .	96

5.6.4	Persoane Importante . . . . .	96
5.6.5	Forumuri Importante . . . . .	97
5.6.6	Dimensiunea locală și dimensiunea globală . . . . .	97
<b>6</b>	<b>Inginerie software</b>	<b>98</b>
6.1	Dezvoltarea software . . . . .	98
6.1.1	Activitățile principale . . . . .	98
6.1.2	Relațiile cu celealte subdomenii . . . . .	99
6.1.3	Probleme importante și probleme deschise . . . . .	99
6.1.4	Persoane importante: cercetători, practicieni, contribuții . . . . .	100
6.1.5	Forumuri importante: reviste, conferințe, forumuri de specialitate . . . . .	100
6.1.6	Dimensiunea locală și globală . . . . .	100
6.2	Ingineria securității software . . . . .	100
6.2.1	Activitățile principale . . . . .	101
6.2.2	Relațiile cu celealte subdomenii . . . . .	101
6.2.3	Probleme importante și probleme deschise . . . . .	101
6.2.4	Persoane importante: cercetători, practicieni, contribuții . . . . .	102
6.2.5	Forumuri importante: reviste, conferințe, forumuri . . . . .	102
6.2.6	Dimensiunea locală și globală . . . . .	102
6.3	Arhitectura software . . . . .	102
6.3.1	Activitățile principale . . . . .	102
6.3.2	Relațiile cu celealte subdomenii . . . . .	103
6.3.3	Probleme importante și probleme deschise . . . . .	103
6.3.4	Persoane importante: cercetători, practicieni, contribuții . . . . .	103
6.3.5	Forumuri importante: reviste, conferințe, forumuri de specialitate . . . . .	103
6.3.6	Dimensiunea locală și globală . . . . .	104
6.4	Testarea și asigurarea calității software . . . . .	104
6.4.1	Activitățile principale . . . . .	104
6.4.2	Relațiile cu celealte subdomenii . . . . .	105
6.4.3	Probleme importante și probleme deschise . . . . .	105
6.4.4	Persoane importante: cercetători, practicieni, contribuții . . . . .	105
6.4.5	Forumuri importante: reviste, conferințe, forumuri . . . . .	106
6.4.6	Dimensiunea locală și globală . . . . .	106
6.5	Ingineria sistemelor software . . . . .	106
6.5.1	Activitățile principale . . . . .	106
6.5.2	Relațiile cu celealte subdomenii . . . . .	107
6.5.3	Probleme importante și probleme deschise . . . . .	108
6.5.4	Persoane importante: cercetători, practicieni, contribuții . . . . .	108

6.5.5	Forumuri importante: reviste, conferințe, forumuri de specialitate . . . . .	108
6.5.6	Dimensiunea locală și globală . . . . .	108
6.6	DevOps și automatizarea software . . . . .	109
6.6.1	Activitățile principale . . . . .	109
6.6.2	Relațiile cu celealte subdomenii . . . . .	110
6.6.3	Probleme importante și probleme deschise . . . . .	110
6.6.4	Persoane importante: cercetători, practicieni, contribuții . . . . .	111
6.6.5	Forumuri importante: reviste, conferințe, forumuri de specialitate . . . . .	111
6.6.6	Dimensiunea locală și globală . . . . .	111
<b>7</b>	<b>Baze de date si regasire de informatii</b>	<b>112</b>
7.1	Modele logice de date . . . . .	112
7.1.1	Activitățile principale . . . . .	113
7.1.2	Relațiile cu celealte subdomenii . . . . .	114
7.1.3	Probleme Importante și Probleme Deschise . . . . .	114
7.1.4	Persoane Importante . . . . .	115
7.1.5	Forumuri Importante . . . . .	115
7.1.6	Dimensiunea locală și dimensiunea globală . . . . .	115
7.2	Mecanisme de Stocare și Indexare . . . . .	115
7.2.1	Activitățile principale . . . . .	115
7.2.2	Relațiile cu celealte subdomenii . . . . .	118
7.2.3	Probleme Importante și Probleme Deschise . . . . .	118
7.2.4	Persoane Importante . . . . .	119
7.2.5	Forumuri Importante . . . . .	119
7.2.6	Dimensiunea locală și dimensiunea globală . . . . .	119
7.3	Acces la date și optimizarea interogărilor . . . . .	120
7.3.1	Activitățile principale . . . . .	120
7.3.2	Relațiile cu celealte subdomenii . . . . .	122
7.3.3	Probleme Importante și Probleme Deschise . . . . .	123
7.3.4	Persoane Importante . . . . .	123
7.3.5	Forumuri Importante . . . . .	123
7.3.6	Dimensiunea locală și dimensiunea globală . . . . .	123
7.4	Controlul concurenței și managementul tranzacțiilor . . . . .	124
7.4.1	Activitățile principale . . . . .	124
7.4.2	Relațiile cu celealte subdomenii . . . . .	126
7.4.3	Probleme importante și probleme deschise . . . . .	126
7.4.4	Persoane importante . . . . .	126
7.4.5	Forumuri importante . . . . .	127
7.4.6	Dimensiunea locală și dimensiunea globală . . . . .	127

7.5	Recuperarea și integritatea bazelor de date . . . . .	127
7.5.1	Activitățile principale . . . . .	127
7.5.2	Relațiile cu celealte subdomenii . . . . .	129
7.5.3	Probleme importante și probleme deschise . . . . .	129
7.5.4	Persoane importante . . . . .	130
7.5.5	Forumuri importante . . . . .	130
7.5.6	Dimensiunea locală și dimensiunea globală . . . . .	130
7.6	Securitate și confidențialitate în bazele de date . . . . .	130
7.6.1	Activitățile principale . . . . .	131
7.6.2	Relațiile cu celealte subdomenii . . . . .	132
7.6.3	Probleme importante și probleme deschise . . . . .	133
7.6.4	Persoane importante . . . . .	133
7.6.5	Forumuri importante . . . . .	133
7.6.6	Dimensiunea locală și dimensiunea globală . . . . .	133
7.7	Big Data și baze de date distribuite . . . . .	134
7.7.1	Activitățile principale . . . . .	134
7.7.2	Relațiile cu celealte subdomenii . . . . .	136
7.7.3	Probleme importante și probleme deschise . . . . .	136
7.7.4	Persoane importante . . . . .	136
7.7.5	Forumuri importante . . . . .	137
7.7.6	Dimensiunea locală și dimensiunea globală . . . . .	137
7.8	Depozitarea datelor și analiza . . . . .	137
7.8.1	Activitățile principale . . . . .	137
7.8.2	Relațiile cu celealte subdomenii . . . . .	139
7.8.3	Probleme importante și probleme deschise . . . . .	139
7.8.4	Persoane importante . . . . .	140
7.8.5	Forumuri importante . . . . .	140
7.8.6	Dimensiunea locală și dimensiunea globală . . . . .	140
7.9	Reprezentări avansate ale datelor și mașini virtuale . . . . .	140
7.9.1	Activitățile principale . . . . .	141
7.9.2	Relațiile cu celealte subdomenii . . . . .	142
7.9.3	Probleme importante și probleme deschise . . . . .	142
7.9.4	Persoane importante . . . . .	143
7.9.5	Forumuri importante . . . . .	143
7.9.6	Dimensiunea locală și dimensiunea globală . . . . .	143
7.10	Integrarea datelor eterogene și multimedia . . . . .	143
7.10.1	Activitățile principale . . . . .	143
7.10.2	Relațiile cu celealte subdomenii . . . . .	145
7.10.3	Probleme importante și probleme deschise . . . . .	145
7.10.4	Persoane importante . . . . .	146

7.10.5	Forumuri importante . . . . .	146
7.10.6	Dimensiunea locală și dimensiunea globală . . . . .	146
<b>8</b>	<b>AI și Robotica</b>	<b>147</b>
8.1	Învățare Automată și Deep Learning . . . . .	147
8.1.1	Activitățile principale . . . . .	148
8.1.2	Relațiile cu celealte subdomenii . . . . .	149
8.1.3	Probleme importante și probleme deschise . . . . .	149
8.1.4	Persoane importante . . . . .	149
8.1.5	Forumuri importante . . . . .	149
8.1.6	Dimensiunea locală și globală . . . . .	150
8.2	Procesarea Limbajului Natural și Viziune Computerizată . . . . .	150
8.2.1	Activitățile principale . . . . .	150
8.2.2	Relațiile cu celealte subdomenii . . . . .	151
8.2.3	Probleme importante și probleme deschise . . . . .	151
8.2.4	Persoane importante . . . . .	151
8.2.5	Forumuri importante . . . . .	151
8.2.6	Dimensiunea locală și globală . . . . .	151
8.3	AI Explicabil și Sisteme Multi-Agent . . . . .	152
8.3.1	Activitățile principale . . . . .	152
8.3.2	Relațiile cu celealte subdomenii . . . . .	152
8.3.3	Probleme importante și probleme deschise . . . . .	153
8.3.4	Persoane importante . . . . .	153
8.3.5	Forumuri importante . . . . .	153
8.3.6	Dimensiunea locală și globală . . . . .	153
8.4	Roboți Autonomi și Industriali . . . . .	153
8.4.1	Activitățile principale . . . . .	153
8.4.2	Relațiile cu celealte subdomenii . . . . .	154
8.4.3	Probleme importante și probleme deschise . . . . .	154
8.4.4	Persoane importante . . . . .	154
8.4.5	Forumuri importante . . . . .	154
8.4.6	Dimensiunea locală și globală . . . . .	155
8.5	Roboți Umanoizi și Colaborativi . . . . .	155
8.5.1	Activitățile principale . . . . .	155
8.5.2	Relațiile cu celealte subdomenii . . . . .	156
8.5.3	Probleme importante și probleme deschise . . . . .	156
8.5.4	Persoane importante . . . . .	156
8.5.5	Forumuri importante . . . . .	156
8.5.6	Dimensiunea locală și globală . . . . .	156
8.6	Robotica Medicală și Soft Robotics . . . . .	157
8.6.1	Activitățile principale . . . . .	157

8.6.2	Relațiile cu celealte subdomenii . . . . .	158
8.6.3	Probleme importante și probleme deschise . . . . .	158
8.6.4	Persoane importante . . . . .	158
8.6.5	Forumuri importante . . . . .	158
8.6.6	Dimensiunea locală și globală . . . . .	158
8.7	Robotica de Tip „Swarm” (Roboți de Roi) . . . . .	158
8.7.1	Activitățile principale . . . . .	159
8.7.2	Relațiile cu celealte subdomenii . . . . .	159
8.7.3	Probleme importante și probleme deschise . . . . .	160
8.7.4	Persoane importante . . . . .	160
8.7.5	Forumuri importante . . . . .	160
8.7.6	Dimensiunea locală și globală . . . . .	160
8.8	Robotica Cognitivă și Sisteme de Control Inteligente . . . . .	160
8.8.1	Activitățile principale . . . . .	160
8.8.2	Relațiile cu celealte subdomenii . . . . .	161
8.8.3	Probleme importante și probleme deschise . . . . .	161
8.8.4	Persoane importante . . . . .	161
8.8.5	Forumuri importante . . . . .	161
8.8.6	Dimensiunea locală și globală . . . . .	162
8.9	Simulare și Realitate Virtuală . . . . .	162
8.9.1	Activitățile principale . . . . .	162
8.9.2	Relațiile cu celealte subdomenii . . . . .	163
8.9.3	Probleme importante și probleme deschise . . . . .	163
8.9.4	Persoane importante . . . . .	163
8.9.5	Forumuri importante . . . . .	163
8.9.6	Dimensiunea locală și globală . . . . .	163
<b>9</b>	<b>Grafică</b>	<b>164</b>
9.1	Grafică 2D (bidimensională) . . . . .	164
9.1.1	Activitățile principale . . . . .	164
9.1.2	Relațiile cu celealte subdomenii . . . . .	165
9.1.3	Probleme Importante și Probleme Deschise . . . . .	165
9.1.4	Persoane Importante . . . . .	166
9.1.5	Forumuri Importante . . . . .	167
9.1.6	Dimensiunea locală și dimensiunea globală . . . . .	167
9.2	Grafică 3D (tridimensională) . . . . .	168
9.2.1	Activitățile principale . . . . .	168
9.2.2	Relațiile cu celealte subdomenii . . . . .	170
9.2.3	Probleme Importante și Probleme Deschise . . . . .	170
9.2.4	Persoane Importante . . . . .	170
9.2.5	Forumuri Importante . . . . .	171

9.2.6	Dimensiunea locală și dimensiunea globală . . . . .	171
9.3	Realitate Virtuală (VR) . . . . .	171
9.3.1	Activitățile principale . . . . .	171
9.3.2	Relațiile cu celealte subdomenii . . . . .	175
9.3.3	Probleme Importante și Probleme Deschise . . . . .	175
9.3.4	Persoane Importante . . . . .	175
9.3.5	Forum-uri Importante . . . . .	175
9.3.6	Dimensiunea locală și dimensiunea globală . . . . .	175
9.4	Realitate Augmentată (AR) . . . . .	176
9.4.1	Activitățile principale . . . . .	176
9.4.2	Relațiile cu celealte subdomenii . . . . .	179
9.4.3	Probleme Importante și Probleme Deschise . . . . .	179
9.4.4	Persoane Importante . . . . .	179
9.4.5	Forum-uri Importante . . . . .	179
9.4.6	Dimensiunea locală și dimensiunea globală . . . . .	179
9.5	Grafică pentru Jocuri Video . . . . .	180
9.5.1	Activitățile principale . . . . .	180
9.5.2	Relațiile cu celealte subdomenii . . . . .	182
9.5.3	Probleme Importante și Probleme Deschise . . . . .	182
9.5.4	Persoane Importante . . . . .	183
9.5.5	Forumuri Importante . . . . .	183
9.5.6	Dimensiunea locală și dimensiunea globală . . . . .	183
9.6	Grafică Computerizată pentru Imagini Generate pe Calculator (CGI)	183
9.6.1	Activitățile principale . . . . .	184
9.6.2	Relațiile cu celealte subdomenii . . . . .	185
9.6.3	Probleme Importante și Probleme Deschise . . . . .	186
9.6.4	Persoane Importante . . . . .	187
9.6.5	Forumuri Importante . . . . .	187
9.6.6	Dimensiunea locală și dimensiunea globală . . . . .	187
9.7	Vizualizarea Datelor . . . . .	188
9.7.1	Activitățile principale . . . . .	188
9.7.2	Relațiile cu celealte subdomenii . . . . .	190
9.7.3	Probleme Importante și Probleme Deschise . . . . .	190
9.7.4	Persoane Importante . . . . .	190
9.7.5	Forum-uri Importante . . . . .	190
9.7.6	Dimensiunea locală și dimensiunea globală . . . . .	191
9.8	Grafică Interactivă(GUI) . . . . .	191
9.8.1	Activitățile principale . . . . .	191
9.8.2	Relațiile cu celealte subdomenii . . . . .	193
9.8.3	Probleme Importante și Probleme Deschise . . . . .	193

9.8.4	Persoane Importante . . . . .	194
9.8.5	Forumuri Importante . . . . .	194
9.8.6	Dimensiunea locală și dimensiunea globală . . . . .	194
<b>10</b>	<b>Interacțiunea Om-Computer</b>	<b>195</b>
10.1	Designul interfețelor utilizatorului (UI/UX Design) . . . . .	195
10.1.1	Activitățile principale . . . . .	195
10.1.2	Relațiile cu celealte subdomenii . . . . .	196
10.1.3	Probleme importante și probleme deschise . . . . .	197
10.1.4	Persoane importante . . . . .	197
10.1.5	Forumuri importante . . . . .	197
10.1.6	Dimensiunea locală și globală . . . . .	197
10.2	Interacțiunea bazată pe inteligență artificială . . . . .	197
10.2.1	Activitățile principale . . . . .	198
10.2.2	Relațiile cu celealte subdomenii . . . . .	199
10.2.3	Probleme importante și probleme deschise . . . . .	199
10.2.4	Persoane importante . . . . .	199
10.2.5	Forumuri importante . . . . .	199
10.2.6	Dimensiunea locală și globală . . . . .	199
10.3	Interacțiunea om-computer în medii colaborative . . . . .	200
10.3.1	Activitățile principale . . . . .	200
10.3.2	Relațiile cu celealte subdomenii . . . . .	201
10.3.3	Probleme importante și probleme deschise . . . . .	201
10.3.4	Persoane importante . . . . .	201
10.3.5	Forumuri importante . . . . .	201
10.3.6	Dimensiunea locală și globală . . . . .	202
10.4	Factori umani și psihologia interacțiunii . . . . .	202
10.4.1	Activitățile principale . . . . .	202
10.4.2	Relațiile cu celealte subdomenii . . . . .	203
10.4.3	Probleme importante și probleme deschise . . . . .	204
10.4.4	Persoane importante . . . . .	204
10.4.5	Forumuri importante . . . . .	204
10.4.6	Dimensiunea locală și globală . . . . .	205
10.5	Interacțiunea om-computer în contexte specifice . . . . .	205
10.5.1	Activitățile principale . . . . .	205
10.5.2	Relațiile cu celealte subdomenii . . . . .	206
10.5.3	Probleme importante și probleme deschise . . . . .	206
10.5.4	Persoane importante . . . . .	207
10.5.5	Forumuri importante . . . . .	207
10.5.6	Dimensiunea locală și globală . . . . .	207
10.6	Securitate și etica în HCI . . . . .	207

10.6.1	Activitățile principale . . . . .	208
10.6.2	Relațiile cu celealte subdomenii . . . . .	208
10.6.3	Probleme importante și probleme deschise . . . . .	208
10.6.4	Persoane importante . . . . .	209
10.6.5	Forumuri importante . . . . .	209
10.6.6	Dimensiunea locală și globală . . . . .	209
<b>11</b>	<b>Știință Computațională</b>	<b>210</b>
11.1	Analiză numerică . . . . .	210
11.1.1	Activitățile principale . . . . .	210
11.1.2	Relațiile cu celealte subdomenii . . . . .	211
11.1.3	Probleme Importante și Probleme Deschise . . . . .	211
11.1.4	Persoane Importante . . . . .	212
11.1.5	Forumuri Importante . . . . .	212
11.1.6	Dimensiunea locală și dimensiunea globală . . . . .	212
11.2	Calcul de Înaltă Performanță . . . . .	212
11.2.1	Activități Principale . . . . .	212
11.2.2	Relații cu Alte Subdomenii . . . . .	213
11.2.3	Probleme Importante și Probleme Deschise . . . . .	214
11.2.4	Persoane Importante . . . . .	214
11.2.5	Forumuri Importante . . . . .	214
11.2.6	Dimensiune Locală și Globală . . . . .	214
11.3	Aplicații în Fizica Computațională . . . . .	215
11.3.1	Activități principale . . . . .	215
11.3.2	Relațiile cu celealte subdomenii . . . . .	216
11.3.3	Probleme Importante și Probleme Deschise . . . . .	216
11.3.4	Persoane Importante . . . . .	216
11.3.5	Forumuri Importante . . . . .	216
11.3.6	Dimensiunea locală și dimensiunea globală . . . . .	216
11.4	Aplicații în Biologia Computațională . . . . .	217
11.4.1	Activități principale . . . . .	217
11.4.2	Relațiile cu celealte subdomenii . . . . .	218
11.4.3	Probleme Importante și Probleme Deschise . . . . .	218
11.4.4	Persoane Importante . . . . .	218
11.4.5	Forumuri Importante . . . . .	218
11.4.6	Dimensiunea locală și dimensiunea globală . . . . .	219
11.5	Aplicații în Chimia Computațională . . . . .	219
11.5.1	Activități principale . . . . .	219
11.5.2	Relațiile cu celealte subdomenii . . . . .	220
11.5.3	Probleme Importante și Probleme Deschise . . . . .	220
11.5.4	Persoane Importante . . . . .	220

11.5.5 Forumuri Importante . . . . .	221
11.5.6 Dimensiunea locală și dimensiunea globală . . . . .	221
11.5.7 Bibliografie . . . . .	221
11.6 Aplicații în Economia Computațională . . . . .	221
11.6.1 Activități principale . . . . .	221
11.6.2 Relațiile cu celealte subdomenii . . . . .	222
11.6.3 Probleme Importante și Probleme Deschise . . . . .	222
11.6.4 Persoane Importante . . . . .	223
11.6.5 Forumuri Importante . . . . .	223
11.6.6 Dimensiunea locală și dimensiunea globală . . . . .	223
11.6.7 Bibliografie . . . . .	223
11.7 Aplicații în Ingineria Computațională . . . . .	223
11.7.1 Activități principale . . . . .	223
11.7.2 Relațiile cu celealte subdomenii . . . . .	224
11.7.3 Probleme Importante și Probleme Deschise . . . . .	225
11.7.4 Persoane Importante . . . . .	225
11.7.5 Forumuri Importante . . . . .	225
11.7.6 Dimensiunea locală și dimensiunea globală . . . . .	225
<b>12 Informatica Organizatoriala</b>	<b>226</b>
12.1 Sisteme de management al informației (MIS) . . . . .	226
12.1.1 Activitățile principale . . . . .	226
12.1.2 Relațiile cu celealte subdomenii . . . . .	228
12.1.3 Probleme importante și probleme deschise . . . . .	228
12.1.4 Persoane importante . . . . .	229
12.1.5 Forumuri importante . . . . .	229
12.1.6 Dimensiunea locală și dimensiunea globală . . . . .	229
12.2 Baze de date și gestionarea informațiilor . . . . .	230
12.2.1 Activitățile principale . . . . .	230
12.2.2 Relațiile cu celealte subdomenii . . . . .	231
12.2.3 Probleme importante și probleme deschise . . . . .	232
12.2.4 Persoane importante . . . . .	232
12.2.5 Forumuri importante . . . . .	232
12.2.6 Dimensiunea locală și dimensiunea globală . . . . .	233
12.3 Arhitectura sistemelor informatice pentru organizații . . . . .	233
12.3.1 Activitățile principale . . . . .	233
12.3.2 Relațiile cu celealte subdomenii . . . . .	235
12.3.3 Probleme importante și probleme deschise . . . . .	235
12.3.4 Persoane importante . . . . .	235
12.3.5 Forumuri importante . . . . .	235
12.3.6 Dimensiunea locală și dimensiunea globală . . . . .	236

12.4	Rețele și securitatea informațiilor în organizații . . . . .	236
12.4.1	Activitățile principale . . . . .	236
12.4.2	Relațiile cu celealte subdomenii . . . . .	238
12.4.3	Probleme importante și probleme deschise . . . . .	238
12.4.4	Persoane importante . . . . .	238
12.4.5	Forumuri importante . . . . .	238
12.4.6	Dimensiunea locală și globală . . . . .	239
12.5	Automatizarea și optimizarea proceselor organizaționale . . . . .	239
12.5.1	Activitățile principale . . . . .	239
12.5.2	Relațiile cu celealte subdomenii . . . . .	241
12.5.3	Probleme importante și probleme deschise . . . . .	241
12.5.4	Persoane importante . . . . .	241
12.5.5	Forumuri importante . . . . .	241
12.5.6	Dimensiunea locală și globală . . . . .	241
12.6	Interacțiunea om-calculator în mediul organizațional . . . . .	242
12.6.1	Activitățile principale . . . . .	242
12.6.2	Relațiile cu celealte subdomenii . . . . .	243
12.6.3	Probleme importante și probleme deschise . . . . .	243
12.6.4	Persoane importante . . . . .	244
12.6.5	Forumuri importante . . . . .	244
12.6.6	Dimensiunea locală și globală . . . . .	244
12.7	Guvernanța IT și managementul strategic al tehnologiilor . . . . .	244
12.7.1	Activitățile principale . . . . .	245
12.7.2	Relațiile cu celealte subdomenii . . . . .	246
12.7.3	Probleme importante și probleme deschise . . . . .	246
12.7.4	Persoane importante . . . . .	246
12.7.5	Forumuri importante . . . . .	247
12.7.6	Dimensiunea locală și globală . . . . .	247
<b>13</b>	<b>Bioinformatică</b>	<b>248</b>
13.1	Forumuri importante . . . . .	248
13.2	Dimensiunea locală și dimensiunea globală . . . . .	250
13.3	Analiza secvențelor biologice . . . . .	251
13.3.1	Activitățile principale . . . . .	252
13.3.2	Relațiile cu celealte subdomenii . . . . .	253
13.3.3	Probleme Importante și Probleme Deschise . . . . .	255
13.3.4	Persoane Importante . . . . .	257
13.4	Genomica . . . . .	258
13.4.1	Activitățile principale . . . . .	258
13.4.2	Relațiile cu celealte subdomenii . . . . .	260
13.4.3	Probleme Importante și Probleme Deschise . . . . .	260

13.4.4 Persoane Importante . . . . .	262
13.5 Proteomica . . . . .	263
13.5.1 Activitățile principale . . . . .	263
13.5.2 Relațiile cu celealte subdomenii . . . . .	264
13.5.3 Probleme Importante și Probleme Deschise . . . . .	265
13.5.4 Persoane Importante . . . . .	266
13.6 Bioinformatica structurală . . . . .	267
13.6.1 Activitățile principale . . . . .	267
13.6.2 Relațiile cu celealte subdomenii . . . . .	269
13.6.3 Probleme Importante și Probleme Deschise . . . . .	270
13.6.4 Persoane Importante . . . . .	272
13.7 Bioinformatica evoluționistă . . . . .	273
13.7.1 Activitățile principale . . . . .	274
13.7.2 Relațiile cu celealte subdomenii . . . . .	275
13.7.3 Probleme Importante și Probleme Deschise . . . . .	275
13.7.4 Persoane Importante . . . . .	276
13.8 Bioinformatica medicală . . . . .	278
13.8.1 Activitățile principale . . . . .	278
13.8.2 Relațiile cu celealte subdomenii . . . . .	279
13.8.3 Probleme Importante și Probleme Deschise . . . . .	280
13.8.4 Persoane Importante . . . . .	281
13.9 Bioinformatica aplicată în farmacologie . . . . .	281
13.9.1 Activitățile principale . . . . .	282
13.9.2 Relațiile cu celealte subdomenii . . . . .	283
13.9.3 Probleme Importante și Probleme Deschise . . . . .	283
13.9.4 Persoane Importante . . . . .	285
13.10 Biotehnologie . . . . .	286
13.10.1 Activitățile principale . . . . .	286
13.10.2 Relațiile cu celealte subdomenii . . . . .	287
13.10.3 Probleme Importante și Probleme Deschise . . . . .	287
13.10.4 Persoane Importante . . . . .	288
13.11 Bioinformatica sistemelor . . . . .	288
13.11.1 Activitățile principale . . . . .	288
13.11.2 Relațiile cu celealte subdomenii . . . . .	289
13.11.3 Probleme Importante și Probleme Deschise . . . . .	289
13.11.4 Persoane Importante . . . . .	290
<b>14 Implementarea aplicației “Map Explorer”</b>	<b>290</b>
14.1 Structura aplicației . . . . .	290
14.2 Ce se află în spatele interfeței?	292



# 1 Introducere

Informatica este un domeniu apărut relativ recent în forma sa actuală. Aceasta se ocupă cu mult mai mult decât pur și simplu programare. În spatele se ocupă cu studiul calculului automat, al programării și al automatizării. [1]. Precum multe alte domenii, domeniul informaticii nu există într-un vid, ci are multe conexiuni cu alte domenii, mai ales cu matematica. Aceasta poate fi împărțit în diferite moduri, în funcție de nevoi, dar cea mai utilizată clasificare este cea a lui Denning. [2]

În concepția lui Denning, informatica este alcătuită din 12 subdomenii, iar fiecare subdomeniu este împărțit în 3 subsecțiuni:

**Teorie** Noțiunile necesare subdomeniului

**Experiment** Ariile de cercetare practică ale subdomeniului

**Design** Produse create pe baza teoriei și experimentului

Bazele domeniului au fost puse încă din secolul al XVII-lea cu primul calculator mecanic construit în 1623. O altă contribuție notabilă o reprezintă munca matematicianului George Boole care a creat sistemul de logică binară folosit de o covârșitoare majoritate a calculatoarelor construite în ultimii 70 de ani.

În Al Doilea Război Mondial, Alan Turing a creat mașinaria Enigma pentru a descifra cifrul folosit de Germania Nazistă. Totodată, el a introdus conceptul pe care noi îl numim mașina Turing, mecanism teoretic cu 3 operații simple (citire element de pe bandă, scriere element pe bandă, mutarea stânga sau dreapta) ce a definitivat teoretic funcționarea tuturor calculatoarelor moderne.

În anii '50 calculatoarele de tip mainframe dominau mediul de afaceri mulțumită invenției tranzistorului ce a făcut ca circuitele pentru memoria de lucru să fie mai fiabile. Atunci au fost create primele limbi de programare precum limbajul assembly și COBOL pentru a ajuta în crearea programelor. De atunci, în ultimii 70 de ani device-urile s-au micșorat, iar performanțele și capabilitățile mecanismelor au crescut razant, ajungând în 2025 să poată simula un proces de raționament.

## 1.1 Motivație

Domeniul informaticii este unul foarte vast, precum observă și Peter J. Denning în [2]. De aceea, acesta a împărțit această disciplină în 12 arii tematice. Chiar și așa, aceste arii tematice sunt la rândul lor foarte vaste, datorită multelor cercetări în domeniul nostru din ultimii 70 de ani. Acest fapt face găsirea unei nișe potrivite pentru un student de licență de anul 1 la specializarea informatică destul de complicat. Ca atare, propunem spre rezolvarea acestei probleme un program scris în Python care să servească drept o „ hartă” digitală a acestui domeniu pentru a

ne putea crea o idee despre ariile importante și noțiunile, precum și aptitudinile necesare pentru a ne specializa spre nișă respectivă. Pentru un studiu mai amănunțit recomandăm citirea acestei lucrări care servește drept o completare a aplicației.

**Din punct de vedere al originalității** informațiile folosite au fost cercetate printr-un efort comun a 9 persoane, iar elementele grafice folosite în program au fost concepute manual de alte 4 persoane din echipă. Totodată, programul care face posibilă existența componentei interactive a proiectului este o implementare originală, realizată de o singură persoană.

Lucrarea de față completează aplicația interactivă disponibilă pe GitHub (<https://github.com/alexandraCsizmadia06/Harta-Informaticii>), oferind informații detaliate despre fiecare domeniu explorat vizual în hartă. După ce utilizatorul accesează aplicația și identifică subdomeniul dorit, poate reveni la această lucrare pentru a aprofunda informațiile teoretice: activități specifice, legături între domenii, probleme deschise, personalități și forumuri relevante. Aplicația servește astfel ca punct de plecare vizual și intuitiv, iar documentul ca resursă complementară, organizată logic pentru studiu aprofundat. Structura lucrării este următoarea: capitolul 1 prezintă introducerea, capitolele 2–13 sunt dedicate domeniilor principale ale informaticii, capitolul 14 detaliază implementarea practică a aplicației, iar capitolul 15 oferă concluziile generale. Această structură facilitează navigarea rapidă și corelarea între partea teoretică și componenta interactivă a proiectului.

## 2 Algoritmi și Structuri de Date

Domeniul „Algoritmi și Structuri de Date” reprezintă una dintre componentele fundamentale ale informaticii teoretice și aplicate. Aceasta se ocupă cu studiul metodelor sistematice de rezolvare a problemelor, cu optimizarea acestora, precum și cu organizarea eficientă a datelor. Este un domeniu esențial în formarea gândirii algoritmice și în dezvoltarea aplicațiilor informatice, oferind un fundament teoretic solid pentru diverse arii ale informaticii: inteligență artificială, baze de date, programare, securitate cibernetică și nu numai.

Subdomeniile principale:

- Teoria Algoritmilor
- Algoritmi Fundamentali
- Structuri de Date Clasice
- Structuri de Date Avansate
- Algoritmi pe Grafuri

- Algoritmi Geometrici
- Algoritmi Probabilistici și Randomizați
- Algoritmi Paraleli și Distribuiți

## 2.1 Teoria Algoritmilor

Teoria algoritmilor reprezintă o ramură fundamentală a informaticii teoretice care studiază proprietățile formale ale algoritmilor, precum corectitudinea, eficiența și complexitatea acestora. Un algoritm este definit ca o secvență finită de pași bine determinați, concepuți pentru a rezolva o problemă sau pentru a efectua o anumită sarcină computațională.

Dezvoltarea teoriei algoritmilor este strâns legată de fundamentele matematicii și logicii, în special de lucrările lui Church și Turing (1936), [3]. Aceștia au introdus, respectiv, calculul lambda și mașina Turing, două modele formale de calcul care au permis definirea riguroasă a noțiunii de funcție calculabilă. Ulterior, contribuțiile lui Gödel (1931), [4], au adus o perspectivă axiomatică asupra limitelor logicii formale, iar lucrările lui Cook (1971), [5], și Karp (1972), [6], au introdus concepte esențiale în analiza complexității algoritmilor, precum clasele P și NP și problema NP-completitudinii.

În prezent, teoria algoritmilor constituie baza pentru evaluarea performanței oricărui sistem computațional, prin studierea comportamentului algoritmilor în funcție de resursele consumate, în special timpul de execuție și memoria utilizată. De asemenea, acest subdomeniu contribuie la înțelegerea limitelor computabilității, demonstrând că anumite probleme sunt indecidabile sau intractabile.

Teoria algoritmilor nu are doar un rol teoretic, ci influențează direct dezvoltarea de soluții eficiente pentru probleme concrete din domenii precum inteligența artificială, criptografie, bioinformatică sau ingineria software. Ea furnizează un cadru abstract de analiză, necesar atât pentru cercetarea fundamentală, cât și pentru optimizarea aplicațiilor reale.

### 2.1.1 Activitățile principale

1. Teorie:

(a) **Fundamentele teoretice ale subdomeniului**

Teoria algoritmilor se bazează pe conceptul de algoritm ca entitate abstractă, tratată independent de detaliile de implementare. Un algoritm este privit ca un proces computabil care transformă un input într-un output, conform unui set de reguli precise. În acest cadru, accentul cade pe caracteristicile esențiale ale algoritmilor: corectitudinea (producerea

unui rezultat corect pentru orice instanță validă), eficiența (utilizarea optimă a resurselor) și terminarea (garantarea unui număr finit de pași).

(b) **Modele formale și paradigmă de analiză**

Principalele cadre teoretice care susțin analiza algoritmilor includ:

- *Modelul RAM (Random Access Machine)*, utilizat pentru estimarea timpului și spațiului de execuție;
- *Mașina Turing*, ca model universal de calcul, folosit pentru studii de calculabilitate și indecidabilitate;
- *Teoria complexității*, care clasifică problemele în funcție de resursele necesare pentru rezolvare.

(c) **Metodologii de demonstrare și analiză algoritmică**

Argumentele teoretice din acest domeniu se bazează pe demonstrații formale, deseori realizate prin inducție, reducere, contradicție sau construcție explicită. Pentru problemele de optimizare, se utilizează metode de analiză amortizată, tehnici probabilistice și evaluări ale cazului mediu sau ale cazului cel mai defavorabil. De asemenea, reducerea între probleme este o tehnică centrală în studiul problemelor NP-complete.

(d) **Întrebări centrale în cercetarea teoretică**

Printre întrebările fundamentale formulate în teoria algoritmilor se regăsesc:

- Ce probleme pot fi rezolvate de o mașină de calcul?
- Ce algoritmi sunt cei mai eficienți pentru o anumită problemă?
- Există probleme care nu pot fi rezolvate în timp polinomial?
- Cum putem demonstra că o problemă este „greu de rezolvat”?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

În cercetarea teoretică a algoritmilor, explorarea modelelor presupune analizarea comportamentului algoritmilor în diverse contexte formale. Modele precum RAM, PRAM, mașina Turing sau rețelele distribuite permit evaluarea performanței algoritmilor în funcție de constrângerile structurale.

(b) **Implementare algoritmică și evaluare empirică**

Deși teoria algoritmilor are un caracter formal, validarea experimentală este esențială pentru a compara comportamentul practic al algoritmilor cu estimările teoretice. Aceasta implică implementarea algoritmilor și analizarea metricilor de performanță.

(c) **Studii de caz și aplicații concrete**

Un exemplu reprezentativ este algoritmul lui Dijkstra pentru determinarea celor mai scurte drumuri într-un graf orientat cu costuri nenegative, [7]. Deși analiza teoretică arată că algoritmul are o complexitate de  $O((V + E) \log V)$ , validarea experimentală permite observarea comportamentului acestuia pe grafuri reale, precum rețele de transport sau hărți digitale, cum este proiectul OpenStreetMap [8].

(d) **Metode consacrate și abordări avansate**

Printre metodele consacrate în validarea experimentală se numără: simularea pe inputuri mari și analiza pe benchmarkuri standardizate. Metodele avansate includ analiza pe date sintetice și instrumentarea automată a codului.

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Numeroase biblioteci software au fost dezvoltate plecând de la concepte teoretice fundamentale din algoritmică. De exemplu, Boost Graph Library (BGL) oferă o colecție extinsă de algoritmi și structuri de date pentru grafuri, implementate în C++ și inspirate din cercetările academice riguroase, [9]. De asemenea, LEDA (Library of Efficient Data types and Algorithms) pune la dispoziție funcționalități pentru algoritmi geometrico-combinatoriali, oferind precizie și eficiență.

(b) **Aplicații software influențate de fundamentele algoritmice**

Conceptele din teoria algoritmilor sunt fundamentale în dezvoltarea unor aplicații software critice. LLVM (Low Level Virtual Machine), un framework extensibil pentru compilatoare, folosește algoritmi de analiză de flux de date și optimizări bazate pe grafuri, inspirate din teoria algoritmilor [10]. Aceste fundamente teoretice sunt aplicate concret și în motoarele de căutare, sistemele de navigație sau rețelele sociale moderne.

(c) **Paradigme de design și metode algoritmice dominante**

Teoria algoritmilor a dat naștere unor paradigmă de rezolvare a problemelor care structurează gândirea computatională. Strategii precum divide et impera, programarea dinamică, backtracking sau greedy sunt esențiale în conceperea soluțiilor eficiente. Un exemplu notabil este algoritmul lui Strassen pentru înmulțirea matricilor, care a redus complexitatea clasica  $O(n^3)$  la  $O(n^{2.81})$ , devenind un punct de plecare pentru dezvoltarea de algoritmi numerici avansați, [11].

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Designul algoritmic contribuie nu doar la performanța software, ci și la siguranță, scalabilitate și robustețe. De exemplu, în domeniul criptografiei moderne, concepte din teoria complexității sunt esențiale pentru definirea protocolelor sigure. Algoritmii randomizați sau de aproximare, inițial studiați doar teoretic, sunt azi utilizati în procesarea volumelor masive de date, în special în aplicații web și big data [12].

#### 4. Extra:

##### (a) **Competențe fundamentale pentru studierea subdomeniului**

Pentru a înțelege în profunzime teoria algoritmilor, sunt necesare cunoștințe solide de matematică discretă (logica propozițională, mulțimi, relații, funcții), structuri de date fundamentale (liste, stive, cozi, arbori, grafuri), precum și o bună înțelegere a programării structurale. Este esențială familiaritatea cu notarea Big-O și cu tehniciile de analiză a complexității. De asemenea, este importantă capacitatea de a rationa algoritmic și de a construi demonstrații logice — abilități dezvoltate gradual prin exerciții, proiecte și studii de caz.

##### (b) **Surse academice esențiale pentru aprofundare**

O lucrare de referință în domeniu este „Introduction to Algorithms” de Cormen, Leiserson, Rivest și Stein (2009), [13], cunoscută și sub acronimul CLRS, utilizată în majoritatea universităților din lume pentru predarea teoriei algoritmilor. O altă lucrare fundamentală este „The Art of Computer Programming” de Donald Knuth (1997–2011), [14], considerată o enciclopedie a algoritmilor, care tratează atât aspecte teoretice, cât și istorice și pragmatice. Pentru o abordare orientată pe complexitate computațională, cartea lui Sipser (2012), [15], „Introduction to the Theory of Computation”, oferă o privire riguroasă asupra conceptelor precum decibilitatea, NP-completitudinea și reducerea între probleme.

În plus, resursele open-source precum MIT OpenCourseWare și Stanford CS Theory sunt valoroase pentru cursuri video, notițe și teme care urmează exact linia academică riguroasă a marilor universități.

#### 2.1.2 Relațiile cu celelalte subdomenii

Teoria algoritmilor reprezintă fundamentalul conceptual care sustine toate celelalte subdomenii ale informaticii teoretice.

Algoritmii fundamentali derivă direct din principiile teoriei, fiind analizați din perspectiva corectitudinii și eficienței. Structurile de date, atât clasice cât și

avansate, sunt selectate și utilizate pe baza criteriilor teoretice privind complexitatea operațiilor.

În domeniul algoritmilor pe grafuri, concepțele teoretice sunt aplicate pentru optimizarea rutelor, acoperirilor sau conectivității. Algoritmi geometrici adaptează paradigmile teoretice precum divide et impera pentru probleme spațiale.

Relația cu algoritmii probabilistici și randomizați reflectă extinderea teoriei către modele stocastice, în timp ce algoritmii paraleli și distribuiți aplică ideile teoretice în contexte de execuție simultană și distribuită.

Astfel, teoria algoritmilor oferă atât fundamentele, cât și instrumentele necesare pentru dezvoltarea tuturor acestor subdomenii.

### 2.1.3 Probleme importante și probleme deschise

Teoria algoritmilor a identificat și abordat o serie de probleme fundamentale care au modelat evoluția informaticii teoretice.

Printre problemele importante rezolvate se numără:

- proiectarea algoritmilor de sortare eficiente (QuickSort, MergeSort);
- căutarea rapidă în structuri ordonate (Binary Search);
- optimizarea rutelor și căilor minime în grafuri (algoritmii lui Dijkstra și Bellman-Ford);
- determinarea subsecvenței comune maxime (Longest Common Subsequence) prin programare dinamică.

În ceea ce privește problemele deschise, una dintre cele mai celebre este problema P vs. NP, formulată formal de Cook (1971),[16]. Aceasta întrebă dacă toate problemele al căror rezultat poate fi verificat rapid pot fi și rezolvate rapid. Deși problema este esențială pentru teoria complexității și securitatea informatică, o soluție general acceptată nu a fost încă descoperită.

Alte probleme deschise majore includ:

- determinarea limitelor exacte pentru algoritmi de aproximare în problemele NP-dure;
- îmbunătățirea algoritmilor pentru multiplicarea matricilor sub  $O(n^{2.37})$ ;
- găsirea unor algoritmi deterministici eficiente pentru probleme de satisfiabilitate booleană (SAT);
- construirea de scheme criptografice sigure pe baze pur teoretice fără presupuneri nevalidate.

Studiul acestor probleme deschise nu doar că stimulează progresul teoretic, dar are și un impact practic major asupra siguranței comunicațiilor, eficienței bazelor de date și dezvoltării inteligenței artificiale.

#### 2.1.4 Persoane importante

Dezvoltarea teoriei algoritmilor este strâns legată de contribuțiile unor personalități remarcabile.

Alonso Church și Alan Turing (1936),[3], au pus bazele teoretice ale calculabilității, definind formal limitele a ceea ce poate fi rezolvat algoritmic. Kurt Gödel (1931),[4], a introdus concepte esențiale privind indecidabilitatea, care au influențat ulterior dezvoltarea teoriei algoritmilor.

Stephen Cook (1971),[5], și Richard Karp (1972), [6], au definit conceptele de NP-completitudine și reducere între probleme, schimbând fundamental înțelegerea complexității computaționale.

Donald Knuth, prin seria monumentală „The Art of Computer Programming” (1997–2011), [14], a sistematizat și popularizat analiza algoritmică, având un impact major asupra modului în care algoritmii sunt proiectați și studiați.

Aceste contribuții au stabilit repere teoretice esențiale care structurează domeniul algoritmilor până astăzi.

#### 2.1.5 Forumuri importante

În domeniul teoriei algoritmilor, principalele canale de diseminare a cunoștințelor sunt conferințele și revistele de specialitate.

Conferințele STOC (Symposium on Theory of Computing) și FOCS (Foundations of Computer Science) sunt recunoscute la nivel internațional pentru publicarea celor mai importante rezultate în teoria algoritmilor și complexitatea computațională.

De asemenea, jurnalul *SIAM Journal on Computing* publică articole esențiale în domeniu, oferind o platformă riguroasă pentru cercetările teoretice avansate.

Alte forumuri notabile includ *Algorithmica* și *Journal of the ACM*, care acoperă o gamă largă de subiecte din teoria algoritmilor.

Prin aceste forumuri, noile descoperiri și metode teoretice sunt validate și propagate în comunitatea științifică internațională.

#### 2.1.6 Dimensiunea locală și dimensiunea globală

- **Local:** În cadrul Departamentului de Informatică al Facultății de Matematică și Informatică, Universitatea de Vest din Timișoara, teoria algoritmilor ocupă un loc central în formarea studentilor. Cursurile și proiectele de cercetare

stimulează analiza critică, proiectarea și optimizarea algoritmilor, asigurând astfel pregătirea pentru activități de cercetare și industrie de înalt nivel.

- **Global:** La nivel internațional, teoria algoritmilor are un impact semnificativ în cercetarea academică și în industrie. Conferințe precum STOC, FOCS și SODA redefineste constant direcțiile de evoluție ale domeniului. De asemenea, competiții precum ICPC (International Collegiate Programming Contest) promovează abilitățile algoritmice la scară mondială. Standardele educaționale stabilite de instituții precum MIT, Stanford și Berkeley includ teoria algoritmilor ca pilon fundamental pentru formarea specialistilor în informatică.

## 2.2 Algoritmi Fundamentali

Algoritmii fundamentali reprezintă o categorie centrală în informatică, formată din acei algoritmi generali care rezolvă probleme clasice, recurente și independente de un domeniu specific de aplicare. Aceștia includ metode de sortare, căutare, selecție, parcurgere și optimizare de bază.

Dezvoltarea acestor algoritmi a fost esențială pentru evoluția informaticii, deoarece ei stau la baza multor aplicații practice și sisteme software. De la sortarea eficientă a datelor în baze de date, până la găsirea celei mai scurte căi într-o rețea, algoritmii fundamentali sunt utilizati zilnic în aproape orice sistem informatic.

Majoritatea acestor algoritmi au fost dezvoltăți începând cu mijlocul secolului XX, în paralel cu apariția primelor limbaje de programare și sisteme de operare. Conceptele teoretice care stau la baza lor, cum ar fi complexitatea temporală și spațială, au fost formalizate în aceeași perioadă, odată cu maturizarea teoriei algoritmilor.

### 2.2.1 Activitățile principale

1. Teorie:

(a) **Fundamentele teoretice ale subdomeniului**

Algoritmii fundamentali se bazează pe concepțele de bază din teoria complexității, precum analiza în cel mai bun, cel mai rău și mediu caz, precum și pe paradigme de rezolvare precum divide et impera, greedy și programarea dinamică. Aceștia servesc drept exemplu ideal pentru înțelegerea principiilor de optimizare și corectitudine.

(b) **Modele formale și paradigmă de analiză**

Modelele de calcul utilizate sunt cele clasice, precum modelul RAM și mașina Turing. Paradigmele dominante sunt: divide et impera (ex.

MergeSort), greedy (ex. algoritmul lui Kruskal) și programarea dinamică (ex. problema rucsacului).

(c) **Metodologii de demonstrare și analiză algoritmică**

Se folosesc demonstrații prin inducție, reducere la contradicție și comparație între complexități. De asemenea, se studiază corectitudinea algoritmilor prin invariante și prin tehnici de recurență.

(d) **Întrebări centrale în cercetarea teoretică**

Ce algoritm este optim pentru o anumită clasă de probleme? Există o limită inferioară pentru sortare? Cum se poate îmbunătăți un algoritm printr-o alegere intelligentă a structurii de date?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Se simulează algoritmii fundamentali în diverse condiții (seturi ordonate, aleatoare, invers ordonate) pentru a observa comportamentul practic în raport cu cel teoretic.

(b) **Implementare algoritmică și evaluare empirică**

Algoritmii de sortare (ex. QuickSort, MergeSort, HeapSort) și de căutare (ex. Binary Search, Jump Search) sunt implementați și comparați în termeni de viteză și eficiență pe date reale.

(c) **Studii de caz și aplicații concrete**

De exemplu, QuickSort este adesea utilizat în bibliotecile standard de programare, datorită performanței excelente în medie. Algoritmul lui Kruskal este implementat în aplicații de rețelistică pentru minimizarea costului de conectivitate.

(d) **Metode consacrate și abordări avansate**

Sunt analizate optimizări precum introsort (combinarea QuickSort și HeapSort), sortări hibride și îmbunătățiri ale căutării binare cu interpolare. Totodată, se testează algoritmi care au comportament adaptiv în funcție de distribuția datelor.

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Algoritmii fundamentali sunt implementați în biblioteci standard precum STL (C++), Java Collections sau Python's `sorted()`. Aceste biblioteci oferă funcții optimizate, bazate pe decenii de cercetare teoretică și practică.

(b) **Aplicații software influențate de fundamentele algoritmice**

Sortarea și căutarea eficientă sunt indispensabile în sisteme de baze de date, motoare de căutare, algoritmi de compresie, rutare și procesare de date.

(c) **Paradigme de design și metode algoritmice dominante**

Se evidențiază utilizarea recursivității, a selecției pivotului în QuickSort, a strategiilor greedy în probleme de acoperire și a programării dinamice în optimizări combinate.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Designul eficient al algoritmilor fundamentali permite procesarea rapidă a volumelor mari de date în sisteme moderne, de la aplicații mobile până la servere distribuite.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Este necesară cunoașterea structurilor de date, a noțiunilor de recursivitate și a logicii de programare. De asemenea, sunt importante noțiunile de complexitate și de analiză matematică a funcțiilor de recurență.

(b) **Surse academice esențiale pentru aprofundare**

- *Introduction to Algorithms* de Cormen et al. (2009) [13]
- *Algorithm Design* de Kleinberg și Tardos (2005) [17]
- *Algorithms* de Sedgewick și Wayne (2011) [18] Aceste lucrări acoperă într-un mod riguros și aplicativ bazele algoritmilor fundamentali.

### 2.2.2 Relațiile cu celelalte subdomenii

Algoritmii fundamentali sunt direct influențați de teoria algoritmilor și servesc drept bază pentru dezvoltarea algoritmilor pe grafuri, geometrii și paraleli. Alegerea unei structuri de date optime pentru un anumit algoritm este esențială, iar tehnici precum backtracking sau divide et impera sunt reutilizate în subdomenii mai avansate.

### 2.2.3 Probleme importante și probleme deschise

Printre problemele clasice se numără identificarea celui mai eficient algoritm de sortare pentru diferite scenarii și găsirea celui mai rapid algoritm pentru selecție. Probleme deschise includ limitele precise ale sortării adaptative și dezvoltarea de algoritmi universali cu performanță constantă indiferent de distribuția datelor.

#### 2.2.4 Persoane importante

Donald Knuth a sistematizat algoritmii de sortare și căutare în lucrarea sa *The Art of Computer Programming*, [14]. Jon Bentley a contribuit la rafinarea acestor metode în contextul optimizării performanțelor în software real. Robert Sedgewick și Kevin Wayne au modernizat predarea algoritmilor fundamentali prin abordări vizuale și interactive.

#### 2.2.5 Forumuri importante

Revista *Journal of Algorithms* și conferințele SODA (Symposium on Discrete Algorithms) și ESA (European Symposium on Algorithms) publică frecvent rezultate relevante în domeniul algoritmilor fundamentali. Aceste forumuri permit diseminarea de metode noi și compararea performanțelor între abordări concurente.

#### 2.2.6 Dimensiunea globală

În mediul internațional, acești algoritmi sunt utilizati intensiv în industrie, dar și în cadrul competițiilor precum ICPC sau Google Code Jam, unde înțelegerea fină a eficienței este crucială.

### 2.3 Structuri de Date Clasice

Structurile de date clasice reprezintă o componentă fundamentală a informaticii, ocupând un rol central în eficiența algoritmilor. Ele definesc moduri standard de organizare, accesare și manipulare a datelor. Printre cele mai cunoscute se numără liste, stivele, cozile, arborii binari și tabelele de dispersie.

Acest subdomeniu s-a conturat începând cu anii 1960–1970, odată cu dezvoltarea limbajelor de programare precum ALGOL și C, în care operarea pe structuri de date a devenit o cerință practică. În prezent, structurile de date clasice sunt studiate în cadrul oricărui curriculum universitar de informatică, fiind esențiale în proiectarea de algoritmi eficienți.

#### 2.3.1 Activitățile principale

1. Teorie:

##### (a) Fundamentele teoretice ale subdomeniului

Structurile de date clasice se bazează pe concepte fundamentale precum alocarea memoriei, pointerii, relațiile de ordine și teoria grafurilor. Fiecare structură este asociată cu o suită de operații caracteristice și cu o analiză a costurilor în cel mai bun, mediu și cel mai rău caz.

(b) **Modele formale și paradigmă de analiză**

Se utilizează modelul RAM pentru analiză, iar operațiile sunt modelate matematic prin funcții de cost. Paradigmele includ abstractizarea datelor și separarea logicii de implementare, evidențiind astfel complexitatea algoritmică a fiecărei operații.

(c) **Metodologii de demonstrare și analiză algoritmică**

Se utilizează inducția pentru demonstrarea corectitudinii operațiilor (ex. menținerea ordinii într-un arbore binar), recurența pentru analiza complexității și invarianții de buclă pentru validarea funcționării.

(d) **Întrebări centrale în cercetarea teoretică**

Care este structura optimă pentru un anumit tip de problemă? Cum se poate reduce costul inserării și căutării? Există un compromis universal între timp și spațiu în contextul stocării datelor?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Structurile clasice sunt testate în scenarii tipice: operații repetate pe stive, cozile în procesare asincronă, arbori în aplicații recursive, tabele de dispersie pentru căutări rapide.

(b) **Implementare algoritmică și evaluare empirică**

Se compară implementările iterative și recursive pentru liste și arbori, precum și variațiile de implementare pentru hash tables: hashing liniar vs. chaining.

(c) **Studii de caz și aplicații concrete**

Cozile sunt utilizate în planificarea proceselor din sisteme de operare, arborii binari de căutare în bazele de date și stivele în parsarea expresiilor matematice.

(d) **Metode consacrate și abordări avansate**

Se explorează variații precum coada cu priorități, arborii echilibrați (AVL, Red-Black), tabelele de dispersie cu rehashing. Se analizează și comportamentul în cazuri de degenerare (ex. listă liniară în loc de arbore).

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Biblioteci standard precum STL (C++), Java Collections Framework sau Python `collections` includ implementări optimizate pentru stive, cozi, liste și hash maps, validate prin cercetare teoretică și testare industrială.

(b) **Aplicații software influențate de fundamentele algoritmice**

Structurile de date clasice sunt utilizate în aproape toate aplicațiile software: motoare de căutare, compilatoare, sisteme de operare și aplicații financiare.

(c) **Paradigme de design și metode algoritmice dominante**

Se aplică principiile modularității, reutilizării codului și encapsulării. Alegerea structurii potrivite se bazează pe analiza formală a operațiilor așteptate.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Structurile de date eficiente permit procesarea în timp real, optimizarea resurselor și scalabilitatea în sisteme mari, de la aplicații mobile până la infrastructuri cloud.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Este necesară înțelegerea conceptelor de pointeri, alocare dinamică, liste înlănuite, recursivitate, traversare și arbori. De asemenea, se recomandă familiaritatea cu analiza complexității operațiilor.

(b) **Surse academice esențiale pentru aprofundare**

- *Data Structures and Algorithm Analysis* de Mark Allen Weiss (2013) [19]
- *Algorithms* de Sedgewick și Wayne (2011)[18]
- *The Algorithm Design Manual* de Steven Skiena (2008) [20].

### 2.3.2 Relațiile cu celelalte subdomenii

Structurile de date clasice sunt strâns legate de algoritmi fundamentali și teoria algoritmilor, reprezentând suportul pentru stocarea și manipularea datelor în cadrul algoritmilor. De asemenea, ele sunt punctul de plecare pentru dezvoltarea structurilor de date avansate, fiind reutilizate în subdomenii precum algoritmi pe grafuri sau algoritmi paraleli.

### 2.3.3 Probleme importante și probleme deschise

Printre problemele fundamentale se numără eficiența operațiilor în funcție de tipul de date și implementarea robustă în memorie limitată. Problemele deschise includ găsirea unor structuri universale eficiente în medii distribuite și dezvoltarea de structuri hibride care se adaptează automat la tipurile de utilizare.

### 2.3.4 Persoane importante

Robert Sedgewick, [18], a contribuit la sistematizarea și eficientizarea structurilor clasice în context educațional. Mark Allen Weiss, [19], este cunoscut pentru claritatea cu care a prezentat conceptele de analiză a structurilor de date, iar Niklaus Wirth a introdus concepte fundamentale privind tipurile de date structurate în programare.

### 2.3.5 Forumuri importante

Revistele *ACM Transactions on Algorithms* și *Journal of Experimental Algorithmics* publică lucrări semnificative privind eficiența și comportamentul structurilor de date. Conferințele SODA și ESA includ adesea secțiuni dedicate structurilor de date și aplicațiilor acestora.

### 2.3.6 Dimensiunea locală și dimensiunea globală

- **Local:** Structurile de date clasice sunt studiate în cadrul cursurilor de Algoritmi și Structuri de Date la UVT, reprezentând baza formării algoritmice.
- **Global:** Aceste structuri sunt parte integrantă din orice program internațional de informatică și constituie standardul pentru optimizarea aplicațiilor în industrie.

## 2.4 Structuri de Date Avansate

Structurile de date avansate reprezintă o extensie a structurilor clasice, concepute pentru a răspunde unor cerințe mai stricte de eficiență și adaptabilitate. Acestea includ structuri precum arbori echilibrați (Red-Black, AVL, Splay), structuri de tip heap, tabele de dispersie avansate, arbori segment și arbori fenwick, precum și structuri persistente și dinamice.

Acest subdomeniu s-a dezvoltat în special în contextul aplicațiilor care necesită răspunsuri rapide la interogări pe seturi mari de date, inclusiv în procesarea fluxurilor de date, baze de date, sisteme de fișiere și aplicații grafice.

### 2.4.1 Activitățile principale

1. Teorie:

#### (a) Fundamentele teoretice ale subdomeniului

Structurile avansate sunt construite pe baza conceptelor de echilibrare, amortizare și compartimentare a datelor. Ele oferă operații eficiente în timp garantat și în condiții de încărcare neuniformă.

(b) **Modele formale și paradigme de analiză**

Se folosesc modele de calcul RAM și analiza amortizată. Paradigmele includ folosirea heapurilor în algoritmi de prioritizare, arborii de intervale pentru interogări rapide și tehnici de hashare dublă pentru acces direct.

(c) **Metodologii de demonstrare și analiză algoritmică**

Se demonstrează prin inducție proprietăți de echilibrare (în AVL), costuri amortizate (în Splay), și prin recurențe structura operațiilor. Se folosește și analiza potențialului pentru justificarea eficienței în cazuri medii.

(d) **Întrebări centrale în cercetarea teoretică**

Cum se menține echilibrul în structuri dinamice? Care este compromisul optim între actualizare și interogare? Se pot combina structuri multiple pentru a obține performanțe generalizate?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Structurile sunt testate pe seturi mari de date cu interogări și actualizări frecvente. Se măsoară eficiența în raport cu modelele teoretice.

(b) **Implementare algoritmică și evaluare empirică**

Arborii AVL și Red-Black sunt implementați și comparați pentru inserare și căutare. Se testează heapuri binare și Fibonacci pentru algoritmi de rutare.

(c) **Studii de caz și aplicații concrete**

Arborii segment sunt utilizati în procesarea întrebărilor de tip range sum/query în aplicații competitive. Heapurile Fibonacci sunt folosite în optimizarea Dijkstra pentru grafuri sparse.

(d) **Metode consacrate și abordări avansate**

Se explorează structuri persistente, skip lists și structuri hibride (ex. segment tree + lazy propagation), precum și alocări eficiente în memorie cache-aware.

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Biblioteci precum Boost, CGAL și LEDA oferă implementări robuste ale acestor structuri. De asemenea, limbaje precum Rust și Go oferă suport optimizat pentru structuri concurrente.

(b) **Aplicații software influențate de fundamentele algoritmice**

Structurile avansate sunt esențiale în baze de date relationale, compilatoare, motoare de căutare și aplicații de vizualizare 3D.

(c) **Paradigme de design și metode algoritmice dominante**

Se folosește design modular, orientat pe performanță, cu accent pe reducerea numărului de alocări și pe reutilizarea spațiului.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Aceste structuri permit dezvoltarea aplicațiilor reactive, a sistemelor distribuite și a bazelor de date scalabile, contribuind la performanța generală a sistemelor moderne.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Sunt necesare cunoștințe solide despre arbori, recursivitate, alocare dinamică și complexitate. De asemenea, sunt utile noțiuni de matematică discretă și de modelare algoritmică.

(b) **Surse academice esențiale pentru aprofundare**

- *Advanced Data Structures* de Peter Brass (2008) [21]
- *Purely Functional Data Structures* de Chris Okasaki (1999) [22]
- Capitolele avansate din *Introduction to Algorithms* (Cormen et al., 2009), [13].

#### 2.4.2 Relațiile cu celelalte subdomenii

Structurile de date avansate extind și optimizează structurile clasice, fiind esențiale în algoritmi pe grafuri, algoritmi geometrici și aplicații paralele. Ele influențează direct proiectarea algoritmilor care necesită eficiență la scară mare.

#### 2.4.3 Probleme importante și probleme deschise

Printre problemele esențiale se numără eficiența în accesarea și actualizarea datelor, echilibrarea automată și controlul amortizării. Problemele deschise includ combinarea structurilor persistente cu cele distribuite și obținerea unor structuri generalizate adaptabile.

#### 2.4.4 Persoane importante

Chris Okasaki, [22], este cunoscut pentru lucrările sale în structuri de date funcționale. Peter Brass, [21], a sistematizat structurile avansate din punct de vedere teoretic și aplicativ. Robert Tarjan a contribuit la dezvoltarea heapurilor Fibonacci și la fundamentele analizei amortizate.

#### **2.4.5 Forumuri importante**

Conferințele ESA și SPAA includ frecvent lucrări despre structuri de date avansate. Revistele *ACM Journal of Experimental Algorithmics* și *SIAM Journal on Computing* oferă spațiu pentru publicarea de metode inovatoare privind aceste structuri.

#### **2.4.6 Dimensiunea globală**

Structurile de date avansate sunt integrate în proiecte open-source, sisteme de baze de date, compilatoare moderne și infrastructuri distribuite.

### **2.5 Algoritmi pe Grafuri**

Algoritmii pe grafuri reprezintă o ramură esențială a informaticii, dedicată rezolvării problemelor definite pe structuri de tip graf — rețelele de noduri conectate prin muchii. Aceștia sunt utilizati pentru analiza rețelelor, optimizarea rutelor, detectarea ciclurilor, parcurgerea grafurilor și găsirea de componente conexe.

Subdomeniul s-a dezvoltat intens începând cu anii 1950, odată cu apariția primelor rețele de comunicații și a problemelor de transport, și rămâne extrem de activ în prezent datorită aplicațiilor vaste în internet, biologie computațională, rețele sociale și robotică.

#### **2.5.1 Activitățile principale**

1. Teorie:

##### **(a) Fundamentele teoretice ale subdomeniului**

Se bazează pe teoria grafurilor, o ramură a matematicii discrete, în care concepțele de conectivitate, ciclicitate și orientare sunt esențiale. Problemele sunt formulate pe grafuri neorientate, orientate, ponderate sau bipartite.

##### **(b) Modele formale și paradigmă de analiză**

Modelele RAM și modele pe flux sunt frecvent utilizate. Paradigmele clasice includ BFS, DFS, backtracking, algoritmi greedy (Dijkstra, Prim), programare dinamică (Bellman-Ford) și reducerea la flux maxim (Ford-Fulkerson).

##### **(c) Metodologii de demonstrare și analiză algoritmică**

Se folosesc inducția și construcția incrementală, recurențele și studiul grafurilor prin matrici și liste de adiacență. Se analizează complexitatea în funcție de numărul de noduri și muchii.

(d) **Întrebări centrale în cercetarea teoretică**

Cum se poate determina eficient drumul minim sau arborele de acoperire de cost minim? Cum se detectează existența unui circuit eulerian sau hamiltonian? Pot fi reprezentate rețele complexe în mod compact și eficient?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Se aplică algoritmii pe grafuri în grafuri generate random, rețele reale și grafuri dense. Se evaluatează timpul de execuție și scalabilitatea.

(b) **Implementare algoritmică și evaluare empirică**

Se implementează BFS, DFS, Dijkstra, Kruskal, Floyd-Warshall și se compară în funcție de dimensiunea grafurilor și de tipul ponderii.

(c) **Studii de caz și aplicații concrete**

Algoritmii sunt utilizati în hărți digitale (ex. Google Maps), rețele sociale (detectarea comunităților), rețele de trafic (rutare optimă) și biologie (modelarea proteinelor).

(d) **Metode consacrate și abordări avansate**

Se folosesc algoritmi apropiati de timp real, precum A\*, algoritmi pe grafuri dinamice și structuri speciale pentru reprezentarea grafurilor (ex. arbori de dominanță).

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Biblioteci precum NetworkX (Python), Boost Graph Library (C++), igraph și LEDA oferă suport robust pentru algoritmi pe grafuri, cu aplicații în cercetare și industrie.

(b) **Aplicații software influențate de fundamentele algoritmice**

Rețelele de comunicație, motoarele de căutare, sistemele de transport și procesarea limbajului natural integrează algoritmi pe grafuri pentru analiză structurală și optimizare.

(c) **Paradigme de design și metode algoritmice dominante**

Se folosesc modularitatea, orientarea pe eficiență și reutilizarea componentelor. Algoritmii sunt deseori implementați generic pentru a putea lucra pe diverse tipuri de grafuri.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Algoritmii pe grafuri permit dezvoltarea aplicațiilor inteligente, a sistemelor de recomandare, rețelelor sociale scalabile și a infrastructurilor optimizate.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Este necesară cunoașterea noțiunilor de matematică discretă, structuri de date, parcurgeri și recursivitate. Familiaritatea cu reprezentările grafurilor și cu analiza de complexitate este esențială.

(b) **Surse academice esențiale pentru aprofundare**

- *Graph Algorithms* de Shimon Even (2011) [23]
- *Network Flows* de Ahuja et al. (1993) [24]
- *Algorithm Design Manual* de Skiena (2008) [20].

### 2.5.2 Relațiile cu celelalte subdomenii

Algoritmii pe grafuri utilizează și extind conceptele din teoria algoritmilor și structurile de date. Sunt frecvent integrați cu algoritmi geometrici și paraleli, datorită aplicabilității în modelarea și analiza sistemelor complexe.

### 2.5.3 Probleme importante și probleme deschise

Probleme clasice includ găsirea drumului minim, acoperirea maximă și fluxul maxim. Probleme deschise includ algoritmi eficienți pe grafuri dinamice, aproximări rapide pentru grafuri foarte mari și combinarea cu metode de învățare automată.

### 2.5.4 Persoane importante

Edsger Dijkstra este considerat un pionier prin algoritmul său pentru drumuri minime. Robert Tarjan a contribuit la algoritmi de parcurgere și structură a componentelor. Jon Kleinberg este cunoscut pentru aplicarea grafurilor în analiza rețelelor și teoriile legate de „navigarea în rețele mici”.

### 2.5.5 Forumuri importante

Conferințele SODA, ICALP și FOCS includ frecvent secțiuni despre algoritmi pe grafuri. Revistele *Algorithmica*, *Journal of Graph Algorithms and Applications* și *ACM Transactions on Algorithms* publică articole relevante.

### 2.5.6 Dimensiunea locală și dimensiunea globală

- **Local:** La UVT, algoritmii pe grafuri sunt studiați în cadrul cursurilor de Algoritmi pe Grafuri, Programare Algoritmică și Tehnici Avansate în Grafuri.
- **Global:** La nivel internațional, acești algoritmi sunt integrați în infrastructuri mari (Facebook, Google, Amazon), competiții internaționale și proiecte de cercetare multidisciplinare.

## 2.6 Algoritmi Geometrici

Algoritmii geometrici sunt o ramură specializată a algoritmicii care se ocupă cu rezolvarea problemelor formulate în termeni geometrici, cum ar fi puncte, segmente, poligoane și arii. Problemele tipice includ intersecția segmentelor, triangularea poligoanelor, cuplajul geometric și determinarea învelitorii convexe.

Acest subdomeniu a evoluat din combinatorica geometrică și geometria compuțională, câștigând vizibilitate începând cu anii 1970 odată cu aplicarea sa în proiectare asistată de calculator (CAD), robotică, grafică pe calculator și GIS (Sisteme Informaționale Geografice).

### 2.6.1 Activitățile principale

1. Teorie:

(a) **Fundamentele teoretice ale subdomeniului**

Algoritmii geometrici se bazează pe geometria euclidiană, algebra liniară și analiza combinatorică. Se analizează complexitatea geometrică și proprietățile de incidență, orientare și convexitate.

(b) **Modele formale și paradigmă de analiză**

Modelele includ planul euclidian discret și continuu. Paradigmele specifice sunt divide et impera, sweeping line, incremental construction și random sampling.

(c) **Metodologii de demonstrare și analiză algoritmică**

Se folosesc argumente geometrice riguroase, inducții pe dimensiune, tehnici probabilistice și dualități geometrice. Complexitatea este exprimată în funcție de numărul de entități geometrice și interacțiuni.

(d) **Întrebări centrale în cercetarea teoretică**

Cum pot fi reprezentate eficient obiectele geometrice? Care este algoritmul optim pentru învelitoare convexă? Cum se pot gestiona intersecțiile multiple între obiecte geometrice în timp sub-liniar?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Se folosesc seturi sintetice de puncte, segmente sau poligoane. Se testează performanța algoritmilor în funcție de distribuția spațială și de degenerate cases.

(b) **Implementare algoritmică și evaluare empirică**

Sunt implementați algoritmi precum Graham Scan, QuickHull (învelitoare convexă), algoritmi de triangulare și intersecție de segmente. Se compară performanțele în funcție de dimensiunea și tipul datelor.

(c) **Studii de caz și aplicații concrete**

În grafică pe calculator, algoritmii de triangulare sunt folosiți pentru randare 3D. În robotică, algoritmii de evitare a obstacolelor folosesc diagrame Voronoi. În GIS, se folosesc pentru hărți și căutări spațiale.

(d) **Metode consacrate și abordări avansate**

Se explorează arbori R, KD-trees, diagrame Voronoi, structuri pentru range searching și metode probabilistice de eșantionare geometrică.

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

CGAL (Computational Geometry Algorithms Library) este una dintre cele mai complete biblioteci open-source. LEDA include și un modul geometric. Boost Geometry oferă suport pentru operații geometrice standardizate.

(b) **Aplicații software influențate de fundamentele algoritmice**

Aplicațiile includ GIS, CAD, robotică, simulare fizică, realitate augmentată și modelare 3D. Ele se bazează pe precizia și eficiența algoritmilor geometrici.

(c) **Paradigme de design și metode algoritmice dominante**

Designul modular, orientat pe obiecte, și geometria robustă sunt elemente centrale. Se folosesc structuri orientate pe topologie și vizibilitate.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Algoritmii geometrici contribuie la precizia navigației autonome, optimizarea sistemelor de cartografiere și modelarea realistă în jocuri și simulări.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Este necesară o bună înțelegere a geometriei plane, algebrei liniare, structurilor de date spațiale și metodelor de analiză algoritmică.

(b) **Surse academice esențiale pentru aprofundare**

- *Computational Geometry: Algorithms and Applications* de de Berg et al. (2008)[25]
- *Geometric Algorithms and Combinatorial Optimization* de Grötschel et al. (1993) [26]
- Documentația oficială a bibliotecii CGAL,[27].

### 2.6.2 Relațiile cu celelalte subdomenii

Algoritmii geometrici folosesc structuri de date avansate pentru stocare și căutare spațială și sunt integrați cu algoritmi pe grafuri în navigație, rețele și robotică. Au legături și cu algoritmii probabilistici pentru eșantionări și optimizări.

### 2.6.3 Probleme importante și probleme deschise

Probleme clasice includ învelitoarea convexă, intersecția segmentelor, triangularea și vizibilitatea. Probleme deschise includ găsirea unor algoritmi optimali în dimensiuni mai mari, gestiunea obiectelor dinamice și robustețea numerică.

### 2.6.4 Persoane importante

Mark de Berg,[25], este coautor al lucrării de referință în domeniu. Leonidas Guibas a contribuit la structuri dinamice și diagrame Voronoi. Herbert Edelsbrunner este cunoscut pentru lucrările sale privind triangularea și complexitatea geometrică.

### 2.6.5 Forumuri importante

Conferințele SoCG (Symposium on Computational Geometry) și EuroCG sunt centrale în acest domeniu. Revistele *Computational Geometry: Theory and Applications* și *Discrete & Computational Geometry* sunt cele mai relevante pentru diseminarea rezultatelor recente.

### 2.6.6 Dimensiunea locală și dimensiunea globală

- **Local:** La UVT, algoritmii geometrici sunt studiați în cadrul cursurilor optionale de grafică pe calculator, robotică și proiecte avansate din informatică.

- **Global:** La nivel internațional, acești algoritmi sunt integrați în sisteme de navigație, hărți digitale, simulatoare 3D, robotică și arhitectură computațională.

## 2.7 Algoritmi Probabilistici și Randomizați

Algoritmii probabilistici și randomizați utilizează aleatorul ca parte integrantă a procesului de calcul, fie pentru a reduce timpul de execuție, fie pentru a obține soluții aproximative cu o probabilitate mare de succes. Ei se disting prin comportamentul care variază în funcție de numere aleatoare generate în timpul execuției.

Acest subdomeniu a fost consacrat în anii 1970–1980, odată cu apariția unor algoritmi revoluționari precum testul de primalitate Miller-Rabin și algoritmii randomizați pentru determinarea medianei. Algoritmii probabilistici sunt esențiali în domenii precum criptografie, baze de date, algoritmi de streaming și învățare automată.

### 2.7.1 Activitățile principale

1. Teorie:

(a) **Fundamentele teoretice ale subdomeniului**

Bazele acestui subdomeniu sunt teoria probabilităților, complexitatea aleatoare și teoria informației. Algoritmii sunt clasificați în Las Vegas (corectitudine garantată, timp variabil) și Monte Carlo (timp fix, probabilitate de eroare controlabilă).

(b) **Modele formale și paradigmă de analiză**

Se folosesc modele de analiză în medie, analiza probabilistă, metoda martingalelor, tehnici de amplificare a probabilității și reducere prin esantionare aleatoare.

(c) **Metodologii de demonstrare și analiză algoritmică**

Demonstrațiile se bazează pe estimarea distribuțiilor de probabilitate, calculul valorilor așteptate, inegalități (Markov, Chebyshev, Chernoff) și argumente inductive probabilistice.

(d) **Întrebări centrale în cercetarea teoretică**

Care este raportul optim între precizie și eficiență? Cum poate fi îmbunătățit un algoritm determinist prin randomizare? Este randomizarea esențială sau poate fi eliminată prin determinism derandomizat?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Algoritmii sunt testați pe multiple instanțe aleatoare și distribuite, pentru a evalua stabilitatea performanței și a probabilității de succes.

(b) **Implementare algoritmică și evaluare empirică**

Sunt implementați algoritmi precum Miller-Rabin, QuickSelect randomizat, Hashing universal, Count-Min Sketch și algoritmi pentru eșantionare uniformă.

(c) **Studii de caz și aplicații concrete**

Algoritmii randomizați sunt utilizați în baze de date pentru optimizarea interogărilor, în rețelele peer-to-peer pentru selecție aleatoare și în algoritmi de streaming pentru rezumarea datelor.

(d) **Metode consacrate și abordări avansate**

Se analizează Bloom filters, hashing universal, amplificare prin repetiție, algoritmi F0-estimatori și algoritmi pentru streaming de frecvențe.

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Biblioteci precum NumPy, SciPy, Boost Random și biblioteci din R și Julia oferă suport pentru generare aleatoare, distribuții și tehnici de randomizare controlată.

(b) **Aplicații software influențate de fundamentele algoritmice**

Randomizarea este utilizată în algoritmi de hashing, criptografie probabilistă, algoritmi genetici, învățare automată și simulări stocastice.

(c) **Paradigme de design și metode algoritmice dominante**

Se folosesc tehnici de reducere probabilistă a erorii, scheme de codare și structuri orientate pe trade-off între spațiu și acuratețe.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Algoritmii probabilistici permit procesarea rapidă a datelor masive, securizarea comunicațiilor și dezvoltarea de soluții scalabile în sisteme distribuite.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Este esențială înțelegerea teoriei probabilităților, calculul valorii așteptate, noțiuni de variabilitate, distribuții discrete și continue, precum și analiza algoritmică clasică.

(b) **Surse academice esențiale pentru aprofundare**

- *Probability and Computing* de Mitzenmacher & Upfal (2005)[12]
- *Randomized Algorithms* de Motwani & Raghavan (1995)[28]
- Resurse online MIT OCW privind analiza probabilistă [29].

### 2.7.2 Relațiile cu celelalte subdomenii

Algoritmii probabilistici completează și îmbunătățesc performanța algoritmilor fundamentali și a celor pe grafuri. Sunt esențiali în structuri de date avansate (hashing, sketching) și au aplicații în algoritmii paraleli și distribuiți.

### 2.7.3 Probleme importante și probleme deschise

Probleme importante includ testarea primalității, hashing eficient și detectia probabilistă a coliziunilor. Probleme deschise implică derandomizarea generalizată, optimizarea ratelor de eroare și echilibrul între acuratețe și memorie.

### 2.7.4 Persoane importante

Michael Mitzenmacher și Eli Upfal sunt autori ai unei lucrări fundamentale în domeniul. Rajeev Motwani și Prabhakar Raghavan au formulat baza algoritmilor randomizați moderni. Andrei Z. Broder a contribuit la hashing și similitudine probabilistice.

### 2.7.5 Forumuri importante

Conferințele RANDOM și STOC conțin secțiuni specializate. Revistele *SIAM Journal on Computing* și *Journal of the ACM* publică articole relevante despre algoritmi randomizați.

### 2.7.6 Dimensiunea locală și dimensiunea globală

- **Local:** La UVT, noțiuni introductive sunt abordate în cursurile de probabilități, algoritmică și structuri de date avansate. Subiectul este inclus în proiecte de cercetare și lucrări de licență.
- **Global:** Algoritmii probabilistici sunt implementați în sisteme de bază ale Google, Facebook și companii de big data, fiind centrali în procesarea fluxurilor, AI și securitate cibernetică.

## 2.8 Algoritmi Paraleli și Distribuiți

Algoritmii paraleli și distribuiți sunt proiectați pentru a rula pe sisteme cu mai multe procesoare sau calculatoare care colaborează pentru a rezolva o problemă. Scopul este creșterea performanței, scalabilității și rezilienței în aplicații care implică volume mari de date sau necesită timp de răspuns redus.

Domeniul a apărut odată cu dezvoltarea calculatoarelor multiprocesor și a rețelelor de calculatoare, fiind intens cercetat în contextul arhitecturilor multicore, cloud computing și sistemelor distribuite moderne.

### 2.8.1 Activitățile principale

1. Teorie:

(a) **Fundamentele teoretice ale subdomeniului**

Se bazează pe modele de calcul paralele (PRAM, BSP) și distribuite (modelul asincron, modelul sincronic, modelul de mesaje). Se analizează concepțele de concurență, consistență, sincronizare și toleranță la erori.

(b) **Modele formale și paradigme de analiză**

Modelele includ EREW, CREW și CRCW pentru PRAM și modelul de mesaje pentru algoritmi distribuiți. Paradigmele includ divide et impera, paralelizat, map-reduce, și algoritmi locali pentru rețele.

(c) **Metodologii de demonstrare și analiză algoritmică**

Se folosesc inducția paralelă, analize de complexitate temporală paralelă (depth, work) și dovezi de corectitudine bazate pe invariante și coerență globală.

(d) **Întrebări centrale în cercetarea teoretică**

Cum se poate minimiza comunicarea între noduri? Care este viteza optimă de paralelizare? Cum pot fi proiectați algoritmi toleranți la erori în sisteme nesigure?

2. Experiment:

(a) **Modele computaționale și scenarii de analiză**

Algoritmii sunt testați pe sisteme multicore, clustere sau simulatoare distribuite. Se evaluatează scalabilitatea, latența, timpul de execuție și costul comunicării.

(b) **Implementare algoritmică și evaluare empirică**

Se implementează sortare paralelă, reducere paralelă, BFS distribuit și algoritmi de consens (ex. Paxos, Raft). Se evaluatează pe platforme precum MPI, OpenMP și Hadoop.

(c) **Studii de caz și aplicații concrete**

Aplicațiile includ baze de date distribuite, sisteme de fișiere distribuite (ex. GFS, HDFS), motoare de procesare big data (Apache Spark) și sisteme blockchain.

(d) **Metode consacrate și abordări avansate**

Se folosesc strategii precum partitionare inteligentă, comunicare asincronă, snapshot global, structuri fault-tolerant și coduri de erori.

3. Design:

(a) **Instrumente și biblioteci construite pe baze teoretice solide**

Frameworkuri precum MPI, OpenMP, CUDA, Hadoop și Spark oferă implementări pentru execuție paralelă și distribuită. Acestea sunt susținute de cercetare teoretică riguroasă privind comunicarea și sincronizarea.

(b) **Aplicații software influențate de fundamentele algoritmice**

Sistemele de streaming de date, motoarele de căutare, aplicațiile cloud și simulările numerice la scară mare se bazează pe algoritmi paraleli și distribuiți.

(c) **Paradigme de design și metode algoritmice dominante**

Se folosesc strategii funcționale, fluxuri de date, execuție lazy și împărțirea sarcinilor în funcție de granularitate și dependențe.

(d) **Contribuția designului algoritmic la progresul tehnologic actual**

Algoritmii paraleli și distribuiți permit procesarea rapidă în cloud, antrenarea rețelelor neuronale mari și operarea sistemelor critice în timp real.

4. Extra:

(a) **Competențe fundamentale pentru studierea subdomeniului**

Cunoștințe de programare concurrentă, sisteme de operare, arhitecturi de calcul, teoria algoritmilor și noțiuni de complexitate paralelă.

(b) **Surse academice esențiale pentru aprofundare**

- *Introduction to Parallel Computing* de Grama et al. (2003)[30]
- *Distributed Algorithms* de Nancy Lynch (1996)[31]
- *Designing Data-Intensive Applications* de Martin Kleppmann (2017)[32].

### 2.8.2 Relațiile cu celelalte subdomenii

Acest subdomeniu combină principii din algoritmii fundamentali, structuri de date, grafuri și algoritmi probabilistici. Integrarea se face prin paraleлизare și comunicare eficientă în medii distribuite.

### **2.8.3 Probleme importante și probleme deschise**

Probleme fundamentale includ sortarea paralelă, căutarea distribuită, acordul în sisteme asincrone și detecția erorilor. Probleme deschise includ scalabilitatea algoritmilor în rețele masive, consens în prezența defectelor bizantine și reducerea costului de comunicare.

### **2.8.4 Persoane importante**

Nancy Lynch este autoare a unei lucrări de referință în algoritmi distribuiți. Leslie Lamport este cunoscut pentru algoritmul Paxos și conceptele de timp logic. Anshul Gupta și Vipin Kumar au contribuit la algoritmi paraleli pentru științe computaționale.

### **2.8.5 Forumuri importante**

Conferințele SPAA, PODC și IPDPS sunt de referință în domeniu. Revistele *Journal of Parallel and Distributed Computing* și *IEEE Transactions on Parallel and Distributed Systems* sunt surse academice importante.

### **2.8.6 Dimensiunea locală și dimensiunea globală**

- **Local:** La UVT, acest subdomeniu este tratat în cursuri opționale avansate de calcul paralel și distribuții, fiind integrat în proiecte de cercetare și licență.
- **Global:** Algoritmii paraleli și distribuiți sunt parte esențială a infrastructurilor cloud, HPC, inteligenței artificiale distribuite și aplicațiilor critice în industrie.

### 3 Limbaje de programare

Domeniul Limbajelor de programare reprezintă un domeniu important din informatică, stând la baza dezvoltării aplicațiilor și a altor domenii. Subdomeniile acestuia sunt strâns legate între ele și contribuie la înțelegerea mai bună a programelor.

#### Subdomenii

1. Programare imperativă
2. Programare orientată pe obiect
3. Programare funcțională
4. Programare logică
5. Programare declarativă
6. Programare reactivă și event-based
7. Programare concurentă și paralelă
8. Metaprogramming and Reflective programming

#### 3.1 Programare imperativă

*Programarea imperativă* este o paradigmă de programare care tratează programul ca pe o secvență de comenzi care își schimbă starea. Aceasta are ca scop principal scăderea costurilor dezvoltării unui program și a menținării acestuia. Este una dintre paradigmile de programare fundamentale. [33]

##### 3.1.1 Activitățile principale

*Programarea imperativă* se bazează pe executarea pas cu pas a comenziilor, folosind structuri de control precum structuri alternative sau repetitive.

1. Teorie:
  - (a) Definirea cadrului conceptual    Programarea imperativă se bazează pe următoarele concepte: stare, instrucțiuni, flux de control, variabile și atribuire.
  - (b) Descrierea cadrului/cadrelor
    - Conceptul de *stare* reprezintă valorile variabilelor din memorie într-un anumit moment.

- Conceptul de *instructiuni* se referă la comenzi care modifică starea unui program.
- Conceptul de *flux de control* este ordinea în care se execută instrucțiunile unui program.
- Conceptul de *variabile* reprezintă locații din memorie care au un nume, în care se stochează date.
- Conceptul de *atribuire* se referă la operația de a modifica o variabilă cu o anumită valoare.

(c) Metode de raționament utilizate

- Manipulare directă a hardware-ului unui computer [34]

(d) Exemplificarea întrebărilor fundamentale

- Cum se verifică corectitudinea unui program imperativ?
- Care sunt metodele eficiente de optimizare a performanței în programele imperative?
- Cum se modifică starea unui program după execuția unei secvențe de instrucțiuni?
- Cum poate fi controlat fluxul de execuție al instrucțiunilor pentru realizarea unei sarcini?

2. Experiment:

(a) Explorarea modelelor:

- *Assembly, 1950-1960* este un limbaj de nivel jos care leagă strâns componentele hardware de instrucțiuni. [35]
- *Fortran, 1957* este un limbaj folosit pentru calcule numerice și științifice.

(b) Implementări și validarea experimentală:

- S-au implementat diferiți algoritmi de căutare, sortare(QuickSort) într-un mod imperativ și de analiză a compilatoarelor.

(c) Exemple practice:

- *Dezvoltarea sistemului de operare UNIX*
- *Calcule numerice în Fortran*

3. Design:

(a) Produse rezultate din cercetare:

- *Sisteme de operare UNIX, MINIX, Plan 9*

- Limbaje de programare Go, Rust
  - Sisteme de compilare
- (b) Instrumente rezultate din cercetare:
- Analizatoare statice Clang Static, Infer
  - IDE-uri Visual Studio, Eclipse
  - Sisteme de benchmarking grproof, perf
- (c) Tehnici rezultate din cercetare:
- Programare structurată
  - Optimizarea codului utilizând memoria eficient
- (d) Impactul designului asupra dezvoltării subdomeniului: Modul de structurare a codului în maniera imperativă este utilizat foarte des în multe limbaje de programare. Această paradigmă stă la baza altor paradigmă de programare(Orientată pe obiect, Paralelă și concurentă).
4. Extra:
- (a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?
    - Programare: C, Pascal, Fortran
  - (b) Literatura de specialitate
    - Concepts of Programming Languages [36]
    - The CProgramming Languages [37]
    - The Handbook of Programming Languages [38]

### 3.1.2 Relațiile cu celelalte subdomenii

Se leagă de *Arhitectura calculatoarelor*, deoarece este strâns legată de configurația von Neumann și de modul în care funcționează procesorul și memoria unui calculator. De asemenea, se leagă și de *Algoritmi și structuri de date*, deoarece influențează modul în care sunt implementați algoritmii și sunt folosite structurile de date.

### 3.1.3 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *paralizarea programării imperative* care poate deveni prea complexă.

### **3.1.4 Persoane Importante**

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu, John von Neumann, Edsger W. Dijkstra, Niklaus Wirth, Dennis Ritchie, Brian Kernighan

### **3.1.5 Forum-uri Importante**

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow

### **3.1.6 Dimensiunea locală și dimensiunea globală**

- **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, sunt abordate aceste subiecte în cadrul cursurilor de Limbaje de Programare, Tehnici Avansate de Programare, și Sisteme de Compilare.
- **Global:**
  - Colaborări internaționale: Proiecte europene(Horizon Europe), Consorții globale(Eclipse Foundation, ISO/IEC JTC 1/SC).
  - Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features ISO C, ISO Fortran
  - Influență globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie. e o metodă fundamentală de structurare a programelor

## **3.2 Programare orientată pe obiect**

*Programarea orientată pe obiect* este o paradigmă de programare care utilizează obiecte. Obiectele conțin date (câmpuri, atribute, proprietăți) și acțiuni specifice (proceduri, metode care sunt implementate în cod). Programele sunt compuse din obiecte care interacționează.

### **3.2.1 Activitățile principale**

*Programarea orientată pe obiect* se bazează pe encapsulare, moștenire, polimorfism și abstractizare.

1. Teorie:

(a) Definirea cadrului conceptual

Programarea orientată pe obiecte se bazează pe următoarele concepte: obiect, clasă, metodă, encapsulare, moștenire, polimorfism și abstractizare.

(b) Descrierea cadrului/cadrelor

- Conceptul de *obiect* se referă la o instanță a unei clase, o "variabilă" care conține attribute și metode
- Conceptul de *clasă* definește o structură care are tipul obiectelor și include variabile și funcții asociate.
- Conceptul de *metodă* se referă la o funcție asociată unei clase, specifică acesteia.

(c) Metode de raționament utilizate

- Modularizare

(d) Exemplificarea întrebărilor fundamentale

- Cum se proiectează o ierarhie de clase eficientă?
- Cum se evită ambiguizarea moștenirilor?

2. Experiment:

(a) Explorarea modelelor:

- *Smalltalk, 1989, Goldberg, Robson* a fost limbajul care s-a dezvoltat alături de această paradigmă, însă nu a devenit aşa de cunoscut ca Java, C++ sau C sharp.

(b) Exemple practice:

- Java
- Python
- C++

3. Design:

(a) Produse rezultate din cercetare:

- *Framework-uri* Spring(Java), Qt(C++)

(b) Instrumente rezultate din cercetare:

- *IDE-uri* IntelliJ IDEA, Eclipse, Visual Studio

(c) Tehnici rezultate din cercetare:

- *Design patterns* Singleton, Factory, Observer

- (d) Impactul designului asupra dezvoltării subdomeniului: A folosit concepte noi care au ajutat la modularizarea software-ului și la creșterea scalabilității.
4. Extra:
- (a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?
    - Programare: C++, Python, Java, C sharp
  - (b) Literatura de specialitate
    - The Object-Oriented Thought Process

### 3.2.2 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *supraîncărcarea arhitecturii calculatorului* care apare atunci când programul devine prea complex.

Câteva probleme deschise sunt legate de *verificarea formală a codului* din limbajele dinamice care obligă tipurile eronate să fie descoperite de abia la run-time.

### 3.2.3 Persoane Importante

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu, Alan Kay, Bjarne Stroustrup, James Gosling

### 3.2.4 Forum-uri Importante

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow IEEE Software și ACM Digital Library

### 3.2.5 Dimensiunea locală și dimensiunea globală

- **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, sunt abordate aceste subiecte în cadrul cursurilor de Limbaje de Programare, Tehnici Avansate de Programare, și Sisteme de Compilare. II, în cadrul cursurilor de OOP, Tehnici avansate de programare.
- **Global:**
  - Colaborări internaționale: Proiecte europene(Horizon Europe), Consorții globale(ISO/IEC JTC1).
  - Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming featuresISO C++, UML

- Influența globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie. Această paradigmă este folosită în foarte multe aplicații enterprise, mobile, web și în dezvoltarea de inteligență artificială.

### 3.3 Programare funcțională

*Programarea funcțională* este o paradigmă de programare care se bazează pe matematică și se folosește de funcții, expresii condiționate, recursivitate.

#### 3.3.1 Activitățile principale

1. Teorie:

(a) Definirea cadrului conceptual

Programarea funcțională se bazează pe concepte precum: pure functional, imutabilitate, recursivitate și expresii.

(b) Descrierea cadrului/cadrelor

- *Pure functional* se referă la returnarea aceluiași rezultat, pentru aceleasi argumente.
- Conceptul de *imutabilitate* se referă la valori constante, care nu pot fi modificate
- *Recursivitatea* e o tehnică de definire a funcțiilor în care funcția se folosește de propria implementare și se auto-apelează până când se îndeplinește condiția de oprire.
- Conceptul de *expresii* sunt combinații de operatori și variabile care produc o valoare.

(c) Metode de raționament utilizate

- Rationamente matematice

(d) Exemplificarea întrebărilor fundamentale

- Cum se garantează corectitudinea unui program funcțional?
- Cum se exprimă structurile repetitive fără bucle?

2. Experiment:

(a) Explorarea modelelor:

- *LISP, 1958* este unul dintre primele limbaje funcționale.
- *Haskell, 1990* este un limbaj pure functional care are un sistem de tipuri statici.

- (b) Implementări și validarea experimentală:
    - Implementarea *MergeSort*
    - *Calcul simbolic* în Haskell
  - (c) Exemple practice:
    - *Compilatoare scrise în Haskell* GHC
    - *Sistem de procesare paralelă* MapReduce
3. Design:
- (a) Produse rezultate din cercetare:
    - *Limbaje de programare* Haskell, LISP, Ocaml, F sharp
    - *Sistem de analiză formală* Coq, Agda
  - (b) Instrumente rezultate din cercetare:
    - *Compilatoare* GHC
    - *Instrumente de verificare formală* QuickCheck
  - (c) Tehnici rezultate din cercetare:
    - *Programare pure functional*
    - *Lazy evaluation and memoization*
  - (d) Impactul designului asupra dezvoltării subdomeniului: S-au descoperit tehnici moderne de paralelism și a influențat Scala, Rust și JavaScript.

#### 4. Extra:

- (a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?
  - Programare: Haskell, Ocaml, F sharp, LISP
  - Logică matematică
- (b) Literatura de specialitate
  - *Introduction to Functional Programming* [39]

#### 3.3.2 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *integrarea cu sistemele imperative* și *abstractizarea side-effects-urilor*.

Câteva probleme deschise sunt legate de *Programare funcțională reactivă la scară largă*.

### 3.3.3 Persoane Importante

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu  
John Backus, John McCarthy, Simon Peyton

### 3.3.4 Forum-uri Importante

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow

### 3.3.5 Dimensiunea locală și dimensiunea globală

- **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, sunt abordate aceste subiecte în cadrul cursurilor de Limbaje de Programare, Tehnici Avansate de Programare, și Sisteme de Compilare. III, semestrul II, în cadrul cursului de Programare logică și funcțională.

- **Global:**

- Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features Haskell 2010, ML
- Influență globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie, educație, cercetare în limbiage

## 3.4 Programare logică

*Programarea logică* este o paradigmă de programare care utilizează logica propozițională. Un program este considerat o colecție de fapte și reguli în care calculul constă în deducerea de concluzii logice.

### 3.4.1 Activitățile principale

*Programarea logică* se utilizează în inteligența artificială, în procesarea limbajului natural.

1. Teorie:

(a) Definirea cadrului conceptual

Programarea logică se bazează pe următoarele concepte: fapt, regulă, interogare, rezoluție și unificare.

(b) Descrierea cadrului/cadrelor

- Conceptul de *fapt* se referă la o propoziție atomică care e considerată mereu adevărată.
- Conceptul de *regulă* se referă la clauze, la implicații între fapte.
- Conceptul de *interrogare* este o întrebare adresată programului ca să verifice dacă o propoziție e adevărată sau nu.
- *Rezoluția* este mecanismul utilizat pentru a deduce noi propoziții pornind de la clauze.
- *Unificarea* este procesul de determinare a substituțiilor care fac doi termeni logici identici.

(c) Metode de raționament utilizate

- Deducre logică

(d) Exemplificarea întrebărilor fundamentale

- Cum se determină validitatea unei propozitii în sistemul logic?
- Ce fapte și reguli sunt necesare pentru a răspunde unei interogări?

2. Experiment:

(a) Explorarea modelelor:

- *Prolog, 1972*: limbaj reprezentativ pentru programarea logică, bazat pe rezoluție și unificare.

(b) Implementări și validarea experimentală:

- Sisteme avansate dezvoltate în Prolog folosite pentru diagnosticarea medicală.
- Motoare logice pentru baze de date.

(c) Exemple practice:

- Procesarea limbajului natural

3. Design:

(a) Produse rezultate din cercetare:

- Sisteme avansate pentru decizii logice (MYCIN)

(b) Instrumente rezultate din cercetare:

- *Sisteme de inferență* bazate pe Prolog.
- *Motoare semantice* pentru RDF/OWL.

(c) Tehnici rezultate din cercetare:

- *Unificare*
- *Backtracking*
- *Rezoluție*

(d) Impactul designului asupra dezvoltării subdomeniului: Se remarcă prin structura organizată sub formă de fapte și reguli care permit dezvoltarea de sisteme bazate pe cunoștințe și prin contribuțiile semnificative în inteligența artificială.

4. Extra:

- (a) Ce cunoștințe sunt necesare pentru a intra în domeniu?
  - Logică matematică, basic inteligență artificială, Prolog
- (b) Literatura de specialitate
  - *Foundations of Logic Programming* - J.W. Lloyd

#### **3.4.2 Relațiile cu celelalte subdomenii**

Se leagă de *Inteligența artificială*, deoarece utilizează logica simbolică.

#### **3.4.3 Probleme Importante și Probleme Deschise**

Câteva probleme deschise sunt legate de *integrarea cu paradigmă moderne*.

#### **3.4.4 Persoane Importante**

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu, Robert Kowalski, Alain Colmerauer

#### **3.4.5 Forum-uri Importante**

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow.

#### **3.4.6 Dimensiunea locală și dimensiunea globală**

- **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, în cadrul cursurilor de inteligență artificială, Programare I, Programare II și Programare logică și funcțională din semestrul II, anul III.
- **Global:**

- Colaborări internaționale: Proiecte europene (Horizon Europe), Consorții globale (ACM SIGPLAN, ISO/IEC JTC1).
- Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features ISO Prolog
- Influență globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie, inteligență artificială, web scrapping

### 3.5 Programare declarativă

*Programarea declarativă* este o paradigmă de programare în care logica programelor este exprimată fără a se specifica efectiv fluxul de control al acestora. Aceasta se concentrează pe descrierea a ce trebuie să se întâmple, și nu cum trebuie să se întâmple.

#### 3.5.1 Activitățile principale

*Programarea declarativă* tratează programele ca fiind descrieri de proprietăți sau relații, iar execuția se realizează printr-un mecanism de evaluare automată.

1. Teorie:

(a) Definirea cadrului conceptual

Programarea declarativă se bazează pe următoarele concepte: expresii, reguli, pure functional, non-determinism, no side effects.

(b) Descrierea cadrului/cadrelor

- Conceptul de *expresii* sunt combinații de operatori și variabile care produc o valoare.
- Conceptul de *regulă* se referă la clauze, la implicații între fapte.
- *Pure functional* se referă la returnarea aceluiași rezultat, pentru aceleași argumente.
- Conceptul de *non-determinism* se referă la existența mai multor soluții valide la aceeași problemă.

(c) Metode de raționament utilizate

- Deduție logică

(d) Exemplificarea întrebărilor fundamentale

- Cum poate fi descrisă logica unui algoritm fără indicarea exactă a pașilor?

- Cum pot fi exprimate restricții astfel încât numărul de soluții să se micșoreze?

2. Experiment:

(a) Explorarea modelelor:

- *SQL, 1974* este un limbaj declarativ care interoghează bazele de date relaționale. item *Prolog, 1972*: limbaj reprezentativ pentru programarea logică, bazat pe rezoluție și unificare.
- *Haskell, 1990* este un limbaj pure functional care are un sistem de tipuri statici.

(b) Implementări și validarea experimentală:

- Sisteme de baze de date (MySQL, PostgreSQL).
- Sisteme de inferență bazate pe reguli (ex: CLIPS).
- Interpretoare și compilatoare Haskell, Mercury.

3. Design:

(a) Produse rezultate din cercetare:

- Limbaje declarative: Haskell, Prolog, Datalog, XQuery.
- Expert systems și knowledge databases

(b) Instrumente rezultate din cercetare:

- Compilatoare și interpretoare declarative.

(c) Tehnici rezultate din cercetare:

- Lazy Evaluation
- Symbolic Evaluation

(d) Impactul designului asupra dezvoltării subdomeniului: A influențat dezvoltarea sistemelor de baze de date, a limbajelor declarative și a oferit o nouă perspectivă de rezolvare a problemelor concentrată pe rezultat, nu pe rezolvare în sine.

4. Extra:

(a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?

- Programare: SQL, Haskell, Prolog, XQuery

### 3.5.2 Relațiile cu celelalte subdomenii

Se leagă de *Inteligenta Artificială*, reprezentând un cadru general de dezvoltarea al inteligenței artificiale simbolice.

### **3.5.3 Probleme Importante și Probleme Deschise**

Câteva probleme importante sunt *performanța scăzută* față de paradigmile imperative.

Câteva probleme deschise sunt legate de *integrarea cu sisteme imperitive*.

### **3.5.4 Persoane Importante**

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu  
Paul Hudak, John Backus, John W. Lloyd

### **3.5.5 Forum-uri Importante**

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow

### **3.5.6 Dimensiunea locală și dimensiunea globală**

- **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, sunt abordate aceste subiecte în cadrul cursurilor de Limbaje de Programare, Tehnici Avansate de Programare, și Sisteme de Compilare. II, semestrul I, în cadrul cursului de Baze de date. O altă opțiune ar fi masteratul de Big Data.
- **Global:**
  - Conferințe: PPDP(Principles and Practice of Declarative Programming), ICFP(Functional Programming), DBPL(Declarative Databases)
  - Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features SQL ISO, ISO Prolog
  - Influență globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie. Paradigma declarativă e folosită în domenii precum big data, data flow și cloud.

## **3.6 Programare reactivă și event-based**

*Programarea reactivă și event-based* este o paradigmă de programare orientată spre modelarea sistemelor dinamice care are reacții la fluxuri de evenimente sau date. Aceasta se bazează pe fluxuri de date și propagarea automată a schimbărilor.

### **3.6.1 Activităile principale**

*Programarea reactivă și event-based* permite descrierea de comportament asincron și dependent de timp într-o manieră declarativă care permite dezvoltarea de aplicații interactive.

#### **1. Teorie:**

##### **(a) Definirea cadrului conceptual**

Programarea reactivă și event-based se bazează pe concepte precum fluxuri de evenimente, dataflow.

##### **(b) Descrierea cadrului/cadrelor**

- Conceptul de *evenimente* se referă la notificări cu privire la schimbările stărilor unui sistem.
- Conceptul de *flux evenimente* se referă la o secvență de evenimente care sunt distribuite în timp.
- Conceptul de *dataflow* se referă la variabile care nu au fost asignate, dar declarate.

##### **(c) Metode de rationament utilizate**

- Rețele de dependență

##### **(d) Exemplificarea întrebărilor fundamentale**

- Cum se poate modela un sistem care reacționează în timp real?
- Cum se pot compune fluxuri de evenimente într-un mod declarativ?

#### **2. Experiment:**

##### **(a) Explorarea modelelor:**

- Observer Design Pattern

##### **(b) Implementări și validarea experimentală:**

- *ReactiveX* este folosit pentru programarea reactivă în mai multe limbiage.
- *Vue.js*, *ReactJs*, *Angular* sunt framework-uri pentru JavaScript care se bazează pe această paradigmă.

##### **(c) Exemple practice:**

- Aplicații UI care răspund la interacțiunea cu userul în timp real (Aplicații mobile, aplicații web)

#### **3. Design:**

- (a) Produse rezultate din cercetare:
- Limbaje și modele reactive Elm, Reflex, Bacon.js
  - Framework-uri reactive RxJava, ReactJs
- (b) Instrumente rezultate din cercetare:
- Sisteme de gestionare a fluxurilor Kafka Streams, Reactive Streams API
  - Toolkits care simulează comportamentul reactiv
- (c) Tehnici rezultate din cercetare:
- Fluxuri de eveniente
  - Lazy Evaluation
  - Flow Control, Back-pressure
- (d) Impactul designului asupra dezvoltării subdomeniului: A ajutat la dezvoltarea de aplicații UI reactive și a redus complexitatea codului asincron.

4. Extra:

- (a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?
- Programare: JavaScript, Java, Python, Kotlin
  - Biblioteci/Frameworks: RxJS, React, Vue, Angular, Kafka
- (b) Literatura de specialitate
- A survey on Reactive Programming [40]

### 3.6.2 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *gestionarea erorilor în sistemele reactive complexe*.

Câteva probleme deschise sunt legate de *optimizarea performanței în sistemele reactive complexe*.

### 3.6.3 Persoane Importante

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu, Erik Meijer, Jonas Boner

### 3.6.4 Forum-uri Importante

Forumuri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow.

### 3.6.5 Dimensiunea locală și dimensiunea globală

- **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, există cursuri de programare a device-urilor mobile și de programare web.
- **Global:**
  - Conferințe: Reactive Summit, Reactive Conf
  - Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features Reactive Streams API, Web Streams API
  - Influență globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie. Este folosit în producerea de aplicații în timp real, IoT și microservicii.

## 3.7 Programare concurrentă și paralelă

*Programarea concurrentă* este o ramură a informaticii care folosește limbaje de programare care permit indicarea explicită a diferitor bucăți de cod să fie executate de diferite procesoare ale calculatorului.

### 3.7.1 Activitățile principale

*Programarea concurrentă* presupune execuția controlată a mai multor threads-uri sau procese care se folosesc de aceleași resurse. Pentru a accelera execuția, *programarea paralelă* împarte sarcinile în mod eficient procesoarelor calculatorului.

1. Teorie:

(a) Definirea cadrului conceptual

Programarea concurentă și paralelă se bazează pe următoarele concepte: thread, proces și race condition.

(b) Descrierea cadrului/cadrelor

- Conceptul de *thread* se referă la o unitate de execuție independentă care poate rula concurent cu alte thread-uri.
- Conceptul de *proces* se referă la o instanță de program independentă care are un spațiu propriu de memorie.
- Conceptul de *race-condition* se referă la starea de concurență non-deterministă între thread-uri.

- (c) Metode de raționament utilizate
  - Teoria sistemelor distribuite
- (d) Exemplificarea întrebărilor fundamentale
  - Cum se proiectează un algoritm paralel eficient și scalabil?
  - Cum se măsoară performanța unui program paralel?

2. Experiment:

- (a) Explorarea modelelor:
  - Thread-based concurrency (Java, POSIX Threads)
  - Message-passing concurrency (Go, Erlang)
  - GPU programming (CUDA, OpenCL)
  - Data parallelism (OpenMP, MPI, Intel TBB)
- (b) Implementări și validarea experimentală:
  - Algoritmi de sortare(QuickSort) și matrix multiplication algoritm
  - Benchmark-uri de performanță pentru mai multe nuclee.
  - Cluser computing
- (c) Exemple practice:
  - *Procesarea audio/videoe în timp real*
  - *Simulații de natură fizică* Modelare meteo și fizică moleculară
  - *Procesarea de date* Big Data

3. Design:

- (a) Produse rezultate din cercetare:
  - Sisteme distribuite scalabile
  - Framework-uri pentru procesare paralelă
  - Sisteme de time-sharing și multitasking
- (b) Instrumente rezultate din cercetare:
  - Instrumente de profilare paralelă (Intel VTune, Valgrind)
  - Biblioteci: OpenMP, MPI, CUDA
  - Sisteme de testare a concurenței (ThreadSanitizer)
- (c) Tehnici rezultate din cercetare:
  - Partitionarea sarcinilor și load balancing
  - Sincronizarea firelor și evitarea deadlock-ului

- Speculația execuției paralele (speculative parallelism)
- (d) Impactul designului asupra dezvoltării subdomeniului:  
A ajutat la folosirea utilă a componentelor hardware ale unui calculator cu arhitectură multicore și a dus la performanțe crescute în aplicații și în procesarea de Big Data
4. Extra:
- Ce cunoștiințe sunt necesare pentru a intra în domeniu?
    - Programare: C++, C, Go, Java
    - Framework-uri: OpenMP, CUDA, MPI
  - Literatura de specialitate
    - *Parallel Programming in C with MPI and OpenMP*

### 3.7.2 Relațiile cu celelalte subdomenii

Se leagă de *sisteme de operare*, deoarece gestionează execuția de task-uri, dar și de *arhitectura calculatoarelor*, folosindu-se de componentele hardware.

### 3.7.3 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *sincronizarea thread-urilor* care apare atunci când programul devine prea complex și *scalabilitatea algoritmilor paraleli*.

Câteva probleme deschise sunt legate de *construcția unor modele deterministe* privind execuția paralelă.

### 3.7.4 Persoane Importante

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu, Tony Hoare, Leslie Lamport, Nir Shavit

### 3.7.5 Forum-uri Importante

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow

### 3.7.6 Dimensiunea locală și dimensiunea globală

- Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, există master-ul de Big Data, dar și cursuri precum
- Global:**

- Colaborări internaționale: ACM PPoPP
- Proiecte europene(Euro-Par Conference)
- Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features OpenMP, MPI, POSIX Threads
- Influența globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie. diverse aplicații folosind computerul, AI, Big Data, Cloud computing

## 3.8 Metaprogramming and Reflective programming

*Metaprogramming* este o paradigmă de programare care scrie programe care manipulează programele ca fiind date sau procese. *Reflective programming* este capacitatea unui program de a-și modifica structura și comportamentul din timpul execuției.

### 3.8.1 Activitățile principale

1. Teorie:

(a) Definirea cadrului conceptual

Metaprogramming se bazează pe următoarele concepte: genrare de cod, transformare, reflectie, generalizare.

(b) Descrierea cadrului/cadrelor

- Conceptul de *generare de cod* se referă automatizarea de task-uri și la crearea de framework-uri sau librării.
- Conceptul de *transformare* se referă la derivarea programelor de la specificații formale și de la versiuni noi la versiuni vechi.
- Conceptul de *reflectie* se referă la abilitatea unui program de a manipula cu ajutorul sărăii unui program și a semanticii sale ca fiind date în timp ce se execută.
- Conceptul de *generalizare* oferă o modalitate de abstractiza datele.

(c) Metode de raționament utilizate

- Abstractizare(template-uri, macros)
- Transformări sintactice și evaluări simbolice

(d) Exemplificarea întrebărilor fundamentale

- Cum se generează automat cod repetitiv?
- Cât de limitată e reflectia?

2. Experiment:

- (a) Explorarea modelelor:
  - *C++ template metaprogramming* este o tehnică bazată pe folosirea proprietăților template-urilor din C++. [41]
  - *Macros din Lisp* sunt similare cu funcțiile din C++ sau Java care iau argumente și returnează o formă Lisp care trebuie evaluată.
- (b) Implementări și validarea experimentală:
  - *Automatizarea generării de test cases*
  - *Serializarea și deserializarea dinamică*
- (c) Exemple practice:
  - *Compilatoare care sunt optimizate automat*
  - *MOP(Protocole pentru metaobiecte*
  - *Decoratorii din Python*

3. Design:

- (a) Produse rezultate din cercetare:
  - *Framework-uri de testare JUnit, pytest*
  - *Framework-uri UI dinamice React, Angular*
- (b) Instrumente rezultate din cercetare:
  - *Generatoare de cod ANTLR, Yeoman*
  - *Compilatoare dinamice LLVM*
- (c) Tehnici rezultate din cercetare:
  - *Protocole de metaobiecte*
  - *Programare orientată pe aspect*
- (d) Impactul designului asupra dezvoltării subdomeniului: A ajutat la automatizarea codului și a testării prin abstractizare și reutilizare de cod.

4. Extra:

- (a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?
  - Programare: Lisp, Python, C++, Java, Rust
- (b) Literatura de specialitate
  - Practical C++ Metaprogramming

### **3.8.2 Relațiile cu celelalte subdomenii**

Se leagă de *inteligenta artificială*, deoarece permite construirea de programe care au un comportament dinamic și generează cod.

### **3.8.3 Probleme Importante și Probleme Deschise**

Câteva probleme importante sunt *securitatea* care apare atunci când pot fi accesate componente interne, *performanță* generarea de cod care poate introduce erori.

Câteva probleme deschise sunt legate de *standardizarea* reflectiei în mod portabil și sigur.

### **3.8.4 Persoane Importante**

Cercetători și pionieri: John McCarthy, Gregor Kiczales, Andrei Alexandrescu  
Gregor Kiczales, Gilad Bracha

### **3.8.5 Forum-uri Importante**

Forum-uri de specialitate: Lambda the Ultimate, Reddit r/programming, Stack Overflow Lambda the ultimate

### **3.8.6 Dimensiunea locală și dimensiunea globală**

• **Local:** În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, în cursul de tehnici avansate de programare

• **Global:**

- Colaborări internaționale: Proiecte europene(Horizon Europe), Consorții globale(ACM SIGPLAN).
- Standardizări: Java Reflection API, C++ Template Metaprogramming, Python metaprogramming features ISO WG21
- Influență globală: Metaprogramarea este folosită extensiv în dezvoltarea de frameworkuri moderne, sisteme dinamice și automatizare de cod în industrie. a fost adoptată în majoritatea framework-urilor moderne

## 4 Arhitectură

Arhitectura în informatică se referă la proiectarea și organizarea componentelor esențiale ale unui sistem de calcul, fie ele hardware sau software. Acest domeniu este esențial pentru buna funcționare, optimizarea performanței, securitatea și scalabilitatea sistemelor informatici moderne. Într-o lume digitală în continuă evoluție, arhitectura în informatică facilitează conectivitatea, procesarea eficientă și gestionarea datelor în medii distribuite și complexe. Este un domeniu interdisciplinar, aflat la granița dintre inginerie, matematică, logică și știința calculatoarelor.

Lista celor subdomeniilor arhitecturii în informatică este:

1. Arhitectura hardware
2. Arhitectura software
3. Arhitectura sistemelor distribuite
4. Arhitectura rețelelor și comunicațiilor
5. Arhitectura bazei de date
6. Arhitectura Cloud și DevOps
7. Arhitectura securității informaticice

### 4.1 Arhitectura hardware

Arhitectura hardware se referă la organizarea fizică și logică a componentelor care alcătuiesc un sistem de calcul. Include procesorul (CPU), memoria (RAM, cache), magistralele, unitățile de stocare și dispozitivele de intrare/ieșire. Evoluția arhitecturii hardware a condus la dezvoltarea sistemelor multi-core, procesoare specializate (GPU, TPU), sisteme încorporate și dispozitive edge computing. De asemenea, implică noțiuni precum paralelismul la nivel de instrucțiune, predicția salturilor și ierarhia memoriei.

#### 4.1.1 Activitățile principale

1. Teorie:
  - (a) Studiază conceptele de instrucțiune, ciclul mașinii
  - (b) Unități aritmetico-logice (ALU)
  - (c) Memorii ierarhizate
  - (d) Seturi de instrucțiuni (ISA).

2. Experiment:
  - (a) Testarea performanței arhitecturilor RISC și CISC
  - (b) Analiza benchmark-urilor
  - (c) Utilizarea simulatoarelor (ex: Logisim, Multi2Sim).
3. Design:
  - (a) Proiectarea și implementarea de microprocesoare (ex: ARM, Intel Core)
  - (b) GPU-uri pentru procesare paralelă
  - (c) FPGA-uri pentru logica reconfigurabilă.
4. Extra:
  - (a) Cunoștințe necesare: logică digitală, circuite electronice, teoria sistemelor, arhitectura calculatoarelor.

#### **4.1.2 Relațiile cu celelalte subdomenii**

Hardware-ul suportă toate celelalte arhitecturi, furnizând baza pe care rulează software-ul, sistemele distribuite și comunicațiile. Arhitectura hardware influențează direct eficiența algoritmilor, timpul de răspuns al aplicațiilor și capacitatea sistemului de a susține sarcini critice sau în timp real. Este interdependentă cu arhitectura software (prin setul de instrucțiuni și drivere), cu sistemele distribuite (prin scalabilitate) și cu securitatea (prin tehnologii ca TPM).

#### **4.1.3 Probleme Importante și Probleme Deschise**

Probleme actuale: consumul de energie, limitările frecvenței de ceas, disiparea căldurii, dimensiunea tranzistorilor (efectul de tunelare), costurile de fabricație. Probleme deschise: computația cuantică (care redefineste principiile hardware), arhitecturi neuromorfice (inspirate de creierul uman), procesoare optice, arhitecturi heterogene și integrate pentru AI.

#### **4.1.4 Persoane Importante**

1. John von Neumann: Modelul arhitectural fundamental (arhitectura Von Neumann). [42]
2. David A. Patterson și John L. Hennessy: Inițiatorii arhitecturii RISC și autori ai unor lucrări de referință. [43]
3. Gordon Moore: Cunoscut pentru „Legea lui Moore”, care a influențat proiectările în dezvoltarea hardware.

#### 4.1.5 Forumuri Importante

Conferințe: ISCA (International Symposium on Computer Architecture), MICRO (IEEE/ACM International Symposium on Microarchitecture), HPCA (High Performance Computer Architecture). Reviste: IEEE Computer Architecture Letters, ACM Transactions on Architecture and Code Optimization, IEEE Micro.

#### 4.1.6 Dimensiunea locală și dimensiunea globală

- **Local:** Arhitectura hardware este studiată în cadrul cursurilor universitare din România, precum "Arhitectura calculatoarelor" la Facultatea de Matematică și Informatică a Universității din București. Sunt implicate și cercetări în optimizarea circuitelor integrate și în sisteme embedded.
- **Global:** Adoptarea standardelor internaționale (IEEE, ISO) în proiectarea de hardware, cercetarea avansată în SUA, China, Coreea de Sud. Mari companii ca Intel, AMD, ARM, NVIDIA investesc constant în dezvoltarea de noi arhitecturi hardware pentru AI, cloud și IoT.

## 4.2 Arhitectura software

Arhitectura software reprezintă structura de ansamblu a unui sistem software, incluzând componentele acestuia, relațiile dintre ele și principiile de proiectare care le guvernează. Este esențială pentru a asigura scalabilitatea, menținabilitatea și performanța aplicațiilor. Arhitectura software definește modul în care aplicațiile sunt construite, organizate și interacționează cu alte sisteme, incluzând aspecte precum modularitatea, reutilizarea codului, separarea responsabilităților și gestionarea dependențelor.

### 4.2.1 Activitățile principale

1. Teorie:
  - (a) Studierea modelelor arhitecturale (MVC, microservicii, monolit, layered architecture)
  - (b) Principiile SOLID
  - (c) Modeler de proiectare (design patterns).
2. Experiment:
  - (a) Implementarea de aplicații conform diverselor stiluri arhitecturale
  - (b) Folosirea containerelor (Docker)

- (c) Integrarea API-urilor
  - (d) Analiza performanței software.
3. Design:
- (a) Proiectarea arhitecturii aplicațiilor web sau enterprise
  - (b) Alegerea tehnologiilor potrivite
  - (c) Elaborarea de diagrame (UML, C4).
4. Extra:
- (a) Necesită cunoștințe de programare, ingineria software, DevOps, baze de date, testare software.

#### **4.2.2 Relațiile cu celelalte subdomenii**

Hardware-ul suportă toate celelalte arhitecturi, furnizând baza pe care rulează software-ul, sistemele distribuite și comunicațiile. Arhitectura hardware influențează direct eficiența algoritmilor, timpul de răspuns al aplicațiilor și capacitatea sistemului de a susține sarcini critice sau în timp real. Este interdependentă cu arhitectura software (prin setul de instrucțiuni și drivere), cu sistemele distribuite (prin scalabilitate) și cu securitatea (prin tehnologii ca TPM).

#### **4.2.3 Probleme Importante și Probleme Deschise**

Probleme actuale: consumul de energie, limitările frecvenței de ceas, disiparea căldurii, dimensiunea tranzistorilor (efectul de tunelare), costurile de fabricație. Probleme deschise: computația cuantică (care redefineste principiile hardware), arhitecturi neuromorfice (inspirate de creierul uman), procesoare optice, arhitecturi heterogene și integrate pentru AI.

#### **4.2.4 Persoane Importante**

1. John von Neumann: Modelul arhitectural fundamental (arhitectura Von Neumann). [42]
2. David A. Patterson și John L. Hennessy: Inițiatorii arhitecturii RISC și autori ai unor lucrări de referință. [43]
3. Gordon Moore: Cunoscut pentru „Legea lui Moore”, care a influențat proiecțiile în dezvoltarea hardware.

#### 4.2.5 Forumuri Importante

Conferințe: ISCA (International Symposium on Computer Architecture), MICRO (IEEE/ACM International Symposium on Microarchitecture), HPCA (High Performance Computer Architecture). Reviste: IEEE Computer Architecture Letters, ACM Transactions on Architecture and Code Optimization, IEEE Micro.

#### 4.2.6 Dimensiunea locală și dimensiunea globală

- **Local:** Arhitectura hardware este studiată în cadrul cursurilor universitare din România, precum "Arhitectura calculatoarelor" la Facultatea de Matematică și Informatică a Universității din București. Sunt implicate și cercetări în optimizarea circuitelor integrate și în sisteme embedded.
- **Global:** Adoptarea standardelor internaționale (IEEE, ISO) în proiectarea de hardware, cercetarea avansată în SUA, China, Coreea de Sud. Mari companii ca Intel, AMD, ARM, NVIDIA investesc constant în dezvoltarea de noi arhitecturi hardware pentru AI, cloud și IoT.

### 4.3 Arhitectura sistemelor distribuite

Arhitectura sistemelor distribuite se referă la proiectarea și organizarea sistemelor în care componentele sunt răspândite pe mai multe noduri de calcul, comunicând între ele prin rețea. Scopul este realizarea unui sistem coerent, unitar și fiabil, chiar dacă elementele componente sunt independente, distribuite geografic și pot eșua individual. Acest subdomeniu este esențial în cloud computing, big data, aplicații web scalabile și rețele de senzori.

#### 4.3.1 Activitățile principale

1. Teorie:
  - (a) Modele de comunicare (RPC, message passing)
  - (b) Coerența datelor
  - (c) Sincronizare
  - (d) Consistență
  - (e) Algoritmi de consens (Paxos, Raft).
2. Experiment:
  - (a) Implementarea de sisteme distribuite simple
  - (b) Folosirea middleware-ului (ex: gRPC, Apache Kafka)

(c) Monitorizarea performanței și a fault tolerance software.

3. Design:

- (a) Proiectarea arhitecturilor distribuite (client-server, peer-to-peer, microservicii)
- (b) Gestionația încărcării
- (c) Replicarea și distribuția datelor.

4. Extra:

- (a) Necesită cunoștințe de programare concurrentă, rețelistică, sisteme de operare, algoritmi distribuiți.

#### **4.3.2 Relațiile cu celelalte subdomenii**

Sistemele distribuite au nevoie de o arhitectură software bine concepută și rulează pe infrastructura hardware existentă. Ele utilizează rețele de comunicații pentru a funcționa și se bazează pe baze de date distribuite. Arhitectura cloud este o extensie naturală a acestui subdomeniu, iar securitatea este esențială pentru protejarea comunicațiilor și datelor între noduri.

#### **4.3.3 Probleme Importante și Probleme Deschise**

Probleme actuale: latența rețelei, partaționarea datelor, fault tolerance, managementul resurselor. Probleme deschise: sisteme auto-scalabile, toleranță la erori fără replicare completă, consens eficient în rețele mari, integrarea cu computația cuantică și edge computing.

#### **4.3.4 Persoane Importante**

1. Andrew S. Tanenbaum: autor fundamental în domeniul sistemelor distribuite.[44]
2. Leslie Lamport: autorul algoritmului de consens Paxos și a conceptului de ordine logică în distribuție.
3. Ken Birman: cercetător în consistența datelor și comunicare fiabilă.

#### **4.3.5 Forumuri Importante**

Conferințe: USENIX Symposium on Networked Systems Design and Implementation (NSDI), ACM Symposium on Operating Systems Principles (SOSP), IEEE ICDCS (International Conference on Distributed Computing Systems).

Reviste: IEEE Transactions on Parallel and Distributed Systems, ACM Transactions on Distributed Systems, Journal of Parallel and Distributed Computing.

#### **4.3.6 Dimensiunea locală și dimensiunea globală**

- **Local:** Universitățile din România predau cursuri precum "Sisteme Distribuite", dezvoltând aplicații de tip client-server și soluții cloud. Există inițiative de cercetare în distribuirea datelor și toleranță la erori.
- **Global:** Sistemele distribuite sunt utilizate pe scară largă de companii globale precum Amazon, Google, Netflix. Cercetările avansează în direcția automatizării, rezilienței și optimizării comunicațiilor între noduri distribuite.

### **4.4 Arhitectura rețelelor și comunicațiilor**

Arhitectura rețelelor și comunicațiilor se referă la structura, designul și principiile de funcționare ale rețelelor de calculatoare. Acest subdomeniu acoperă protocoalele de comunicație, topologiile rețelelor, nivelurile modelului OSI și TCP/IP, precum și infrastructura fizică (switch-uri, routere, fire optice). Rețelele formează coloana vertebrală a oricărei arhitecturi informatiche moderne, facilitând schimbul de date între sisteme și utilizatori din întreaga lume.

#### **4.4.1 Activitățile principale**

1. Teorie:
  - (a) Studiul modelelor OSI și TCP/IP
  - (b) Protocole precum HTTP, FTP, TCP, UDP, IP, DNS, BGP.
2. Experiment:
  - (a) Simularea rețelelor folosind Cisco Packet Tracer, Wireshark sau GNS3
  - (b) Măsurarea latenței, throughput-ului, pierderii de pachete.
3. Design:
  - (a) Proiectarea de rețele locale (LAN)
  - (b) Rețele geografice (WAN)

- (c) Rețele wireless (Wi-Fi, 5G)
  - (d) VPN-uri
  - (e) Infrastructuri SDN.
4. Extra:
- (a) Cunoștințe din teoria grafurilor, sisteme de operare, securitate informatică, programare rețelistică.

#### **4.4.2 Relațiile cu celelalte subdomenii**

Rețelele permit funcționarea sistemelor distribuite, accesul la baze de date la distanță, operarea în cloud și implementarea soluțiilor DevOps. Fără o arhitectură robustă de comunicații, software-ul modern nu poate funcționa eficient, iar securitatea trebuie aplicată adesea la nivel de rețea.

#### **4.4.3 Probleme Importante și Probleme Deschise**

Probleme actuale: congestia rețelelor, securitatea comunicațiilor (interceptări, spoofing), latență ridicată în medii distribuite. Probleme deschise: rețele definite prin software (SDN), rețele autonome bazate pe AI, comunicații cuantice, extinderea IPv6 și conectivitatea globală.

#### **4.4.4 Persoane Importante**

1. Vint Cerf și Bob Kahn: Creatorii protocolului TCP/IP.
2. Radia Perlman: „Mama Internetului”, inventatoarea protocolului Spanning Tree.
3. Tim Berners-Lee: Inventatorul World Wide Web.

#### **4.4.5 Forumuri Importante**

Conferințe: IEEE INFOCOM, ACM SIGCOMM, Network and Distributed System Security Symposium (NDSS). Reviste: IEEE/ACM Transactions on Networking, Computer Networks Journal, Journal of Network and Systems Management.

#### **4.4.6 Dimensiunea locală și dimensiunea globală**

- **Local:** Cursuri universitare precum „Rețele de calculatoare” sau „Sisteme de comunicații” la majoritatea facultăților tehnice din România. Participare la proiecte de cercetare precum ROEduNet.

- **Global:** Progresul rețelelor globale (5G, Internetul Satelitar), organizații internaționale (IETF, IEEE), infrastructura furnizată de companii precum Cisco, Huawei, Juniper.

## 4.5 Arhitectura Bazei de Date

Arhitectura bazei de date se referă la modul în care sunt proiectate, organizate și gestionate bazele de date într-un sistem informatic. Aceasta include niveluri de abstractizare (intern, conceptual, extern), modele de date (relaționale, orientate pe obiecte, NoSQL), motoare de stocare, sisteme de gestiune a bazelor de date (SGBD) și mecanisme de acces și securitate a datelor. Scopul este de a asigura stocarea eficientă, integritatea, coherența și disponibilitatea datelor în sisteme informatici complexe.

### 4.5.1 Activitățile principale

1. Teorie:

- Studiul Modele de date
- Normalizare
- Tranzacții
- Integritate referențială
- ACID
- Arhitecturi client-server sau multi-tier.

2. Experiment:

- Proiectarea de baze de date relaționale
- Utilizarea SGBD-urilor (MySQL, PostgreSQL, Oracle)
- Testarea performanței interogărilor SQL
- Indexare.

3. Design:

- Arhitecturi scalabile (sharding, replicare)
- Baze de date distribuite
- Modele NoSQL (MongoDB, Cassandra)
- Baze de date in-memory.

4. Extra:

- (a) Cunoștințe necesare: teoria grafurilor, algebra relațională, sisteme de operare, arhitectura calculatoarelor.

#### 4.5.2 Relațiile cu celelalte subdomenii

Arhitectura bazei de date interacționează direct cu arhitectura software și sistemele distribuite, întrucât aplicațiile depind de accesul eficient la date. În infrastructurile moderne cloud sau DevOps, gestionarea datelor este centrală, iar în securitate, protecția bazelor de date este esențială. De asemenea, structura hardware influențează performanța bazelor de date.

#### 4.5.3 Probleme Importante și Probleme Deschise

Probleme actuale: consistența datelor în baze distribuite, scalabilitate, latență în accesul la date, controlul concurenței. Probleme deschise: modele de stocare hibride, baze de date post-relationale, integrarea bazelor de date cu AI/ML, protecția vietii private (ex: GDPR), gestionarea datelor în edge computing.

#### 4.5.4 Persoane Importante

1. E. F. Codd: Inventatorul modelului relațional.
2. Michael Stonebraker: Creatorul PostgreSQL și promotor al bazelor de date moderne. [45], [46]
3. Jim Gray: Premiul Turing pentru contribuții în baze de date tranzacționale și fiabilitate.

#### 4.5.5 Forumuri Importante

Conferințe: VLDB (Very Large Data Bases), SIGMOD (ACM Special Interest Group on Management of Data), ICDE (IEEE International Conference on Data Engineering). Reviste: ACM Transactions on Database Systems, Information Systems, The VLDB Journal.

#### 4.5.6 Dimensiunea locală și dimensiunea globală

- **Local:** Universitățile din România includ cursuri de "Bazele Datelor", cercetări privind optimizarea interogărilor și securitatea datelor.
- **Global:** Marile companii IT (Google, Amazon, Microsoft) dezvoltă sisteme avansate de baze de date (BigQuery, Aurora, CosmosDB). Cloud computing a dus la emergența serviciilor DBaaS (Database-as-a-Service), iar cercetarea în baze de date rămâne una dintre cele mai active ramuri ale informaticii.

## 4.6 Arhitectura Cloud și DevOps

Arhitectura Cloud și DevOps se referă la proiectarea și gestionarea infrastructurilor informatici distribuite în medii cloud, împreună cu metodele moderne de livrare continuă a software-ului. Cloud-ul oferă servicii la cerere (IaaS, PaaS, SaaS) prin centre de date globale, în timp ce DevOps integrează dezvoltarea (Dev) cu operațiunile (Ops) pentru a automatiza și optimiza procesul de livrare și monitorizare a aplicațiilor. Această arhitectură urmărește scalabilitatea, disponibilitatea, reziliența și agilitatea serviciilor IT.

### 4.6.1 Activitățile principale

1. Teorie:
  - (a) Studiul Modele de servicii Cloud (IaaS, PaaS, SaaS)
  - (b) Arhitecturi multi-cloud și hybrid cloud
  - (c) Principii DevOps (CI/CD, Infrastructure as Code, observabilitate)
2. Experiment:
  - (a) Folosirea unor platforme ca AWS, Azure, Google Cloud
  - (b) Automatizări cu Ansible, Terraform
  - (c) Monitorizare cu Prometheus sau Grafana.
3. Design:
  - (a) Arhitecturi serverless
  - (b) Microservicii
  - (c) Containere (Docker)
  - (d) Orchestrare cu Kubernetes
  - (e) Arhitecturi resilient și fault-tolerant.
4. Extra:
  - (a) Cunoștințe necesare: sisteme distribuite, securitate, rețele, scripting, software engineering.

#### **4.6.2 Relațiile cu celelalte subdomenii**

Arhitectura Cloud are legături directe cu sistemele distribuite și cu arhitectura software, oferind infrastructura elastică necesară rulării aplicațiilor moderne. DevOps se bazează pe instrumente de automatizare și versionare, esențiale în livrarea rapidă a aplicațiilor. Securitatea în cloud este o componentă critică, iar optimizarea performanței depinde și de arhitectura rețelelor.

#### **4.6.3 Probleme Importante și Probleme Deschise**

Probleme actuale: gestionarea costurilor în cloud, vendor lock-in, managementul configurațiilor complexe, reziliența la fault-uri. Probleme deschise: edge computing și integrarea cu cloud-ul central, computație autonomă, guvernanța datelor în multi-cloud, DevSecOps, standardizarea arhitecturilor cloud-native.

#### **4.6.4 Persoane Importante**

1. Martin Fowler: Popularizator al microserviciilor și practicilor DevOps. [47]
2. Gene Kim: Coautor al „The Phoenix Project” și promotor al culturii DevOps. [48]
3. Werner Vogels: CTO Amazon, arhitectul principal al AWS, influent în dezvoltarea arhitecturii cloud moderne.

#### **4.6.5 Forumuri Importante**

Conferințe: KubeCon, AWS re:Invent, Google Cloud Next, DevOpsDays, HashiConf. Reviste și platforme: ACM Queue, IEEE Cloud Computing, InfoQ, The New Stack.

#### **4.6.6 Dimensiunea locală și dimensiunea globală**

- **Local:** Companii din România adoptă din ce în ce mai mult tehnologii DevOps și soluții cloud (ex: UiPath, Endava). Universitățile încep să includă cursuri despre cloud computing și DevOps în curricula lor.
- **Global:** Giganți tehnologici ca Amazon, Microsoft, Google sau IBM oferă infrastructuri globale de cloud. DevOps este standardul de facto pentru dezvoltarea modernă, iar standardele internaționale (ISO/IEC 27017 pentru cloud) reglementează calitatea și securitatea acestor arhitecturi.

## **4.7 Arhitectura securității informative**

Arhitectura securității informative definește modul în care sunt proiectate, implementate și gestionate politicile, măsurile și sistemele de protecție a informațiilor într-un mediu IT. Aceasta implică o abordare holistică a confidențialității, integrității, disponibilității și autenticității datelor și serviciilor, prin infrastructuri și protocoale de securitate. Subdomeniul acoperă atât aspecte tehnice (criptografie, firewall-uri, autentificare) cât și aspecte organizaționale (politici de acces, audit, conformitate).

### **4.7.1 Activitățile principale**

1. Teorie:
  - (a) Modele de securitate (Bell-LaPadula, Biba)
  - (b) Criptografie (simetrică, asimetrică)
  - (c) Autentificare
  - (d) Controlul accesului
  - (e) Managementul risurilor.
2. Experiment:
  - (a) Testarea penetrării (penetration testing)
  - (b) Simularea atacurilor cibernetice
  - (c) Securizarea rețelelor
  - (d) Criptare cu TLS
  - (e) Folosirea de IDS/IPS.
3. Design:
  - (a) Arhitecturi zero trust
  - (b) Segmentarea rețelei
  - (c) Aplicații rezistente la atacuri (defensive programming)
  - (d) Sistemele SIEM (Security Information and Event Management)
4. Extra:
  - (a) Necesită cunoștințe de sisteme de operare, rețele, criptografie aplicată, conformitate legală.

#### **4.7.2 Relațiile cu celelalte subdomenii**

Arhitectura securității informaticice este transversală, fiind esențială pentru toate celelalte subdomenii. În cloud, asigură protecția datelor distribuite. În rețele, implică criptarea și controlul traficului. În hardware, presupune implementarea de enclave securizate (ex: Intel SGX). În software, implică dezvoltarea de cod sigur și actualizări regulate.

#### **4.7.3 Probleme Importante și Probleme Deschise**

Probleme actuale: atacuri ransomware, vulnerabilități zero-day, inginerie socială, lipsa de specialiști. Probleme deschise: securizarea IoT-ului și a edge computing-ului, protejarea datelor personale în AI, criptografie post-cuantică, confidențialitate în medii federate/distribuite, securitate by design.

#### **4.7.4 Persoane Importante**

1. Bruce Schneier: Expert în securitate informatică și autor prolific. [49]
2. Whitfield Diffie și Martin Hellman: Pionieri ai criptografiei cu cheie publică. [50]
3. Ron Rivest, Adi Shamir și Leonard Adleman: Creatori ai algoritmului RSA.
4. Mikko Hyppönen: Specialist în securitate cibernetică, analist de malware recunoscut.

#### **4.7.5 Forumuri Importante**

Conferințe: Black Hat, DEF CON, RSA Conference, OWASP Global AppSec. Reviste și surse: IEEE Security & Privacy, ACM Transactions on Privacy and Security, OWASP, NIST.

#### **4.7.6 Dimensiunea locală și dimensiunea globală**

- **Local:** CERT-RO (Centrul Național de Răspuns la Incidente de Securitate Cibernetică), universități și organizații din România dezvoltă programe de masterat și certificări în securitate.
- **Global:** Agenții ca ENISA (UE), NIST (SUA), ISO/IEC stabilesc standarde globale. Companii ca Kaspersky, Palo Alto Networks, Cisco, Google și Microsoft investesc în cercetare și practici avansate de securitate.

## 5 Sisteme de operare și rețelistică

Sistemele de operare sunt componenta software fundamentală a unui sistem de calcul modern. Întrucât sistemele de calcul au crescut foarte rapid în complexitate, programarea manuală a acestora a devenit foarte complexă. Acest fapt a dus la crearea unor sisteme de abstractizare a hardware-ului cu scopul de a simplifica scrierea de software. [51]

1. Teoria planificării proceselor
2. Sisteme de fișiere
3. Gestiunea memoriei
4. Rețele de calculatoare
5. Teoria concurenței
6. Programare concurrentă
7. Analiza Comportamentelor Programelor

### 5.1 Teoria planificării proceselor

Teoria planificării proceselor se ocupă cu metodele de gestionare a mai multor programe care se execută pe un singur procesor. Acest subdomeniu reprezintă o arie fundamentală a unui sistem de operare general modern.

#### 5.1.1 Activități principale

1. Teorie: În acest subdomeniu se lucrează într-un cadru conceptual oarecum abstract, întrucât ideea fundamentală reprezintă crearea unui context „virtual” în care un program se poate executa. Din punct de vedere al raționamentului folosit, acest subdomeniu se bazează foarte mult pe **modele**. Aceste modele conțin totul de la modul de reprezentare a unui „procesor virtual” până la relația dintre procese, relația cu hardware-ul și multe altele. Concepte importante:
  - (a) Program - o „rețetă” folosită pentru a procesa anumite date
  - (b) Proces - o structură de date ce reține mai multe informații despre un anumit program. Procesul se comportă ca un „procesor virtual” care execută programul.

- (c) Schimbare de context - procedeul prin care procesorul fizic trece de la un anume proces la altul
  - (d) Monoprogramare - crearea unui mediu de lucru care conține doar un singur proces
2. Experiment: Teoria planificării are o importantă componentă experimentală. Aceasta este reprezentată, în general, de multitudinea de sisteme de gestiune a proceselor din sistemele de operare moderne. Se consideră un algoritm de gestiune eficient unul care se poate folosi de toate resursele procesorului real.
3. Design: Din perspectiva design-ului s-au realizat mai mulți algoritmi de planificare a proceselor, folosite și astăzi
- (a) First-Come-First-Served este un algoritm de planificare a proceselor ce implică o coadă de procese din care se iau procesele și se execută în ordinea inserării în coadă
  - (b) Shortest-Job-First este un algoritm de planificare a proceselor ce implică o coadă de priorități de procese din care se iau procesele și se execută în ordine crescătoare a timpului de execuție
  - (c) Longest-Job-First este similar cu Shortest-Job-First, dar procesele se execută în ordine descrescătoare a timpului de execuție.
  - (d) Round Robin este un algoritm de planificare a proceselor în care fiecărui proces î se atribuie în mod ciclic un anumit interval de timp în care se poate executa. După ce intervalul expiră, procesul este pus înapoi în coada de execuție pentru a î se atribui un alt interval de execuție
4. Literatură de specialitate
- The MINIX book de A. S. Tannenbaum

### 5.1.2 Relațiile cu celelalte subdomenii

Teoria planificării are o relație puternică cu teoria concurenței și teoria sistemelor distribuite la scară largă

### 5.1.3 Probleme Importante și Probleme Deschise

Cea mai importantă problemă a planificării proceselor este, precum îi zice numele, problema planificării eficiente. Această problemă este în continuare una deschisă, încrucișată nu există un algoritm ideal în toate cazurile. În cazul sistemelor multi-core sau chiar multi-processor, cum putem planifica procese astfel încât să optimizăm latența dintre nuclee sau dintre procesor.

#### **5.1.4 Persoane Importante**

1. Ken Thompson și Dennis Ritchie - creatorii sistemului de operare UNIX, pionierii conceptelor de comunicare între procese, gestiune a proceselor
2. Fernando J. Corbato - pionierul sistemelor de operare cu partitioanare a timpului de execuție al proceselor.

#### **5.1.5 Forumuri Importante**

- OSDev Wiki - for de creare a unui sistem de operare, printre care și componenta de gestiune a proceselor

#### **5.1.6 Dimensiunea locală și dimensiunea globală**

- **Local:** Relația subdomeniului cu activitățile și structura din cadrul Departamentului de Informatică, FMI, UVT. - Avem curs de sisteme de operare în care se predau anumite fundamente ale teoriei gestiunii
- **Global:** Contextul internațional: colaborări, standarde și influența globală a subdomeniului.- Teoria planificării stă la baza tuturor sistemelor de operare, oferind anumite proceduri standard de gestiune a proceselor precum preempția

### **5.2 Sisteme de fișiere**

Sistemele de fișiere se ocupă cu stocarea eficientă a datelor pe un suport non-volatile de date.

#### **5.2.1 Activitățile principale**

1. Teorie:
  - (a) Geometria unui disk - Metodă de atribuire a adreselor unui disk fizic
  - (b) Fișier - O secvență de date corespunzător unui anume conținut
  - (c) Folder - O grupare de fișiere
  - (d) Partiție - O secvență pe un disk.
  - (e) Fragmentări - Stare în care conținutul unui fișier este împrăștiat pe suprafața unui harddisk
  - (f) Cum stochează calculatorul datele?
  - (g) Reziliența la perturbări de natură fizică. Proprietate a sistemului de fișiere la degradarea materialului fizic.

2. Experiment:

- (a) Formate Copy-on-Write - Formate ce copiază un element doar atunci când copia este modificată.
- (b) Formate ce înglobează mai multe diskuri fizice - Formate care acceptă mai multe diskuri pentru a paraleliza stocarea datelor, făcând-o, astfel, mai rapidă
- (c) Sistem de permisiuni a fișierelor - Algoritm ce gestionează operațiile permise ale unui fișier.
- (d) Criptare on-the-fly - criptarea instantanee a conținutului unui sistem de fișiere.

3. Design:

- (a) ext4 - Format folosit de o bună parte din distribuțiile Linux.
- (b) XFS - Format dezvoltat de Silicon Graphics pentru sistemul de operare IRIX. Se remarcă prin performanța foarte bună, inclusiv pe harddiskuri.
- (c) NTFS - Format folosit de toate sistemele de operare din gama Windows NT de la 3.1 la 2000 și Windows, în general, de la Windows XP la 11 inclusiv
- (d) btrfs - Format relativ nou, implicit pe majoritatea distribuțiilor Linux și se remarcă prin Copy-on-Write, *subvolume* (subpartiții) și reziliență mai mare la intemperii fizice.
- (e) ZFS - Format de fișiere conceput pentru gestiunea a mai multor medii de stocare.

4. Extra:

- (a) Algoritmi și Structuri de date, Arhitectura calculatoarelor, Știința materialelor
- (b) Operating Systems: Internals and Design Principles [52]

### 5.2.2 Relațiile cu celelalte subdomenii

Acest subdomeniu este unul esențial care permite celorlalte subdomenii să își poată prelua datele de lucru.

### **5.2.3 Probleme Importante și Probleme Deschise**

1. Cum putem stoca un fișier astfel încât să poată supraviețui cât mai mult pe un mediu fizic?
2. Cum putem să legăm mai multe discuri împreună?

### **5.2.4 Persoane Importante**

1. Chris Mason Creatorul BTRFS
2. Mingming Cao, Andreas Dilger, și echipa de creație ext4 Creatorii ext4
3. Kent Overstreet Creatorul bcachefs
4. Steve Lord Inginerul ce a portat sistemul XFS de pe IRIX pe Linux

### **5.2.5 Forumuri Importante**

- LKML Mailing List-ul kernelului Linux, cel mai folosit kernel sub GPL 2.0
- FMS Summitul Future of Memory and Storage este interesant întrucât aici se prezintă ultimele noutăți despre memorie și stocare, ceea ce poate influența metodele de gestiune a memoriei

### **5.2.6 Dimensiunea locală și dimensiunea globală**

- **Local:** În cadrul UVT avem acces la o sferă tangentă cu sfera sistemelor de fișiere, anume, baze de date. C
- **Global:** La nivel global, ext4 și btrfs au devenit standard pe sistemele de operare Linux. NTFS este de foarte mulți ani standard pe Windows, iar acum tema de cercetare o reprezintă sistemele de fișiere Copy-On-Write

## **5.3 Gestiunea memoriei**

Prezentarea subdomeniului. Contextul istoric și evoluția acestuia.

### **5.3.1 Activitățile principale**

1. Teorie:
  - (a) Pagini de memorie - Secțiuni din memoria fizică pe care sistemul de operare le gestionează

- (b) Maparea memoriei - Procesul de generare a paginilor de memorie din memoria fizică
- (c) Spațiu swap - Spațiu de depozitare temporară a proceselor ce nu mai încap în memorie
- (d) Partiționarea memoriei - Procedeul prin care se stabilesc paginile de memorie
- (e) Alocare dinamică - Proces prin care se solicită mai multă memorie din partea sistemului de operare
- (f) Zone de memorie - Subsecțiuni din memoria asociată unui proces.
- (g) Siguranța memoriei - Conceptul ce asigură că un program nu accesează secțiuni nepermise.

2. Experiment:

- (a) Algoritmi de gestiune a paginilor de memorie - Metode prin care sistemul atribuie și elimină anumite asocieri ale memoriei unui proces.
- (b) Cache de disk - Spațiu de stocare a datelor din memorie temporar pe disk.
- (c) Alocare dinamică (malloc și prietenii săi) - Alocare în timpul execuției programului a unui spațiu mai mare de memorie.

3. Design:

- (a) Alocatoare dinamice - programe ce interacționează cu sistemul de operare pentru a solicita mai multă memorie.
- (b) Zram - Sistem ce alocă o secțiune a memoriei RAM pentru swap.
- (c) Sisteme de compresie a memoriei - Sistem ce aplică un algoritm de compresie asupra conținutul memoriei pentru a elibera spațiu

4. Extra:

- (a) Algoritmi și Structuri de Date, Matematică Statistică
- (b) Operating Systems: Internals and Design Principles [52]

### 5.3.2 Relațiile cu celelalte subdomenii

Gestiunea memoriei este un subdomeniu cheie ce are legătură atât cu teoria planificării cât și cu teoria concurenței, ceea ce

### 5.3.3 Probleme Importante și Probleme Deschise

1. Cum putem accesa sau aloca o cantitate mare de memorie ( $>=4\text{GB}$ ) în mod eficient?
2. Cum putem să ne asigurăm că aranjăm procesele în mod cât mai eficient în memorie?

### 5.3.4 Persoane Importante

1. Donald Knuth - Creatorul „The Art of Programming” în care se prezintă un alocator dinamic. [14]
2. Lorenzo Stoakes - Maintainerul sistemului de mapare a memoriei din Linux

### 5.3.5 Forumuri Importante

- ASPLOS - Conferința internațională pentru suport arhitectural a limbajelor de programare și sisteme de operare acoperă o plajă mare de domenii, acceptând, totuși, deseori, lucrări despre suport hardware pentru gestiunea memoriei
- LKML - Mailing List-ul kernelului Linux, cel mai folosit kernel sub GPL 2.0
- FMS - Summitul Future of Memory and Storage este interesant întrucât aici se prezintă ultimele noutăți despre memorie și stocare, ceea ce poate influența metodele de gestiune a memoriei

### 5.3.6 Dimensiunea locală și dimensiunea globală

- **Local:** La nivel local acest subdomeniu este abordat sumar în cadrul cursului de sisteme de operare și este menționat pe alocuri în cursurile de programare.
- **Global:** Sistemele de management al memoriei sunt într-o continuă dezvoltare pentru a oferi aplicațiilor care necesită multe resurse acele resurse în mod eficient

## 5.4 Rețele de calculatoare

Rețelele de calculatoare se ocupă cu legarea a mai multor calculatoare împreună cu scopul de a împărți resurse.

### **5.4.1 Activitățile principale**

1. Teorie:
  - (a) Rețea - Un graf conex format din calculatoare
  - (b) Subnet - Un subgraf al grafului menționat mai sus
  - (c) Latență - Durata de timp de care semnalul electric are nevoie pentru a ajunge dintr-un nod A al rețelei într-un nod B.
  - (d) Rutare - Procedura de trimite pachetele pe nodurile corecte pentru a ajunge la destinație
  - (e) Calcul distribuit - Noțiunea de a executa o procedură pe mai multe sisteme de calcul interconectate
2. Experiment:
  - (a) Packet-switching - Sistem de spargere a unui mesaj în mai multe pachete și trimitera sa pe mai multe drumuri până la destinație
  - (b) Protocole de negociere - O convenție prin care 2 sisteme creează o convenție ad-hoc de transmisie a datelor.
  - (c) Sisteme de corectare a erorilor de transmisie - permit verificarea mesajului asamblat la final pentru erori
3. Design:
  - (a) Internetul - Rețea globală de calculatoare fundamentată pe procesul de packet-switching.
  - (b) Compilare distribuită - procedeul de a împărți procesul de compilare pe mai multe calculatoare
  - (c) iPXE - protocol de încărcare a unui sistem de operare din rețea
4. Extra:
  - (a) Avem nevoie de ASD și noțiuni elementare de Teoria Semnalelor și știința materialelor
  - (b) Computer networks: a systems approach

### **5.4.2 Relațiile cu celelalte subdomenii**

Rețelele de calculatoare se bazează pe celelalte subdomenii pentru a putea extinde plaja de date pe care un calculator poate să le acceseze

#### **5.4.3 Probleme Importante și Probleme Deschise**

1. Cum putem să ne asigurăm că mesajul trimis a fost primit?
2. Cum putem să verificăm cu o precizie cât mai mare dacă datele preluate nu au fost deteriorate pe parcursul transferului de date?

#### **5.4.4 Persoane Importante**

1. Vint Cerf și Bob Kahn - Creatorii TCP-IP (folosit și azi)
2. Tim Burners Lee - Creatorul World Wide Web-ului
3. Leonard Kleinrock - Un pionier al părții teoretice a științei rețelelor. Teoria sa pentru cozi de pachete a făcut ca ARPANET să fie viabil

#### **5.4.5 Forumuri Importante**

- ACM SIGCOMM - Cea mai prestigioasă conferință de rețelistincă - se ocupă cu partea practică a aplicațiilor și protocoalelor
- IETF (Internet Engineering Task Force) - Comunitate internațională de ingineri ce stabilesc standarde (RFC) de comunicație
- IEEE/ACM Transactions on Networking (ToN) - Jurnal academic în care se publică rezultate importante, verificate, în rețelistică

#### **5.4.6 Dimensiunea locală și dimensiunea globală**

- **Local:** În cardrul ciclului de licență există un curs de rețele de calculatoare
- **Global:** Rețelele de calculatoare au devenit coloana vertebrală a societății secolului XXI datorită Internetului și protocolului Hyper-Text Transfer Protocol (Secured) (HTTP(S))

### **5.5 Programare concurrentă**

Programarea concurrentă se ocupă cu gestiunea execuției unui program pe mai multe fire de execuție sau în mai multe subprocese.

### **5.5.1 Activitățile principale**

1. Teorie:

- (a) Elemente de sincronizare - sunt anumite structuri de date ce „protejează” accesul la o anumită resursă pentru a nu apărea comportamente nedefinite atunci când 2 fire de execuție încearcă să modifice resursa respectivă.
- (b) Fir de control - o grupare de resurse din cadrul unui proces
- (c) Fir de execuție - un pointer la instrucțiunea ce urmează a fi executată
- (d) Variabile atomice - Variabile care se modifică într-o unitate de timp, firele de execuție putând vedea doar valoarea de înainte sau de după orice modificare [53]
- (e) Variabile volatile - Variabile ale căror modificări sunt vizibile instantaneu de toate celelalte proceze [53]
- (f) Operații atomice - Operații care se execută într-un singur bloc
- (g) Fibre de execuție - Secțiune din program ce își poate salva și relua starea și ceda controlul înapoi la programul principal
- (h) *Blocking* - A „bloca” un fir de execuție înseamnă înștiințarea sistemului de gestiune al firelor de execuție că un anumit fir nu mai poate fi executat până când o anumită operație se execută. Firul este pus în starea de așteptare până când operația respectivă se termină.
- (i) Context de execuție - Starea în care procesorul real se află la un moment dat.

2. Experiment:

- (a) Cum putem decupla o componentă ce durează mult de programul principal
- (b) Validarea mecanismelor de gestiune a memoriei folosite de firele de execuție
- (c) Cât de mult putem paraliza un algoritm până când nu mai avem o îmbunătățire a timpului de execuție?

3. Design:

- (a) Mutex - Structură de date ce permite sau blochează accesul la o resursă.

- (b) Semafoare - O valoare întreagă ce permite accesul la date, dacă valoarea este mai mare sau egală cu 0. La fiecare acces se decrementează valoarea cu 1. După terminarea procesării de către un fir de execuție se incrementează cu 1.[54]
- (c) Canale de comunicare între fire de execuție - Un mod de concretizare a modelului „producer-consumer”. Zonă de memorie protejată de primitive de sincronizare ce facilitează transferul corect de date între procese

4. Extra:

- (a) Ce cunoștințe sunt necesare pentru a intra în domeniu? Algoritmi și structuri de date, Arhitectura calculatoarelor,
- (b) Literatura de specialitate:
  - *Modern Operating Systems* de A. S. Tannenbaum [55]
  - *The MINIX Book* de A. S. Tannenbaum
  - Explorations in Parallel Distributed Processing: A Handbook of Models
  - *The little book of semaphores* de Allen B. Downing

### 5.5.2 Relațiile cu celelalte subdomenii

Teoria concurenței are foarte mare legătură cu teoria planificării proceselor, împărțind multe metode de rationament.

### 5.5.3 Probleme Importante și Probleme Deschise

O importantă problemă este problema sincronizării datelor. Se poate întâmpla ca 2 procese să actualizeze în același timp aceeași dată, ducând la o actualizare imprecisă în lipsa unor elemente de sincronizare

O altă problemă importantă o reprezintă scalarea unui algoritm. De multe ori, adăugarea mai multor nuclee / fire de execuție nu este suficient pentru a observa o creștere clară a rapidității obținerii unei soluții, din cauza fenomenului de „cross-talk”.

### 5.5.4 Persoane Importante

1. Edsger Wybe Dijkstra - a inventat conceptele de mutex, semafor și „îmbrățișarea morții” - deadlock-ul, atunci când 2 procese se așteaptă unul pe altul.
2. C.A.R. Hoare - a dezvoltat limbajul CSP (Communicating Sequential Processes) pentru descrierea formală a sistemelor concurente
3. Per Brinch Hansen - Inventatorul a unuia dintre primele sisteme multitasking

### 5.5.5 Forumuri Importante

- ACM European Conference on Computer Systems (EuroSys) - Conferință europeană pentru sisteme de operare, programare concurrentă. Se ocupă cu explorarea implicațiilor ultimelor descoperiri asupra hardware-ului și software-ului
- ACM Symposium on Operating Systems Principles (SOSP) - una dintre cele mai prestigioase conferințe pentru noțiuni de sisteme de operaare, printre care și programarea concurrentă
- Concurrency and Computation: Practice and Experience (Wiley) - jurnal ce publică research original în materie de calcul paralel și distribuit

### 5.5.6 Dimensiunea locală și dimensiunea globală

- **Local:** În cadrul ciclului de licență există un curs optional în anul 3 numit Sisteme de calcul distribuit. Totodată, în anul 2, cursul de sisteme de operare se ocupă la nivel conceptual de anumite elemente ale programării concurente.
- **Global:** Din punct de vedere global, teoria concurenței este un domeniu de foarte mare importanță, încrucișând tot mai multe procese care conțin cel puțin 2 nuclee de procesare. Așadar, execuția în paralel a cât mai multor secțiuni dintr-un program devine o idee cheie a programelor.

## 5.6 Analiza Comportamentelor Programelor

Analiza comportamentelor programelor se ocupă cu analiza modului în care un program folosește resursele unui sistem

### 5.6.1 Activitățile principale

1. Teorie:
  - (a) Analiza fluxului de control - Analiza acțiunilor unui proces.
  - (b) Automate finite - Sisteme teoretice de recunoaștere a unui model sau limbaj
  - (c) Algebra de proces - Metode matematice de modelare a unui sistem de calcul paralel
  - (d) Modele statistice probabiliste - Modele ce prezic tipologia unui proces
2. Experiment:

- (a) Urmărirea apelurilor în nucleu - Procedeu prin care se observă cum reacționează nucleul unui sistem de operare la primirea unor cereri
- (b) Urmărirea apelurilor de sistem -Procedeu prin care se observă interacțiunea unui program cu nucleul sistemului de operare.
- (c) Fuzzing - Proces de testare a aplicațiilor pe intrări ce sunt cunoscute că pot aduce probleme.

3. Design:

- (a) Programme antivirus - Programme ce determină natura unui program și elimină programele cu un comportament malitios.
- (b) Analizatoare ale integrității memoriei - Programme ce analizează accesările în memorie ale unui program dat.
- (c) Fuzzere - Programme ce fac operațiunea de fuzzing.

4. Extra:

- (a) ASD, Assembly
- (b) „The Art of Programming” - D. Knuth, [14]

### **5.6.2 Relațiile cu celelalte subdomenii**

Analiza comportamentală oferă siguranță unui sistem de operare și, implicit, celorlalte subsisteme reprezentate de domeniile sale

### **5.6.3 Probleme Importante și Probleme Deschise**

1. Cum putem verifica un program să execute ceea ce ne-a promis autorul? (Caz particular: nu dăunează sistemului nostru)
2. Cum putem verifica faptul că programul nostru nu se va comporta neașteptat în cazul primirii unei anumite intrări?

### **5.6.4 Persoane Importante**

1. Frances E. Allen - A pus bazele compilatoarelor în anul 1960-1970
2. Manuel Blum și Sampath Kannan - Programme ce se verifică prin autocertificare [56]
3. Bryan Cantrill și echipa DTrace - Sistemul de instrumentare dinamică și cercetare de performanță a kernelului

### 5.6.5 Forumuri Importante

- ACM SOSP - Simpozion pentru cercetarea ultimelor rezultate în materie de securitate a unui sistem de operare, analizia performanței și unelte la nivel de kernel
- IEEE S and P - Simpozionul IEEE de securitate și protecția datelor se
- CAV (Computer Aided Verification) - Un eveniment important cu tematici pe verificarea modelelor, execuția simbolurilor și alte tehnici formale aplicate pe software

### 5.6.6 Dimensiunea locală și dimensiunea globală

- **Local:** În cadrul ciclului de licență UVT, cursul de sisteme de operare are în componența sa anumite elemente de cercetare de performanță. Totodată, în cadrul cursului de programare 3 se predau testarea unitară, ce reprezintă o mică intrare în analiza comportamentală a unui program
- **Global:** Analiza comportamentală a unui program și, implicit, a unui sistem de operare stă la baza unui întreg lanț de încredere mutuală și se asigură de integritatea și corectitudinea bucășilor de software folosite.

## 6 Inginerie software

Ingineria software este disciplina inginerească care se ocupă de toate aspectele producției de software. Datorită importanței tot mai mari a sistemelor software în diverse sectoare economice și sociale, precum și complexității ridicate a acestora, domeniul a evoluat pentru a acoperi o gamă largă de activități și subdiscipline. Ingineria software îmbină metode științifice și tehnici ingineresci pentru a proiecta, implementa, valida și menține aplicații software de înaltă calitate, fiabile și eficiente.

Lista celor 6 subdomenii ale ingineriei software este:

1. Dezvoltarea software
2. Ingineria securității software
3. Arhitectura software
4. Testarea și asigurarea calității software
5. Ingineria sistemelor software
6. DevOps și automatizarea software

### 6.1 Dezvoltarea software

Dezvoltarea software reprezintă procesul organizat de creare a sistemelor software, incluzând etapele de specificare a cerințelor, proiectare, codare, testare și mențenanță. Ea face parte din ciclul de viață al produselor software și implică atât aspecte tehnice, cât și manageriale. Istoric, acest subdomeniu a fost marcat de tranziția de la modele rigide, precum modelul Waterfall, la metode iterative și agile (Scrum, Kanban, XP). Necesitatea abordării metodice a apărut după „criza software” din anii '60. În anii '90 și 2000, filozofii precum Agile au redefinit paradigma dezvoltării pentru a crește colaborarea și eficiența echipei.

#### 6.1.1 Activitățile principale

1. Teorie:
  - (a) Modele de proces (Waterfall, V, spirală, Agile)
  - (b) Estimări de cost/durată
  - (c) Metrici de complexitate
  - (d) Principii de proiectare (SOLID)

- (e) Analiza sistemică
  - (f) Ingineria cerințelor
2. Experiment:
- (a) Compararea metodologiilor (Scrum vs. Waterfall)
  - (b) DevSecOps
  - (c) Explorarea de limbaje (Kotlin, Rust)
  - (d) Validare prin studii de caz și prototipuri (ex. aplicații mobile, kernel-ul Linux)
3. Design:
- (a) Produse software (cod, documentație)
  - (b) IDE-uri, Git, CI/CD
  - (c) Programare în pereche
  - (d) Code review
  - (e) impactul asupra productivității și calității livrabilelor.
4. Extra:
- (a) Necesită cunoștințe despre algoritmi, limbaje, paradigme, unelte de colaborare.

### **6.1.2 Relațiile cu celelalte subdomenii**

Dezvoltarea software se corelează cu arhitectura software (structură), testarea (prin CI, TDD), securitatea (DevSecOps), ingineria sistemelor (integrarea componentelor) și DevOps (automatizare, monitorizare).

### **6.1.3 Probleme importante și probleme deschise**

Probleme: complexitatea proiectelor mari, estimări greșite, mențenabilitate, datorie tehnică. Probleme deschise: scalarea Agile, AI în dezvoltare (ex. code assistants), adaptarea la cloud/edge computing.

#### **6.1.4 Persoane importante: cercetători, practicieni, contribuții**

- Fred Brooks – „Mythical Man-Month”
- Barry Boehm – modelul spirală, COCOMO
- Grady Booch – OOP, UML
- Michael Fowler – design patterns
- Kent Beck – Extreme Programming
- Ken Schwaber, Jeff Sutherland – Scrum
- Ian Sommerville și Roger Pressman – lucrări de sinteză

#### **6.1.5 Forumuri importante: reviste, conferințe, forumuri de specialitate**

Conferințe: ICSE, FSE, ASE, ICSM. Reviste: IEEE TSE, ACM TOSEM, IEEE Software. Forumuri: Stack Overflow, comunități locale. În România: conferințe naționale IT, evenimente universitare.

#### **6.1.6 Dimensiunea locală și globală**

- **Local:** Universitatea de Vest din Timișoara (FMI) – programe în Inginerie Software, cercetare aplicată și colaborări industriale.
- **Global:** Standardizare prin ISO/IEC 12207, CMMI; proiecte open-source (ex. Eclipse Foundation); inițiativa SWEBOK; impactul companiilor mari (Google, Microsoft, Amazon).

## **6.2 Ingineria securității software**

Ingineria securității software este subdisciplina care se ocupă de proiectarea și construcția de sisteme software rezistente la atacuri și erori, integrând cerințele de securitate în toate etapele ciclului de viață al produsului [57]. Ea aplică principii din securitatea informațională și criptografie pentru a minimiza vulnerabilitățile. Istorice, abordarea securității a evoluat de la tratarea ei ca un aspect adițional, după finalizarea dezvoltării, la integrarea încă din fazele timpurii de proiectare (*Shift Left Security*). Creșterea amenințărilor cibernetice și a reglementărilor de securitate (de exemplu, GDPR în UE) au impulsionat consolidarea securității software ca disciplină distinctă.

### **6.2.1 Activitățile principale**

1. Teorie: Se bazează pe modele formale de securitate (ex. modelul Bell–LaPadula pentru confidențialitate) și pe principii de criptografie. Cadrul conceptual include triada CIA (Confidențialitate, Integritate, Disponibilitate), gestionarea riscurilor (analiza amenințărilor, evaluarea vulnerabilităților) și standarde recunoscute (ISO/IEC 27001, OWASP Top 10). Întrebările fundamentale privesc asigurarea că sistemul recunoaște și rezistă atacurilor, tolerând evenualele compromisuri și recuperându-se din acestea [57].
2. Experiment: Cercetarea explorează modele și tehnici de detecție și prevenire a atacurilor. Se testează modele de *threat modeling* (ex. STRIDE), simulări de atac (*penetration testing, red teaming*) și noi algoritmi criptografici sau mecanisme de autentificare. Exemple: incidente de securitate (ex. ransomware) sau securizarea sistemelor IoT.
3. Design:
  - (a) Arhitecturi securizate (VPN, rețele segmentate)
  - (b) Instrumente (ex. SonarQube, Fortify)
  - (c) Tehnici (codare defensivă, input validation)
  - (d) Protocole (TLS, OAuth2) și concepte de reziliență.
  - (e) Impactul designului: reducerea riscurilor și creșterea încrederii în aplicații.
4. Extra: Necesită cunoștințe solide de criptografie, protocole, sisteme de operare, rețele, managementul riscurilor și standarde (ISO 27001, NIST). Bibliografie: [58], [57], OWASP, articole recente.

### **6.2.2 Relațiile cu celelalte subdomenii**

Ingineria securității este transversală: interacționează cu dezvoltarea software (securizarea SDLC), arhitectura (modele de amenințări), testarea (pen testing), ingineria sistemelor (sisteme critice) și DevOps (DevSecOps). Exemple: scanări automate CI/CD, politici stricte în microservicii.

### **6.2.3 Probleme importante și probleme deschise**

Provocări: vulnerabilități zero-day, securizarea lanțului de aprovisionare, adaptarea la tehnologii emergente (IoT, blockchain, quantum). Probleme deschise: automatizarea completă a securității, AI pentru detecția amenințărilor, securitatea arhitecturilor distribuite globale (cloud, edge).

#### **6.2.4 Persoane importante: cercetători, practicieni, contribuții**

Contribuții majore: Bruce Schneier, Peter Neumann, Andrew van der Stock (NIST), OWASP, Ross Anderson, Gene Spafford, Howard și Lipner (SDL la Microsoft). Comunități: CIA, agenții guvernamentale, OWASP.

#### **6.2.5 Forumuri importante: reviste, conferințe, forumuri**

- Conferințe: IEEE S&P, USENIX Security, ACM CCS, Black Hat, DEF CON.
- Reviste: *ACM TOPS, IEEE S&P*.
- Forumuri: OWASP, Security StackExchange, DefCamp (România).

#### **6.2.6 Dimensiunea locală și globală**

- **Local:** În România, securitatea software se predă în universități (ex. UVT), există colaborări cu industria (aplicații guvernamentale) și comunități locale (cluburi, laboratoare).
- **Global:** Sprijinită de standarde internaționale (ISO/IEC 27000, NIST), reglementări (GDPR, NIS), organizații globale (Microsoft, Google, OWASP). Colaborări internaționale: CERT/EU, NATO.

### **6.3 Arhitectura software**

Arhitectura software se referă la structurile la nivel înalt ale unui sistem software și la disciplina care le definește. Conform [59], arhitectura unui sistem este „structurile (una sau mai multe) ale sistemului, compuse din elemente software, proprietățile vizibile extern ale acestor elemente și relațiile dintre ele”. A fost formalizată în anii '90 (Shaw și Garlan), odată cu creșterea complexității sistemelor.

#### **6.3.1 Activitățile principale**

1. Teorie:
  - (a) Modele de calitate (performanță, modifiabilitate)
  - (b) Limbaje (UML, ADL)
  - (c) Modele arhitecturale (stratificată, pe evenimente, microservicii)
  - (d) Pattern-uri (MVC, Broker)
  - (e) Scenarii de calitate.
2. Experiment:

- (a) Studii de caz (monolit vs microservicii)
  - (b) Prototipuri distribuite
  - (c) Simulări (rețele pentru Big Data/IoT)
  - (d) Evaluări automate ale arhitecturii.
3. Design:
- (a) Documentații (diagrame, specificații subsisteme)
  - (b) Unelte CASE (Sparx EA), ArchiMate, xADL.
  - (c) Tehnici: revizuire arhitecturală, guvernare arhitecturală.
  - (d) Impact: scalabilitate, întreținere facilă.
4. Extra:
- (a) Necesită modelare UML, design orientat pe obiect, cunoașterea pattern-urilor, experiență cu TOGAF.
  - (b) Literatură de specialitate: [59], Martin Fowler, ISO/IEC 42010.

### **6.3.2 Relațiile cu celelalte subdomenii**

Deciziile arhitecturale afectează implementarea, testabilitatea, securitatea (ex. criptare, izolarea serviciilor), integrarea cu hardware (ingineria sistemelor), DevOps (automatizare prin microservicii).

### **6.3.3 Probleme importante și probleme deschise**

Probleme: documentarea arhitecturii reale (cod legacy), evaluarea automată a calităților, adaptarea la cerințe emergente. Probleme deschise: arhitecturi pentru AI, blockchain, tranziția monolit–microservicii, integrarea cerințelor nefuncționale.

### **6.3.4 Persoane importante: cercetători, practicieni, contribuții**

Contribuții: Mary Shaw, David Garlan, Grady Booch (UML), Ralph Johnson (design patterns), Len Bass, Paul Clements, Rick Kazman, [59], Martin Fowler, arhitecți de la Netflix, Amazon.

### **6.3.5 Forumuri importante: reviste, conferințe, forumuri de specialitate**

- Conferințe: WICSA, ECSA, ICSE, FSE.
- Reviste: *IEEE Software*, *Journal of Systems and Software*.
- Forumuri: grupuri LinkedIn, comunități de arhitecți.

### 6.3.6 Dimensiunea locală și globală

- **Local:** La UVT se predă în cursuri precum *Proiectarea Avansată a Software-ului*, se colaborează cu industria locală (aplicații enterprise).
- **Global:** Standardele (IEEE 1471 / ISO/IEC 42010), OMG (UML, SOA), Kubernetes (open-source). Provocări: interoperabilitate, calitate arhitecturală globală.

## 6.4 Testarea și asigurarea calității software

*Testarea și asigurarea calității software* (Quality Assurance – QA) este subdisciplina ingineriei software care urmărește verificarea, validarea și îmbunătățirea calității produselor software pe parcursul întregului ciclu de viață. Scopul principal este identificarea defectelor înainte ca produsul să ajungă la utilizator și garantarea conformității cu cerințele specificate. Istoric, testarea software a început ca activitate post-dezvoltare (Waterfall), dar a evoluat către o integrare continuă în procesele agile și DevOps (ex. testare continuă, TDD – Test-Driven Development). Accentul a trecut de la „testarea finală” la „calitatea construită din faza de design”.

### 6.4.1 Activitățile principale

1. **Teorie:** Se bazează pe concepte precum testarea funcțională vs. nefuncțională, testare statică vs. dinamică, niveluri de testare (unitate, integrare, sistem, acceptanță) și tehnici de proiectare a testelor (ex. partitionare echivalentă, analiza valorilor-limită). Calitatea este abordată prin atrbute precum fiabilitate, menenabilitate, eficiență, conformitate, definite de standarde (ex. ISO/IEC 25010). Se urmărește reducerea incertitudinii și creșterea încrederii în produs.
2. **Experiment:**
  - (a) Metode automate de testare (ex. testare automată UI/API)
  - (b) Tehnici bazate pe inteligență artificială (ex. test generation, defect prediction)
  - (c) Evaluarea acoperirii codului (code coverage).
  - (d) Exemple includ testarea aplicațiilor web cu Selenium sau testarea de performanță cu JMeter.
3. **Design:** Rezultatele activităților de testare conduc la sisteme mai robuste și mai ușor de întreținut. Se elaborează suite de teste, strategii de testare (ex. testare exploratorie, regresie) și cadre de testare (ex. JUnit, TestNG).

Integrarea testării în pipeline-urile CI/CD contribuie la detectarea timpurie a erorilor și la livrarea rapidă a versiunilor stabile.

#### 4. Extra:

- (a) Necesită cunoștințe de programare, algoritmi, metriki de calitate, procese de dezvoltare software și unelte (Postman, Jenkins, Allure, SonarQube).
- (b) Bibliografie:
  - *Software Security: Building Security In* - G. McGraw (2006) [60]
  - *The Art of Software Testing (3rd ed.)* - Myers, G. J., Sandler, C., & Badgett, T. (2011)[61]
  - *Software Engineering: A Practitioner's Approach (8th ed.)* - Pressman, R. S. (2014) [62]
  - *Software Engineering (10th ed.)* - Sommerville, I. (2016).[63]
  - *Software Engineering Body of Knowledge* - SWEBOK, IEEE Computer Society, 3.0 (2014) [64]
  - Articole IEEE, ISTQB Foundation Syllabus.

#### 6.4.2 Relațiile cu celelalte subdomenii

Testarea este interconectată cu toate aspectele dezvoltării software: cu arhitectura (testabilitate), cu securitatea (ex. testare de penetrare), cu DevOps (CI/CD pipelines), cu cerințele (validarea acestora), și cu proiectarea UX (teste de uzabilitate). Exemple: testare automată în microservicii, mocking pentru testarea serviciilor externe.

#### 6.4.3 Probleme importante și probleme deschise

**Provocări:** gestionarea testării în sisteme complexe, testarea aplicațiilor distribuite (cloud, edge), testarea IA, costul automatizării.

**Probleme deschise:** testarea complet autonomă, generarea automată de scenarii relevante, predicția erorilor pe baza istoricului, testarea în medii dinamice (ex. CI/CD distribuit).

#### 6.4.4 Persoane importante: cercetători, practicieni, contribuții

1. Contribuții notabile: Cem Kaner (principii moderne de testare) , James Bach (testare exploratorie) , Lisa Crispin & Janet Gregory (Agile testing), Kent Beck (TDD).
2. Comunități active: ISTQB, Ministry of Testing, Test Guild.

#### **6.4.5 Forumuri importante: reviste, conferințe, forumuri**

- **Conferințe:** ICST (IEEE International Conference on Software Testing), Agile Testing Days, TestBash, EuroSTAR, STAREAST.
- **Reviste:** Journal of Systems and Software, IEEE Software.
- **Forumuri:** Stack Overflow (taguri QA/test), Ministry of Testing, Reddit r/QualityAssurance.

#### **6.4.6 Dimensiunea locală și globală**

- **Local:** În România, testarea software este un domeniu bine reprezentat în industrie și educație (ex. cursuri la UPB, UTCN, UVT). Există comunități active (Meetup-uri QA, conferințe locale – Romanian Testing Conference).
- **Global:** Susținută de certificări internaționale (ISTQB, ASTQB), standarde (ISO 29119), și companii majore (Microsoft, Google, IBM). Se practică la scară largă în outsourcing și în centrele R&D internaționale.

### **6.5 Ingineria sistemelor software**

Ingineria sistemelor software se ocupă cu proiectarea, dezvoltarea și integrarea sistemelor software complexe, ca parte a unor sisteme mai largi. Inspirată din ingineria sistemelor, această disciplină pune accent pe viziunea holistică și pe interdependentă software-hardware-uman. Este necesară în contexte precum industria auto (software în vehicule inteligente), telecomunicații și apărare, unde numeroase module software trebuie să coopereze într-un întreg. Ingineria sistemelor software subliniază conceptul de sistem de sisteme și managementul cerințelor complexe, asigurând interoperabilitatea și fiabilitatea întregului sistem.

#### **6.5.1 Activitățile principale**

1. Teorie: Se bazează pe principii sistemic (modelare de sistem, UML/SysML) și pe managementul proiectelor complexe. Cadrul conceptual include procesarea cerințelor de sistem (funcționale și nefuncționale), designul multi-disciplinar și standarde precum ISO/IEC 15288 (procesele de viață ale sistemelor). Metodele de raționament implică gădire de sistem (ex. diagrame de bloc funcțional), gestionarea configurațiilor de sistem și analiza modului în care schimbările într-un subsistem afectează întregul (managementul impactului).
2. Experiment: Exemple de modele explorate includ simulări de sisteme integrate (ex. simulatoare de zbor pentru integrarea subsistemelor avionului),

implementări pilot de rețele IoT la scară (smart cities) și prototipuri de sisteme de control complex (ex. sisteme SCADA pentru utilități). Validarea se face prin teste de integrare complicate și simulări multi-domeniu. Un exemplu avansat este infrastructura de vehicule autonome, unde software-ul vehiculului trebuie să interacționeze cu rețelele de trafic și infrastructura orașului.

3. Design: Produsele rezultate sunt arhitecturi la nivel de sistem (ex. diagrame SysML, arhitecturi multi-strat), protocole de comunicație între subsisteme și platforme de integrare (middleware, Enterprise Service Bus). Instrumente: CASE-uri de inginerie a sistemelor (ex. IBM Rational Rhapsody, Cameo Systems Modeler), simulatoare de sisteme (MATLAB/Simulink). Tehnici rezultate: abstractizare la nivel de sistem, sincronizarea multiplor procese operaționale, orchestrare de sisteme (ex. Kubernetes pentru aplicații distribuite). Impactul designului este în creșterea eficienței și fiabilității sistemului global, permitând funcționarea coordonată a componentelor diverse.
4. Extra:
  - (a) Sunt necesare cunoștințe interdisciplinare: fundamente de inginerie (electronică, mecanică, control), analiza domeniului (cerințe specifice industriei – auto, aviatic, de apărare, telecom), și coordonarea echipelor mari.
  - (b) Literatură de specialitate:
    - *Systems Engineering: Principles and Practice* -Kossiakoff & Sweet, (2011) [65]
    - materiale INCOSE (International Council on Systems Engineering).
    - Standarde cheie: ISO/IEC 15288 (procesele ciclului de viață al sistemului) și 42010 (arhitectura sistemelor).

### 6.5.2 Relațiile cu celelalte subdomenii

Ingineria sistemelor software suprapune software-ul obișnuit în contexte de sistem mai larg. Cerințele de securitate și calitate sunt impuse la nivel de sistem în ansamblu. Arhitectura software este parte integrantă a arhitecturii generale a sistemului (ex. arhitectura software a unui avion integrată cu hardware și cu rețelele de comunicație). DevOps și automatizarea pot fi privite ca extensii la nivel de sistem (ex. automatizarea infrastructurii fizice și a rețelelor). Testarea se realizează la nivel de sistem complet (teste de acceptanță pe întregul sistem integrat).

### **6.5.3 Probleme importante și probleme deschise**

Gestionarea complexității sistemelor integrate și colaborarea între echipe interdisciplinare sunt provocări majore. Securitatea și siguranța în sisteme critice (aviatic, medical) rămân probleme stringente. Probleme deschise: formalizarea proceselor MBSE (Model-Based Systems Engineering) și adaptarea rapidă a acestora în context agil, precum și integrarea standardelor de interoperabilitate specifice fiecărui domeniu (ex. AUTOSAR în auto). De asemenea, adoptarea DevOps la nivel de sistem (ex. DevOps for IoT) și noile modele de licitații de sisteme (SaaS și sistemele ca serviciu) sunt în curs de explorare.

### **6.5.4 Persoane importante: cercetători, practicieni, contribuții**

Aaron Kossiakoff (MBSE), Eberhardt Rechtin (space systems) și Peter Keen se numără printre pionierii ingineriei sistemelor. Organizația INCOSE (fondată în anii '90) reunește experți mondiali în ingineria sistemelor. În România, ingineria sistemelor apare în proiecte de infrastructură și în institute tehnice (de ex. Avioane Craiova, ICMET), deși nu există personalități cunoscute internațional în mod specific. La nivel global, cadre precum Andrew P. Sage și Judy F. (fondatori INCOSE) au pus bazele metodologilor de sistem.

### **6.5.5 Forumuri importante: reviste, conferințe, forumuri de specialitate**

- Conferințe: INCOSE International Symposium (principalul eveniment anual INCOSE), IEEE International Systems Conference, ICAM-Systems (Engineering Systems with Applications).
- Reviste: IEEE Systems Journal, Systems Engineering (Wiley).
- România nu are conferințe dedicate exclusiv ingineriei sistemelor software; subiecte relevante apar la conferințe interdisciplinare (ex. Roedunet – rețele și sisteme)

### **6.5.6 Dimensiunea locală și globală**

- **Local:** La UVT, ingineria sistemelor apare în cursuri precum Ingineria Sistemelor și proiecte de cercetare multi-disciplinare (robotică avansată, sisteme embedded). Universitățile tehnice colaborează cu industria (ex. automotive, energie) pentru dezvoltarea sistemelor complexe care includ componente software. Totuși, focusul este mai mult pe software individual, iar conceptul de sistem de sisteme este abordat în context de proiecte aplicate.

- **Global:** Ingineria sistemelor software este dominată de organizații internaționale (INCOSE) și standarde (ISO/IEC 15288, ISO/IEC 12207 pentru procese). Colaborările internaționale în cercetare includ proiecte europene (Horizon Europe) și parteneriate industriale (ex. Aerospace Valley, Smart-Grid). Standardele de interoperabilitate (ex. AUTOSAR, ARINC în avionics) reflectă influența globală. Companii precum Lockheed Martin, Boeing, Airbus investesc masiv în MBSE, influențând întreaga industrie.

## 6.6 DevOps și automatizarea software

DevOps reprezintă ansamblul de practici, culturi și instrumente care integrează echipele de dezvoltare (Dev) și cele de operațiuni (Ops) pentru a livra aplicații și servicii la viteze ridicate [66]. Accentul se pune pe automatizare, colaborare și feedback continuu. Termenul a apărut în jurul anului 2009 (Patrick Debois) ca reacție la silozurile dintre dezvoltatori și operatorii de infrastructură. DevOps împrumută principii din metodologiile Agile și Lean, urmărind eliminarea barierelor între roluri și accelerarea ciclului de livrare [66].

### 6.6.1 Activitățile principale

1. Teorie: Se bazează pe modelul CALMS (Cultură, Automatizare, Lean, Măsurare, Sharing) și pe principiile Lean (flux continuu, eliminarea risipei). Elemente teoretice includ Continuous Delivery Pipeline, Infrastructure as Code și Continuous Testing. Modele de performanță precum devops metrics (lead time, frecvența livrării, MTTR etc.) sunt studiate pentru a evalua eficiența. Întrebările fundamentale: Cum creștem viteza și stabilitatea livrării software fără a compromite calitatea? Cum organizăm cultura organizațională pentru a facilita aceste obiective?
2. Experiment: Studii empirice (ex. raportul Accelerate – DORA) demonstrează corelații între practicile DevOps și performanța echipelor. Exemple includ implementarea pipeline-urilor CI/CD complete (de ex. Jenkins + Docker + Kubernetes) și automatizarea configurației de mediu (Ansible, Terraform). Exemple inedite: utilizarea Inteligenței Artificiale pentru automatizarea recomandărilor în pipeline, chatops (controlul pipeline-ului prin chatbots) și testare continuă cu containere ephemeral. Abordări avansate: GitOps (managementul infrastructurii prin Git) și chaos engineering (injectarea deliberată de defecte pentru a testa reziliența).
3. Design: Rezultatele includ fluxuri de lucru automatizate (scripturi pentru build, test, deploy), modele de containerizare (Docker) și platforme de orchestrare (Kubernetes, Docker Swarm). Instrumente-cheie: sisteme de CI/CD

(Jenkins, GitLab CI/CD), tool-uri de management al configurației (Ansible, Chef, Puppet), tehnologii de container orchestration. Tehnici: Infrastructure as Code (definirea infrastructurii ca cod versionat), pipeline-uri automatizate de testare și deploy, și monitorizare continuă (Prometheus, Grafana). Impactul acestor practici este semnificativ: creșterea vitezei de livrare, scăderea erorilor umane și îmbunătățirea fiabilității aplicațiilor [66]

#### 4. Extra:

- (a) Sunt necesare cunoștințe de administrare a sistemelor (Linux, rețelistică), de scripting (bash, Python) și de cloud computing (AWS, Azure, GCP). Cunoașterea conceptelor Agile și Lean este de asemenea importantă.
- (b) Literatura de specialitate:
  - *Accelerate* - Forsgren et al., (2018) [66]
  - *The DevOps Handbook* - Kim et al., (2016) [48]
  - Mai recent, rapoartele industriei (State of DevOps) și lucrările din IEEE Software sau CACM furnizează perspective științifice asupra DevOps.

#### 6.6.2 Relațiile cu celelalte subdomenii

DevOps servește ca liant între dezvoltare, testare și operațiuni. Automatizarea și cultura DevOps asigură faptul că testarea (QA) se întâmplă continuu în pipeline-ul de livrare. Echipele DevOps integrează securitatea în proces (DevSecOps), astfel încât noile vulnerabilități să fie detectate rapid. Arhitectura software influențează adoptarea DevOps (ex. arhitecturile bazate pe microservicii facilitează pipeline-uri CI/CD paralele). Ingineria sistemelor se intersecează cu DevOps în orchestrarea sistemelor distribuite și a infrastructurii ca un întreg unitar.

#### 6.6.3 Probleme importante și probleme deschise

Provocări: gestionarea complexităților dependențelor în lansări frecvente, administrarea infrastructurii hibride (on-premises + cloud), și schimbarea culturală necesară în organizații (rezistență la schimbare). Probleme deschise includ definirea indicatorilor KPI (ex. DORA metrics) care să măsoare în mod obiectiv performanța DevOps, integrarea completă a lanțului de aprovisionare (software bill of materials) pentru trasabilitatea codului, și adoptarea DevOps în industrii reglementate (bănci, sănătate) fără a compromite conformitatea. Adaptarea la noile modele de infrastructură (serverless, edge computing) rămâne o zonă activă de cercetare.

#### **6.6.4 Persoane importante: cercetători, practicieni, contribuții**

1. Patrick Debois este considerat unul din pionierii culturii DevOps.
2. Gene Kim, Jez Humble și Nicole Forsgren au popularizat DevOps și au condus cercetarea formală [48], [66]
3. Alți lideri includ Damon Edwards și John Willis [48]
4. Cercetători precum Anders Wallgren (studii de adoptare DevOps) contribuie la înțelegerea academică.
5. Mulți experti AWS, Google și Microsoft împărtășesc practici prin conferințe și articole.

#### **6.6.5 Forumuri importante: reviste, conferințe, forumuri de specialitate**

- Evenimente principale: DevOpsDays (evenimente regionale), DevOps Enterprise Summit, conferințe mixte de cloud și Agile (ex. Agile + DevOps).
- Reviste academice (IEEE Software, Journal of Systems and Software) publică lucrări despre DevOps.
- Comunități și forumuri online (CNCF – Cloud Native Computing Foundation, Kubernetes Forum) reprezintă platforme de colaborare tehnică continuă.

#### **6.6.6 Dimensiunea locală și globală**

- **Local:** În România, DevOps a câștigat teren prin meetup-uri și conferințe locale (ex. Timișoara Cloud Native Meetup). UVT include elemente DevOps în laboratoarele de administrare de sistem și proiectele capstone. Totodată, multe companii locale de outsourcing software și startup-uri folosesc intensiv DevOps (Git, Docker, CI/CD), influențând cerințele educaționale.
- **Global:** DevOps este acceptat la scară mondială în companiile de tehnologie. Organizații precum CNCF și DevOps Institute sunt lideri în definirea de bune practici și certificări. Standardele informale (Continuous Delivery, Agile, cloud-native architectures) sunt promovate prin studii de caz globale. Devenirea DevOps de bază pentru dezvoltare este vizibilă în tendințele globale de angajare și proiectare de infrastructură (Infrastructure as Code, platform engineering).

## 7 Baze de date si regasire de informatii

Bazele de date sunt o componenta esentiala a societatii, fiind regasite in aproape fiecare domeniu de activitate. Acestea faciliteaza stocarea si accesarea de date, fie ca este vorba despre cumpararea unor obiecte, rezervari online sau datele salvate in banchi. Daca intr-un tip de activitate se foloseste un calculator, cel mai probabil se folosesc si bazele de date.

1. Modele Logice de Date
2. Mecanisme de Stocare și Indexare
3. Acces la Date și Optimizarea Interrogărilor (Query optimization)
4. Controlul Concurenței și Managementul Tranzacțiilor
5. Recuperarea și Integritatea Bazelor de Date
6. Securitate și Confidențialitate în Bazele de Date
7. Big Data și Baze de Date Distribuite
8. Depozitarea Datelor și Analiza
9. Reprezentări Avansate ale Datelor și Mașini Virtuale
10. Integrarea Datelor Eterogene și Multimedia

### 7.1 Modele logice de date

Modelele logice de date generalizează modelele relaționale, ierarhice și de rețea. Acestea reprezintă schema unei baze de date sub forma unui graf orientat, în care frunzele reprezintă datele, iar nodurile interne reprezintă conexiunile. Modelul face distincția între numele obiectelor și valorile acestora, permitând astfel definirea clară a semanticii.

Modelele logice definesc structura conceptuală a datelor (tabele, grafuri, documente) și stabilesc cum sunt legate entitățile din lumea reală. Evoluția a pornit de la modelul relational (1970, E.F.Codd, [67]) și a continuat cu modele graf (nod-arc) și document (JSON).

### 7.1.1 Activitățile principale

1. Teorie:

(a) Definirea cadrului conceptual

Modelele logice de date reprezintă o descriere detaliată a structurii datelor, incluzând entitățile, atributele și relațiile dintre acestea. Ele sunt independente de sistemul de gestionare a bazelor de date (SGBD) și servesc ca bază pentru proiectarea modelelor fizice. Modelul relațional (tabele + relații) extins cu graf (noduri + muchii) și document-store (JSON/XML) explică ce stocăm, tabel (rânduri + coloane), cheie primară (identifică unică), cheie externă (leagă tabele).

(b) Descrierea cadrului/cadrelor

În centrul său schemă = entități (obiecte reale) + atrbute (proprietăți) + relații, la care se mai adaugă și constrângeri reprezentate în algebra relațională (set de operatori formali) sau cardinalitatea.

(c) Metode de raționament utilizate.

- normalizarea (eliminarea redundanțelor) prin dependențe funcționale
- logica predicatelor
- deducția logică (Datalog)
- validarea ontologică (OWL)

(d) Exemplificarea întrebărilor fundamentale (ex.: Cum funcționează...?, De ce...?). Comparăm property-graph (Neo4j) cu document (MongoDB) pe un set de rețele sociale.

- „Este schema suficient de clară pentru aplicație?”
- „Ce pierdem când migrăm de la relațional la graf?”
- Ce model aleg dacă am multe relații între obiecte?

2. Experiment:

(a) Explorarea modelelor.

Desenăm aceeași aplicație (catalog școlar) în trei modele: tabelar, graf și document; observăm plusuri și minusuri.

(b) Implementări și validarea experimentală.

Se încarcă 1.000 de înregistrări și se verifică dacă găsim rapid „Cursurile la care e înscris Ana”.

(c) Exemplu practic (ex.: metode de sortare și relevanța lor).

Transformarea catalogului universitar într-un knowledge-graph ce răspunde la: „Cine predă cursuri avansate după 2023?”

3. Design:
  - (a) Produse rezultate din cercetare.  
generator Excel→SQL. (SQLizer, TableConvert etc)
  - (b) Instrumente rezultate din cercetare.  
editor online de diagrame ER (Entity-Relationship) (Draw-SQL, Lucid-Chart etc)
  - (c) Tehnici rezultate din cercetare.  
mapare automată obiect-relațională (ORM)
  - (d) Impactul designului asupra dezvoltării subdomeniului.
    - Explicarea modelului prin pictograme simplifică discuția dintre programatori și clienți.
    - reduce timpul de lansare a noilor aplicații.

4. Extra:
  - (a) Ce cunoștințe sunt necesare pentru a intra în domeniu?
    - cunoștințe de matematică discretă
    - noțiuni elementare de logică
  - (b) Literatura de specialitate
    - Codd E. F. (1970) — “A Relational Model of Data for Large Shared Data Banks”. [67]
    - Chen P. P. (1976) — “The Entity-Relationship Model: Toward a Unified View of Data”. [68]
    - Libkin L. (2014) — “Foundations of Data Exchange”. [69]

### 7.1.2 Relațiile cu celelalte subdomenii

Descrierea interdependențelor și influențelor reciproce. Exemplificarea tipurilor de relații: dependență, influență, colaborare etc.

### 7.1.3 Probleme Importante și Probleme Deschise

- Evoluția schemelor fără downtime (schema-less vs. schema-later).
- Unificarea modelelor relațional-graf-document într-un singur SGBD multi-model.
- Validarea semantică automată a datelor folosind ontologii.

#### **7.1.4 Persoane Importante**

1. E.F.Codd (părintele modelului relațional)
2. Peter Chen (model ER).
3. Leonid Libkin (bazele logicei pentru baze de date)

#### **7.1.5 Forumuri Importante**

- Conferințe: DMZ (Data modelling zone), ACM SIGMOD (se discută cele mai noi cercetări, tendințe și inovații din domeniul gestionării datelor și al bazelor de date), VLDB
- Jurnale: ACM TODS.
- Forumuri: Idera Community – Data Modeling Tools

#### **7.1.6 Dimensiunea locală și dimensiunea globală**

- **Local:** cursul „Modelarea BD” (FMI-UVT).
- **Global:** standardele ISO SQL și W3C RDF.

### **7.2 Mecanisme de Stocare și Indexare**

Este ramura informaticii care se ocupă cu felul în care datele sunt păstrate pe disc, în memorie sau în cloud și modul în care găsim rapid ceea ce căutăm. Istoric, aceste mecanisme au inceput în anii 1950 de la benzi și discuri cu citire secvențială, iar apoi la fișiere index-secvențiale (ISAM) în anii 1960, apoi modelul relațional și arborii B+ (IBM System R, 1970). Web-ul anilor 1990 a adus indicele inversat pentru căutare text, iar din 2000 încep arhitecturile distribuite (Bigtable, Dynamo), bazele “in-memory” (Redis) și serviciile cloud (S3) scalează pe zeci de mii de servere.

#### **7.2.1 Activitățile principale**

##### **1. Teorie:**

- (a) Definirea cadrului conceptual – Stocarea este privită ca procesul de păstrare persistentă a datelor, iar indexarea ca mecanism de localizare rapidă. Obiectivele principale sunt integritatea informației, folosirea eficientă a spațiului și minimizarea latenței de acces.
- (b) Descrierea cadrelor – Sunt luate în considerare patru niveluri:

- suportul fizic (SSD, HDD, memorie volatilă)
- sistemul de fișiere
- motorul de stocare (ex.: arbori B+, LSM-tree, tabele hash)
- modelul de date (relațional, cheie-valoare, coloanar, document)

Pentru fiecare nivel se evaluează costul de I/O, complexitatea algoritmică și garanțiile de consistență.

(c) Metode de raționament utilizate – Se aplică

- analiza complexității (notația  $O$ )
- modele de cost I/O (metodă de estimare a performanței unei operații într-o bază de date, bazată pe numărul de citiri și scrieri pe disc necesare pentru a executa acea operație)
- teoreme de consistență (ACID, CAP)
- validare empirică prin benchmark-uri

(d) Întrebări fundamentale

- Cum se determină optimul dintre spațiul ocupat de un index și câștigul de performanță adus de acesta?
- Cum știe sistemul unde să găsească rapid o informație anume?
- Cum se actualizează indexul când adăugăm, ștergem sau modificăm date?

## 2. Experiment:

(a) Explorarea modelelor

- *Wisconsin Benchmark* (1983) a comparat primele două prototipuri relaționale – System R și Ingres – și a arătat variații de  $3\text{--}7 \times$  la selecții și îmbinări, explicând practic de ce implementarea internă (arbori B+) contează la fel de mult ca limbajul SQL folosit de ambele sisteme.
- Evaluările *Google Bigtable* (2006) au dovedit că un model pe coloane distribuit poate susține peste 500 MB/s la scriere pe sute de noduri, păstrând citiri sub 200 ms; rezultatele au popularizat ideea de “NoSQL” scalabil orizontal.
- Seria de teste  *RocksDB* (2013) a pus față în față B-tree și LSM-tree: la flux intens de inserări, LSM-tree a atins de zece ori debitul la scriere, acceptând însă o dublare a latenței la citiri punctuale – compromisul clasic între scriere și citire.

(b) Implementări și validarea experimentală

- Suitele *TPC-C* (1992) și *TPC-H* (1999) sunt standardul de măsurare: recordurile recente depășesc 10 milioane tpmC la un cost sub 1 USD/tpmC pe hardware x86 obișnuit, confirmând că optimizările de jurnal și cache reușesc să țină pasul cu creșterea capacitații SSD-urilor.
- Pentru replicare globală, studiul *Azure Cosmos DB* (2018) a raportat coerență menținută în mai puțin de 15 ms pe patru regiuni; cifra a validat practic predicțiile teoretice CAP privind latența necesară unei consistențe puternice la distanță mare.
- Experimentele *LSM-trie* (2018) au folosit YCSB și au arătat că, într-un flux predominant de citiri, faza de „compactare” ajunge principalul consumator de timp, identificând astfel punctul în care LSM-ul își pierde avantajul față de B-tree.

(c) Exemplu practic

- Recordurile *Daytona / PennySort* (2009–2024) ilustrează diferența între sortarea în memorie și cea externă: 100 GB sortați pe SSD NVMe cu un algoritm paralel extern termină în 30 s, în timp ce pe HDD durează peste 12 minute – un argument direct pentru alegerea sortării externe pe medii rapide.
- În *PostgreSQL 14* (2021), un test pg\_bench pe 10 milioane de rânduri a arătat că adăugarea unui index B-tree pe coloana de timp reduce o scanare de la 140 ms la 4 ms, demonstrând efectul imediat al indexării corecte asupra timpului de răspuns.

3. Design:

(a) Produse rezultate din cercetare

- PostgreSQL (bază relațională cu arbori B-tree și GiST)
- Elasticsearch (motor de căutare bazat pe indicele inversat Lucene)
- Google Bigtable (sistem columnar distribuit)

(b) Instrumente rezultate din cercetare – Suitele TPC-C și YCSB evaluatează debitul tranzacțiilor, iar perf și flamegraph localizează secțiunile lente de cod în execuții reale.

(c) Tehnici rezultate din cercetare

- Replicare asincronă cu re-alegere de lider sub 2 s.
- Compresie pe coloane cu dublu-dicționar care reduce spațiul ocupat de logurile JSON până la 75 %.
- Partiționare automată pe intervale de timp și re-echilibrare fără oprire de serviciu.

(d) Impactul designului asupra subdomeniului

- Compresia pe coloane a făcut posibilă interogarea interactivă a teraoctetilor de date în dashboard-uri BI.
- Replicarea automată și failover-ul rapid au ridicat așteptările de disponibilitate la „cinci nouă” (99,999 %).
- Partiționarea și re-echilibrarea dinamică permit scalarea liniară pe sute de noduri, influențând arhitectura modernă de microservicii.

(e) Extra:

- i. Ce cunoștiințe sunt necesare pentru a intra în domeniu?
  - structuri de date
  - sisteme de operare
- ii. Literatura de specialitate.
  - Bayer R., McCreight E. (1972) — “Organization and Maintenance of Large Ordered Indexes”.
  - Guttman A. (1984) — “R-trees: A Dynamic Index Structure for Spatial Searching”. [70]
  - O’Neil P. et al. (1996) — “The Log-Structured Merge-Tree (LSM-tree)”.
  - Knuth D. E. (1998 ed.) — “The Art of Computer Programming, Vol. 3: Sorting & Searching”. [14]

### 7.2.2 Relațiile cu celelalte subdomenii

- sisteme de operare
- arhitectura sistemelor distribuite
- inteligenței artificiale

### 7.2.3 Probleme Importante și Probleme Deschise

- Eficiență vs. consistență: găsirea echilibrului între viteza de acces și garanțiile ACID în sisteme geo-distribuite.
- Indexare pe date comprimate: cum se menține căutarea rapidă fără a decomprima integral fișierul.
- Stocare pentru date criptate: proiectarea de indexuri care funcționează direct pe text cifrat, păstrând confidențialitatea.

- Autotuning: baze de date capabile să-și adapteze singure schema de stocare și indexurile la fluxurile de lucru în schimbare.
- Energie și sustenabilitate: reducerea consumului electric al centrelor de date fără a sacrifică performanța.

#### 7.2.4 Persoane Importante

1. Edgar F. Codd – a introdus modelul relațional, stabilind baza teoretică pentru indexarea modernă. [67]
2. Michael Stonebraker – a condus proiecte-cheie (Ingres, Postgres) și a promovat concepțile de column-store și stream processing. [45], [46]
3. Jeffrey Dean & Sanjay Ghemawat – au conceput Bigtable și modele care stau la baza stocării scalabile din cloud. [71]
4. Patrick O’Neil – a formalizat B-tree-ul și a contribuit la metoda de *log-structured merge*.
5. Michael Burrows – a creat indicele inversat al lui Google, revoluționând căutarea pe web.

#### 7.2.5 Forumuri Importante

- *ACM SIGMOD* și *VLDB* – conferințe de top pentru baze de date și sisteme de stocare.
- *USENIX FAST* și *USENIX ATC* – forumuri axate pe sisteme de fișiere, I/O și performanță.
- Revistele *ACM TODS* și *VLDB Journal* – publică articole extinse, revizuite.
- Workshop-urile *CIDR* și *HotStorage* – idei emergente și direcții de cercetare timpurii.

#### 7.2.6 Dimensiunea locală și dimensiunea globală

- Local:
  - *Curs „Baze de date avansate” (UVT, anul III)* – laboratoare dedicate construirii și evaluării indexurilor B+, GiST și LSM pe PostgreSQL și RocksDB.

- *Laborator „Sisteme Distribuite” (UVT)* – replicare și partitioanare Bigtable-like pe clusterul HPC al facultății, cu măsurarea latențelor de I/O.
- *Grant CDI „OPTISTORE” (2024–2026)* – optimizarea scrierilor pe SSD NVMe pentru fluxuri IoT, în parteneriat cu Continental Automotive Timișoara; include prototipuri cu log-structured merge și compresie pe coloane.
- *Hackathon „Data Hack Vest”* – secțiune dedicată microserviciilor de persistență: concurs de proiectare a unui magazin cheie-valoare peste RocksDB și Kafka, realizat de studenți UVT cu mentorat Nokia și Endava.
- **Global:** La nivel internațional, cercetarea se aliniază standardelor SQL/ISO, inițiatiivelor de tip open-source (PostgreSQL, RocksDB) și proiectelor europene de supercomputing. Colaborări cu centre precum TU Munich, ETH Zürich și consorțiul LF AI sporesc impactul global al rezultatelor.

## 7.3 Acces la date și optimizarea interogărilor

Se studiază algoritmii, strategiile și mecanismele care permit extragerea rapidă a informației din baze de date, prin transformarea unei interogări declarative (ex. SQL) într-un plan de execuție eficient. Evoluția a început cu optimizatorul cost-based Selinger (1979) și a continuat cu tehnici pentru sisteme paralelizate, colonare și distribuite.

### 7.3.1 Activitățile principale

1. Teorie:
  - (a) Cadru conceptual
    - *Model de cost* (estimează câte citiri de disc, operații CPU și transferuri de rețea consumă un plan).
    - *Algebra relațională* (set minimal de operații SELECT, PROJECT, JOIN pe care SQL le transpunе).
  - (b) Cadre de optimizare
    - *Heuristic Optimizer* (aplică reguli fixe: mută filtrele înainte, combină proiectările).
    - *Cost-based Optimizer* – Selinger/Volcano (generează mai multe planuri, alege cel mai ieftin conform modelului de cost).
  - (c) Metode de raționament

- *Estimarea cardinalității* (ghicește câte rânduri produce fiecare pas; folosește statistici și histograme = distribuția valorilor pe coloane).
- *Enumerarea planurilor* (alcătuiește „join tree” – arborele care arată ordinea îmbinărilor).
- *Reguli de comutare a îmbinărilor* (permite schimbarea ordinii deoarece JOIN este asociativ și comutativ).

(d) Întrebări fundamentale

- Cum se stabilește ordinea îmbinărilor pentru a evita explozia de date intermediare?
- De ce aplicarea filtrului înainte de JOIN reduce drastic costul total?
- Când este *index nested-loop* (caută rând cu rând printr-un index) mai bun decât *hash join* (încarcă tablele în memorie și creează tabele hash)?

2. Experiment:

- Explorarea modelelor – comparație publicată TPC-H între PostgreSQL (execuță plan clasic cu hash-join) și DuckDB (execuță vectorizat: procesează datele pe loturi pentru mai bună folosire a cache-ului).
- Implementări și validare – lucrări recente arată că estimările EXPLAIN ANALYZE în PostgreSQL 16 au eroare medie sub 10%
- Exemplu practic – un index compus (`customer_id, order_date`) reduce interogarea „cel mai bun client lunar” de la 220 ms la 9 ms ( $15 \times$  mai rapid) prin evitarea sortării și a scanării integrale a tablei.

3. Design:

- Produse rezultate – PostgreSQL Optimizer (cost-based clasic), Spark Catalyst (reescrie planul ca arbore logic, apoi fizic), Google F1 (optimizează interogări SQL cu tranzacții distribuite pe Spanner).
- Instrumente – EXPLAIN PLAN (arată pașii), pg\_visualize (graf interactiv), Index Advisor (propune indexuri pe baza istoricului de interogări).
- Tehnici – reordonarea automată a îmbinărilor, *predicate push-down* (trimite filtrele cât mai aproape de sursa de date), materializare incrementală (refolosește rezultate partiale), optimizatori „learned” (modele ML care prezic costul).
- Impact – latențe sub 50 ms pentru analitice complexe și scădere cost cloud cu 40%

4. Extra:

(a) **Cunoștințe necesare**

- SQL avansat
- Algebra relațională (operațiile de bază SELECT, PROJECT, JOIN)
- Structuri de date clasice (tabele hash, arbori B/B+)
- Statistică elementară (estimări de cardinalitate, distribuții)
- Programare practică (C/C++, Python)

(b) Literatură de specialitate

- Selinger *et al.* (1979) — *Access Path Selection in a Relational Database Management System*. [72]
- Graefe (1993) — *The Volcano Optimizer Generator: Extensibility and Efficient Search*. [73]
- Fent, Moerkotte, Neumann (2023) — *Asymptotically Better Query Optimization Using Indexed Algebra*. [42]
- Kleppmann (2017) — *Designing Data-Intensive Applications*, cap. 12. [32]

### 7.3.2 Relațiile cu celelalte subdomenii

- Algoritmi și structuri de date (planurile de interogare folosesc arbori, grafuri și tabele hash; analiza costurilor se bazează pe tehnici clasice de complexitate).
- Limbaje de programare (SQL este un limbaj declarativ; optimizatorul rescrie expresiile în algebră relațională, iar ORMs generează automat interogări ce trebuie apoi optimizate).
- Arhitectură (execuția vectorizată și instrucțiunile SIMD influențează forma planului fizic; set-based processing profită de lărgimea registrelor CPU).
- Sisteme de operare și rețele (planurile țin cont de ierarhia memorie–cache–SSD și de costul transferului de date între noduri în interogări distribuite).
- Inginerie software (profilarea interogărilor, „query tracing” și instrumentele CI/CD includ pași de tuning automat pentru planuri).
- Inteligență artificială & Robotica (modelele „learned optimizer” folosesc tehnici de ML pentru a prezice cardinalități și a alege planuri mai bune).
- Interacțiune om–computer (vizualizatoarele de planuri și dashboard-urile de performanță ajută dezvoltatorii să înțeleagă și să ajusteze strategiile de acces).

- Bioinformatică (bazele de date coloanare și sistemele de interogare paralelă sunt esențiale pentru analizele din fizică, biologie sau climatologie).
- Informatică organizațională (sistemele de suport decizional ERP/BI depind de optimizarea interogărilor pentru rapoarte rapide pe volume mari).

### 7.3.3 Probleme Importante și Probleme Deschise

- Estimarea cardinalității pe date foarte skew-ate (eroare  $>10\times$ ).
- Optimizare pentru hardware heterogen (CPU + GPU, SSD + NVM).
- Integrarea regulilor de cost cu modele ML fără pierderea predictibilității.
- Planuri adaptative care se ajustează la execuție (re-optimizare on-the-fly).

### 7.3.4 Persoane Importante

- Pat Selinger (pionier cost-based), Goetz Graefe (Volcano, Columbia), Thomas Neumann (HyPer, optimizare vectorială), Matei Zaharia (Spark Catalyst).

### 7.3.5 Forumuri Importante

- Conferințe – ACM SIGMOD, VLDB, ICDE, CIDR (idei emergente).
- Reviste – ACM TODS, VLDB Journal.
- Workshop-uri – *QUESTA*, *Ai4DB* (optimizatori «learned»).

### 7.3.6 Dimensiunea locală și dimensiunea globală

- Local:

- Curs „Optimizarea interogărilor” (UVT, MSc) – proiecte cu EXPLAIN ANALYZE și tuning pe PostgreSQL.
- Proiect „QOptVest” (2025–2027) – optimizator ML pentru date bancare, colaborare UVT + Banca Transilvania.
- Hackathon „Query Challenge” – task: reducerea timpului TPC-H Q13 sub 50 ms pe hardware local.

- Global:

- Contribuții open-source la PostgreSQL și DuckDB (module de statistică extinsă).

- Participare la standardizarea ISO SQL/PGQ (proceduri de optimizare pentru grafuri).
- Colaborări de cercetare cu TU Munich (HyPer) și UC Berkeley (StarSchema Benchmark).

## 7.4 Controlul concurenței și managementul tranzacțiilor

Subdomeniul descrie regulile și mecanismele prin care un sistem de baze de date se asigură că mai multe operații simultane (tranzacții) nu se influențează greșit între ele. Ideea-cheie este principiul ACID, formulat la sfârșitul anilor '70, care garantează că fiecare tranzacție se comportă „ca și cum” ar rula singură. De la primele scheme cu blocări (System R, 1979) s-a evoluat spre tehnici non-blocante (MVCC), protocole distribuite (two-phase commit) și izolare configurabilă (snapshot isolation).

### 7.4.1 Activitățile principale

1. Teorie:
  - (a) Cadru conceptual
    - ACID – atomaritate, coherență, izolare, durabilitate (asigură integritatea).
    - Serializabilitate – model teoretic în care rezultatul concurrent este echivalent cu o ordine secvențială.
  - (b) Cadre
    - Blocare în doi pași (2PL) – tranzacția își ia toate blocările înainte de a le elibera.
    - Control optimist (OCC) – se rulează fără blocări, se verifică la final dacă au apărut conflicte.
    - Multiversion concurrency control (MVCC) – fiecare scriere creează o versiune; citirile văd un instantaneu consistent.
  - (c) Metode de raționament
    - Grafuri de precedență (depistează cicluri ce încalcă serializabilitatea).
    - Analiza protocolului de commit (ex. corectitudinea 2-PC se dovedește cu diagrama stărilor).
  - (d) Întrebări fundamentale
    - Cum se previn anomaliiile „lost update” și „dirty read”?
    - De ce 2PL garantează serializabilitate, iar snapshot isolation nu?

- Care este limita scalării atunci când se utilizează locking versus MVCC?

2. Experiment:

- Explorarea modelelor – comparația clasică 2PL vs. OCC pe TPC-C (Berkeley, 2008) arată că OCC oferă throughput mai mare când rata de conflict este sub 10
- Implementări și validare – testele Jepsen 2020 pe CockroachDB confirmă respectarea serializabilității după remedierea bug-ului de split-brain identificat în versiunea 19.2.
- Exemplu practic – în PostgreSQL, trecerea de la „read committed” la „repeatable read” elimină anomalia phantom, dar crește durata medie a blocării de la 3 ms la 12 ms într-un micro-benchmark de conturi bancare.

3. Design:

- Produse – PostgreSQL (MVCC), Google Spanner (clock-SI distribuit), FoundationDB (serializabilitate strictă cu OCC).
- Instrumente – Jepsen (injectează partiții de rețea), TPC-C (stres tranzacțional), Perfetto (urmărește latențe în commit).
- Tehnici – journaling ARIES, replicate state machine, determinism (Calvin), adaptarea izolării la conflict (adaptive CC).
- Impact – timpi de nefuncționare sub un minut după failover și coerentă globală pe centre de date aflate pe continente diferite.

4. Extra:

- Cunoștințe necesare
  - Baze de date (SQL, modele de izolare)
  - Structuri de date concurente (liste blocate, skip-list MVCC)
  - Sisteme de operare (primitive de sincronizare, journal)
  - Rețelistică distribuită (consens Paxos/Raft)
- Literatură de specialitate
  - Gray & Reuter – *Transaction Processing* (1993) [74]
  - Bernstein, Hadzilacos, Goodman – *Concurrency Control and Recovery* (1987)
  - Bailis et al. – *Highly Available Transactions* (2014) [75]
  - Adya – teză MIT *Weak Consistency: A Generalized Theory* (1999)

#### **7.4.2 Relațiile cu celelalte subdomenii**

- Algoritmi și structuri de date (protocolul de blocare 2-PL, grafurile de precedență și structurile multiversion—skip-list, B-tree MVCC—sunt aplicații directe ale algoritmicii concurente).
- Sisteme de operare și rețele (primitivele de sincronizare, mmap, jurnalul pe disc și protocoalele de commit peste TCP determină performanța și corectitudinea tranzacțiilor).
- Arhitectură (modelele de memorie la nivel CPU și coerentă cache stabilesc limitele scalării; hardware transactional memory extinde controlul concurenței la nivel de procesor).
- Limbaje de programare (construcțe precum `atomic`, `async/await` sau „software transactional memory” oferă dezvoltatorilor acces direct la izolarea tranzacțională).
- Inginerie software (modelele de „unit of work”, „saga” și „event sourcing” folosesc tranzacțiile pentru a asigura consistența microserviciilor).
- Informatică organizațională (ERP și sistemele bancare depind de tranzacții ACID pentru a proteja datele financiare în medii cu mii de utilizatori concurenți).

#### **7.4.3 Probleme importante și probleme deschise**

- Serializabilitate economică la scară globală (coordonare minimă).
- Eliminarea blocajelor (deadlock-free) fără sacrificarea debitului.
- Combinarea izolării puternice cu latență sub 10 ms pe WAN.
- Reguli automate de adaptare a izolării în funcție de conflict.

#### **7.4.4 Persoane importante**

- Jim Gray – log ARIES, principii ACID în practică.
- Andreas Reuter – teorii de recuperare.
- Pat Helland – 2PC în sisteme de mesaje.
- Peter Bailis – tranzacții cu disponibilitate înaltă. [75]
- Daniel Abadi – Calvin, determinism concurrent. [76]

#### **7.4.5 Forumuri importante**

- Conferințe – SIGMOD, VLDB, ICDE, SOSP.
- Workshop-uri – HotStorage, LADIS.
- Jurnale – ACM TODS, VLDB J.

#### **7.4.6 Dimensiunea locală și dimensiunea globală**

- Local:

- Curs „Sisteme distribuite” (UVT) – laborator cu două-phase commit și testare Jepsen.
- Proiect „TransSafe” (2025–2027) – analiză de consistență pentru fintech locale (UVT + Banca Transilvania).
- Colaborare cu Continental Timișoara – prototip OCC pe date senzor auto.

- Global:

- Contribuții la PostgreSQL (serializable snapshot isolation).
- Participare în Cloud Native Computing Foundation pentru standarde de observabilitate a tranzacțiilor.
- Schimb de cercetători cu ETH Zürich și MIT-PDOS pe teme de determinism concurent.

### **7.5 Recuperarea și integritatea bazelor de date**

Se ocupă de menținerea corectitudinii și refacerii datelor după erori de software, căderi de curent sau defecte hardware. Își are rădăcinile în anii ’70, când s-a introdus jurnalizarea write-ahead, și a evoluat prin algoritmul ARIES (1992) și prin mecanisme moderne de verificare a integrității (checksum, scrubbing online).

#### **7.5.1 Activitățile principale**

1. Teorie:

- (a) cadru conceptual
  - durabilitate (D din ACID: datele confirmate nu se pierd)
  - integritate logică (chei primare, strâine, constrângeri CHECK)
  - jurnalizare (log) și puncte de control (checkpoint)

- (b) cadre
  - write-ahead logging (WAL: logul se scrie pe disc înaintea paginii modificate)
  - shadow paging (copie de pagini, evită logul)
  - restricții declarative și declanșatoare (trigger-uri)
- (c) metode de raționament
  - dovada idempotentă a redo/undo din ARIES
  - verificare formală a integrității referențiale cu logica relațională
- (d) întrebări fundamentale
  - cum garantează WAL că tranzacțiile par atomice după un crash?
  - de ce o cheie străină împiedică introducerea de „date orfane”?
  - cât timp durează refacerea după defect de disc și cum se minimizează?

## 2. Experiment:

- (a) explorarea modelelor – comparație publicată ARIES vs. shadow paging (VLDB 1993) pe un jurnal de 1 GB, unde ARIES recuperează în 17 s iar shadow paging în 45 s
- (b) implementări și validare – testele Jepsen 2022 pe MySQL InnoDB au injectat opriri de curent; la repornire, niciun update confirmat nu s-a pierdut, însă s-au observat 0,5
- (c) exemplu practic – simularea unui crash pe PostgreSQL 16 cu comenzi `pg_bench` arată că punctele de control la 30 s limitează timpul de recovery sub 8 s la 100 K tranzacții/minut

## 3. Design:

- (a) produse – PostgreSQL (WAL + checksums), Oracle Data Guard (redo shipping), Amazon Aurora (redo log pe SSD partajat)
- (b) instrumente – `pg_dump` (backup logic), `wal_g` (backup incremental pe cloud), `db_verify` (detectare corupție)
- (c) tehnici – point-in-time recovery, erasure coding, online scrub care recitește periodic datele și logul
- (d) impact – pierdere de date sub 1 s în centre cloud cu replicare și restaurări de terabiti în mai puțin de o oră prin backup diferențial

## 4. Extra:

- (a) Cunoștințe necesare
  - concepte ACID și modele de consistență
  - organizarea pe disc (pagini, blocuri, jurnale)
  - sisteme de fișiere și comenzi de backup
- (b) Literatură de specialitate
  - Gray, J.; Reuter, A. — *Transaction Processing: Concepts and Techniques*, 1993 [74]
  - Mohan, C.; Haderle, D.; Lindsay, B.; et al. — “ARIES: A Transaction Recovery Method”, 1992 [77]
  - Stonebraker, M.; Wong, E.; Kreps, P.; Held, G. — “Implementation of Integrity Constraints and Views by Query Modification”, 1975
  - PostgreSQL Global Development Group — “PostgreSQL Documentation (Backup & Restore; Reliability)”, 2024

### 7.5.2 Relațiile cu celelalte subdomenii

- sisteme de operare și rețele – depinde de semantica `fsync` și de replicarea la distanță
- algoritmi și structuri de date – logul este o listă legată pe disc, iar verificarea checksum folosește funcții hash
- arhitectură – disponibilitatea NVRAM reduce latența jurnalizării
- inginerie software – strategiile DevOps de backup continuu și teste de chaos engineering
- informatică organizațională – reglementări GDPR/BCP cer planuri de recuperare documentate

### 7.5.3 Probleme importante și probleme deschise

- refacere instantanee pe baze de date de zeci de terabiți fără oprirea traficului
- detectarea pro-activă a corupției cauzate de hardware (silent data corruption)
- protecția împotriva ransomware prin backup imutabil și restaurare rapidă

#### **7.5.4 Persoane importante**

1. Jim Gray – pionier ACID și „careful restart”
2. C. Mohan – autorul ARIES [77]
3. Michael Stonebraker – mecanisme de integritate în Ingres și Postgres [46]
4. Markus Pflüger – scrubbing online la Google Spanner

#### **7.5.5 Forumuri importante**

- conferințe – SIGMOD, VLDB, USENIX FAST
- workshop – HotStorage (idei timpurii de fiabilitate)
- jurnale – ACM TODS, VLDB Journal

#### **7.5.6 Dimensiunea locală și dimensiunea globală**

- **Local:**

- curs „Administrarea bazelor de date” (UVT) – laboratoare cu backup și point-in-time recovery
- proiect „SafeDB Vest” (2024–2026) – evaluarea integrității pe SSD-uri QLC, în parteneriat cu Atos Timișoara

- **Global:**

- contribuții la utilitarul `wal_g` și la codul de checksum PostgreSQL
- colaborare cu ETH Zürich pe metodologii de recovery instant folosind NVRAM
- participare la grupul ISO SQL/Part 15 „Integrity and recovery facilities”

### **7.6 Securitate și confidențialitate în bazele de date**

Studiază metodele prin care datele sunt protejate împotriva accesului neautorizat și prin care se păstrează caracterul privat al informațiilor sensibile. De la primele comenzi SQL GRANT/REVOKE (anii '80) s-a trecut la control pe roluri (RBAC), criptare la nivel de disc (TDE) și tehnici moderne de anonimizare (differential privacy, homomorphic encryption).

### 7.6.1 Activitățile principale

1. Teorie:
  - (a) cadru conceptual
    - triada confidențialitate–integritate–disponibilitate (CIA)
    - autentificare, autorizare, audit (AAA)
    - control de acces: discret (DAC), obligatoriu (MAC), bazat pe rol (RBAC)
    - criptare la repaus și în tranzit, hashing pentru parole
  - (b) cadre
    - model Bell–LaPadula (confidențialitate pe nivele)
    - polită de etichetare a informației (information-flow control)
    - difuzoare de zgomot pentru privacy diferențială
  - (c) metode de rationament
    - analiză formală a fluxurilor de informație (dovezi că datele sensibile nu ies pe canale publice)
    - threat-modeling STRIDE și evaluarea riscurilor prin matrice DREAD
  - (d) întrebări fundamentale
    - de ce un hash simplu nu este suficient pentru stocarea parolelor?
    - cum limitează securitatea la nivel de rând accesul între chiriași (multitenancy)?
    - la ce volum de date devine fezabilă criptarea completă a coloanelor fără impact semnificativ de performanță?
2. Experiment:
  - (a) explorarea modelelor – studiu 2021 PostgreSQL TDE vs. MySQL InnoDB TDE: penalizare medie 4–6 % la citire, 8 % la scriere pe SSD NVMe de 1 TB
  - (b) implementări și validare – testele Jepsen (2023) pe CockroachDB au injectat atacuri de privilegii; verificarea audit-log confirmă negarea accesului ilicit în 100 % cazuri
  - (c) exemplu practic – activarea „row level security” și criptarea coloanei „card\_nr” în PostgreSQL reduce viteza unei interogări cu agregare de la 30 ms la 45 ms (1,5×) dar elimină expunerea numerelor de card
3. Design:

- (a) produse – Oracle Transparent Data Encryption; Microsoft Always Encrypted; Google BigQuery differential privacy
  - (b) instrumente – sqlmap (test penetrabilitate), pgaudit (log detaliat), HashiCorp Vault (gestionare chei)
  - (c) tehnici – tokenizare, mascare dinamică, policies-as-code (ex. Open Policy Agent), căutare peste date criptate (searchable encryption)
  - (d) impact – reducerea breşelor de date, conformitate cu GDPR/PCI-DSS, încredere sporită a utilizatorilor
4. Extra:
- (a) Cunoştinţe necesare
    - bazele criptografiei simetrice şi asimetrice
    - sintaxa GRANT/REVOKE, politicile row/column-level
    - reţelistică sigură (TLS, VPN)
  - (b) Literatură de specialitate
    - Sandhu, R., Coyne, E. J., Feinstein, H. L., & Youman, C. E. – „Role-Based Access Control Models”, *IEEE Computer*, vol. 29, nr. 2, pp. 38–47, 1996. [78]
    - Popa, R. A., Redfield, C. M. S., Zeldovich, N., & Balakrishnan, H. – „CryptDB: Processing Queries on an Encrypted Database”, *Communications of the ACM*, vol. 55, nr. 9, pp. 103–111, 2012. [79]
    - Johnson, N., Near, J. P., & Song, D. – „Towards Practical Differential Privacy for SQL Queries”, *Proceedings of the VLDB Endowment*, vol. 11, nr. 5, pp. 526–539, 2018. [80]
    - Schneier, B. – *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed., Wiley, 1996. (Capitolul 24: Protecţia criptografică a bazelor de date) [81]

### **7.6.2 Relaţiile cu celelalte subdomenii**

- algoritmi şi structuri de date – implementarea criptării şi a indexurilor sigure necesită funcţii hash şi arbori verificabili
- sisteme de operare şi reţele – securitatea kernelului şi TLS protejează fişierele şi traficul de replicare
- limbi de programare – ORM-urile trebuie să transmită parametri securizaţi şi să evite injectia SQL

- inginerie software – DevSecOps integrează teste de penetrare și scanarea codului în pipeline-urile CI/CD
- IA & robotică – detectarea anomalieiilor în loguri cu modele ML ajută la descoperirea acceselor suspecte
- informatică organizațională – politicile de confidențialitate influențează desigurarea rolurilor și a jurnalelor de audit

#### **7.6.3 Probleme importante și probleme deschise**

- execuția interogărilor direct pe date criptate cu latență sub 100 ms
- asigurarea confidențialității în baze de date coloane-plus-vector (AI embeddings)
- prevenirea scurgerilor prin canale colaterale (timing, cache) în medii cloud multi-tenant

#### **7.6.4 Persoane importante**

1. Ravi Sandhu – definițor al RBAC [78]
2. Shafi Goldwasser – pionier al criptografiei probabilistice și a conceptului de confidențialitate diferențială
3. Hakan Hacigümüş – căutare pe date criptate și sistemul CryptDB [82]
4. Dan Boneh – schemă de criptare deterministică folosită în baze de date comerciale

#### **7.6.5 Forumuri importante**

- conferințe – IEEE S&P, ACM CCS, DBSec, SIGMOD Workshop on DBSecurity
- jurnale – ACM Transactions on Privacy and Security, VLDB Journal (secțiunea Data Security)

#### **7.6.6 Dimensiunea locală și dimensiunea globală**

- **Local:**

- curs „Securitatea bazelor de date” (UVT) – laborator cu criptare TDE și politici row-level

- proiect „DataShield Vest” (2025–2028) – prototip de mascare în timp real pentru date medicale (UVT + Spitalul Județean Timișoara)

- **Global:**

- contribuții la modulul pgcrypto și la standardul ISO/IEC 18033-6 (tokenisation)
- colaborare cu TU Darmstadt pe private-SQL și cu Google Confidential Computing Group pe execuție în enclavă

## 7.7 Big Data și baze de date distribuite

Se ocupă cu stocarea și procesarea unor volume de date ce depășesc capacitatele unui singur server, folosind clustere scalabile pe orizontală. A pornit de la fișierul distribuit Google GFS (2003) și modelul MapReduce (2004), a continuat cu Dynamo (2007) și Bigtable (2006) și a ajuns la sisteme „planet-scale” precum Spanner și Snowflake.

### 7.7.1 Activitățile principale

1. Teorie:

- (a) cadru conceptual
  - CAP (consistență-disponibilitate-toleranță la partiziție)
  - partiziționare pe shard-uri, replicare și re-echilibrare automată
  - modele de consistență: eventuală, snapshot, serializabilitate globală
- (b) cadre
  - arhitecturi shared-nothing (HDFS, Cassandra)
  - procesare lot MapReduce vs. streaming continuu (Kafka Streams, Flink)
  - paradigme NewSQL (Spanner, CockroachDB) pentru tranzacții distribuite
- (c) metode de raționament
  - analize de cost pe I/O și trafic de rețea (model „rule of thumb”  $10 \times$  rețea vs. disc)
  - dovezi de corectitudine pentru Paxos/Raft (consens) și pentru algoritmi de re-partiziționare online
- (d) întrebări fundamentale
  - Ce se sacrifică pentru a obține disponibilitate în prezența partiziilor?

- Cum se menține consistența pe mai multe regiuni fără latențe mari?
- De ce schema columnară reduce costul scanării analitice pe zeci de terabiți?

2. Experiment:

- explorarea modelelor – raportul original Google Bigtable demonstrează 500 MB/s scriere și latență sub 6 ms pe clustere cu 1000 de noduri
- implementări și validare – benchmark YCSB 2020: Cassandra depășește 1,2 M operații/sec cu consistență „quorum”, CockroachDB 0,45 M oper./sec cu serializabilitate; latență p99 este totuși mai mică la Cockroach (20 ms vs. 45 ms)
- exemplu practic – sortarea 10 TB (TeraSort): Spark 3 pe 32 noduri NVMe termină în 18 minute, în timp ce același job pe Hadoop classic durează 45 minute

3. Design:

- produse – Apache Hadoop, Apache Cassandra, Google Spanner, Snowflake, Apache Iceberg
- instrumente – YCSB (workload key-value), TPCx-BB (benchmark analitic big data), Jepsen (test consistență)
- tehnici – compresie pe coloane + coduri erasure, re-echilibrare online, query-pushdown pe worker, format lakehouse (Parquet + Delta/Apache Iceberg)
- impact – interogări interactive pe petabytes, scalare liniară la adăugarea nodurilor și reducerea costurilor prin stocare separată de compute

4. Extra:

- Cunoștințe necesare
  - programare distribuită (map, reduce, actor model)
  - concepte de rețea (latență, Throughput TCP)
  - baze de date NoSQL/NewSQL și sisteme de fișiere distribuite
- Literatură de specialitate
  - Dean & Ghemawat – „MapReduce: Simplified Data Processing on Large Clusters” (2004) [71]
  - Brewer – „Towards Robust Distributed Systems” (PODC Keynote, 2000) [83]

- Lakshman & Malik – „Cassandra: A Decentralized Structured Storage System” (2009) [84]
- Corbett et al. – „Spanner: Google’s Globally-Distributed Database” (2012) [85]
- Kleppmann – „Designing Data-Intensive Applications”, cap. 2 și 9 [86]

### 7.7.2 Relațiile cu celelalte subdomenii

- algoritmi și structuri de date – hashing consistent, arbori log-structured, algoritmi de consens
- sisteme de operare și rețele – scheduling la nivel de cluster, RPC, balansare de încărcare
- arhitectură – exploatarea SSD-urilor NVMe și a magistralelor RDMA pentru throughput ridicat
- inginerie software – practicile DevOps gestionează clustere de sute de noduri cu CI/CD continuu
- știința computațională – simulări climatice și genomice folosesc HDFS și Spark pentru a procesa petabytes
- IA & robotică – modelele de antrenare distribuie datele pe clustere (ex. pipelines de feature store)

### 7.7.3 Probleme importante și probleme deschise

- tranzacții ACID pe tabelă distribuită fără cost de latență prohibitiv
- gestionarea datelor calde vs. reci cu mutare automată între stocări heterogene
- guvernanța și reproducibilitatea datelor în lacuri partajate de echipe multiple

### 7.7.4 Persoane importante

1. Jeff Dean & Sanjay Ghemawat – creatori GFS, MapReduce, Bigtable
2. Eric Brewer – formularea teoremei CAP
3. Avinash Lakshman – Dynamo și Cassandra
4. Peter Bailis – consistență și disponibilitate la scară mare
5. Ion Stoica – Apache Spark și Ray

### 7.7.5 Forumuri importante

- conferințe – SIGMOD, VLDB, NSDI, SOSP, IEEE BigData
- workshop-uri – CIDR, HotStorage, EuroSys BigSys
- jurnale – VLDB Journal, ACM TODS

### 7.7.6 Dimensiunea locală și dimensiunea globală

- Local:

- laborator „Big Data” (UVT) – clustere Hadoop/Spark de 16 noduri pentru cursuri și disertații
- proiect „SmartCity-DataVest” (2024–2027) – analiză fluxuri IoT urbane pe Apache Flink (UVT + primăria Timișoara)
- hackathon „Big Data Vest” – provocare: procesarea unui set de 1 TB logs în sub 10 minute pe cluster universitar

- Global:

- contribuții la Apache Flink (conector Kafka) și Cassandra (noul gossiping)
- participare în GAIA-X și în CNCF Data on Kubernetes TAG
- colaborări cu TU Munich (lagoDB) și UC Berkeley (SkyhookDM)

## 7.8 Depozitarea datelor și analiza

Se concentrează pe colectarea, modelarea și interogarea unor volume mari de date istorice pentru raportare și decizie. Primele depozite (Bill Inmon, 1990) foloseau scheme în stea și cuburi OLAP. Evoluția a dus la colonar (C-Store, 2005), la sisteme cloud elastice (Redshift, BigQuery, Snowflake) și la „lakehouse”, unde formate tip Parquet/Delta combină lacuri de date cu structuri de depozit.

### 7.8.1 Activitățile principale

1. Teorie:

- (a) cadru conceptual
  - ETL/ELT (extract–transform–load) și actualizare incrementală
  - model în stea și fulg (tablou de fapte + dimensiuni)
  - agregări OLAP (drill-down / roll-up, slice-and-dice)
- (b) cadre

- depozit „enterprise” (Inmon) vs. depozit „datamart” (Kimball)
  - colonar vs. row-store; lakehouse (Delta, Iceberg)
- (c) metode de raționament
- calcule de selectivitate pentru filtrarea pe coloane
  - cost-model TPC-DS (I/O secvențial, compresie, latență rețea)
- (d) întrebări fundamentale
- De ce schemele în stă accelerează interogările agregate?
  - Cum reduc coloanele compresia și traficul?
  - Când este utilă materializarea unui cub agregat?

## 2. Experiment:

- explorarea modelelor – C-Store (2005) arată de  $8\times$  mai puține citiri decât depozitul row-store pe același TPC-H
- implementări și validare – test TPC-DS 10 TB (2023): Snowflake termină query 64 în 18 s, BigQuery în 22 s, Redshift în 45 s; diferența provine din cache-ul colonar și algoritmul de partajare de date calde
- exemplu practic – adăugarea partiționării pe col. `order_date` și a formatului Parquet reduce un job Spark de agregare de la 7 min la 55 s ( $8\times$ ) pe 1 TB log-uri clickstream

## 3. Design:

- produse – Snowflake (lakehouse elastic), Google BigQuery, Amazon Redshift Spectrum, ClickHouse (OLAP în memorie)
- instrumente – dbt (transformări SQL ca-cod), Airflow (orchestrare ETL), Data Build Validator, Superset (vizualizare)
- tehnici – coloane + dicționar de compresie, materialized views refresh la minut, clustering automat, query-pushdown în storage
- impact – rapoarte interactive sub 5 s pe zeci de miliarde de rânduri, cost redus prin stocare comprimată și compute on-demand

## 4. Extra:

- Cunoștințe necesare
  - SQL analitic (GROUP BY CUBE, window functions)
  - modele de date dimensionale și noțiuni de ETL
  - procesare paralelă (MapReduce, Spark)

(b) Literatură de specialitate

- Kimball & Ross – *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* (2013) [87]
- Abadi, Madden & Hachem – „Column-Stores vs. Row-Stores: How Different Are They Really?” (SIGMOD 2008) [76]
- Stonebraker et al. – „C-Store: A Column-oriented DBMS” (VLDB 2005) [46]
- Armbrust, Ghodsi, Xin & Zaharia – „Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics” (CIDR 2021) [88]

### 7.8.2 Relațiile cu celelalte subdomenii

- algoritmi și structuri de date – arbori de partitie, bitmap indexuri, algoritmi de agregare paralelă
- sisteme de operare și rețele – throughput I/O secvențial și bandwidth rețea influențează timpii ETL
- inginerie software – pipeline-uri CI/CD pentru transformări dbt și testare de calitate a datelor
- știința computațională – analiză statistică și machine-learning rulează direct peste depozite colonare
- IA & robotică – feature stores pentru modele ML sunt construite pe aceleasi mecanisme de depozitare
- informatică organizațională – BI și dashboard-urile executive consumă date din depozit pentru decizii zilnice

### 7.8.3 Probleme importante și probleme deschise

- integrarea datelor în timp real fără compromiterea istoricului (lambda-/kappa-architecture)
- guvernanță și catalogarea automată în lacuri de date mixte (schema-on-read)
- reducerea costului de stocare la exaocetei cu păstrarea performanței interactive

#### **7.8.4 Persoane importante**

1. Bill Inmon – „părintele” data warehouse
2. Ralph Kimball – modele dimensionale și datamarts
3. Michael Stonebraker – C-Store și Vertica
4. Matei Zaharia – Apache Spark SQL & Delta Lake
5. Jordan Tigani – BigQuery design și optimizare

#### **7.8.5 Forumuri importante**

- conferințe – SIGMOD, VLDB, CIDR, Strata Data, Gartner DA
- jurnale – VLDB Journal, ACM TODS (secțiunea Analytics)

#### **7.8.6 Dimensiunea locală și dimensiunea globală**

- **Local:**

- curs „Depozite de date și BI” (UVT) – workshop cu Snowflake și dbt
- proiect „Retail-Analytics Vest” (2023–2026) – data lake Delta + Spark pentru lanțuri de magazine regionale

- **Global:**

- contribuții la Apache Iceberg și dbt-core
- parteneriat academic cu TU Munich pe optimizare Hyrise-Lakehouse
- implicare în Open Metadata și OpenLineage pentru standarde de catalog de date

### **7.9 Reprezentări avansate ale datelor și mașini virtuale**

Vizează modelele de date care depășesc tabelul relațional (grafuri, spațiale, temporale, documente JSON, obiecte) și mecanismele de execuție pe mașini virtuale dedicate (interpretori de byte-code pentru interogări, VM Java/LLVM în nuclee de baze de date). Primele extinderi (Postgres, 1986) au adăugat tipuri spațiale; mașina virtuală SQLite (2000) a popularizat byte-code pluggable; astăzi, sisteme precum Neo4j (graf) și DuckDB (vectorized VM) ilustrează evoluția.

### 7.9.1 Activitățile principale

1. Teorie:

- (a) cadru conceptual
  - tipuri complexe (array, JSON, XML, GIS, timp)
  - modele graf (nod–muchie–proprietăți) și obiect (OODB)
  - mașină virtuală de interogare (byte-code + registru)
- (b) cadre
  - extensii SQL/MM (multimedia) și SQL/PGQ (grafe)
  - VM vectorizată (HyPer, DuckDB) vs. VM pe tuplu (SQLite)
- (c) metode de raționament
  - transformarea expresiilor SQL graf în operații de pattern-matching (Cypher-to-algebră)
  - analiza costului pentru operatori vectorizați (procesare pe lot vs. tuplu-cu-tuplu)
- (d) întrebări fundamentale
  - De ce grafurile nu se mapează eficient pe tabelă relațională simplă?
  - Când este rentabil să vectorizezi execuția în loc să interpretezi tuplu-cu-tuplu?
  - Cum se păstrează ordinea temporală fără a duplica date?

2. Experiment:

- (a) explorarea modelelor – studiu LDBC (2022): Neo4j atinge 3,2 k traversări/s, iar PostgreSQL cu PGGraph 0,9 k; diferența provine din stocarea nod-centrică
- (b) implementări și validare – test DuckDB vs. SQLite pe TPC-H 100 GB: DuckDB (VM vectorizată) rulează Q6 în 0,11 s, SQLite (byte-code tuplu) 1,4 s
- (c) exemplu practic – adăugarea indexului R-tree la poligoane GIS în PostGIS reduce căutarea „poligon conține punct” de la 450 ms la 25 ms

3. Design:

- (a) produse – Neo4j (graf), MongoDB (JSON), PostGIS (spațial), DuckDB/Hyper VM vectorizată
- (b) instrumente – LDBC SNB (benchmark graf), GeoBench, SQLite `explain` (byte-code)

- (c) tehnici – compresie coloană + dicționar pentru JSON, R-tree/GiST pentru GIS, compilare JIT cu LLVM pentru operatori personalizați
- (d) impact – interogări graf sub 10 ms, analitice JSON 4× mai rapide, suport nativ pentru date multimedia/temporale în aplicații moderne

4. Extra:

- (a) Cunoștințe necesare
  - modele de date semi-structurate și graf
  - noțiuni GIS, temporal și multimedia
  - programare C/C++ sau Rust (scriere de extensii, JIT)
- (b) Literatură de specialitate
  - Stonebraker – *Object-Relational DBMSs: The Next Great Wave* (1996)
  - Guttman – „R-Trees: A Dynamic Index Structure for Spatial Searching” (1984) [70]
  - Boncz, Zukowski & Nes – „MonetDB/X100: Hyper-Pipelining Query Execution” (CIDR 2005) [89]
  - Angles & Gutierrez – „Survey of Graph Database Models” (ACM Computing Surveys, 2008) [90]

### 7.9.2 Relațiile cu celelalte subdomenii

- algoritmi și structuri de date – R-tree, B-tree GiST, graf traversal BFS/DFS
- arhitectură – SIMD și cache liniarizează operatorii vectorizați
- sisteme de operare și rețele – extensiile VM pot folosi mmap și JIT cu protecții DEP
- IA & robotică – grafuri de cunoștințe și scene spațiale pentru planificarea robotilor
- interacțiune om-computer – vizualizarea grafurilor și a hărților GIS depinde de acces rapid

### 7.9.3 Probleme importante și probleme deschise

- suport tranzacțional ACID eficient pentru grafuri masive
- combinarea vectorizării cu execuție GPU în mașina virtuală
- reprezentare compactă pentru date temporal-versiune + graf (bi-temporal property graph)

#### **7.9.4 Persoane importante**

1. Michael Stonebraker – obiect-relațional și extensii tip PostGIS
2. Thomas Neumann – HyPer și VM vectorizată/JIT
3. Renee Miller – transformări și integrare XML/JSON avansate
4. Emil Eifrem – co-fondator Neo4j și promotor al grafurilor la scară industrială

#### **7.9.5 Forumuri importante**

- conferințe – SIGMOD, VLDB, ICDE, SSTD (spațial), GRADES (graf)
- jurnale – VLDB Journal, ACM TODS (secțiunea Data Models)

#### **7.9.6 Dimensiunea locală și dimensiunea globală**

- Local:

- curs „Modelare graf și spațială” (UVT) – laborator Neo4j și PostGIS
- proiect „RouteGraph Vest” (2025–2028) – optimizare rute transport Timișoara pe graf property + GIS

- Global:

- contribuții la Neo4j APOC și la DuckDB spatial extension
- colaborări cu Tsinghua Graph Computing Lab și TU Munich (HyPer)

### **7.10 Integrarea datelor eterogene și multimedia**

Tratează conectarea și unificarea datelor provenite din surse diferite (SQL, NoSQL, fișiere, API-uri, stream-uri audio-video) într-o vedere coerentă, ușor de interogat. Primele sisteme (TSIMMIS, 1995) au propus „wrapper–mediator”, urmate de XML/RDF pentru schimbare de schemă și, mai recent, de pipeline-uri graf (NiFi, Airbyte) și lacuri multimedia (MPEG-7, Deep Feature Stores).

#### **7.10.1 Activitățile principale**

1. Teorie:
  - (a) cadru conceptual
    - mapare schemă–schemă (schema mapping)
    - extragere–transformare–încărcare (ETL) și varianta ELT

- date multimedia: metadate (MPEG-7) + caracteristici de conținut (vectori imagine, spectru audio)

(b) cadre

- wrapper–mediator (înfășoară sursa + limbaj intermedier)
- integrare pe grafo–triplu RDF (ontologii, Linked Data)
- pipeline declarativ (NiFi/Apache Beam) pentru fluxuri video/audio

(c) metode de raționament

- potrivire de schemă (string similarity, machine-learning)
- rezolvare entități (duplicate entities resolution)
- aliniere ontologii (logica descriptivă)

(d) întrebări fundamentale

- Cum se unifică două tabele cu coloane denumite diferit dar cu același sens?
- De ce este nevoie de „late binding” (ELT) în lucrările de date?
- Cum se caută „imagini similare” fără etichete text exacte?

## 2. Experiment:

- explorarea modelelor – comparație 2022: FedBench arată că mediatorul Ontop + RDF răspunde la 120 QPS, iar Drill (SQL-on-files) la 250 QPS, dar fără semantica OWL
- implementări și validare – benchmark LUBM: integrarea 100 GB RDF în Apache Jena Fuseki cu 3 noduri livrează interogări în p95 = 90 ms; cu mapping RML, p95 scade la 65 ms
- exemplu practic – pipeline NiFi care unește fișiere CSV vânzări + imagini produs: extragerea histogramelor de culoare reduce timpul de căutare „produs asemănător” de la 8 s la 0,7 s după indexare în Elasticsearch

## 3. Design:

- produse – Apache NiFi (integrare fluxuri), Talend, Airbyte, Elastic Search + ingest-pipeline, Google DataFusion
- instrumente – OpenRefine (curățare), De-Dup (rezolvare entități), DBpedia Spotlight (linking semantic)
- tehnici – transformări declarative (RML), vector-search multimedia, cache rezultat inter-sursă, data-catalog automat (OpenMetadata)
- impact – reducere timp integrare de la luni la zile, căutare multimodală (text + imagine) sub 1 s, bază unică de adevăr pentru BI și AI

4. Extra:

- (a) Cunoștințe necesare
  - formate date (JSON, XML, AVRO, Parquet)
  - bazele RDF/SPARQL și ontologii OWL
  - noțiuni de procesare imagine/audio și vector-search
- (b) Literatură de specialitate
  - Garcia-Molina et al. – „The TSIMMIS Approach to Mediation: Data Models and Languages” (1997) [91]
  - Halevy, Rajaraman & Ordille – „Data Integration: The Teenage Years” (VLDB 2006) [92]
  - Doan, Halevy & Ives – *Principles of Data Integration* (2012) [93]
  - Lew, Sebe, Djeraba & Jain – „Content-Based Multimedia Information Retrieval: State of the Art and Challenges” (ACM TOMCCAP 2006) [94]

#### 7.10.2 Relațiile cu celelalte subdomenii

- algoritmi și structuri de date – filtre Locality-Sensitive Hashing pentru căutare multimedia
- limbaje de programare – DSL-uri ETL bazate pe Python/SQL
- sisteme de operare și rețele – debite mari de fișiere media, protocol S3/GCS
- inginerie software – orchestrare CI/CD a pipeline-urilor de date (Airflow)
- IA & robotică – modele de viziune și NLP se antrenează pe date integrate
- interacțiune om-computer – dashboard-uri care combină grafice numerice cu previzualizări media

#### 7.10.3 Probleme importante și probleme deschise

- potrivire automată de schemă cu acuratețe  $> 95\%$  fără etichete manuale
- stocarea vectorilor de caracteristici multimedia la scară petabyte și căutare sub-secundă
- guvernanță semantică a datelor (tracking provenance + respect GDPR)

#### **7.10.4 Persoane importante**

1. Alon Halevy – mediere semnatică și Google Fusion Tables [92]
2. AnHai Doan – potrivire de schemă asistată de ML [93]
3. Hector Garcia-Molina – proiectele TSIMMIS și ACME [91]
4. Ramesh Jain – pionier căutare multimedia bazată pe conținut [95]

#### **7.10.5 Forumuri importante**

- conferințe – SIGMOD, VLDB, ICDE, ACM MM (multimedia), ISWC (Web semnatic)
- jurnale – VLDB J., ACM TOMM (multimedia)

#### **7.10.6 Dimensiunea locală și dimensiunea globală**

- **Local:**

- curs „Integrare de date” (UVT) – proiect practică NiFi + OpenRefine
- proiect „MediVest” (2024–2027) – integrare imagini CT + fișiere HL7 pentru diagnostic asistat (UVT + Spital Județean)

- **Global:**

- contribuții la Apache Arrow Flight și la proiectul OpenMetadata
- colaborare cu Universitatea Wisconsin-Madison (laborator Data Integration @ Doan) și cu EBU (European Broadcasting Union) pe standarde multimedia RDF

## 8 AI și Robotica

Inteligenta Artificiala (AI) reprezinta domeniul informaticii care se ocupă cu dezvoltarea de algoritmi și sisteme capabile să imite, într-o anumită măsură, procesele cognitive umane (învățare, raționament, luarea deciziilor). Din AI pornesc subramurile ce includ învățarea automată, deep learning-ul, procesarea limbajului natural sau sisteme multi-agent, fiecare implicând metode matematice și algoritmice sofisticate.

Robotica este domeniul ingineresc ce se axează pe proiectarea, construcția și operarea roboților – sisteme automate sau semi-autonome ce interactionează cu mediul prin senzori și actuatori. De la roboți industriali folosiți pe liniile de asamblare, până la roboți umanoizi destinați asistenței în situații sensibile (medical, educational), evoluția roboticii a fost profund influențată de progresele din AI.

Combinarea AI cu robotică conduce la sisteme ce nu doar execută sarcini, ci și învață din experiență și se adaptează la noi condiții. De exemplu, roboții medicali asistați de AI pot interpreta imagini radiologice și oferi diagnostice, iar sistemele de tip ‘swarm’ pot coordona mii de agenți autonomi, inspirându-se din comportamentul natural al furnicilor sau al păsărilor. Astfel, această lucrare explorează subdomeniile principale, structurându-le pe ramuri teoretice, experimentale și de design, și evidențiind atât provocările cât și contribuțiile pionierilor în domeniu.

1. Învățare automată și deep learning
2. Procesarea limbajului natural și viziune computerizată
3. AI explicabil și sisteme multi-agent
4. Roboți autonomi și industriali
5. Roboți umanoizi și colaborativi
6. Robotica medicală și soft robotics
7. Robotica de tip „Swarm” (Roboți de Roi)
8. Robotica cognitivă și sisteme de control inteligente
9. Simulare de realitate viruală

### 8.1 Învățare Automată și Deep Learning

Învățarea automată și deep learning reprezintă ramuri fundamentale ale AI, prin care se extrag modele complexe din seturi mari de date, folosind rețele neuronale multiple. Evoluția tehnicielor pornește din apariția perceptronului în anii '60 și a

evaluat odată cu creșterea capacitaților de calcul și extinderea datelor disponibile. Aceasta permite, de exemplu, recunoașterea imaginilor, analiza limbajului și luarea deciziilor în timp real.

### 8.1.1 Activitățile principale

#### 1. Teorie

- (a) **Definirea cadrului conceptual:** Se prezintă conceptele de bază—rețele neuronale, funcții de activare și algoritmi de antrenare.
- (b) **Descrierea cadrelor teoretice:** Se explică metode precum backpropagation, algoritmi de optimizare (ex. gradient descent, Adam) și tehnici de regularizare (dropout, batch normalization).
- (c) **Metode de raționament:** Este analizat modul de funcționare al rețelelor complexe și cum acestea pot învăța trăsături din date.
- (d) **Întrebări fundamentale:** De exemplu, „Cum recunoaște o rețea profundă structuri complexe în imagini?” sau „De ce este importantă reducerea overfitting-ului?”

#### 2. Experiment

- (a) **Explorarea modelelor:** Se compară arhitecturi clasice (ex.: AlexNet, VGG) cu modele moderne (ex.: ResNet, Transformers).
- (b) **Implementări și validarea experimentală:** Sunt realizate experimente pe seturi standard (ex.: MNIST, CIFAR-10, ImageNet), iar performanțele se evaluatează prin precizie, recall și F1-score.
- (c) **Exemplu practic:** Un proiect concret poate fi implementarea unui sistem de clasificare a imaginilor, unde se monitorizează îmbunătățirea performanței prin diferite tehnici de augmentare a datelor.
- (d) **Listarea metodelor cunoscute:** De exemplu, este prezentată trecerea de la rețelele simple la cele adânci, precum și inovațiile din domeniul GAN.

#### 3. Design

- (a) **Produse rezultate:** Prototipuri software care pot fi integrate în aplicații, cum ar fi sisteme de recunoaștere facială.
- (b) **Instrumente rezultate:** Utilizarea unor biblioteci importante (TensorFlow, PyTorch, Keras) care facilitează rapid dezvoltarea de modele.

- (c) **Tehnici inovative:** Noile arhitecturi permit o mai bună interpretabilitate a deciziilor rețelelor.
- (d) **Impactul designului:** Acest design transformă aplicațiile din industrie, de la sisteme de securitate la asistenți personali, contribuind la automatizarea multor procese.

#### 4. Extra

- (a) **Cunoștințe necesare:** Fundamentele matematicii, statistică și programare sunt esențiale. De exemplu, lucrările precum „Deep Learning” de Goodfellow et al. oferă o bază teoretică solidă.
- (b) **Literatura de specialitate:** Se recomandă studierea lucrărilor publicate în conferințe precum NeurIPS sau ICML; de exemplu, articolele ce prezintă primele rezultate ale rețelelor generative adversariale.

##### 8.1.2 Relațiile cu celelalte subdomenii

Învățarea automată se intersecțează cu metodele de procesare a limbajului natural și aplicațiile în robotică (pentru sisteme de control și percepție), facilitând colaborarea interdisciplinară.

##### 8.1.3 Probleme importante și probleme deschise

- **Provocări tehnice:** Necesitatea unor seturi de date foarte mari, probleme de scalabilitate și un consum energetic ridicat la antrenarea modelelor.
- **Probleme deschise:** Interpretabilitatea modelelor (black-box), biasul algoritmic și nevoia de explicare a deciziilor luate într-un mod inteligibil pentru utilizator.

##### 8.1.4 Persoane importante

Pionieri ca Geoffrey Hinton, Yann LeCun, Yoshua Bengio și Andrew Ng au avut un impact decisiv, punând bazele teoretice și practice ale acestei ramuri.

##### 8.1.5 Forumuri importante

- **Conferințe:** NeurIPS, ICML, CVPR.
- **Reviste:** IEEE Transactions on Neural Networks și Journal of Machine Learning Research.

### 8.1.6 Dimensiunea locală și globală

- **Local:** Proiectele universitare și laboratoarele din România implementează aceste tehnologii, colaborând la proiecte interinstituționale.
- **Global:** Există numeroase colaborări și standarde internaționale, iar tehnologiile dezvoltate găsesc rapid aplicabilitate în industrie.

## 8.2 Procesarea Limbajului Natural și Viziune Computerizată

Acest subdomeniu se concentrează pe dezvoltarea de algoritmi capabili să înțeleagă, genereze și interpreteze limbajul natural, precum și să proceseze imagini și video-clipuri pentru a extrage informații semnificative. Evoluția a trecut de la sisteme bazate pe reguli la utilizarea modelelor de deep learning moderne, cum ar fi Transformers.

### 8.2.1 Activitățile principale

#### 1. Teorie

- (a) Prezentarea conceptelor de tokenizare, embedding-uri și modele de sevențe (RNN, LSTM, Transformers).
- (b) Diferențierea abordărilor tradiționale față de cele moderne.

#### 2. Experiment

- (a) Testarea tehniciilor pe seturi de date standard, de exemplu utilizarea setului COCO pentru viziune și WikiText pentru NLP.
- (b) Evaluarea performanțelor folosind metrii specifice precum BLEU pentru traduceri sau IoU pentru recunoașterea obiectelor.

#### 3. Design

- (a) Crearea de interfețe interactive integrate cu module NLP și de viziune, personalizate în funcție de datele utilizatorilor.
- (b) Proiectarea sistemelor care permit rafinarea continuă a interacțiunii, similar cu interfețele adaptive.

#### 4. Extra

- (a) **Cunoștințe necesare:** Este necesară o bază solidă în lingvistică computațională, programare avansată, matematică și statistică.

- (b) **Literatura de specialitate:** Se recomandă studierea lucrărilor publicate în ACL și EMNLP, de exemplu articole care prezintă noile tendințe ale procesării limbajului natural și tehniciile de prelucrare a imaginilor pentru aplicații industriale.

### 8.2.2 Relațiile cu celelalte subdomenii

Tehnicile din acest subdomeniu sunt fundamentale pentru dezvoltarea aplicațiilor AI, fiind integrate cu sistemele conversaționale și cu roboții capabili să interacționeze cu mediul în mod intuitiv.

### 8.2.3 Probleme importante și probleme deschise

- **Provocări:** Adaptarea la limbi cu resurse limitate, reducerea bias-ului în modelele de limbaj și asigurarea interpretabilității rezultatelor.
- **Probleme deschise:** Integrarea contextului cultural și social în modelele de limbaj și optimizarea performanței în medii cu zgomot informațional.

### 8.2.4 Persoane importante

Cercetători remarcăți ca Christopher Manning și Yoav Goldberg au adus contribuții esențiale în avansarea metodelor moderne de NLP.

### 8.2.5 Forumuri importante

- **Conferințe:** ACL, EMNLP.
- **Reviste:** Computational Linguistics, IEEE Transactions on Pattern Analysis and Machine Intelligence.

### 8.2.6 Dimensiunea locală și globală

- **Local:** Integrarea acestor tehnologii în proiectele de cercetare și în aplicațiile industriale din România.
- **Global:** Colaborări internaționale extinse și standarde adoptate la nivel global.

## 8.3 AI Explicabil și Sisteme Multi-Agent

Acest subdomeniu se ocupă de dezvoltarea de modele AI care oferă explicații clare asupra deciziilor lor, precum și de studierea sistemelor multi-agent, unde mai mulți agenți autonomi colaborează pentru rezolvarea unor sarcini complexe.

### 8.3.1 Activitățile principale

#### 1. Teorie

- (a) Elaborarea cadrului conceptual necesar pentru interpretarea deciziilor AI.
- (b) Studierea interacțiunilor dintre agenți, cu accent pe strategii de colaborare descentralizată.

#### 2. Experiment

- (a) Testarea prototipurilor care integrează componente de explainable AI în sisteme multi-agent, de exemplu în simulări de trafic autonom.
- (b) Evaluarea impactului asupra încrederii utilizatorilor și performanței generale a sistemului.

#### 3. Design

- (a) Crearea interfețelor transparente care explică modul de decizie al algoritmilor.
- (b) Implementarea dashboard-urilor pentru monitorizarea eficienței sistemelor multi-agent.

#### 4. Extra

- (a) **Cunoștințe necesare:** Este importantă o înțelegere inter-disciplinară a AI, psihologiei utilizatorului și eticii.
- (b) **Literatura de specialitate:** De exemplu, lucrările prezentate la conferințele ACM IUI și AAAI oferă studii de caz concrete privind clarificarea deciziilor algoritmice.

### 8.3.2 Relațiile cu celelalte subdomenii

Acest domeniu contribuie la creșterea transparentei în AI și facilitează colaborarea între agenți autonomi, fiind strâns legat de celelalte ramuri ale inteligenței artificiale și aplicat în robotică.

### **8.3.3 Probleme importante și probleme deschise**

- **Provocări:** Explicarea deciziilor complexe și menținerea unui echilibru între performanță și transparență.
- **Probleme deschise:** Integrarea armonioasă a sistemelor multi-agent într-un mediu dinamic și complex.

### **8.3.4 Persoane importante**

Pionieri precum Saleema Amershi și Ben Shneiderman au propus principii care au influențat practicile actuale în AI explicabil.

### **8.3.5 Forumuri importante**

- **Conferințe:** ACM IUI, AAAI, NeurIPS.
- **Reviste:** Journal of Artificial Intelligence Research, AI Magazine.

### **8.3.6 Dimensiunea locală și globală**

- **Local:** Aplicații și proiecte la nivel universitar și cercetare în domenii interdisciplinare.
- **Global:** Standarde internaționale și colaborări multinaționale intense.

## **8.4 Roboti Autonomi și Industriali**

Robotii autonomi și industriali folosesc algoritmi de AI și sisteme de control pentru a executa sarcini fără intervenția umană constantă. Evoluția acestui domeniu a fost determinată de nevoile industriei moderne, unde automatizarea, siguranța și eficiența sunt prioritare.

### **8.4.1 Activitățile principale**

#### **1. Teorie**

- (a) Stabilirea cadrului conceptual: modelele de percepție, planificare și navigare specifice robotilor autonomi.
- (b) Analiza tehniciilor de învățare aplicate pentru adaptarea robotilor la diverse medii dinamice.

#### **2. Experiment**

- (a) Implementarea prototipurilor în medii simulate și reale pentru testarea preciziei și siguranței.
- (b) Evaluarea performanței în condiții operaționale diferite, de exemplu în medii industriale.

### 3. Design

- (a) Proiectarea sistemelor de control integrate în platforme robotice.
- (b) Utilizarea unor soluții inovatoare pentru optimizarea traseelor și evitarea obstacolelor.

### 4. Extra

- (a) **Cunoștințe necesare:** Este necesară o bază în mecanică, electronică, programare și inginerie de control.
- (b) **Literatura de specialitate:** De exemplu, lucrări despre robotică industrială și sisteme autonome, cum ar fi studiile publicate în IEEE Transactions on Robotics, oferă exemple concrete de implementare.

#### 8.4.2 Relațiile cu celelalte subdomenii

Acești roboți se interconectează cu dezvoltările din deep learning și interacțiunea om-computer, integrându-se în sisteme complexe de automatizare și control.

#### 8.4.3 Probleme importante și probleme deschise

- **Provocări:** Adaptarea la medii dinamice și creșterea securității operaționale.
- **Probleme deschise:** Optimizarea costurilor și dezvoltarea unor protocoale eficiente de colaborare om-robot.

#### 8.4.4 Persoane importante

Cercetători precum Rodney Brooks și Oussama Khatib sunt figuri reprezentative în dezvoltarea paradigmelor moderne de robotică.

#### 8.4.5 Forumuri importante

- **Conferințe:** ICRA, IROS, RoboCup.
- **Reviste:** IEEE Transactions on Robotics, International Journal of Robotics Research.

#### **8.4.6 Dimensiunea locală și globală**

- **Local:** Proiecte de cercetare în universitățile din România, precum și colaborări cu parteneri industriali.
- **Global:** Standardele internaționale și colaborările transnaționale intensifică adoptarea tehnologiilor autonome.

### **8.5 Roboți Umanoizi și Colaborativi**

Acest subdomeniu se axează pe dezvoltarea roboților care imită mișcările și comportamentele umane, facilitând interacțiunea naturală și colaborativă între om și mașină, de la uz casnic până la aplicații industriale și cercetare.

#### **8.5.1 Activitățile principale**

##### **1. Teorie**

- (a) Studiul conceptelor de imitație a mișcării și sincronizare între acțiunile umane și cele ale roboților.
- (b) Analiza impactului biomecanicii și ergonomiei în proiectarea roboților umanoizi. Această teorie se referă la studiul modului în care parametrii fiziologici (postură, distribuția forțelor, ritmul natural al mișcării) influențează designul roboților, astfel încât aceștia să se miște în mod natural și să minimizeze riscul de accidentări.

##### **2. Experiment**

- (a) Testarea roboților în medii simulate și reale pentru a evalua interacțiunea și siguranța.
- (b) Măsurarea performanței în sarcini colaborative, de exemplu prin tehnici de eye-tracking și analize ergonomice.

##### **3. Design**

- (a) Dezvoltarea interfețelor de comunicare între om și robot, pentru a facilita un flux natural de interacțiune.
- (b) Implementarea sistemelor de feedback vizual și haptic pentru a îmbunătăți coordonarea.

##### **4. Extra**

- (a) **Cunoștințe necesare:** Este esențială cunoașterea domeniilor de robotică, psihologie a interacțiunii și ergonomie.
- (b) **Literatura de specialitate:** De exemplu, studiile publicate precum *The Design of Everyday Things* de Donald Norman [96] și lucrări experimentale despre interacțiunea om-robot oferă exemple concrete despre modul de evaluare a ergonomiei și impactul biomecanicii.

#### **8.5.2 Relațiile cu celelalte subdomenii**

Roboții umanoizi se leagă de dezvoltările în AI pentru a îmbunătăți autonomia și adaptabilitatea, precum și de sistemele de interacțiune om-computer pentru a asigura o colaborare naturală.

#### **8.5.3 Probleme importante și probleme deschise**

- **Provocări:** Sincronizarea mișcărilor cu activitățile umane și adaptabilitatea la diverse medii.
- **Probleme deschise:** Stabilirea unor standarde de siguranță și optimizarea costurilor de producție.

#### **8.5.4 Persoane importante**

Cercetători ca Hiroshi Ishiguro au contribuit semnificativ la dezvoltarea și înțelegerea roboților umanoizi.

#### **8.5.5 Forumuri importante**

- **Conferințe:** Humanoids, RoboCup, European Robotics Forum.
- **Reviste:** Advanced Robotics, Robotics and Autonomous Systems.

#### **8.5.6 Dimensiunea locală și globală**

- **Local:** Colaborări între institutele de cercetare și companiile tehnice din România.
- **Global:** Parteneriate internaționale și implementarea standardelor de interoperabilitate.

## 8.6 Robotica Medicală și Soft Robotics

Robotica Medicală combină ingineria robotică, AI și medicina pentru a crea sisteme ce asistă specialiștii, optimizează tratamentele și, în anumite cazuri, efectuează intervenții chirurgicale minim invazive. Pe de altă parte, Soft Robotics se concentrează pe construcția roboților din materiale flexibile și moi, inspirându-se din biologia naturală pentru a obține mișcări sigure și naturale, esențiale în medii medicale delicate.

### 8.6.1 Activitățile principale

#### 1. Teorie

- (a) Elaborarea modelelor AI pentru analiza imagistică medicală și pentru planificarea intervențiilor chirurgicale.
- (b) Studiul materialelor flexibile și a mecanismelor de mișcare adaptate, pentru a obține sisteme de soft robotics potrivite în aplicații medicale.

#### 2. Experiment

- (a) Testarea roboților medicali în simulări și proceduri pilot, de exemplu în asistență chirurgicală robotizată.
- (b) Evaluarea interacțiunii între roboți și personalul medical, incluzând studii clinice pentru verificarea siguranței.

#### 3. Design

- (a) Proiectarea de exoschelete și roboți asistivi care sprijină recuperarea și mobilitatea pacienților.
- (b) Dezvoltarea interfețelor de colaborare între operator și robot, pentru o manipulare precisă în intervenții.

#### 4. Extra

- (a) **Cunoștințe necesare:** Se impune o combinație de cunoștințe în biomecanică, inginerie biomedicală, AI aplicată și neuroștiințe.
- (b) **Literatura de specialitate:** De exemplu, articole din IEEE Transactions on Biomedical Engineering detaliază studii experimentale care evaluatează precizia și siguranța intervențiilor robotizate.

### **8.6.2 Relațiile cu celelalte subdomenii**

Robotica medicală se conectează cu tehniciile NLP (pentru interacțiunea cu pacientul), AI explicabil (pentru interpretarea datelor clinice) și simularea VR (pentru antrenamentul medicilor).

### **8.6.3 Probleme importante și probleme deschise**

- **Provocări:** Asigurarea siguranței maxime în intervenții și obținerea unei precizii ridicate în proceduri chirurgicale.
- **Probleme deschise:** Integrarea eficientă a AI în diagnostic și adaptarea roboților la diversitatea fiziologică a pacienților.

### **8.6.4 Persoane importante**

Cercetători importanți din domeniul roboticii medicale, care au contribuit la dezvoltarea sistemelor chirurgicale asistate, au avut un impact semnificativ asupra siguranței și preciziei intervențiilor.

### **8.6.5 Forumuri importante**

- **Conferințe:** IEEE International Conference on Robotics and Automation (ICRA), Medical Image Computing and Computer-Assisted Intervention (MICCAI).
- **Reviste:** IEEE Transactions on Medical Robotics and Bionics.

### **8.6.6 Dimensiunea locală și globală**

- **Local:** Implementări pilot în centre clinice și colaborări între facultăți de inginerie și medicină.
- **Global:** Standardele internaționale de siguranță și colaborările multinaționale amplifică adoptarea tehnologiilor în domeniul medical.

## **8.7 Robotica de Tip „Swarm” (Roboți de Roi)**

Acest subdomeniu se inspiră din comportamentul natural al grupurilor (ex.: furnici, albine, păsări) pentru a dezvolta grupuri de roboți care colaborează fără un control centralizat. Roboții de tip swarm își coordonează mișcările și comunică între ei, adaptându-se dinamic la schimbări de mediu, astfel încât să rezolve împreună sarcini complexe.

### 8.7.1 Activitățile principale

#### 1. Teorie

- Studierea algoritmilor de coordonare autonomă și a comportamentului emergent în grupuri.
- Modelarea interacțiunilor descentralizate și a strategiilor de comunicare între agenți.

#### 2. Experiment

- Testarea prototipurilor în medii dinamice, prin simulări și experimente în condiții reale.
- Evaluarea eficienței comunicării și coordonării între roboți, de exemplu, prin monitorizarea răspunsurilor la situații de urgență.

#### 3. Design

- Crearea de roboți de mici dimensiuni, modulari, capabili să se reconfigureze și să colaboreze pentru diverse sarcini.
- Dezvoltarea algoritmilor care permit o comunicare robustă și adaptabilitate în timp real, asigurând o coordonare eficientă fără un nod central.

#### 4. Extra

- **Cunoștințe necesare:** Sunt esențiale cunoștințele din AI distribuită, robotică modulară și dinamica sistemelor naturale.
- **Literatura de specialitate:** De exemplu, lucrări precum *Swarm Robotics: From Biology to AI* - Bonabeau, E., Dorigo, M., & Theraulaz, G. [97] și publicații IEEE în domeniu oferă studii experimentale care validează concepțile algoritmice pentru coordonarea descentralizată.

### 8.7.2 Relațiile cu celelalte subdomenii

Roboții de tip swarm se leagă de AI explicabil (pentru a gestiona controlul descentralizat), de sistemele multi-agent și de robotică autonomă, contribuind la realizarea unor operațiuni colaborative eficiente.

### **8.7.3 Probleme importante și probleme deschise**

- **Provocări:** Menținerea unei comunicări eficiente între toți agentii și gestionarea dinamică a grupului.
- **Probleme deschise:** Dezvoltarea unor strategii complet autonome care pot ajusta comportamentul roiului în funcție de schimbările de mediu.

### **8.7.4 Persoane importante**

Cercetători specializați în sisteme distribuite și comportamente colective, inspirați din studiile naturaliste, au contribuit la formularea strategiilor de coordonare pentru roboții swarm.

### **8.7.5 Forumuri importante**

- **Conferințe:** IEEE International Conference on Swarm Intelligence (ICSI), RoboCup.
- **Reviste:** IEEE Transactions on Systems, Man, and Cybernetics.

### **8.7.6 Dimensiunea locală și globală**

- **Local:** Proiecte interdisciplinare în cadrul universităților din România, împreună cu colaborări regionale.
- **Global:** Inițiative internaționale și colaborări cu laboratoare de vârf din țară și din străinătate amplifică impactul acestui subdomeniu.

## **8.8 Robotica Cognitivă și Sisteme de Control Inteligente**

Robotica cognitivă integrează metodele AI cu tehnologiile de robotică pentru a crea sisteme capabile să ia decizii autonome într-un mediu complex. Această abordare combină percepția, planificarea și controlul pentru a permite o colaborare eficientă între om și robot.

### **8.8.1 Activitățile principale**

#### **1. Teorie**

- (a) Definirea modelelor cognitive ce ghidează procesul decizional.
- (b) Integrarea algoritmilor de învățare în sisteme de control în timp real.

## 2. Experiment

- (a) Testarea prototipurilor pe platforme multi-senzoriale.
- (b) Evaluarea performanței sistemelor în medii dinamice.

## 3. Design

- (a) Dezvoltarea interfețelor de monitorizare și control.
- (b) Crearea protocolelor de comunicare între componente hardware și software.

## 4. Extra

- (a) **Cunoștințe necesare:** Este necesară o înțelegere solidă a roboticii, AI și ingineriei de control, completată de elemente de psihologie cognitivă.
- (b) **Literatura de specialitate:** De exemplu, lucrările publicate de cercetaitori de la Carnegie Mellon și MIT evidențiază integrarea algoritmilor de machine learning în controlul în timp real, oferind studii de caz concrete.

### 8.8.2 Relațiile cu celelalte subdomenii

Această secțiune se leagă de aplicațiile în deep learning, robotică autonomă și interacțiunea om-computer, contribuind la integrarea armonioasă a sistemelor autonome.

### 8.8.3 Probleme importante și probleme deschise

- **Provocări:** Interpretabilitatea deciziilor sistemelor și coordonarea în timp real.
- **Probleme deschise:** Dezvoltarea unor sisteme complet integrate și adaptive, capabile să răspundă la condiții imprevizibile.

### 8.8.4 Persoane importante

Cercetaitori de la instituții de prestigiu precum Carnegie Mellon și MIT contribuie la formularea paradigmelor moderne în robotica cognitivă.

### 8.8.5 Forumuri importante

- **Conferințe:** IJCAI, RoboCup.
- **Reviste:** Autonomous Robots, IEEE Transactions on Automation Science and Engineering.

### 8.8.6 Dimensiunea locală și globală

- **Local:** Proiecte pilot la universități și colaborări interdepartamentale.
- **Global:** Standardizarea sistemelor autonome și colaborările internaționale intensifică aplicabilitatea tehnologiilor.

## 8.9 Simulare și Realitate Virtuală

Acest subdomeniu explorează utilizarea simulărilor computerizate și a tehnologiilor de realitate virtuală (VR) pentru a testa, valida și optimiza sistemele de AI și robotică înainte de implementarea lor în medii reale.

### 8.9.1 Activitățile principale

#### 1. Teorie

- (a) Construirea modelelor de medii virtuale care reproduc scenarii complexe și imită condițiile reale.
- (b) Studierea avantajelor și limitărilor simulărilor pentru evaluarea performanțelor sistemelor.

#### 2. Experiment

- (a) Crearea simulărilor detaliate pentru testarea sistemelor, permitând detectarea timpurie a deficiențelor.
- (b) Analiza comparativă a performanței între medii virtuale și implementări reale.

#### 3. Design

- (a) Dezvoltarea interfețelor dedicate manipulării și monitorizării simulărilor.
- (b) Integrarea tehnologiilor VR în procesul de training și validare a prototipurilor.

#### 4. Extra

- (a) **Cunoștințe necesare:** Sunt esențiale cunoștințele de grafică pe calculator și programare.
- (b) **Literatura de specialitate:** De exemplu, studiile publicate la IEEE VR și SIGGRAPH prezintă cazuri concrete despre cum simulările pot reduce costurile de dezvoltare și identifica probleme înainte de implementarea fizică.

### **8.9.2 Relațiile cu celelalte subdomenii**

Simularea și VR sunt complementare tuturor domeniilor din AI și robotică, oferind un cadru de testare și dezvoltare care minimizează riscurile la implementare.

### **8.9.3 Probleme importante și probleme deschise**

- **Provocări:** Realismul mediilor virtuale și sincronizarea precisă cu sistemele fizice.
- **Probleme deschise:** Dezvoltarea unor medii virtuale care reflectă fidel complexitatea realității și care pot fi integrate seamlessly cu prototipurile robotice.

### **8.9.4 Persoane importante**

Cercetători din domeniul VR și simulărilor contribuie la perfecționarea tehnicielor care permit evaluarea precisă a sistemelor.

### **8.9.5 Forumuri importante**

- **Conferințe:** IEEE VR, SIGGRAPH.
- **Reviste:** Presence: Teleoperators and Virtual Environments, Virtual Reality.

### **8.9.6 Dimensiunea locală și globală**

- **Local:** Laboratoare de simulare implementate la instituții educaționale din România.
- **Global:** Colaborările internaționale și standardele globale contribuie la evaluarea precisă a sistemelor.

## 9 Grafică

Grafica computerizată este un domeniu vast și interdisciplinar, cu aplicații în divertisment, educație, știință și industrie. Aceasta include subdomenii precum grafica 2D și 3D, realitatea virtuală și augmentată, grafica pentru jocuri video, CGI, vizualizarea datelor și interfețele grafice. Fiecare dintre aceste ramuri contribuie la crearea unor experiențe vizuale interactive, realiste și eficiente. De la modelarea matematică la designul vizual și aplicațiile practice, grafica joacă un rol esențial în comunicarea vizuală și în dezvoltarea tehnologiilor moderne.

Subdomeniile graficii sunt:

1. Grafică 2D (bidimensională)
2. Grafică 3D (tridimensională)
3. Realitate Virtuală (VR)
4. Realitate Augmentată (AR)
5. Grafică pentru Jocuri Video
6. Grafică Computerizată pentru Imagini Generate pe Calculator (CGI)
7. Vizualizarea Datelor
8. Grafică Interactivă(GUI)

### 9.1 Grafică 2D (bidimensională)

Grafica 2D este o tehnică de redare simplificată a obiectelor reale, care au trei dimensiuni, înălțime, lățime și adâncime, renunțând la una dintre ele (de obicei adâncimea), rezultând astfel o reprezentare schematică în două dimensiuni, lipsită de profunzime, dar totuși satisfăcătoare. Imaginele/desenele bidimensionale există încă din preistorie.

Aceasta a luat naștere în anii 1950 și se baza pe dispozitive de grafică vectorială, care au fost ulterior înlocuite în mare parte de dispozitive bazate pe raster în următoarele decenii. Limbajul PostScript și protocolul X Window System au fost evoluții marcante în domeniu.

#### 9.1.1 Activitățile principale

1. Teorie: Grafica pe computer 2D este utilizată în principal în aplicații care au fost dezvoltate inițial pe tehnologiile tradiționale de imprimare și desen, precum tipografia, cartografia, desenul tehnic, publicitatea, etc. Modelele

bidimensionale sunt deci preferate, deoarece oferă un control mai direct asupra imaginii decât grafica pe computer 3D.

2. Experiment: Modelele grafice 2D pot combina modele geometrice (grafică vectorială), imagini digitale (grafică raster), textul care urmează să fie formatat, funcții și ecuații matematice și multe altele. Aceste componente pot fi modificate și manipulate prin transformări geometrice bidimensionale, cum ar fi translația, rotirea și scalarea. În grafica orientată pe obiect, imaginea este descrisă indirect de un obiect dotat cu o metodă de auto-redare, o procedură care atribuie culori pixelilor imaginii printr-un algoritm arbitrar.
3. Design: În multe domenii, cum ar fi desktop publishing, inginerie și afaceri, un document bazat pe tehnici de grafică computerizată 2D poate fi mult mai mic decât imaginea digitală corespunzătoare; adesea la o scară de 1/1000 sau mai mult. Această reprezentare este mai flexibilă, deoarece poate fi redată la rezoluții diferite pentru a se potrivi cu diferite dispozitive de ieșire. Din aceste motive, documentele și ilustrațiile sunt adesea stocate sau transmise ca fișiere grafice 2D. Ca și exemplu avem programele de proiectare CAD, cum ar fi ArchiCAD sau AutoCAD (fișiere DWG), folosite pentru designul în proiectare, utilizând designul scalar și vectorial.

### 9.1.2 Relațiile cu celelalte subdomenii

- Grafică 3D
  - Deși dedicată redării tridimensionale, grafica 3D se bazează pe elemente 2D și folosește grafica 2D pentru elemente ajutătoare în crearea de elemente 3D (texturi bidimensionale aplicate pe suprafețe 3D, crearea de machete 2D pentru obiecte 3D).
- Interfețe grafice (GUI)
  - Sistemele grafice de operare și aplicațiile software folosesc elemente 2D pentru o interfață user-friendly și se bazează pe grafica 2D pentru elemente de tipul iconuri, meniuri, butoane, pictogramele de pe desktop, ferestre și grafice.

### 9.1.3 Probleme Importante și Probleme Deschise

- Crearea și manipularea formelor
  - Problema: Crearea formelor complexe fără a pierde detalii sau a supraîncărca fișierelor de ieșire este o problemă importantă mai ales în animație sau grafica pentru jocuri.

- Soluție parțială: Programele avansate pentru grafică și design, precum Adobe Illustrator sau CorelDRAW, utilizează tehnici eficiente, cum ar fi ”Anti Aliasing” care constă în adăugarea pixelilor în nuanțe de gri lângă cei negrii pentru a crea o imagine mai finisată, eliminând ”efectul de scară”.
- Compresia imaginilor
  - Formatele imaginilor trebuie să fie cât mai eficiente(png, jpg, jpeg, etc.), iar tehniciile principale folosite se bazează pe transformate Fourier sau wavelet (descompun imaginile în funcții sinusoidale) și selecție de coeficienți, însă prezervarea detaliilor la rezoluții diferite și a diverselor domenii de culoare rămâne o problemă deschisă.

#### 9.1.4 Persoane Importante

- Ivan Sutherland
  - Este considerat un pionier al graficii computerizate. El a creat în 1963 primul sistem de grafică interactivă, numit Sketchpad, care permite utilizatorului să deseneze și să manipuleze obiecte 2D pe un ecran.
- Walt Disney
  - Este un pionier al industriei americane de animație, care a dezvoltat mult animația 2D și a inițiat evoluția în producția de desene animate. El deține recordul pentru cele mai multe premii Oscar câștigate(22) și nominalizări(59) de către o persoană.
- John Warnock și Charles Geschke
  - Sunt fondatorii companiei Adobe, care a revoluționat designul grafic 2D și procesarea imaginilor digitale, dezvoltând software-uri precum Photoshop sau Illustrator. Aceste aplicații sunt utilizate de milioane de designeri grafici pentru crearea, editarea și manipularea imaginilor 2D.
- Shigeru Miyamoto
  - A fost unul dintre pionierii în utilizarea graficii 2D în designul jocurilor video. Acesta este cunoscut pentru crearea jocurilor pentru Nintendo, cum ar fi Super Mario, The Legend of Zelda și Donkey Kong, care au avut un mare impact asupra industriei de gaming, iar grafica 2D a jucat un rol esențial în succesul lor.

### **9.1.5 Forumuri Importante**

- Conferința SIGGRAPH
  - Este numele conferinței anuale despre grafică computerizată (CG) convocată de Organizația ACM SIGGRAPH. Prima conferință SIGGRAPH a avut loc în 1974. La ele participă zeci de mii de profesioniști. În ultimii ani, conferințele SIGGRAPH au avut loc în Dallas, Seattle, Los Angeles, New Orleans, San Diego și în alte orașe din Statele Unite. (SIGGRAPH este acronimul pentru Special Interest Group on GRAPHics and Interactive Techniques).
- Adobe MAX
  - Este o conferință anuală de creativitate organizată de Adobe Inc, adesea în Los Angeles, dar și online. Evenimentul ajută Adobe să prezinte noile dezvoltări ale suitei sale de aplicații și să construiască o comunitate de profesioniști creativi.
- Offf Festival
  - Este un festival, organizat în principal la Barcelona, care se concentrează pe inovație vizuală și tehnici de design. În cadrul acestuia se organizează multe workshopuri care abordează grafica 2D, ilustrația, tipografia și animațiile digitale. Participanții pot învăța de la experți și pot face networking cu alți profesioniști din domeniu.
- The 2D Animation and Illustration Conference
  - Această conferință este dedicată exclusiv animării 2D și ilustrației și se desfășoară adesea în Statele Unite, dar și online. Este un loc excelent pentru a învăța despre tehnici noi, tool-uri și pentru a cunoaște alți profesioniști din domeniu, oferind studenților oportunitatea de a învăța noțiunile fundamentale ale tehniciilor de ilustrare și ale principiilor de animație.

### **9.1.6 Dimensiunea locală și dimensiunea globală**

1. Local:
  - UVT oferă prin Facultatea de Arte și Design specializări în Grafică, Design, Arte Digitale și cursuri axate pe desen tradițional, ilustrație, compozitie, tipografie, design de personaje 2D.

- În Facultatea de Matematică și Informatică, se abordează grafica 2D din punct de vedere tehnic (OpenGL, SVG, procesare de imagine).

2. Global:

- Studio-uri de top: Cartoon Network, Nickelodeon, Netflix Animation, DisneyTV Animation Riot Games.

## 9.2 Grafică 3D (tridimensională)

Grafica 3D este o tehnică bazată pe reprezentarea carteziană geometrică a obiectelor care redă într-un mod realist obiecte care au trei dimensiuni (înălțime, lățime și adâncime).

### 9.2.1 Activitățile principale

Grafica 3D se ocupă cu reprezentarea obiectelor tridimensionale într-un spațiu virtual, utilizând coordonate carteziene ( $x, y, z$ ). Aceasta permite crearea de modele care pot fi modificate cu ajutorul diferitor opțiuni.

1. Teorie:

- (a) Definirea cadrului conceptual

Conceptul de *modelare 3D* se referă la procesul de dezvoltare a reprezentării unui obiect într-un spațiu tridimensional pe baza coordonatelor matematice.

Conceptul de *rendering 3D* se referă la conversia unui model 3D într-o imagine 2D care să includă efecte fotorealistice și non-fotorealistice.

Conceptul de *animație 3D* reprezintă tehnica de a crea imagini realisticе în mișcare. [98]

- (b) Descrierea cadrului/cadrelor

*Modelare 3D* se împarte în modele solide și modele shell. Un model 3D poate fi reprezentat prin modelare poligonală, modelare a curburilor și sculptură digitală.

Conceptul de *rendering 3D* implică generarea de imagini de calitate sau animații, luând în considerare lumina, umbrele și alte efecte vizuale.

Conceptul de *animație 3D* se împarte în două tipuri: unul care redă în timp real animații cum ar fi jocurile video și unul care redă o singură dată ceea ce conține un videoclip.

- (c) Metode de raționament utilizate.

Simularea și modelarea

- (d) Exemplificarea întrebărilor fundamentale
- Cum funcționează proiecțiile 3D pe ecrane 2D?
  - Cum influențează alegerea metodei de modelare rezultatul final?
  - De ce e importantă structura geometrică în reprezentarea obiectelor?

2. Experiment:

- (a) Explorarea modelelor:

*Sketchpad*(1960, Ivan Sutherland) Este unul dintre primele software-uri de 3D drawing care a comercializat folosirea modelelor 3D.[99]

*Autodesk 3D studio*(1990, Hudson and Dan Silva) Este un software de 3D design, engineering și entertainment.

*AutoCad*(1982)

*Rhinoceros*(1980, NURBS)

- (b) Implementări și validarea experimentală:

*Hardware*: Dispozitive de scanare 3D, imprimante 3D

*Software*: Software-uri de modelare 3D, algoritmi de redare și simulare.

- (c) Exemple practice

*Cinematografie*: O primă exemplificare constă în efectuarea filmelor de animații 3D pentru copii și a efectelor vizuale(VFX). [100]

*Arhitectură și design*: Un exemplu constă efectuarea modelelor 3D de locuințe care prezintă clientului într-un mod realist cum ar arăta ceea ce propune arhitectul/designer-ul în viață reală. *Industria auto*:

Grafica 3D se folosește pentru a simula diferite scenarii care să ajute la antrenarea pilotului automat. [101]

3. Design: [102]

- (a) Produse rezultate din cercetare:

- *Imprimante 3D* (Bambu Lab, Prusa, Creali)

- (b) Instrumente rezultate din cercetare.

- *Modelare 3D* (Blender, Autodesk Maya)

- *Animare 3D* (RenderMan, Houdini)

- (c) Tehnici rezultate din cercetare.

- *Ray tracing și path tracing* care permit iluminarea realistă în scene 3D și sunt folosite în filme și jocuri video.

- *Sculptură digitală* este o tehnică derivată din modelarea 3D

- *Motion capture și retargeting* sunt folosite ca să facă realistică mișcarea personajelor 3D din jocurile video și animațiilor.

(d) Impactul designului asupra dezvoltării subdomeniului.

- *Calitate vizuală* a animațiilor mult mai realistă.
- *Performanță optimizată* prin algoritmi de rendering mai eficienți.

4. Extra:

5. Ce cunoștințe sunt necesare pentru a intra în domeniu?

- Grafică 3D: Blender, Autodesk
- Matematică: Geometrie vectorială, Algebră liniară
- Fizică

6. Literatura de specialitate

- IEEE Transactions on Visualization and Computer-Graphics [103]
- Journal of Computer Graphics Techniques [104]

### 9.2.2 Relațiile cu celelalte subdomenii

Se leagă de *z*, deoarece grafica 3D permite utilizatorilor să interacționeze cu obiecte virtuale prin gesturi și mișcări care îmbunătățesc experiența. Se leagă de *inteligenta artificială*, deoarece folosește algoritmi care să facă iluminare, umbre și animații realiste.

### 9.2.3 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *limitări hardware* care impiedică randarea unor scene complexe, *consum mare de energie* și *conținut limitat și scump* care e produs greu, limitând domeniul la o evoluție rapidă.

Câteva probleme deschise sunt legate de *accesibilitate*, fiind greu de învățat la început și *automatizarea modelării 3D*, care poate fi rezolvată cu generarea unor modele realiste cu ajutorul inteligenței artificiale.

### 9.2.4 Persoane Importante

Cercetători și pionieri: Edwin Catmull [105], James Blinn [106]

Cercetători și ingineri: Turner Whitted [107]

### **9.2.5 Forumuri Importante**

Forum-uri de specialitate: SigGraph[108], BlenderArtists [109]

### **9.2.6 Dimensiunea locală și dimensiunea globală**

1. Local:

2. Global:

Colaborări internaționale: Proiecte de cercetare(ACM SigGraph, NVIDIA research), Consortii globale(Eurographics).

Standardizări: OpenGL

Influența globală: Jocuri video, filme și animații, arhitectură și design interior

## **9.3 Realitate Virtuală (VR)**

*Realitatea virtuală* reprezintă utilizarea tehnologiei computerizate pentru a crea efectul unei lumi tridimensionale interactive în care obiectele au propria prezență în spațiu.

Încă din 1950, oamenii de știință și cercetătorii au încercat să creeze dispozitive care să ajute utilizatorii să experimenteze diferite medii. Nintendo este considerată ca fiind prima companie care a lansat headset-ul VR Nintendo Virtual Boy în 1995. În 2006, Nintendo lansează consola Wii care folosește accelerometrii și senzori cu infraroșu pentru a prelua date de la utilizatori. Ulterior, în 2014, Google lansează Cardboard, un headset VR din carton, care se folosește de telefonul utilizatorului. În 2015, Oculus și Samsung lansează Gear VR de la care urmează să se inspire și alte companii. [110]

### **9.3.1 Activitățile principale**

*Realitatea virtuală* implică crearea și manipularea unor medii artificiale tridimensionale și are scopul de a oferi utilizatorului o experiență imersivă, interactivă și multisenzorială.

1. Teorie:

(a) Definirea cadrului conceptual

Conceptul de *imersiune* se referă la nivelul stimulilor senzoriali pe care utilizatorul îl experimentează.

Conceptul de *prezență* se referă la senzația utilizatorului că este

„acolo” în mediul virtual, ca și cum ar face parte din el.

Conceptul de *interactivitate* se referă la capacitatea unui sistem, unei tehnologii sau unui mediu de a permite utilizatorului să interacționeze activ cu acesta.

(b) Descrierea cadrului/cadrelor

*Imersiunea* se împarte în imersiune reprezentativă, imersiune participativă, imersiune afectivă și imersiune narativă. Imersiunea reprezentativă se referă la realismul vizual și auditiv al mediului virtual. Imersiunea participativă implică gradul de interactivitate oferit utilizatorului, cât de mult poate acesta să influențeze mediul virtual prin acțiunile lui. Imersiunea afectivă se referă la emoțiile pe care le creează experiența virtuală, contribuind la implicarea utilizatorului. Imersiunea narativă se referă la cât de prins este utilizatorul în mediul virtual, la cât de captivat este de poveste, determinându-l pe acesta să exploreze mai mult. [111]

*Prezența* este influențată de realismul vizual, interactivitatea și răspunsul senzorial. Aceasta conține diferite iluzii care ajută la o experiență mult mai realistă precum iluzia de a te afla într-un spațiu stabil, iluzia propriei întruchipări, iluzia interacțiunii fizice și iluzia comunicării sociale. [112]

*Interactivitatea* se manifestă prin capturarea de mișcare, urmărirea ochiului, feedback haptic, interacțiune vocală și urmărirea gesturilor. Capturarea de mișcare se bazează pe capturarea unui obiect 3D în spațiu, în timp real. Urmărirea ochiului și a mișcărilor acestuia care, atunci când utilizatorul reacționează la un stimул, acesta își focusează privirea pe el, iar astfel calculatorul detectează mișcarea ochiului. Feedback-ul haptic este resimțit de utilizator atunci când face diferite acțiuni precum deschiderea unei uși, apăsarea unui buton sau ridicarea unui obiect. Interacțiunea vocală permite utilizatorilor să vorbească între ei și să folosească comenzi vocale. Urmărirea gesturilor se realizează prin senzori diferenți poziționați pe mâinile sau picioarele utilizatorului. [113]

(c) Metode de raționament utilizate.

- Raționamentul logic [114]
- Simularea și modelarea

(d) Exemplificarea întrebărilor fundamentale

- Cum funcționează realitatea virtuală?
- Cum percep oamenii mediile virtuale?
- De ce se simt oamenii „prezenți” în mediile virtuale?
- Cum afectează realitatea virtuală comportamentele utilizatorilor?

- Ce impact are realitatea virtuală asupra sănătății fizice și mentale?

## 2. Experiment:

### (a) Explorarea modelelor:

*The Sensorama*(1955-1962, Morton Heilig) Este unul dintre primele sisteme de realitate virtuală care oferă o experiență multisenzorială utilizatorului, combinând imagini stereoscopice, sunet binaural, miroșuri, vibratii și senzații Primul exemplu Sensorama a fost o plimbare virtuală cu bicicleta prin New York în care utilizatorul experimentează sunetul străzii, miroșurile orașului și vântul generat de ventilatoare.[115]

*The Sword of Damocles*(1968, Ivan Sutherland) Este primul HMD (Head-mounted display). Acesta folosește două ecrane mici și un sistem de urmărire a mișcărilor capului pentru a ajusta imaginile afișate în funcție de poziția utilizatorului.[99]

*Flight simulators*(Tom Furness) Furness alături de colegii săi a dezvoltat un sistem care simulează bordul virtual pentru piloții de avioane care conține informații despre zbor. [116]

*NASA Ames*(1984, Dr Mike McGreevy and Jim Humphries) Cei doi au inventat setup-ul VR VIVED(Virtual Environment Display) care a fost folosit mai apoi de către astronauți [116]

### (b) Implementări și validarea experimentală:

- *Hardware*: Căști VR, Controllere, Senzori de mișcare.
- *Software*: Motoare grafice(Unity, Unreal Engine), algoritmi de redare și simulare.

### (c) Exemple practice:

*Educație*: O primă exemplificare constă în simulări interactive care ajută adolescenții să realizeze o serie de evenimente la job-uri și care li se potrivește cel mai bine. O altă exemplificare ar fi ora de geografie care poate fi asistată de un headset VR, astfel încât elevii să viziteze diferite obiecte turistice pentru a le observa caracteristicile. De asemenea, ochelarii VR pot fi folosiți și la matematică, elevii trebuind să descopere proprietățile formelor geometrice din ceea ce văd cu ochelarii VR.[117] *Medicina*: O primă exemplificare constă în terapie. Pacienții ar fi introduși într-un mediu virtual controlat în care să-și vindece fobiile, stresul post-traumatic și să trateze durerea cronică din timpul unor proceduri. De asemenea, ochelarii VR pot fi folosiți și la exemplificarea unor proceduri pentru cei care sunt la început și nu au experiență. Aceștia pot exersa operații chirurgicale cu ajutorul realității virtuale.[118]

3. Design:

- (a) Produse rezultate din cercetare:
- *Headset-uri VR* (Oculus Rift, HTC Vive, Playstation VR)
  - *Controller VR* (Valve Index Controller)
  - *Sisteme de urmărire a gesturilor* (Leap Motion, VR Gloves)
  - *Platforme de realitate virtuală* (VR Chat, Altspace VR)
- (b) Instrumente rezultate din cercetare:
- *Motoare grafice* (Unity, Unreal Engine)
  - *Software de modelare 3D* (Blender, Autodesk Maya)
  - *Tehnologii de captare și simulare*
- (c) Tehnici rezultate din cercetare:
- *Rendering în timp real* pentru a asigura grafică fluidă, fără întârzieri.
  - *Reducere motion sickness* prin optimizarea cadrelor pe secundă.
  - *Haptică avansată* pentru simularea senzațiilor tactile în interacțiunea cu obiectele virtuale.
  - *Tehnici de urmărire a mișcării ochiului* pentru a adapta focalizarea și interacțiunea utilizatorului în mediul virtual.
- (d) Impactul designului asupra dezvoltării subdomeniului:
- *Experiențe* mai captivante și mai realiste care cresc gradul de imersiune.
  - *Accesibilitate* îmbunătățită.
  - *Aplicații* în educație, medicină, antrenamente industriale și divertisment.
  - *Evoluția interacțiunii om-calculator*.

4. Extra:

- (a) Ce cunoștiințe sunt necesare pentru a intra în domeniu? [119]
- Programare: Python, C sharp, C++, Java
  - Grafică 3D: Blender, Autodesk
  - Matematică: Geometrie vectorială, Algebră liniară
- (b) Literatura de specialitate
- The Journal of Virtual Reality and Broadcasting [120]
  - Virtual Reality from Springer [121]

### **9.3.2 Relațiile cu celelalte subdomenii**

Se leagă de *interacțiunea om-computer*, deoarece realitatea virtuală se folosește de gesturi și eye-tracking pentru a funcționa. Se leagă de *inteligenta artificială*, deoarece într-un mediu virtual interacționezi cu personaje NPC care au un comportament realist generat de AI. Se leagă de *arhitectura calculatorului*, deoarece pentru a folosi un mediu virtual, este nevoie de o placă video puternică pentru a procesa frame-urile dintr-un mediu virtual.

### **9.3.3 Probleme Importante și Probleme Deschise**

Câteva probleme importante sunt *motion-sickness* care apare atunci când utilizatorul folosește prea mult timp ochelarii VR și pierde contactul cu realitatea, *limitări hardware* care scad din imersiunea utilizatorului în mediul virtual și *continut limitat și scump* care e produs greu, limitând domeniul la o evoluție rapidă.

Câteva probleme deschise sunt legate de *etică și consumămant* privind acțiunile realizate de utilizator în mediul virtual și *protecția datelor personale*, ochelarii VR folosindu-se în permanență de date biometrice pentru a oferi o experiență imersivă.

### **9.3.4 Persoane Importante**

Cercetători și pionieri: Ivan Sutherland [99], Morton Heilig [115], Tom Furness [116]

Cercetători și psihologi: Mel Slater [122], Jeremy Bailenson [123]

### **9.3.5 Forum-uri Importante**

Forum-uri de specialitate: VRChat[124], VR/AR Association [125]

### **9.3.6 Dimensiunea locală și dimensiunea globală**

1. Local:

În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, în anul III, semestrul II, studenții pot opta pentru optionalul de Programarea jocurilor pe calculator și realitate virtuală [126]

2. Global:

Colaborări internaționale: Proiecte europene(Horizon Europe), Consorții globale(IEEE VR, EuroXR Association).

Standardizări: OpenXR, WebXR

Influența globală: Educație, Sănătate, Industrie și producție, Cultură și patrimoniu

## 9.4 Realitate Augmentată (AR)

*Realitatea augmentată* reprezintă procesul prin care se suprapune realitatea virtuală peste lumea reală.

Încă din 1901, L. Frank Baum a definit conceptul de realitate augmentată în cartea sa *The Master Key* prin intermediul unor ochelari care ajutau utilizatorul să observe că urșii erau buni prin litera G, malefici prin litera E și naivi prin litera F atașată lângă aceștia. În 1957, Morton Heilig dezvoltă *Sensorama* care oferă o experiență multisenzorială utilizatorului, combinând imagini stereoscopice, sunet binaural, mirosluri, vibrații și senzații. Ulterior, în 1966 apare primul HMD(head-mounted display) dezvoltat de Ivan Sutherland ajutat de studentul său Bob Sproull. [127]

### 9.4.1 Activitățile principale

*Realitatea augmentată* implică combinarea realității cu lumea virtuală, interacțiunea în timp real și obiecte 3D. [128]

1. Teorie:

(a) Definirea cadrului conceptual

Conceptul de *imersiune* se referă la integrare a obiectelor virtuale în viața reală.

Conceptul de *prezență* se referă la senzația utilizatorului că obiectele virtuale fac parte din lumea reală.

Conceptul de *interactivitate* se referă la capacitatea unui sistem, unei tehnologii sau unui mediu de a permite utilizatorului să interacționeze activ cu acesta. *Înregistrarea contextuală* implică capacitatea sistemelor RA de a adapta informațiile afișate în funcție de contextul fizic înconjurător. [128]

(b) Descrierea cadrului/cadrelor

*Imersiunea* se împarte în imersiune reprezentatională, imersiune participativă, imersiune afectivă și imersiune narativă. Imersiunea reprezentatională se referă la realismul vizual și auditiv al mediului înconjurător. Imersiunea participativă implică gradul de interactivitate oferit utilizatorului, cât de mult poate acesta să influențeze mediul înconjurător prin acțiunile lui. Imersiunea afectivă se referă la emoțiile pe care le creează experiența virtuală, contribuind la implicarea utilizatorului. Imersiunea narrativă se referă la cât de prins este utilizatorul în mediul înconjurător, la cât de captivat este de poveste, determinându-l pe acesta să exploreze mai mult. [111]

*Prezența* este influențată de realismul vizual, interactivitatea și

răspunsul senzorial. Aceasta conține diferite iluzii care ajută la o experiență mult mai realistă precum iluzia de a te afla într-un spațiu stabil, iluzia propriei întruchipări, iluzia interacțiunii fizice și iluzia comunicării sociale. [112]

*Interactivitatea* se manifestă prin capturarea de mișcare, urmărirea ochiului, feedback haptic, interacțiune vocală și urmărirea gesturilor. Capturarea de mișcare se bazează pe capturarea unui obiect 3D în spațiu, în timp real. Urmărirea ochiului și a mișcărilor acestuia care, atunci când utilizatorul reacționează la un stimул, acesta își focusează privirea pe el, iar astfel calculatorul detectează mișcarea ochiului. Feedback-ul haptic este resimțit de utilizator atunci când face diferite acțiuni precum deschiderea unei uși, apăsarea unui buton sau ridicarea unui obiect. Interacțiunea vocală permite utilizatorilor să vorbească între ei și să folosească comenzi vocale. Urmărirea gesturilor se realizează prin senzori diferenți poziționați pe mâinile sau picioarele utilizatorulu. [113]

(c) Metode de raționament utilizate.

- Raționamentul logic
- Modelare 3D și simulare în timp real

(d) Exemplificarea întrebărilor fundamentale

- Cum funcționează realitatea augmentată?
- Cum percep oamenii obiectele virtuale?
- De ce se simt oamenii „prezenți” când interacționează cu obiecte virtuale?
- Cum afectează realitatea augmentată comportamentele utilizatorilor?

2. Experiment:

(a) Explorarea modelelor:

*The Sword of Damocles*(1968, Ivan Sutherland) Este primul HMD (Head-mounted display). Acesta folosește două ecrane mici și un sistem de urmărire a mișcărilor capului pentru a ajusta imaginile afișate în funcție de poziția utilizatorului.[99]

*ARToolkit*(1999, Hirokazu Kato) Este primul framework open-source pentru dezvoltarea aplicațiilor RA. [129]

*Google Glass*(2013, Babak Parviz) Este un dispozitiv head-mounted care oferă suprapunerile digitale direct în câmpul vizual al utilizatorului. [130]

*Microsoft HoloLens*(2016, Alex Kipman) Sunt căști de tip mixed-

reality care combină realitatea augmentată cu realitatea virtuală pentru a crea holograme interactive.

(b) Implementări și validarea experimentală:

*Hardware*: ochelari AR (HoloLens, Magic Leap), telefoane mobile, tablete

*Software*: ARKit (Apple), ARCore (Google), Unity, Unreal Engine

(c) Exemple practice

*Educatie*: Se folosește la discipline precum biologie(vizualizarea organelor 3D), chimie(vizualizarea atomilor), geografie(vizualizarea reliefului unei țări sau a sistemului solar) și matematică(vizualizarea figurilor geometrice).[131]

*Medicina*: Se folosește la ghidare în timp real pentru intervenții chirurgicale, vizualizarea organelor interne. [132]

*Retail*: Aplicații care îți permit să încerci hainele sau mobila înainte să le cumperi. [133]

3. Design:

(a) Produse rezultate din cercetare:

- *Headset-uri AR*(Microsoft HoloLens, Magic Leap)
- *Ochelari inteligenți*(Google Glass, Vuzix Blade)
- *Sisteme de operare pentru telefoane cu AR*(iPhone cu ARKit, Android cu ARCore)

(b) Instrumente rezultate din cercetare:

- *Urmărire spațială* prin recunoaștere de imagini, marcatori sau SLAM
- *Ochelari inteligenți*(Google Glass, Vuzix Blade)
- *Sisteme de operare pentru telefoane cu AR*(iPhone cu ARKit, Android cu ARCore)

(c) Tehnici rezultate din cercetare:

- *Rendering în timp real* pentru a asigura grafică fluidă, fără întârzieri.
- *Spatial tracking* pentru recunoașterea de obiecte în timp real.
- *Haptică avansată* pentru simularea senzațiilor tactile în interacțiunea cu obiectele virtuale.

(d) Impactul designului asupra dezvoltării subdomeniului:

- *Experiențe* mai captivante și mai realiste care cresc gradul de imersiune.
- *Accesibilitate* îmbunătățită.

- Aplicații în educație, medicină, industrie și divertisment.
- Interacțiune avansată cu obiecte reale și virtuale.

4. Extra:

(a) Ce cunoștiințe sunt necesare pentru a intra în domeniu? [134]

- Programare: Python, C, C++, Java
- Software AR: ARToolkit, FlarToolkit
- Grafică 3D: Unity, Sketchup
- Matematică: Geometrie, Algebră liniară

(b) Literatura de specialitate

- International Journal of Virtual and Augmented Reality [135]

#### **9.4.2 Relațiile cu celelalte subdomenii**

Se leagă de *interacțiunea om-computer*, deoarece realitatea augmentată se folosește de gesturi și eye-tracking pentru a funcționa.

#### **9.4.3 Probleme Importante și Probleme Deschise**

Câteva probleme importante sunt *limitări hardware* care scad din imersiunea utilizatorului în mediul virtual și *conținut limitat și scump* care e produs greu, limitând domeniul la o evoluție rapidă.

Câteva probleme deschise sunt legate de *etică și consumămant* privind acțiunile realizate de utilizator în mediul virtual și *protecția datelor personale*, ochelarii AR folosindu-se în permanență de date biometrice pentru a oferi o experiență imersivă.

#### **9.4.4 Persoane Importante**

Cercetători: Mark Bolas [136], Steven Feiner [137], Mark Billinghurst [138]

#### **9.4.5 Forum-uri Importante**

Forum-uri de specialitate: Beyond3D, VR/AR[139]

#### **9.4.6 Dimensiunea locală și dimensiunea globală**

1. Local:

2. Global:

Colaborări internaționale: AWE (Augmented World Expo).

Organizații: IEEE ISMAR (International Symposium on Mixed and Augmented Reality), ARIA (Augmented Reality for Industry and Academia).

Standarde: OpenXR

Influența globală: Aplicații extinse în retail (IKEA Place), medicină (asistență AR în operații), educație STEM, turism virtual ghidat. [140]

## 9.5 Grafică pentru Jocuri Video

Subdomeniul graficii pentru jocuri video se ocupă cu generarea de conținut vizual (2D/3D) în timp real pentru aplicații interactive. Conținutul vizual trebuie optimizat la zeci sau sute de cadre pe secundă, astfel îngineria grafică de joc pune accent pe algoritmi și tehnologii pentru optimizare de performanță, fără a sacrifica realismul sau stilul artistic. După spusele specialiștilor efectele și grafica vizuală din jocurile moderne necesită atât talent artistic, cât și cunoștințe tehnice despre software-ul și hardware-ul utilizat. Grafica pentru jocuri video trebuie să echilibreze continuu calitatea imaginii cu elementele hardware și celelalte sisteme (inteligentă artificială, fizică de joc etc.) pentru funcționalitatea în timp real.

### 9.5.1 Activitățile principale

1. Teorie:

(a) Definirea cadrului conceptual

Se studiază modele matematice și fizice pentru reprezentarea scenelor.

Printre concepții fundamentale se numără pipeline-ul grafic de bază (stadii de transformare geometrică, rasterizare, shadere) și sisteme de lămpi virtuale care urmează legile fizicii optice. În general, se dezvoltă cadre de referință care includ geometria 3D, modele de textură și de lumină, precum și structuri de scene.

(b) Descrierea cadrului/cadrelor

Se analizează algoritmi de redare, modele de shaders (ex. Cook-Torrance pentru materiale metalice) și tehnici de gestionare a volumelor de date și se efectuează simulări și calcule grafice.

(c) Metode de raționament utilizate.

- Matematică aplicată
- fizică
- algoritmi grafici

(d) Exemplificarea întrebărilor fundamentale

- Cum simulăm lumina realist?
- Cum optimizăm pentru frame rate?

2. Experiment:

(a) Explorarea modelelor.

Implică testarea de modele grafice noi și compararea lor cu cele tradiționale. Se testează diferenți algoritmi de iluminare, umbre și reflectanță (ex. „tapetare” a suprafăcătorilor, modelări SSS – subsurface scattering) și eficiență compresiei de texturi sau a filtrării inteligente.

(b) Implementări și validarea experimentală.

Implementare de prototipuri în motoare grafice (folosind API-uri precum DirectX, Vulkan, OpenGL/GLSL)  
validate pe hardware real (GPU)

(c) Exemplu practic

Gestionarea obiectelor transparente: În randarea scenei, suprafăcătorile transparente necesită algoritmi speciali de ordonare. De exemplu, Painter's algorithm sortează poligoanele după adâncime (de la cele mai îndepărtate la cele mai apropiate) pentru a determina suprafăcătorile vizibile.

3. Design:

(a) Produse rezultate din cercetare.

- Motoare grafice (Unreal, Unity)
- API-uri (DirectX, Vulkan)

(b) Instrumente rezultate din cercetare.

- Shadere
- editoare de materiale PBR
- tooluri de profilare GPU
- motoare fizice integrate (de exemplu NVIDIA PhysX)

(c) Tehnici rezultate din cercetare.

- anormal mapping
- parallax occlusion mapping
- upscaling cu rețele neuronale DLSS
- ray tracing în timp real

(d) Impactul designului asupra dezvoltării subdomeniului.

- Implementarea algoritmilor de iluminare avansată a crescut semnificativ realismul vizual al jocurilor moderne. De asemenea, integrarea cunoștințelor de cercetare în produsele comerciale a permis apariția de hardware specializat și API-uri care acceptă noi paradigme grafice.

### 9.5.2 Relațiile cu celelalte subdomenii

- AI și CGI  
Pentru generare procedurală, texturi, neural rendering, simulări pentru lichide, păr, coliziuni(integrate în randare).
- Inginerie software  
Optimizări GPU, limbaje HLSL/GLSL.
- VR/AR  
Cerințe grafice stricte (latență, stereo, realism).

### 9.5.3 Probleme Importante și Probleme Deschise

- Randare globală în timp real
  - Obținerea iluminării realiste la o calitate înaltă pe diferite cadre este dificilă, iar realizarea unor metode de simulare a luminii indirecte fără penalizări de performanță este un obiectiv încă deschis.
- Performanță și resurse
  - Graficienii trebuie să balanseze în permanență calitatea vizuală cu capacitatele hardware. Tehnologiile noi (ray tracing în timp real, neural rendering) sunt de multe ori reprimate de capacitatea actuală a GPU-ului.
- Redarea obiectelor transparente
  - Gestionarea transparenței complexe (de ex. a fumului sau a apei) și a efectelor volumetrice e complicată în timp real. Metodele clasice de sortare profundă (ex. Painter's algorithm) au limitări când scenele devin dinamice. Producătorii de jocuri se confruntă cu astfel de probleme în producție.

#### **9.5.4 Persoane Importante**

- John Carmack - pionier 3D (Doom, Quake)
- Tim Sweeney - fondator Unreal Engine
- Tomas Akenine-Möller - autor "Real-Time Rendering"
- Bart Wronski - specialist randare real-time (NVIDIA)

#### **9.5.5 Forumuri Importante**

- Conferințe: SIGGRAPH, GDC, Eurographics, Dev.Play (RO).
- Publicații: ACM TOG, Computer Graphics Forum.
- Studiouri: Epic, Ubisoft, Naughty Dog, Gameloft.

#### **9.5.6 Dimensiunea locală și dimensiunea globală**

1. Local:

- UVT oferă cursuri de grafică în cadrul programelor de Informatică și Arte Digitale, precum: Grafică pe calculator, Animatie 2D/3D, Proiectare de jocuri.
- Facultatea de Arte și Design oferă un program de 2 ani de masterat în arta jocurilor video.

2. Global:

- Studiouri AAA: Epic Games, CD Projekt Red, Rockstar, Sony, Naughty Dog, Blizzard

### **9.6 Grafică Computerizată pentru Imagini Generate pe Calculator (CGI)**

Grafica computerizată reprezintă subdomeniul care se ocupă cu generarea de conținut vizual static sau animat folosind software de imagistică”, și este folosită pentru o multitudine de aplicații precum artă vizuală, publicitate, modelare anatomică, design arhitectural, inginerie, televiziune, jocuri video și efecte speciale de film, precum și aplicații AR/VR. CGI-ul include atât grafica 2D, cât și 3D, fiind folosit în special pentru designul de personaje, medii virtuale și scene cu efecte speciale pentru filme, televiziune, reclame și jocuri video.

### 9.6.1 Activitățile principale

#### 1. Teorie:

##### (a) Definirea cadrului conceptual

CGI-ul se bazează pe modele matematice și fizice ale luminii și materialelor. Cel mai important instrument este ecuația de randare, o ecuație integrală în care radianța care părăsește un punct este exprimată prin suma dintre radianța emisă și cea reflectată sub o aproximare optică geometrică. Aceasta rezultă din principiul conservării energiei și stă la baza majorității tehniciilor de iluminare realistă.

O altă componentă esențială este BRDF-ul (funcția de distribuție bidirectională a reflectanței), care descrie matematic cum se reflectă lumina pe o suprafață opacă.

Aceste modele fizice permit caracterizarea materialelor și asigură că simularea respectă legile opticii.

##### (b) Descrierea cadrului/cadrelor

Există diferite „cadre” de studiu, cum ar fi pipeline-ul grafic tradițional (modelare → animație → iluminare → randare) și metodele de redare fizică.

Se utilizează la scară largă PBR-ul (Physically Based Rendering), care folosește algoritmi și formule bazate pe comportamentul real al luminii pentru a obține imagini fotorealiste.

##### (c) Metode de raționament utilizate.

Se utilizează atât deducții analitice cât și metode experimentale, precum simulările Monte Carlo, care sunt folosite pentru a rezolva ecuația de randare, permitând simularea indirectă a luminii prin multiple reflexii.

##### (d) Exemplificarea întrebărilor fundamentale

- Cum simulăm lumina realist?
- Cum redăm materiale (ex. piele, metal)?

#### 2. Experiment:

##### (a) Explorarea modelelor.

Cercetarea experimentală implică testarea practică a modelelor teoretice. De exemplu în simulările pe fluide sunt comparate două abordări principale: simularea pe grilă (punkte fixe Euleriene, cu acuratețe numerică ridicată) și simularea pe particule, care e mai rapidă dar rezultatul arată mai puțin neted.

##### (b) Implementări și validarea experimentală.

- Prototipuri software: scripturi, plugin-uri, sisteme dedicate
- Compararea imaginilor generate cu date reale (fotografii, scanări 3D)
- Comparare cu capturi reale, măsurători vizuale

(c) Exemplu practic

Motion capture facial sau reconstruirea 3D a unui om dintr-un video

3. Design:

(a) Produse rezultate din cercetare.

- Filme CGI (Jurassic Park, Avatar, Toy Story)
- reclame
- jocuri

(b) Instrumente rezultate din cercetare.

- Blender(open source), RenderMan(dezvoltat de Pixar), Unreal Engine(motor grafic în timp real)
- Cadre de simulare fizică (ex. biblioteci de simulare a fluidelor sau a țesăturilor) și platforme de dezvoltare a efectelor vizuale

(c) Tehnici rezultate din cercetare.

- PBR (asigură consistența materialelor, simulând comportamentul real al luminii pe suprafete)
- simulare păr/haine
- ray tracing
- neural rendering

(d) Impactul designului asupra dezvoltării subdomeniului.

- A dus la realism vizual ridicat în industrie prin designul de noi produse, instrumente și motoare grafice

### 9.6.2 Relațiile cu celelalte subdomenii

• Grafică 3D

- CGI folosește la bază algoritmi și concepte din grafica 3D (reprezentări geometrice, texturi, shaderi) și realizează animații complexe.

• Inteligență artificială

- AI-ul este utilizat pentru generarea sau îmbunătățirea imaginilor CGI.

- De asemenea, scenele generate cu ajutorul CGI-ului sunt o sursă importantă pentru antrenarea AI. Generarea automată de imagini și video ajută la crearea seturilor de date de antrenament pentru percepție computerizată, simulând și situații dificil de capturat în lumea reală.
- Vizualizări medicale
  - În medicină, CGI contribuie la vizualizarea datelor complexe (ex. reconstruirea 3D a organelor din imagini CT/MRI) și la simulări științifice (ex. modelarea fluxului sangvin).
- Film, televiziune și publicitate

#### **9.6.3 Probleme Importante și Probleme Deschise**

- Realismul pielii
  - Redarea realistă a pielii umane necesită simularea subsurface scattering. Dacă nu se modelează acest efect, pielea 3D ar părea plastică și nenaturală. 94% din lumina reflectată de pe piele provine din reflexia internă, adică scattering în interiorul materialului, nu din reflexia directă de la suprafață.
- Simularea părului și a firelor de iarbă
  - Părul, blana și iarba sunt formate din mii de elemente foarte subțiri, fiecare cu comportament optic particular. Problemele deschise includ calculul interacțiunilor dinamice (coliziuni, mișcare sub influența vântului) și randarea eficientă (în timp real este extrem de costisitor).
  - Simularea și iluminarea multor fire semi-transparente, păstrând detaliu și fluiditate, este încă o provocare practică majoră.
- Redarea fluidelor și a altor elemente
  - Simularea vizuală a apei, focului sau a norilor presupune rezolvarea ecuațiilor Navier–Stokes și a altor ecuații complexe. În cercetare se testează modele Euleriene (grilă) și Lagrangeiene (particule). Deși acestea sunt algoritmi bine definiți, necesitatea găsirii unui compromis între acuratețe și performanță face ca obținerea efectelor complet realiste să rămână încă o problemă deschisă.

#### **9.6.4 Persoane Importante**

- Edwin Catmull
  - Co-fondator Pixar și pionier al graficii computerizate, câștigător al multiplor premii prestigioase, printre care și un Oscar pentru progrese semnificative în domeniul randării filmelor, precum RenderMan de la Pixar.
- Pat Hanrahan
  - Co-fondator Pixar și câștigător a trei premii Oscar pentru munca sa în cercetarea randării și graficii pe calculator.
- James Kajiya
  - Autorul ecuației de randare
- John Knoll
  - Director de creație la ILM și co-creator al Adobe Photoshop. A supravegheat efecte vizuale pentru multe filme Star Wars și Star Trek. A avut o contribuție esențială pentru folosirea CGI în producțiile cinematografice.

#### **9.6.5 Forumuri Importante**

- Conferințe: SIGGRAPH, Eurographics.
- Reviste: ACM TOG, Computer Graphics Forum.
- Studiouri: Pixar, ILM, Weta Digital, Sony Imageworks.

#### **9.6.6 Dimensiunea locală și dimensiunea globală**

1. Local:

- La UVT, elemente de grafică computerizată pot fi întâlnite în mai multe facultăți:
  - Facultatea de Matematică și Informatică:
    - \* Programe de licență/master care includ cursuri precum: Grafică pe calculator, Vizualizare 3D, Procesare de imagini, Inteligență artificială aplicată în CGI
  - Facultatea de Arte și Design:

- \* Domenii precum arte digitale, design grafic, animație, arte multimedia

2. Global:

- Industria cinematografică și de divertisment
  - Hollywood și studiourile internaționale folosesc CGI pentru efecte vizuale în filme precum Avatar, Avengers, Dune sau The Lion King
  - Studiouri de animație precum Pixar, DreamWorks, Disney, Weta Digital, Industrial Light & Magic (ILM), Sony Imageworks

## 9.7 Vizualizarea Datelor

*Vizualizarea datelor* constă în reprezentarea vizuală a datelor cu ajutorul graficelor, charturilor, ploturilor sau a animațiilor.

Primele reprezentări vizuale din istorie se regăsesc în reprezentarea stelelor de pe cer și în hărțile pentru exploratori și marinari. Claudius Ptolemy [c.85-c.165] a realizat prima reprezentare sferică a Pământului care a reprezentat un standard până în secolul XIV. Pe la jumătatea secolului XIX, începe The Golden Age of Data Visualization care folosește acest procedeu în organizarea de evenimente, industrializare, comerț și transport. În 1957, apare Fortran, primul limbaj de nivel înalt care permite realizarea de statistici de date. Un alt limbaj esențial folosit în acest domeniu este R care apare în 1993. [141]

### 9.7.1 Activitățile principale

Vizualizarea datelor se poate realiza în mai multe moduri. *Line graph* arată relațiile dintre obiecte și compară schimbările pe durata unei perioade de timp. *Bar chart* se folosesc pentru a compara cantități din diferite categorii. *Scatter plot* este un plot 2-dimensional care arată variația a două obiecte. *Pie chart* se folosește pentru a compara părți dintr-un întreg. *Heat Map* e utilă când trebuie observată distribuția unor valori într-o matrice sau într-un tabel. Culoarea fiecărei celule reprezintă magnitudinea unor date

*SciVis* se referă la vizualizarea datelor științifice care ajută cercetătorii să înțeleagă mai bine anumite procedee. [142]

1. Teorie:

(a) Definirea cadrului conceptual

*SciVis* se referă la vizualizarea datelor științifice care ajută cercetătorii să înțeleagă mai bine anumite procedee.

*InfoVis* se referă la vizualizarea datelor numerice și non-numerice

care contribuie la observarea rapidă a unor pattern-uri.

*Visual Analytics* se referă la analizarea datelor și combină data mining cu statistică. [143]

(b) Descrierea cadrului/cadrelor

*Scivis* se ocupă de animație pe computer, simulare pe computer și vizualizare de informații. Are aplicații în științe ale naturii, geografie, ecologie și matematică.

*InfoVis* ajută la înțelegerea datelor prin hărți, grafice, clădiri 3D și town plan designs. [144]

*Visual Analytics* combină tehnici de automatizare a analizei datelor și vizualizări interactive a unor seturi de date largi și complexe. [145]

(c) Metode de raționament utilizate.

Raționamentul logic

Analiza datelor și perceptia vizuală

(d) Exemplificarea întrebărilor fundamentale

- Cum pot fi reprezentate vizual date complexe?
- Ce tipuri de reprezentări sunt eficiente pentru ce tipuri de date?

2. Experiment:

(a) Explorarea modelelor.

(b) Implementări și validarea experimentală.

(c) Exemplu practic (ex.: metode de sortare și relevanța lor).

3. Design:

(a) Produse rezultate din cercetare

- *Power BI* este o platformă de analiză a datelor afacerilor dezvoltată de Microsoft.
- Instrumente rezultate din cercetare.[146]
- *Tableau* este un instrument de vizualizare interactivă a datelor care este utilizat în știință și finanțe.
- *D3.js* e o bibliotecă JavaScript care produce vizualizări de date dinamice în browsere web.
- *Google Data Studio* e un instrument web-based data vizualisation.

(b) Tehnici rezultate din cercetare: t-SNE, PCA

(c) Impactul designului asupra dezvoltării subdomeniului: Designul eficient al organizării datelor a ajutat la o înțelegere mai bună a acestora. În domeniul *sănătății*, oferă abilitatea de analiză și prezentare a datelor într-o manieră ușor de înțeles. În domeniul *financiar*, vizualizarea de date se folosește pentru a descoperi posibile cazuri de fraudă, posibile pattern-uri. În *industria*, ajută la interpretarea ușoară a datelor cantitative ale multor companii mari.

4. Extra:

(a) Ce cunoștiințe sunt necesare pentru a intra în domeniu?

- Programare: Python, JavaScript
- Statistică
- Design

(b) Literatura de specialitate

- International Journal of Visualization[147]

### 9.7.2 Relațiile cu celelalte subdomenii

Se leagă de *interacțiunea om-computer* și *inteligentă artificială*, deoarece ajută la vizualizarea datelor și la interpretarea acestora.

### 9.7.3 Probleme Importante și Probleme Deschise

Câteva probleme importante sunt *gestionarea de big data* care face domeniul mai greu accesibil pentru programatori și *vizualizarea de big data*.

Câteva probleme deschise sunt *dezvoltarea de metode automate* care să genereze reprezentări utile pentru diferite seturi de date și *integrarea de feedback* în timp real a erorilor întâmpinate de utilizatori.

### 9.7.4 Persoane Importante

Cercetători: Jeffrey Heer [148], Tamara Munzner [149], Hanspeter Pfister [150]

### 9.7.5 Forum-uri Importante

Forum-uri de specialitate: Visualizing[151]

### **9.7.6 Dimensiunea locală și dimensiunea globală**

#### **1. Local:**

În cadrul Facultății de Matematică și Informatică a Universității de Vest din Timișoara, la master, studenții pot opta pentru Big Data [152]

#### **2. Global:**

Colaborări internaționale: Consorții globale(IEEE Vis).

Influența globală: Sănătate, Industrie și producție

## **9.8 Grafică Interactivă(GUI)**

Interfața grafică cu utilizatorul (GUI) reprezintă partea de interacțiune om-computer bazată pe elemente vizuale (ferestre, iconițe, meniuuri, butoane etc.) și dispozitive de pointare (pointerul mouse-ului). GUI-ul permite utilizatorilor să comunice cu calculatorul folosind simboluri și metafore vizuale familiare. Conceptul cheie al GUI este manipularea directă: utilizatorul acționează direct asupra obiectelor vizuale și obține feedback imediat. Acest lucru transformă calculatoarele moderne (ex. Windows, macOS) în medii intuitive, înlocuind vechiul mod de operare bazat pe text.

### **9.8.1 Activitățile principale**

#### **1. Teorie:**

##### **(a) Definirea cadrului conceptual**

În proiectarea interfețelor, se definește un model conceptual al sistemului, care descrie modul de organizare și comportamentul artefactului interactiv. Model conceptual face legătura dintre acțiunile vizuale ale utilizatorului și structura internă a aplicației.

##### **(b) Descrierea cadrului/cadrelor**

GUI-urile se bazează pe cadre și teorii din științele cognitive și HCI. De exemplu, Modelul Procesorului Uman (dezvoltat de Stuart K. Card, Thomas P. Moran și Allen Newell în 1983) este o metodă utilizată pentru a calcula timpul necesar pentru a efectua o anumită sarcină. Alte concepții includ modelul GOMS (care analizează pașii cogniției), legi empirice precum Legea lui Fitts (timpul de selectare a unui obiect crește cu distanța și scade cu dimensiunea obiectului), etc.

##### **(c) Metode de raționament utilizate.**

Cercetătorii folosesc modele cognitive și raționamente formale pentru a anticipa comportamentul utilizatorului. Se calculează timpii cu

ajutorul Modelul Procesorului Uman și apoi se aplică Legea lui Fitts pentru optimizarea designului.

(d) Exemplificarea întrebărilor fundamentale

- Cum facem interfețele ușor de înțeles?
- Ce dimensiuni au butoanele ideale?

2. Experiment:

(a) Explorarea modelelor.

În faza de experimentare se construiesc diferite modele și prototipuri ale interfeței pentru a prezice și testa comportamentul utilizatorilor.

(b) Implementări și validarea experimentală.

Implementare începe cu crearea unui prototip care permite evaluarea scenariilor de utilizare. De exemplu, se pot folosi toolkit-uri UI (Qt, WPF, Android SDK etc.) pentru a construi interfețe cu ferestre, meniuuri și controale interactive pe care se execută apoi teste pentru validare experimentală.

Pentru validare echipele de cercetare efectuează testări de uzabilitate cu grupuri de utilizatori reali. Aceștia primesc anumite sarcini de îndeplinit și se măsoară ușurința cu care îndeplinesc sarcinile din punctul de vedere al timpului de completare, erori, satisfacție, etc.

3. Design:

(a) Produse rezultate din cercetare.

- Sisteme de operare (Windows, macOS, Android)
- aplicații cu UI modern
- biblioteci grafice ( Qt, GTK, Swing)

(b) Instrumente rezultate din cercetare.

- instrumente de analiză și monitorizare a interacțiunii (eye-tracker, loguri de evenimente UI)

(c) Tehnici rezultate din cercetare.

- Responsive design
- animații UI
- design centrat pe utilizator
- manipulare directă

(d) Impactul designului asupra dezvoltării subdomeniului.

- Interfețele grafice definesc experiența interacțiunii cu software-ul și de aceea s-au stabilit convenții care asigură consistența aplicațiilor. De asemenea, impactul interfeței grafice al diferitelor programe a dus la apariția unor inovații precum ecranele tactile sensibile la presiune și gesturile spațiale VR/AR.

### 9.8.2 Relațiile cu celelalte subdomenii

- Interacțiunea Om-Calculator
  - GUI se axează pe designul centrat pe utilizator, design-ul interfețelor vizuale și experiența utilizatorului. Acesta facilitează interacțiunile om-calculator, fănd aplicațiile să fie mai ușor de utilizat, fiind mai intuitive.
- Ingineria software
  - Designul interfeței cu utilizatorul este un aspect crucial al ingineriei software, deoarece acesta determinează modul în care utilizatorii interacționează cu aplicația.
- Realitate Virtuală / Realitate Augmentată (VR/AR)

### 9.8.3 Probleme Importante și Probleme Deschise

- Accesibilitate:
  - Se estimează că 15-20% din populație are o dizabilitate, iar alt 15–20% din populație este neurodivergentă. Dezvoltatorii trebuie să implementeze design inclusiv, cum ar fi text contrastant, navigare compatibilă cu tastatura, elemente compatibile cu cititoare de ecran etc.
  - Chiar dacă se observă progrese din acest punct de vedere, testarea accesibilității și adaptarea interfețelor la nevoi variate rămân probleme deschise majore.
- Limitarea timpului de răspuns:
  - Timpii de răspuns ai interfețelor interactive sunt criticați constant. Regula generală spune că o întârziere sub 0,1 secunde este percepță ca reacție instantanee, până la 1 secundă utilizatorul o observă dar este suportabilă, iar peste 10 secunde utilizatorul își pierde complet răbdarea.
  - GUI-urile trebuie optimizate astfel încât să prezinte reacții foarte rapide: de la timpul de redare a animațiilor, la întârzierile induse de rețea sau hardware.

- Consistență cross-platform:
  - Utilizatorii comută constant între dispozitive diferite (PC, telefon, tabletă, VR) și se așteaptă să găsească interfețe și comportamente similare independent de platformă.
  - Realizarea acestei consistențe este dificilă din cauza dimensiunilor și rapoartelor de aspect diferite între diferite ecrane, metodelor de pointare diferite (touch vs cursor), și anumite limitări de performanță. Dezvoltatorii trebuie să decidă ce elemente rămân comune și unde se adaptează interfața pentru fiecare mediu.

#### **9.8.4 Persoane Importante**

- Douglas Engelbart: inventatorul mouse-ului.
- Alan Kay: pionier în domeniul GUI.
- Ben Shneiderman: dezvoltator al conceptului de manipulare directă.
- Donald Norman & Jakob Nielsen: au dezvoltat principii de design și uzabilitate.

#### **9.8.5 Forumuri Importante**

- Conferințe: ACM CHI, UIST, IUI.
- Reviste: ToCHI, Int. Journal of Human-Computer Studies.

#### **9.8.6 Dimensiunea locală și dimensiunea globală**

1. Local:

- UVT oferă în cadrul Facultății de Matematică și Informatică cursuri de Interacțiune om-calculator, Grafică pe calculator, Web design.
- Facultatea de Arte și Design oferă noțiuni de design vizual, compozиie, layout, utile pentru GUI.

2. Global:

- Marile companii, cum ar fi Apple, Google, Microsoft, Tesla, Meta, investesc masiv în cercetare și inovație în GUI, fiind esențial în toate sectoarele tech: aplicații mobile, software, AI, etc.

## 10 Interacțiunea Om-Computer

Interacțiunea om-computer (HCI) este un domeniu interdisciplinar care studiază modul în care utilizatorii interacționează cu sistemele informaticе. HCI a apărut la începutul anilor 1980 ca o specializare a informaticii, înglobând în egală măsură știința cognitivă și ergonomia (factorii umani)[153]. Din 1980 încoace, domeniul s-a extins rapid, atrăgând specialiști din diverse discipline și formând o aglomerare de subdomenii semi-autonome. În HCI se pune accent pe proiectarea centrată pe utilizator, pe utilizabilitate și experiența utilizatorului.

Subdomeniile principale ale HCI includ:

- Interacțiunea om-computer în medii colaborative
- Factorii umani și psihologia interacțiunii
- Interacțiunea om-computer în contexte specifice
- Securitate și etică în HCI
- Designul interfețelor utilizatorului (UI/UX Design)
- Interacțiunea bazată pe inteligență artificială

Fiecare dintre aceste subdomenii este abordat în continuare detaliat.

### 10.1 Designul interfețelor utilizatorului (UI/UX Design)

Acest subdomeniu se axează pe crearea interfețelor prin care utilizatorii interacționează cu aplicații și sisteme informaticе. UI (User Interface) se referă la aspectele vizuale și interactive ale interfeței — butoane, meniuri, culori, layout — în timp ce UX (User Experience) vizează experiența completă a utilizatorului, inclusiv ușurința în utilizare, satisfacția și eficiența în îndeplinirea sarcinilor.

UI/UX a evoluat de la interfețe bazate pe linie de comandă (CLI) în anii '70, la interfețe grafice (GUI) în anii '80-'90 și, mai recent, la interfețe naturale (NUI) bazate pe atingere, voce sau gesturi. Modele conceptuale precum WIMP (Windows, Icons, Menus, Pointer) au definit multe dintre paradigmile moderne de design.

#### 10.1.1 Activitățile principale

##### 1. Teorie:

- (a) Cadre conceptuale: Design Thinking, User-Centered Design, Activity Theory.

(b) Întrebări fundamentale: Cum percep utilizatorul informația? De ce abandonează o pagină? Ce influențează satisfacția?

## 2. Experiment:

- (a) Explorarea prototipurilor: mockup-uri, wireframe-uri, interfețe interactive (Figma, Adobe XD).
- (b) Validare: teste A/B, usability testing, eye-tracking.
- (c) Exemplu: compararea a două interfețe de login pentru eficiență și rată de eroare.
- (d) Wow: interfețe care se adaptează dinamic comportamentului utilizatorului (ex. Spotify).

## 3. Design:

- (a) Produse: aplicații mobile, dashboard-uri web, interfețe enterprise.
- (b) Instrumente: Figma, Adobe XD, Sketch, InVision.
- (c) Tehnici: microinteracțiuni, responsive design, color theory.
- (d) Impact: interfețe intuitive reduc rata de abandon și cresc satisfacția.

## 4. Extra:

- (a) Cunoștințe necesare: psihologie cognitivă, design vizual, ergonomie, HTML/CSS/JS.
- (b) Literatură de specialitate:
  - Donald A. Norman - *The Design of Everyday Things*. [96]
  - Alan Cooper et al - *About Face: The Essentials of Interaction Design*. [154]
  - Susan Weinschenk - *100 Things Every Designer Needs to Know About People*.[155].

### 10.1.2 Relațiile cu celelalte subdomenii

UI/UX este strâns legat de:

- Inteligență artificială – interfețe adaptive, personalizare automată.
- Interacțiunea colaborativă – design pentru lucru simultan, vizibilitate și sincronizare.
- Factorii umani – ergonomie, percepție vizuală, limitări cognitive.

### **10.1.3 Probleme importante și probleme deschise**

- Provocări: interfețe accesibile pentru dizabilități, evitarea dark patterns.
- Probleme deschise: UI-uri care învață, măsurarea satisfacției, interfețe afective.

### **10.1.4 Persoane importante**

1. Don Norman – design centrat pe utilizator. [96]
2. Alan Cooper – creatorul conceptului de personas.[154]
3. Susan Weinschenk – expertă în psihologia utilizatorului.[155]
4. Jakob Nielsen – autor al euristicilor UI. [156]

### **10.1.5 Forumuri importante**

- Conferințe: CHI, UXPA International, Interaction (IxDA).
- Reviste: TOCHI, UXPA Journal, Interactions.
- Comunități: SIGCHI, UX Design Community.

### **10.1.6 Dimensiunea locală și globală**

- **Local (UVT):** Cursuri precum „Proiectarea interfețelor grafice”, lucrări de licență și masterat în UX.
- **Global:** Standardele ISO 9241, comunități ca ACM SIGCHI, companii de top: Google, Apple, Microsoft.

## **10.2 Interacțiunea bazată pe inteligență artificială**

Interacțiunea om-computer bazată pe inteligență artificială (AI-HCI) reprezintă un subdomeniu în plină dezvoltare, aflat la intersecția dintre tehnologiile inteligente și principiile designului centrata pe utilizator. Aceasta explorează moduri prin care sistemele interactive pot deveni nu doar reactive, ci și proactive, personalizabile și sensibile la contextul utilizatorului. În loc ca sistemul să răspundă doar la comenzi, interacțiunea bazată pe AI încearcă să anticipateze nevoile utilizatorului, să-i înțeleagă preferințele și chiar să colaboreze cu el.

Evoluția acestui subdomeniu este direct legată de avansul rapid în domenii precum machine learning, procesarea limbajului natural (NLP), viziune computerizată,

sisteme conversaționale și modele generative. Astăzi, aplicații precum asistenții vocali (Siri, Alexa), sistemele de recomandare (Netflix, Spotify) sau chatbot-urile inteligente integrează AI pentru a oferi o interacțiune fluidă, adaptată și mai naturală.

### 10.2.1 Activitățile principale

#### 1. Teorie:

- (a) Interacțiunea bazată pe AI este fundamentată de modele de învățare automată
- (b) User modeling (modelarea comportamentului utilizatorului)
- (c) Conceptul de explainable AI (XAI),
- (d) Teoriile legate de Human-AI teaming — colaborarea eficientă între om și sistemul intelligent.
- (e) Întrebările esențiale vizează încrederea utilizatorului în sistem, gradul de autonomie acceptabil al AI și nevoia de transparentă în deciziile automate.

2. **Experiment:** Activitatea experimentală include construirea și testarea de prototipuri — chatboturi conversaționale, asistenți digitali, sisteme de predicție a comportamentului sau de recomandare. Acestea sunt evaluate prin studii de utilizabilitate, măsurători de performanță, evaluări de încredere și acceptare socială. De exemplu, un experiment poate compara două asistenți vocali: unul bazat pe reguli simple și altul pe AI avansat, observând diferențele de satisfacție și eficiență în utilizare.

3. **Design:** Designul în AI-HCI implică nu doar aspecte vizuale, ci și modul în care AI-ul comunică deciziile sale și se face înțeles de utilizator. Un exemplu este integrarea explicațiilor automate (de ce a fost făcută o recomandare), personalizarea dinamică a interfeței în funcție de context și comportament, sau suportul decizional în aplicații complexe. Produsele rezultate includ aplicații vocale, interfețe adaptive, și platforme inteligente care învăță și se transformă în timp.

#### 4. Extra:

- (a) Pentru a lucra eficient în acest subdomeniu este nevoie de competențe în AI, machine learning, interacțiune om-calculator, dar și etică și reglementări.
- (b) Literatura de bază include:

- „Artificial Intelligence: A Modern Approach” de Russell și Norvig
- Lucrările lui Ben Shneiderman despre HCI și AI [157]
- Ghiduri practice precum cele propuse de Saleema Amershi privind integrarea AI în produse interactive. [158]

#### **10.2.2 Relațiile cu celelalte subdomenii**

AI-HCI influențează direct UI/UX prin crearea de interfețe adaptive și inteligente, care se personalizează automat. În medii colaborative, agenții inteligenți pot sprijini coordonarea echipelor, oferî asistență contextuală și pot intermedia comunicarea. Totodată, AI ridică noi provocări pentru designul etic și pentru înțelegerea factorilor umani implicați în acceptarea automatelor.

#### **10.2.3 Probleme importante și probleme deschise**

Provocările actuale includ lipsa transparentei algoritmilor (black-box models), biasul algoritmic, lipsa de control perceput de utilizator și lipsa de încredere. Problemele deschise se referă la modul în care putem crea AI care oferă explicații clare, care menține controlul utilizatorului și care colaborează eficient cu oameni în sarcini complexe, menținând în același timp etica și confidențialitatea.

#### **10.2.4 Persoane importante**

1. **Ben Shneiderman** – susținător al interacțiunii centrate pe om în AI. [157]
2. **Ece Kamar** – cercetătoare Microsoft, preocupată de AI echitabil.
3. **Saleema Amershi** – autoarea principiilor pentru designul interfețelor AI. [158]

#### **10.2.5 Forumuri importante**

- Conferințe: CHI, ACM IUI (Intelligent User Interfaces), NeurIPS, AAAI.
- Reviste: ACM TOCHI, AI & Society, International Journal of Human–Computer Studies.

#### **10.2.6 Dimensiunea locală și globală**

- **Local (UVT):** Cursurile și laboratoarele de inteligență artificială, interacțiune om-calculator și programare avansată oferă baza pentru cercetare în AI-HCI. Studenții pot participa la proiecte interdisciplinare sau pot dezvolta aplicații cu componente AI în lucrările de disertație.

- **Global:** La nivel internațional, AI-HCI este un domeniu strategic, cu impact în industrie (Google, Meta, Amazon), în reglementări (AI Act în UE), și în etică (prin consilii de AI responsabile). Colaborările internaționale între laboratoare HCI și centre de AI sunt esențiale pentru definirea standardelor de transparentă și design etic.

## 10.3 Interacțiunea om-computer în medii colaborative

Interacțiunea om-computer în medii colaborative, cunoscută academic și sub numirea de CSCW (Computer-Supported Cooperative Work), este o ramură esențială a domeniului HCI care se concentrează pe sprijinirea colaborării dintre oameni prin intermediul tehnologiei. Obiectivul principal este de a permite indivizilor și grupurilor să colaboreze eficient, indiferent de locație, timp sau dispozitiv, utilizând sisteme informatiche interactive.

CSCW s-a dezvoltat în anii '80, odată cu apariția sistemelor de poștă electronică și a primelor platforme de colaborare în rețea. În prezent, odată cu tranziția accelerată către munca la distanță și în echipe distribuite, acest subdomeniu capătă o importanță strategică, explorând noi forme de interacțiune asincronă, sincronică, hibridă și augmentată digital.

### 10.3.1 Activitățile principale

1. **Teorie:** Teoriile din CSCW includ concepte precum awareness (conștientizarea activității celorlalți), workspace sharing, co-prezență, sincronizare și coordonare. Sunt investigate barierele comunicătionale, modul în care tehnologia afectează dinamica grupurilor și ce înseamnă „prezență” într-un mediu digital. Se utilizează teoria activității distribuite și modelele socio-tehnice pentru a înțelege colaborarea mediată.
2. **Experiment:** În cercetare, se testează platforme colaborative (ex. Google Docs, Miro, Slack, Notion), analizând impactul asupra coordonării, încrederei și performanței de echipă. Se folosesc studii de caz longitudinale, observarea interacțiunii live, analiza logurilor de activitate și chestionare privind satisfacția și utilitatea.
3. **Design:** Designul interfețelor colaborative presupune crearea de spații parțajate, feedback în timp real, indicatori de prezență, istoricul modificărilor și mecanisme de revenire (undo colaborativ). Produsele includ aplicații de videoconferință (Zoom, MS Teams), tablouri interactive (Jamboard), platforme educaționale colaborative sau chiar sisteme de realitate augmentată în echipă.

**Extra:** Lucrul în acest domeniu presupune competențe în psihologie socială, comunicare mediată de calculator, etnografie digitală și design participativ. Sunt de interes aspecte culturale (diferențe interculturale în colaborare), juridice (proprietate intelectuală în munca partajată), și etice (monitorizarea utilizatorilor în echipe).

#### 10.3.2 Relațiile cu celelalte subdomenii

CSCW interacționează direct cu UI/UX prin nevoia de interfețe ușor de învățat și folosit în grup. Inteligența artificială este integrată în sisteme colaborative prin agenți asistenți, recomandări automate sau gestionarea conflictelor de editare. Etica, securitatea și designul centrat pe utilizator rămân în centrul atenției.

#### 10.3.3 Probleme importante și probleme deschise

Provocările actuale includ fragmentarea atenției, suprasarcina informațională, lipsa contextului social, barierele tehnice și culturale. Problemele deschise se referă la crearea unor spații digitale care susțin colaborarea creativă, scalarea colaborării distribuite și integrarea AR/VR pentru co-prezență sporită.

#### 10.3.4 Persoane importante

1. **Irene Greif și Paul Cashman** – definiția inițială a domeniului CSCW.
2. **Jonathan Grudin** – autor al teoriei celor opt provocări în designul de groupware. [159], [160]
3. **Wendy Mackay** – cercetătoare în colaborare augmentată digital.
4. **Thomas Malone** – studii privind structura și dinamica grupurilor online.

#### 10.3.5 Forumuri importante

- Conferințe: ACM CSCW, ECSCW, ACM CHI.
- Reviste: Journal of Computer-Supported Cooperative Work, ACM TOCHI.
- Comunități: SIGCHI, special interest groups pe colaborare distribuită și învățare colaborativă.

### 10.3.6 Dimensiunea locală și globală

- **Local (UVT):** În cadrul UVT, proiectele colaborative în HCI se dezvoltă în contextul cursurilor despre comunicare digitală, design participativ și interacțiune în echipe hibride. Studenții folosesc platforme ca Figma, Miro sau GitHub pentru proiecte distribuite.
- **Global:** CSCW este susținut global de rețele de cercetare (ex. COST, Horizon Europe), de laboratoare precum Microsoft Research, și de inițiative de standardizare a interoperabilității în colaborare digitală. Domeniul este strategic pentru educație, telemuncă și inovație digitală în echipe.

## 10.4 Factori umani și psihologia interacțiunii

Această ară HCI pune accent pe studierea utilizatorilor ca ființe umane cu caracteristici psihofiziologice și cognitive specifice. Factorii umani (ergonomia) provin din nevoie de a proiecta echipamente utilizabile încă din perioada celui de-al Doilea Război Mondial, focalizându-se pe aspecte senzorio-motorii ale interacțiunii (*interaction-design.org*).

În anii '80, cercetătorii HCI (din domeniul ergonomiei și al psihologiei cognitive) au început să pună accentul pe procesarea informației și pe cuplarea cognitivă dintre om și calculator (*interaction-design.org*). Astfel, psihologia interacțiunii se preocupă de procese mentale precum percepția, memoria, atenția și rezolvarea problemelor în interacțiunea cu interfața (*interaction-design.org*). Se studiază cum limitele cognitive și euristicile umane modeleză comportamentul utilizatorului și cum pot fi anticipate aceste aspecte la proiectare.

### 10.4.1 Activitățile principale

1. **Teorie:** Elaborarea cadrelor conceptuale de interpretare a comportamentului utilizatorilor (de exemplu model mental, modelul Omului-processor). Se formulează principii pentru interfețe eficiente pornind de la teoria percepției și a memoriei (de exemplu, „regula de recunoaștere vs. reamintire”). Întrebările fundamentale includ:
  - Cum construiesc oamenii modele mentale ale sistemelor?
  - Cum influențează capacitatea de procesare a informației proiectarea?
  - Ce limite de atenție sau memorie trebuie luate în calcul?
2. **Experiment** Metode experimentale și evaluări formale cu utilizatori (studiu de utilizare, testarea prototipurilor, experimente de urmărire a ochilor etc.). Se folosesc metode ale psihologiei cognitive (de exemplu experimente controlate

de tip *think-aloud*) și statistici pentru a măsura erori de utilizare, eficiență și satisfacție. Studiile verifică, de pildă, cât de intuitiv sunt afișate informațiile și cum performanța umană se modifică în funcție de design.

### 3. Design Crearea de guideline-uri și standarde de interfață bazate pe factorii umani. De exemplu:

- Ben Shneiderman a propus reguli de bază (manipulare directă, feedback, coherență);
- Nielsen a elaborat euristici de evaluare.

Produse precum ERP-urile, interfețele medicale, sistemele de control al traficului sunt proiectate ținând cont de ergonomie cognitivă. Impactul acestui design se vede în reducerea erorilor umane și îmbunătățirea siguranței (de ex. în aviație sau chirurgie).

### 4. Extra

- **Cunoștințe necesare:** psihologie cognitivă, ergonomie fizică și cognitivă, fiziologie senzorială.
- **Literatură de specialitate:** monografii clasice de HCI (Donald Norman, *The Design of Everyday Things* [96]), articole din *Human Factors*, *Ergonomics* și rapoarte tehnice ISO privind interfețele.

#### 10.4.2 Relațiile cu celelalte subdomenii

Factorii umani sunt fundația științifică pentru toate subdomeniile HCI:

- în colaborativ se aplică la evaluarea muncii în echipă;
- în contexte specifice se adaptează la scenarii particulare (ex. atenție scăzută în medii de realitate augmentată);
- în securitate/etică determină comportamentul utilizatorilor în fața riscurilor (ex. ignorarea avertismentelor de securitate).

Designul UI este întotdeauna influențat de capacitațile și limitările umane studiate aici. De asemenea, psihologia interacțiunii beneficiază de tehnologiile colaborative sau speciale (ex. realitatea virtuală pentru studii cognitive).

#### 10.4.3 Probleme importante și probleme deschise

##### Provocări importante:

- Definirea limitelor cognitive în interacțiunea cu interfețe complexe (ex. multitasking pe dispozitive mobile);
- Personalizarea interfeței conform capacitatii utilizatorului (ex. adaptarea pentru dizabilități).

##### Probleme deschise:

- identificarea celor mai relevanți factori umani pentru noile tehnologii (ex. AR);
- integrarea metodologiilor psihologice moderne (ex. neuroștiințe) în design;
- dezvoltarea de modele de predicție a comportamentului pentru anticiparea erorilor în interfețe sofisticate.

#### 10.4.4 Persoane importante

1. **Donald A. Norman** – autor al lucrării *The Design of Everyday Things*, teoretician al modelelor mentale și al designului centrat pe utilizator; [96]
2. **Ben Shneiderman** – pionier HCI, autor al regulilor de design și al cărții *Designing the User Interface*; [161]
3. **Stuart Card, Thomas Moran, Allen Newell** – autori ai modelului *Model Human Processor* și GOMS; [162]
4. **Paul Fitts** – formularea *Fitts' Law* (1954), lege esențială în evaluarea interfețelor; [163]
5. **Alți autori notabili:** I. L. Goldstein, J. Nielsen, Yvonne Rogers.

#### 10.4.5 Forumuri importante

- **Conferințe:** ACM CHI, HCI International, INTERACT, HFES Annual Meeting.
- **Reviste:** *Human Factors*, *International Journal of Human-Computer Studies*, *Ergonomics*, *Behaviour & Information Technology*, *Applied Ergonomics*.
- **Forumuri online:** HFES, SIGCHI, grupuri de discuții despre ergonomie.

#### 10.4.6 Dimensiunea locală și globală

- **Local:** La UVT (Departamentul de Informatică, FMI) se predau cursuri de proiectare interfețe și ergonomie software. Comunitatea HCI din România este activă prin conferința RoCHI și revista de profil.
- **Global:** Factorii umani și psihologia interacțiunii sunt reglementate de standarde și consorții (ex. ISO 9241, IEEE). Colaborările internaționale includ proiecte europene și rețele academice (ex. SIGCHI), influențând practicile din industrie (Apple, Microsoft etc.).

### 10.5 Interacțiunea om-computer în contexte specifice

Interacțiunea om-computer în contexte specifice reprezintă o ramură aplicată a HCI, care se concentrează pe proiectarea, testarea și implementarea interfețelor în domenii critice precum medicina, educația, transporturile, apărarea, siguranța publică sau realitatea augmentată. În aceste medii, interacțiunea nu mai este doar o chestiune de confort și estetică — ci devine esențială pentru acuratețe, siguranță, accesibilitate și performanță sub presiune.

Acest subdomeniu presupune adaptarea principiilor HCI generale la cerințele riguroase ale fiecărui domeniu. Astfel, interacțiunea este influențată de factori precum stresul, medii zgomotoase, multitasking intensiv, responsabilitate legală și vieți omenești în joc (în medicină sau transporturi). Donald Norman, în lucrările sale, a subliniat nevoia de „design centrat pe eroare” în sisteme critice — interfețe care nu doar funcționează corect, ci previn greșelile umane anticipate.

#### 10.5.1 Activitățile principale

1. **Teorie:** Se bazează pe ergonomie cognitivă, teoria activității distribuite, teoria controlului erorilor, analiza sarcinilor și ingineria interacțiunii în sisteme complexe. Un rol important îl are și psihologia utilizatorului expert vs. începător și principiile de compatibilitate contextuală.
2. **Experiment:** Studiile sunt frecvent desfășurate in situ — în spitale, simulatoare de zbor, săli de operații sau vehicule autonome. Se utilizează metode precum simulări de urgență, evaluări formative, analiza sarcinilor, time-on-task, scoruri NASA-TLX pentru măsurarea încărcării cognitive și protocoale de tip think-aloud. Coiera și Hochheiser, în articolele lor despre HCI și informatică medicală, au subliniat importanța prototipurilor testate iterativ în medii reale, nu doar în laborator.
3. **Design:** Interfețele trebuie să fie robuste, clare, adaptate contextului de utilizare. De exemplu:

- (a) Sisteme de monitorizare medicală cu coduri de culoare și feedback auditiv;
- (b) Dashboarduri de control pentru drone sau vehicule autonome;
- (c) Aplicații educaționale adaptative, folosite în medii cu acces limitat la resurse;
- (d) Interfețe AR pentru mențenanță sau chirurgie asistată, cu lag minim și ancore vizuale stabile.

În astfel de cazuri, designul vizual se subordonează priorităților: siguranță, vizibilitate, predictibilitate, ușurință în interpretare — valori reflectate în standarde precum ISO 9241-210 sau ISO 62366.

#### 4. Extra:

- (a) Necesară colaborare interdisciplinară între specialiști HCI, ingineri de sistem, medici, piloți, profesori sau militari. Aspectele legale și etice sunt omniprezente — de la păstrarea confidențialității în spitale [164] până la auditabilitatea deciziilor AI în contexte militare. Este esențial ca proiectarea să fie inclusivă, adaptată și persoanelor cu dizabilități.

#### 10.5.2 Relațiile cu celelalte subdomenii

Interacțiunea în contexte specifice este un teren fertil pentru sinteza dintre UI/UX (pentru interfețe clare), AI (pentru asistență proactivă) și CSCW (pentru colaborare în timp real sau asincron). De exemplu, în telemedicina, trebuie integrate sisteme de vizualizare AI, platforme de colaborare și interfețe accesibile pentru personal medical, pacienți și administratori.

#### 10.5.3 Probleme importante și probleme deschise

Printre cele mai presante provocări se află:

- Asigurarea siguranței utilizatorului în condiții de stres;
- Prevenirea erorilor critice (ex: supradoxaj, coliziuni automate);
- Etichetarea clară a informațiilor automate generate de AI;
- Designul pentru medii imprevizibile sau offline;
- Integrarea echitabilă a AI fără excluderea controlului uman.

Problemele deschise includ designul în realitate mixtă (XR), interfețele în domenii emergente (explorare spațială, climate tech) și auditabilitatea sistemelor în fața legii.

#### 10.5.4 Persoane importante

1. **Donald A. Norman** – susține designul centrat pe prevenirea greșelilor umane în sisteme critice. [96]
2. **Enrico Coiera** – pionier în interacțiunea HCI în informatică medicală.
3. **Yvonne Rogers** – studii în interacțiunea în educație, AR și sisteme tangibile.
4. **Ben Shneiderman** – promotor al AI sigur, explicabil și aplicabil în domenii critice. [165], [161]

#### 10.5.5 Forumuri importante

- Conferințe: MedInfo, IEEE ICHI, AutomotiveUI, CHI, ACM Health.
- Reviste: JAMIA (Journal of the American Medical Informatics Association), Interacting with Computers, International Journal of Medical Informatics.
- Organisme și standarde: WHO, ISO 62366, European Accessibility Act, IEEE SA, GDPR.

#### 10.5.6 Dimensiunea locală și globală

- **Local (UVT):** În cadrul UVT, studenții pot aborda teme precum aplicații pentru medii educaționale speciale, sisteme de evaluare online adaptativă sau interfețe pentru simulări medicale. Se pot realiza colaborări cu spitale, ONG-uri și parteneri industriali pentru validarea sistemelor.
- **Global:** În plan internațional, interacțiunea în contexte specifice este cercetată în centre ca MIT Media Lab, Stanford HCI, Oxford Digital Health, dar și în agenții precum NASA, UE sau organizații non-profit ce lucrează în domeniul umanitar. Tema este prioritară în strategiile de incluziune digitală, reziliență și inovație sustenabilă.

### 10.6 Securitate și etica în HCI

Acest subdomeniu se ocupă de integrarea aspectelor de securitate, confidențialitate și etică în proiectarea interfețelor și a sistemelor interactive. Etica în HCI acoperă teme precum protecția bunăstării utilizatorului, dreptul la confidențialitate, eliminarea bias-ului, accesibilitate universală, transparență și responsabilitate ([pmc.ncbi.nlm.nih.gov](http://pmc.ncbi.nlm.nih.gov)).

În paralel, securitatea informației pune în prim-plan dimensiunea umană: utilizatorii sunt adesea considerați „veriga slabă” în lanțul securității, așa încât HCI-ul

securității (usable security) caută soluții de design care să facă mecanismele de securitate eficiente și ușor de folosit (researchgate.net). Domeniul analizează probleme de încredere și protecție a datelor în interacțiunea om-calculator.

#### 10.6.1 Activitățile principale

1. **Teorie** Dezvoltă cadre etice și normative (de ex. modelul contextual integrity al Helen Nissenbaum, principii precum *Privacy by Design*, coduri de etică profesională ACM/IEEE). Se studiază dilemele morale (de ex. colectarea datelor personale vs. utilitate) și modele cognitive de evaluare a riscului. În securitate, se elaborează modele teoretice de comportament al atacatorului și al utilizatorului (de ex. teoria „lanțului de încredere uman”).
2. **Experiment** Studii empirice despre reacția utilizatorilor la protocoale de securitate (testarea parolelor, 2FA, mesaje de avertizare), experimentarea efectelor privind confidențialitatea (cum percep utilizatorii notificările GDPR). Se folosesc experimente de factori umani în scenarii de simulare a atacurilor (phishing, inginerie socială).
3. **Design** Crearea de interfețe și instrumente care promovează securitatea și respectarea normelor etice. Exemple: widget-uri de confidențialitate, UI-uri pentru criptare, design incluziv, evitarea *dark patterns*. Concept-cheie: *usable security*. Impactul se măsoară prin nivelul de securitate real și încrederea publică în tehnologie.
4. **Extra**
  - (a) Cunoștințe necesare: securitate informatică (criptografie, autentificare), legislație (GDPR), filozofie morală.
  - (b) Literatură: SOUPS , lucrările lui Bruce Schneier [81], cărți despre valori în design și bias în AI.

#### 10.6.2 Relațiile cu celelalte subdomenii

Securitatea și etica în HCI sunt interconectate cu toate celelalte arii. De exemplu, factorii umani explică ocolirea mecanismelor de securitate. În medii colaborative se cercetează granițele de încredere online. Contextul specific (ex. medical) impune cerințe suplimentare – interfețe care asigură protecția datelor pacienților.

#### 10.6.3 Probleme importante și probleme deschise

Provocări:

- Protejarea confidențialității într-un mediu digital omniprezent;
- Menținerea securității în fața atacurilor sofisticate (ex. deepfakes);
- Reconcilierea dintre ușurința în utilizare și robustețea securității;
- Consumământ informat și eliminarea bias-ului algoritmic;
- Responsabilitatea dezvoltatorului pentru consecințele utilizării produsului;
- Reducerea vulnerabilității umane și autentificare biometrică ușor de utilizat.

#### 10.6.4 Persoane importante

1. **Bruce Schneier** – expert în criptografie și securitate, promotor al conceptului de „securitate centrată pe om”. [81]
2. **Lorrie Faith Cranor** – expertă în confidențialitate și usable privacy. [166]
3. **Helen Nissenbaum** – autoarea teoriei *privacy as contextual integrity*. [164]
4. **Alan Patrick, Chris Long** – promotori ai studiului interfețelor de securitate.
5. **Gene Spafford, Dan Geer** – influențează politicile de securitate și etica profesională.

#### 10.6.5 Forumuri importante

- **Conferințe** CHI, SOUPS, IEEE S&P Workshops, Interact.
- **Reviste** IEEE TIFS, Computers & Security, Ethics and Information Technology, Journal of Business Ethics, ACM TOCHI, Journal of Human-Computer Studies.
- **Standardizare** ISO/IEC 27001, ISO 9241-210. Comunități: ACM SIGSEC, organizații de protecție a datelor.

#### 10.6.6 Dimensiunea locală și globală

- **Local** În România: cursuri de securitate, RoCHI, colaborări între instituții și mediul academic (ex. UVT).
- **Global** Influențe internaționale: GDPR, Privacy by Design, standarde ISO/IEC, rețele de cercetare (ex. COST Action), consorții de etică în AI.

# 11 Știință Computațională

Știința computațională este un domeniu interdisciplinar aflat la intersecția dintre matematică, informatică și științele aplicate, dedicat rezolvării problemelor complexe prin metode numerice și simulări computerizate. Ea reunește subdomenii precum analiza numerică, calculul de înaltă performanță, fizica computațională, biologia computațională, chimia computațională, economia computațională și ingeriera computațională. Acestea contribuie la modelarea, simularea și analiza fenomenelor reale, având aplicații esențiale în cercetare, industrie și dezvoltare tehnologică. Evoluția rapidă a puterii de calcul a transformat acest domeniu într-un pilon fundamental al științei moderne.

Subdomenii:

1. Analiză numerică
2. Calcul de înaltă performanță (HPC)
3. Aplicații în fizica computațională
4. Aplicații în biologia computațională
5. Aplicații în chimia computațională
6. Aplicații în economia computațională
7. Aplicații în ingeriera computațională

## 11.1 Analiză numerică

Analiza numerică se ocupă cu dezvoltarea și studiul metodelor numerice pentru rezolvarea problemelor matematice care sunt dificil sau imposibil de rezolvat analitic. Acest domeniu a evoluat odată cu dezvoltarea calculatoarelor, devenind esențial în multe domenii științifice și ingereresti.

### 11.1.1 Activitățile principale

1. Teorie:
  - (a) Studiul erorilor (trunchiere, rotunjire) și stabilității algoritmilor
  - (b) Analiza convergenței metodelor numerice
  - (c) Metode de aproximare și interpolare
  - (d) Întrebări fundamentale: Cum putem aproxima soluții cu precizie? Cât de rapid converg algoritmii?

2. Experiment:
  - (a) Implementarea metodelor numerice (diferențe finite, element finit)
  - (b) Validarea numerică a modelelor matematice
  - (c) Exemplu: compararea metodelor de rezolvare a sistemelor liniare
3. Design:
  - (a) Biblioteci numerice (LAPACK, NumPy)
  - (b) Instrumente de calcul științific (MATLAB, Octave)
  - (c) Tehnici de optimizare numerică
  - (d) Impact: permite simulări complexe în inginerie și știință
4. Extra:
  - (a) Literatura de specialitate:
    - Burden, R. L., & Faires, J. D. (2010). Numerical Analysis. Cengage Learning. [167]
    - Stoer, J., & Bulirsch, R. (2002). Introduction to Numerical Analysis. Springer [168]

### 11.1.2 Relațiile cu celelalte subdomenii

Analiza numerică este strâns legată de:

- Matematica aplicată - furnizează probleme de rezolvat
- Informatica teoretică - oferă algoritmi eficienți
- Fizica computațională - aplică metode numerice

### 11.1.3 Probleme Importante și Probleme Deschise

Probleme importante:

- Rezolvarea numerică a ecuațiilor diferențiale
- Optimizarea numerică

Probleme deschise:

- Dezvoltarea de algoritmi stabili pentru probleme mal-condiționate
- Adaptarea metodelor pentru calculatoare cu arhitecturi noi

#### **11.1.4 Persoane Importante**

- John von Neumann - analiza stabilității von Neumann
- James H. Wilkinson - a evidențiat necesitatea analizării erorilor în calcul
- Gene H. Golub - a creat NA-Net ,NA-Digest și a ajutat la dezvoltarea International Congress on Industrial and Applied Mathematics

#### **11.1.5 Forumuri Importante**

Forumuri relevante:

- Conferințe: ICNAAM 23rd International Conference of Numerical Analysis and Applied Mathematics,ICNACAM :International Conference on Numerical Analysis, Computational and Applied Mathematics
- Reviste:SIAM Journal on Numerical Analysis,ACM Transactions on Mathematical Software

#### **11.1.6 Dimensiunea locală și dimensiunea globală**

1. Local: Cursuri de Analiză Numerică sunt predate în cadrul FMI UVT. Cercetări în domeniul metodei elementelor finite.
2. Global: Colaborări între universități și centre de cercetare pentru dezvoltarea de algoritmi numerici.

### **11.2 Calcul de Înaltă Performanță**

Calculul de Înaltă Performanță (HPC) se referă la practica agregării puterii de calcul pentru a rezolva probleme computaționale complexe care depășesc capacitatele calculatoarelor standard. Istoric, HPC a apărut din necesitățile de calcul științific din mijlocul secolului XX și a evoluat odată cu avansurile în arhitecturi paralele și sisteme distribuite.

#### **11.2.1 Activități Principale**

1. Teorie:
  - (a) Definirea paradigmelor de calcul paralel
  - (b) Descrierea cadrului de analiză a complexității computaționale
  - (c) Metode de analiză a performanței și modelare a scalabilității

- (d) Întrebări fundamentale: Cum se poate obține paralelism maxim? Care sunt limitele teoretice ale accelerării computaționale?
2. Experiment:
- (a) Explorarea arhitecturilor parallele (clustere, griduri, cloudui)
  - (b) Implementarea și măsurarea performanței algoritmilor paralleli
  - (c) Exemplu practic: Compararea operațiilor colective MPI
3. Proiectare:
- (a) Stive software HPC și middleware
  - (b) Instrumente: biblioteci MPI, analizatoare de performanță
  - (c) Tehnici: decompoziția domeniului, echilibrarea încărcării
  - (d) Impact: Permite descoperiri importante în cercetarea științifică
4. Extra:
- (a) Cunoștințe necesare: programare paralelă, arhitectură de calculatoare, algoritmi
  - (b) Literatură de specialitate:
    - Grama, A., et al. (2003). *Introducere în Calculul Paralel*. Pearson. [30]
    - Dongarra, J., et al. (2020). *Sourcebook of Parallel Computing*. Morgan Kaufmann. [169]
    - TOP500.org. (2023). *Lista celor mai puternice supercomputere din lume*. [170]

### 11.2.2 Relații cu Alte Subdomenii

HPC menține relații strânse cu:

- Analiza Numerică – furnizează algoritmi pentru implementare
- Arhitectura Calculatoarelor – stimulează inovațiile hardware
- Sisteme Distribuite – împărtășește concepte de gestionare a resurselor
- Știința Computatională – servește ca tehnologie de suport

### **11.2.3 Probleme Importante și Probleme Deschise**

Probleme cheie:

- Atingerea nivelului de exascale computing
- Eficiență energetică în supercomputere
- Toleranța la defecte la scară mare

Provocări deschise:

- Modele de programare pentru arhitecturi eterogene
- Procesarea în timp real a datelor la scară extremă
- Integrarea calculului cuantic-clasic

### **11.2.4 Persoane Importante**

- Seymour Cray : supercalculatoarele Cray
- Jack Dongarra : LINPACK, TOP500
- Gene Amdahl : Legea lui Amdahl

### **11.2.5 Forumuri Importante**

Conferințe și publicații cheie:

- SC (Conferința de Supercomputing)
- IEEE Transactions on Parallel and Distributed Systems
- Conferința Internațională de Calcul de Înaltă Performanță
- Journal of Parallel and Distributed Computing

### **11.2.6 Dimensiune Locală și Globală**

1. Local: Cursuri HPC oferite de Departamentul de Informatică, FMI UVT; proiecte de cercetare în domeniul calculului paralel.
2. Global: Participarea la colaborări internaționale (EuroHPC); alinierea la standardele globale (MPI, OpenMP).

## 11.3 Aplicații în Fizica Computațională

Fizica computațională este o ramură a fizicii care utilizează metode numerice și algoritmi computaționali pentru modelarea și simularea sistemelor fizice. Acest subdomeniu a evoluat odată cu dezvoltarea calculatoarelor, devenind esențial în investigația sistemelor complexe care nu pot fi studiate analitic.

### 11.3.1 Activități principale

1. Teorie:
  - (a) Definirea modelelor matematice pentru fenomene fizice
  - (b) Descrierea metodelor de discretizare a ecuațiilor diferențiale
  - (c) Metode de analiză numerică aplicate în fizică
  - (d) Întrebări fundamentale: Cum putem simula sisteme fizice complexe? Ce limite există în modelarea numerică?
2. Experiment:
  - (a) Simulații numerice ale sistemelor fizice
  - (b) Validarea experimentală a modelelor computaționale
  - (c) Exemplu practic: Simularea mișcării fluidelor folosind metode CFD
3. Design:
  - (a) Software-uri de simulare (ANSYS, COMSOL)
  - (b) Biblioteci computaționale pentru fizică (ROOT, LAMMPS)
  - (c) Tehnici de paralelizare a calculului
  - (d) Impact: Revoluționează cercetarea în fizică teoretică și aplicată
4. Extra:
  - (a) Cunoștințe necesare: fizică, matematică aplicată, programare
  - (b) Literatură de specialitate:
    - Landau, R. H., Páez, M. J., & Bordeianu, C. C. (2015). Computational Physics. Wiley. [171]
    - Thijssen, J. M. (2013). Computational Physics. Cambridge University Press. [172]

### **11.3.2 Relațiile cu celelalte subdomenii**

Fizica computațională are legături strânse cu:

- Analiza numerică - furnizează algoritmi de calcul
- Calculul de înaltă performanță - oferă resurse computaționale
- Fizica teoretică - oferă modele de studiat
- Ingineria computațională - aplică rezultatele în practică

### **11.3.3 Probleme Importante și Probleme Deschise**

Probleme importante:

- Simularea sistemelor cuantice multi-corpusculeare
- Modelarea turbulenței în dinamica fluidelor

Probleme deschise:

- Simularea sistemelor aflate departe de echilibru
- Cresterea preciziei simulărilor la scară atomică

### **11.3.4 Persoane Importante**

- Richard Feynmann - ideea simularii proceselor fizice pe calculator (apariția ideii de calculator cuantic)
- Sauro Succi - contribuții în simularea dinamicii fluidelor

### **11.3.5 Forumuri Importante**

Forumuri relevante:

- Journal of Computational Physics
- International Conference on Computational Physics
- Physical Review E (secțiunea de fizică computațională)

### **11.3.6 Dimensiunea locală și dimensiunea globală**

1. Local: Cursuri de fizică computațională la FMI UVT; colaborări cu facultățile de fizică
2. Global: Proiecte internaționale de simulare (ITER pentru fizica plasmei)

## 11.4 Aplicații în Biologia Computațională

Biologia computațională este un domeniu interdisciplinar care aplică metode matematice, statistice și computaționale pentru studiul sistemelor biologice. Acest subdomeniu a cunoscut o evoluție rapidă în ultimele decenii odată cu avansul tehnologicilor de secvențiere genomică.

### 11.4.1 Activități principale

1. Teorie:
  - (a) Modelarea proceselor biologice
  - (b) Algoritmi de aliniere a secvențelor
  - (c) Metode de învățare automată aplicată în genomică
  - (d) Întrebări fundamentale: Cum putem prezice structura proteinelor? Ce modele matematice descriu evoluția speciilor?
2. Experiment:
  - (a) Analiza datelor genomice
  - (b) Simularea rețelelor metabolice
  - (c) Exemplu practic: Identificarea genelor asociate bolilor folosind machine learning
3. Design:
  - (a) Baze de date biologice (GenBank, UniProt)
  - (b) Instrumente: BLAST, Bioconductor, PyMOL
  - (c) Tehnici de vizualizare moleculară
  - (d) Impact: Revoluționează cercetarea medicală și farmaceutică
4. Extra:
  - (a) Cunoștințe necesare: biologie moleculară, statistică, programare (Python/R)
  - (b) Literatură de specialitate:
    - Durbin, R., et al. (1998). Biological Sequence Analysis. Cambridge University Press. [173]
    - Lesk, A. M. (2019). Introduction to Bioinformatics. Oxford University Press. [174]

#### **11.4.2 Relațiile cu celelalte subdomenii**

Biologia computațională interacționează cu:

- Bioinformatică - analiza datelor biologice
- Biostatistica - metode de analiză a datelor
- Chimia computațională - modelarea moleculară
- Inteligență artificială - clasificarea pattern-urilor biologice

#### **11.4.3 Probleme Importante și Probleme Deschise**

Probleme importante:

- Predicția structurii 3D a proteinelor
- Reconstrucția arborelui filogenetic

Probleme deschise:

- Modelarea sistemelor biologice multi-scalare
- Interpretarea variantelor genetice rare

#### **11.4.4 Persoane Importante**

- Margaret Dayhoff și Carl Sagan - pionieri în bioinformatică: au realizat o baza de date vastă pentru proteine
- Michael Waterman și Temple Smith - algoritmul Smith-Waterman pentru alinierea secvențelor de nucleotide sau proteine

#### **11.4.5 Forumuri Importante**

Forumuri relevante:

- Journal of Computational Biology
- RECOMB (Research in Computational Molecular Biology)
- ISMB (Intelligent Systems for Molecular Biology)

#### **11.4.6 Dimensiunea locală și dimensiunea globală**

1. Local: Cursuri de bioinformatică la FMI UVT; colaborări cu institute medicale
2. Global: Proiecte internaționale (Genomul Uman, ENCODE)

### **11.5 Aplicații în Chimia Computațională**

Chimia computațională reprezintă ramura chimiei care utilizează metode numerice și simulări computerizate pentru a studia structura și proprietățile sistemelor chimice. A apărut în anii 1950 odată cu dezvoltarea primelor calculatoare și a evoluat semnificativ odată cu creșterea puterii de calcul.

#### **11.5.1 Activități principale**

1. Teorie:
  - (a) Dezvoltarea metodelor de mecanică cuantică și moleculară
  - (b) Modelarea interacțiunilor intermoleculare
  - (c) Metode de simulare a dinamicii moleculare
  - (d) Întrebări fundamentale: Cum putem prezice proprietățile moleculelor? Ce metode sunt cele mai eficiente pentru diferite sisteme chimice?
2. Experiment:
  - (a) Simularea reacțiilor chimice
  - (b) Calculul energiilor de legătură
  - (c) Exemplu practic: Studiul mecanismelor de reacție folosind DFT (Teoria funcțională a densității)
3. Design:
  - (a) Software-uri specializate (Gaussian, ORCA, VASP)
  - (b) Baze de date cu proprietăți moleculare
  - (c) Tehnici de vizualizare moleculară
  - (d) Impact: Accelerarea descoperirii de noi medicamente și materiale
4. Extra:
  - (a) Cunoștințe necesare: chimie, fizică cuantică, programare, metode numerice

(b) Literatură de specialitate:

- Cramer, C. J. (2004). Essentials of Computational Chemistry. Wiley.  
[175]
- Jensen, F. (2017). Introduction to Computational Chemistry. Wiley.  
[176]

### 11.5.2 Relațiile cu celelalte subdomenii

Chimia computațională interacționează cu:

- Fizica computațională - metode de simulare
- Biologia computațională - modelarea biomoleculelor
- Sciencele materialelor - designul de noi materiale
- Farmacologia computațională - designul de medicamente

### 11.5.3 Probleme Importante și Probleme Deschise

Probleme importante:

- Predicția structurii electronice
- Simularea sistemelor cu mari numere de atomi

Probleme deschise:

- Modelarea sistemelor cuantice complexe
- Dezvoltarea de metode precise pentru sisteme biologice mari

### 11.5.4 Persoane Importante

- Walter Kohn - dezvoltatorul DFT (Premiul Nobel 1998)
- John Pople - dezvoltarea metodelor de procesare în chimia cuantică
- Martin Karplus - dezvoltarea modelor pentru sisteme chimice complexe(Premiul Nobel 2013)
- David Baker - design computațional de proteine(premiu Nobel 2024)

### **11.5.5 Forumuri Importante**

Forumuri relevante:

- Journal of Chemical Theory and Computation
- Journal of Computational Chemistry
- International Conference on Computational Chemistry

### **11.5.6 Dimensiunea locală și dimensiunea globală**

1. Local: Cursuri de chimie computațională în cadrul FMI UVT; colaborări cu facultățile de chimie
2. Global: Proiecte internaționale în domeniul materialelor și farmaceutic

### **11.5.7 Bibliografie**

## **11.6 Aplicații în Economia Computațională**

Economia computațională reprezintă o ramură interdisciplinară care aplică metode computaționale și tehnici de simulare pentru studiul fenomenelor economice. A apărut în anii 1970 odată cu creșterea puterii de calcul și a evoluat semnificativ în ultimele decenii.

### **11.6.1 Activități principale**

1. Teorie:
  - (a) Modelarea sistemelor economice complexe
  - (b) Teoria jocurilor computaționale
  - (c) Metode de optimizare economică
  - (d) Întrebări fundamentale: Cum putem simula piețele financiare? Ce modele prezic cel mai bine comportamentul economic?
2. Experiment:
  - (a) Simularea piețelor financiare
  - (b) Testarea politicilor economice
  - (c) Exemplu practic: Modelarea propagării șourilor economice folosind agenți computaționali

3. Design:
  - (a) Platforme de simulare economică
  - (b) Instrumente de analiză a serilor temporale
  - (c) Tehnici de prognoză economică
  - (d) Impact: Îmbunătățirea politicilor economice și a strategiilor de investiții
4. Extra:
  - (a) Cunoștințe necesare: economie, statistică, programare (Python/R), metode numerice
  - (b) Literatură de specialitate:
    - Tesfatsion, L., & Judd, K. L. (2006). Handbook of Computational Economics. Elsevier. [177]
    - Kendrick, D. A., et al. (2006). Computational Economics. Princeton University Press. [178]

### **11.6.2 Relațiile cu celelalte subdomenii**

Economia computațională se intersectează cu:

- Matematica financiară - modele de evaluare a activelor
- Inteligența artificială - algoritmi de trading
- Știința datelor - analiza datelor economice
- Cercetarea operațională - probleme de optimizare

### **11.6.3 Probleme Importante și Probleme Deschise**

Probleme importante:

- Modelarea sistemelor economice adaptive
- Simularea crizelor financiare

Probleme deschise:

- Integrarea factorilor sociali în modelele economice
- Prognoza pe termen lung a sistemelor economice complexe

#### **11.6.4 Persoane Importante**

- Thomas Sargent - Premiul Nobel pentru metodele dezvoltate în economia computațională
- Leigh Tesfatsion - recunoscută pentru propunerea metodelor bazate pe agenți în economie
- Kenneth Judd - aplicarea metodelor numerice în economie

#### **11.6.5 Forumuri Importante**

Forumuri relevante:

- Journal of Economic Dynamics and Control
- Society for Computational Economics
- International Conference on Computing in Economics and Finance

#### **11.6.6 Dimensiunea locală și dimensiunea globală**

1. Local: Cursuri de econometrie și analiză computațională la FMI UVT
2. Global: Colaborări cu instituții financiare internaționale (FMI, Banca Mondială)

#### **11.6.7 Bibliografie**

### **11.7 Aplicații în Ingineria Computațională**

Ingineria computațională reprezintă aplicarea metodelor numerice și a simulărilor computerizate în rezolvarea problemelor de inginerie. Acest domeniu a cunoscut o dezvoltare accelerată odată cu creșterea puterii de calcul și a performanțelor hardware.

#### **11.7.1 Activități principale**

1. Teorie:
  - (a) Modelarea matematică a sistemelor fizice
  - (b) Metode cu elemente finite și diferențe finite
  - (c) Algoritmi de optimizare în inginerie

- (d) Întrebări fundamentale: Cum putem simula sisteme ingineresci complexe? Care sunt limitele modelării numerice?
2. Experiment:
- (a) Simulări CFD (Computational Fluid Dynamics)
  - (b) Analiza structurilor prin FEM
  - (c) Exemplu practic: Simularea comportării unui pod în condiții seismice
3. Design:
- (a) Software-uri de inginerie (ANSYS, COMSOL, SolidWorks)
  - (b) Instrumente de simulare multi-fizică
  - (c) Tehnici de vizualizare 3D a rezultatelor
  - (d) Impact: Reducerea costurilor de prototipare și testare
4. Extra:
- (a) Cunoștințe necesare: mecanica solidelor/fluidelor, metode numerice, programare
  - (b) Literatură de specialitate:
    - Zienkiewicz, O.C. (2005). The Finite Element Method. Butterworth-Heinemann. [179]
    - Anderson, J.D. (1995). Computational Fluid Dynamics. McGraw-Hill. [180]

### **11.7.2 Relațiile cu celelalte subdomenii**

Ingineria computațională interacționează cu:

- Mecanica computațională - analiza structurilor
- Dinamica fluidelor computațională - simulări CFD
- Electromagnetismul computațional - simulări EM
- Sciencele materialelor - proprietăți ale materialelor

### **11.7.3 Probleme Importante și Probleme Deschise**

Probleme importante:

- Cuplarea multi-fizică
- Simulări la scară mare (HPC)

Probleme deschise:

- Modelarea incertitudinilor în simulări
- Integrarea AI în simulările ingineresci

### **11.7.4 Persoane Importante**

- Olgierd Zienkiewicz - recunoscut pentru utilizarea metodei elementului finit în alte zone decât mecanica solidului
- Klaus-Jürgen Bathe - dezvoltator de software pentru avansarea metodei elementului finit
- John C. Houbolt - a condus echipa pentru dezvoltarea metodei Lunar Orbit Rendezvous; ce a permis aterizarea modulului Apollo 11 pe lună și recuperarea lui

### **11.7.5 Forumuri Importante**

Forumuri relevante:

- International Journal for Numerical Methods in Engineering
- Journal of Computational Physics
- Conferința Internațională de Inginerie Computațională

### **11.7.6 Dimensiunea locală și dimensiunea globală**

1. Local: Cursuri de metode numerice în inginerie la FMI UVT; colaborări cu universități tehnice
2. Global: Standarde internaționale în simulări (FEM, CFD); proiecte europene

## **12 Informatica Organizatoriala**

Informatica organizațională este un domeniu interdisciplinar care se află la intersecția dintre tehnologia informației, științele organizaționale și management. Scopul său principal este de a analiza, proiecta și implementa sisteme informatice care sprijină procesele și structurile organizaționale. Acest domeniu ajută organizațiile să-și îmbunătățească eficiența operațională, să ia decizii mai informate și să se adapteze la schimbările din mediul economic și tehnologic.

Subdomenii ale informaticii organizatoriale:

1. Sisteme de management al informației (MIS)
2. Baze de date și gestionarea informațiilor
3. Arhitectura sistemelor informative pentru organizații
4. Rețele și securitatea informațiilor în organizații
5. Automatizarea și optimizarea proceselor organizaționale
6. Interacțiunea om-calculator în mediul organizațional
7. Guvernanța IT și managementul strategic al tehnologiilor

### **12.1 Sisteme de management al informației (MIS)**

Sistemele de management al informației (MIS) reprezintă un subdomeniu esențial al informaticii organizaționale, fiind axate pe proiectarea, dezvoltarea și utilizarea sistemelor informative pentru sprijinirea proceselor decizionale, operaționale și strategice ale unei organizații. MIS combină elemente din informatică, management, economie și științe organizaționale, pentru a transforma datele brute în informații relevante, utile și accesibile.

Istoric, primele sisteme MIS au apărut în anii 1960, fiind orientate către automatizarea rapoartelor financiare. Odată cu avansul tehnologic, MIS a evoluat într-un sistem complex de sprijin managerial, integrând baze de date, rețele, interfețe grafice, sisteme ERP și platforme web. În prezent, MIS acoperă o gamă largă de aplicații, de la dashboarduri de business intelligence până la platforme cloud integrate.

#### **12.1.1 Activitățile principale**

1. Teorie:

(a) **Fundamente conceptuale și definiții**

Un sistem MIS este definit ca un ansamblu de resurse (hardware, software, personal, proceduri) organizate pentru a colecta, procesa, stoca și distribui informații care sprijină luarea deciziilor la nivel managerial. El se diferențiază de sistemele operaționale prin faptul că este orientat spre sinteză și analiză, nu doar stocare sau procesare tranzacțională [181].

(b) **Tipologii de sisteme MIS**

Sistemele MIS pot fi clasificate în:

- *Sisteme pentru managementul operațional* – oferă rapoarte standardizate despre activități curente;
- *Sisteme pentru suport decizional (DSS)* – permit analiza scenariilor și predicții;
- *Sisteme executive (EIS)* – sintetizează date strategice pentru top management;
- *Sisteme ERP* – integrează MIS în toate departamentele organizației.

(c) **Întrebări teoretice esențiale**

Printre întrebările cheie analizate în literatura de specialitate se regăsesc:

- Cum influențează calitatea informației performanța managerială?
- Ce modele de arhitectură sunt cele mai eficiente pentru MIS moderne?
- Cum se echilibrează securitatea și accesibilitatea informațiilor într-un sistem MIS?

2. Experiment:

(a) **Studii de caz și implementări reale**

Un exemplu relevant este implementarea sistemului SAP ERP în cadrul companiei Nestlé, care a urmărit standardizarea proceselor de achiziții și distribuție în peste 70 de locații globale. Studiul a arătat cum o implementare centralizată MIS poate reduce costurile operaționale și îmbunătăți consistența datelor [182].

(b) **Validarea eficienței sistemelor MIS**

Eficiența unui sistem MIS este adesea evaluată prin indicatori precum reducerea timpului de răspuns managerial, creșterea calității deciziilor și gradul de utilizare de către personalul non-tehnic.

3. Design:

(a) **Platforme și instrumente reprezentative**

Printre cele mai utilizate platforme MIS se numără: SAP, Oracle NetSuite, Microsoft Dynamics 365 și Zoho One. Aceste sisteme oferă funcționalități integrate pentru raportare, analiză, automatizare și integrare interdepartamentală.

(b) **Tehnici și modele de proiectare**

Designul unui sistem MIS presupune selectarea unui model arhitectural (centralizat, distribuit sau cloud), integrarea surselor de date și stabilirea fluxurilor de informație între nivelurile ierarhice.

4. Extra:

(a) **Cunoștințe necesare pentru studiu**

Este esențială înțelegerea bazelor de date, analiza sistemelor, programarea web și noțiunile de management organizațional. Familiaritatea cu SQL, modelele ER și analize SWOT este de asemenea utilă.

(b) **Literatură de specialitate**

- *Management Information Systems* de Laudon & Laudon (2020) [181]
- *Decision Support and Business Intelligence Systems* de Turban et al. (2010) [183]
- *Enterprise Systems for Management* de Motiwalla & Thompson (2012) [184]

### 12.1.2 Relațiile cu celelalte subdomenii

Sistemele MIS sunt interconectate cu toate celelalte componente ale informaticii organizaționale. Baza de date este nucleul tehnologic al unui MIS, iar arhitectura sistemului determină scalabilitatea și performanța. Rețelele asigură distribuția informațiilor, iar securitatea lor este vitală pentru confidențialitate. Automatizarea proceselor depinde direct de un MIS bine proiectat. Interacțiunea om-calculator influențează gradul de utilizare al sistemului, iar guvernanța IT definește standardele de funcționare și audit ale acestuia.

### 12.1.3 Probleme importante și probleme deschise

Printre provocările actuale ale sistemelor MIS se numără:

- asigurarea interoperabilității între platforme eterogene;
- menținerea securității în condiții de acces distribuit (ex. muncă remote);

- adaptarea la volume mari de date generate în timp real;
- integrarea inteligenței artificiale pentru sprijin decizional automatizat.

Printre problemele deschise se află dezvoltarea de modele predictive explicabile (XAI), reducerea dependenței de infrastructuri centralizate și evaluarea impactului cultural asupra utilizării MIS în organizații globale.

#### **12.1.4 Persoane importante**

Printre cercetătorii relevanți în domeniul MIS se numără Kenneth C. Laudon, autor al uneia dintre cele mai utilizate lucrări de specialitate, și Efraim Turban, cunoscut pentru contribuțiile privind sisteme de sprijin decizional și business intelligence. De asemenea, Robert I. Benjamin a definit concepte timpurii de transformare digitală prin MIS încă din anii '80.

#### **12.1.5 Forumuri importante**

Printre cele mai relevante conferințe și reviste pentru MIS se numără:

- *MIS Quarterly* – revistă academică de referință în domeniu;
- *Information Systems Research*;
- *HICSS* – Hawaii International Conference on System Sciences;
- *ICIS* – International Conference on Information Systems.

#### **12.1.6 Dimensiunea locală și dimensiunea globală**

- **Local:** La nivelul Departamentului de Informatică, FMI – UVT, studiul sistemelor MIS este integrat în discipline precum „Sisteme informaticice pentru organizații”, „Bazele de date” și „Tehnologii informaționale în management”, având aplicații în proiectele studenților și în parteneriate cu mediul economic regional.
- **Global:** La nivel internațional, MIS reprezintă un pilon strategic în digitalizarea organizațiilor, fiind susținut de standarde ISO, modele ITIL și trenduri globale precum Industry 4.0 și transformarea digitală. Universități precum MIT, Harvard sau Wharton oferă cursuri specializate, iar companii ca SAP și Microsoft contribuie activ la cercetarea și dezvoltarea sistemelor MIS.

## **12.2 Baze de date și gestionarea informațiilor**

Subdomeniul „Baze de date și gestionarea informațiilor” se ocupă cu metodele, tehnologiile și teoriile care permit stocarea, organizarea, regăsirea și manipularea eficientă a datelor în scopul utilizării lor în procese decizionale, operaționale și analitice. Acesta stă la baza tuturor sistemelor informatiche moderne, oferind infrastructura pentru colectarea și procesarea informației în mod structurat.

Istoric, domeniul a luat ampioare în anii 1970 odată cu introducerea modelului relațional de către Edgar F. Codd, care a revoluționat modul de organizare a datelor și a condus la dezvoltarea limbajului SQL. De-a lungul decadelor, au fost dezvoltate și modele alternative precum cele orientate pe obiecte, NoSQL și grafuri, adaptate la cerințele aplicațiilor web, big data și inteligență artificială.

### **12.2.1 Activitățile principale**

1. Teorie:

(a) **Fundamentele teoretice ale domeniului**

Domeniul bazelor de date se bazează pe modele matematice și formale pentru reprezentarea relațiilor între date. Modelul relațional definește datele sub formă de tabele, fiecare tabel având un set de atrbute și chei primare pentru identificare unică. Reguli precum formele normale ghidează procesul de proiectare pentru a evita redundanțele și anomaliiile [185].

(b) **Clasificarea sistemelor de baze de date**

Tipurile majore includ:

- Baze de date relaționale (ex. PostgreSQL, Oracle);
- Baze de date NoSQL (ex. MongoDB, Cassandra);
- Baze de date graf (ex. Neo4j);
- Baze de date în memorie (ex. Redis).

(c) **Întrebări teoretice esențiale**

- Cum asigurăm consistența și integritatea datelor?
- Care sunt limitele de scalabilitate ale diferitelor modele de baze de date?
- Cum echilibrăm performanța și securitatea în sisteme distribuite?

2. Experiment:

(a) **Studii de caz și validare practică**

Un exemplu notabil este utilizarea sistemului Oracle în cadrul NASA pentru gestionarea datelor științifice din misiuni spațiale. Acest sistem permite integrarea datelor provenite de la multiple instrumente și platforme, garantând acuratețea și disponibilitatea acestora în timp real pentru cercetători (Oracle, 2020).

(b) **Optimizarea interogărilor și a stocării**

Tehnici precum indecșii B+ tree, partitionarea datelor și caching-ul contribuie semnificativ la eficiența execuției interogărilor, în special în aplicații critice de tip OLAP (Online Analytical Processing).

3. Design:

(a) **Instrumente și tehnologii reprezentative**

Sisteme moderne precum PostgreSQL, Microsoft SQL Server și MongoDB oferă funcționalități extinse pentru proiectare, replicare, securitate și analiză. De asemenea, SQL este completat de limbaje precum PL/pgSQL, T-SQL sau limbaje de mapare obiect-relațională (ORM).

(b) **Modele de proiectare a bazelor de date**

Proiectarea bazei de date se bazează pe modele conceptuale (ER), logice și fizice. Fiecare etapă definește structura datelor, relațiile dintre entități și optimizările la nivelul sistemului de gestiune (Date, 2003).

4. Extra:

(a) **Cunoștințe necesare**

Este necesară înțelegerea logicii formale, algebrei relaționale, limbajelor de interogare, dar și a paradigmelor de programare pentru integrarea bazelor de date în aplicații. Familiaritatea cu structuri de date, sisteme de operare și rețele este de asemenea importantă.

(b) **Literatură de specialitate**

- *Database Systems* de Elmasri & Navathe (2015) [185]
- *An Introduction to Database Systems* de C. J. Date (2003) [186]
- *Seven Databases in Seven Weeks* de Redmond & Wilson (2012). [187]

### 12.2.2 Relațiile cu celelalte subdomenii

Bazele de date sunt esențiale pentru funcționarea sistemelor MIS, constituind depozitul central de informații. Arhitectura sistemelor informaticice definește modul în care bazele de date sunt accesate și replicate. Rețelele și securitatea definesc

canalele prin care datele circulă și sunt protejate. Automatizarea proceselor depinde de integritatea și disponibilitatea datelor, iar interfața om-calculator este influențată de modul de acces la informație. Guvernanta IT stabilește regulile de stocare, audit și protecție a datelor.

### 12.2.3 Probleme importante și probleme deschise

- Gestionarea datelor distribuite și replicarea fără pierdere de consistență;
- Modelarea datelor nestructurate în mod performant;
- Scalarea sistemelor la nivel de petabytes;
- Integrarea automatizată cu instrumente de învățare automată.

Probleme deschise:

- Optimizarea interogărilor în baze NoSQL fără limbaj standardizat;
- Dezvoltarea unor modele hibride relațional-NoSQL robuste;
- Definirea unor metamodeluri universale pentru big data;
- Îmbunătățirea transparentei în algoritmii de replicare distribuită.

### 12.2.4 Persoane importante

1. Edgar F. Codd este considerat părintele modelului relațional.[67]
2. C. J. Date a contribuit la formalizarea și popularizarea principiilor relaționale. [186]
3. Michael Stonebraker a dezvoltat primele sisteme de baze de date comerciale și a fost implicat în proiecte ca Ingres și PostgreSQL. [45]
4. Jeff Dean, prin lucrările Google Bigtable și Spanner, a influențat profund gestiunea datelor la scară largă. [71]

### 12.2.5 Forumuri importante

- *ACM SIGMOD Conference*;
- *VLDB* (Very Large Data Bases);
- *IEEE Transactions on Knowledge and Data Engineering*;
- *Journal of Database Management*.

### 12.2.6 Dimensiunea locală și dimensiunea globală

- **Local:** La UVT, bazele de date sunt predate în ciclurile de licență și master prin cursuri de „Bazele de date”, „Sisteme avansate de baze de date” și proiecte practice care implică PostgreSQL, MongoDB și modelare ER.
- **Global:** La nivel global, domeniul este susținut de organizații precum ACM și IEEE, cu standarde (ex. SQL:2023), proiecte open-source și conferințe internaționale. Tehnologiile bazelor de date sunt integrate în cloud computing, big data și aplicații AI la scară largă.

## 12.3 Arhitectura sistemelor informaticice pentru organizații

Arhitectura sistemelor informaticice desemnează structura generală, componentele majore și relațiile dintre acestea din cadrul sistemelor informaticice utilizate în organizații. Acest subdomeniu se ocupă cu definirea, modelarea și gestionarea infrastructurilor informaticice care susțin obiectivele de afaceri. Include elemente hardware, software, rețele, baze de date, politici de securitate și fluxuri informaționale.

Evoluția arhitecturii informaticice a trecut de la modele monolitice (mainframe), la arhitecturi client-server, apoi la modele distribuite și, în prezent, la soluții bazate pe cloud și microservicii. Această tranziție reflectă nevoia de flexibilitate, scalabilitate și integrare în cadrul organizațiilor moderne.

### 12.3.1 Activitățile principale

1. Teorie:

#### (a) Fundamente conceptuale și modelele arhitecturale

Arhitectura sistemelor informaticice este definită printr-un set de principii, standarde și modele care ghidează proiectarea și dezvoltarea sistemelor IT. Modelele de referință precum TOGAF, Zachman sau Gartner sunt utilizate pentru a asigura alinierea între tehnologie și strategia organizațională.

#### (b) Tipologii arhitecturale

Modelele principale includ:

- *Arhitectura în trei niveluri (three-tier)* – separă prezentarea, logica aplicației și accesul la date;
- *Arhitectura orientată pe servicii (SOA)* – oferă interoperabilitate între sisteme prin interfețe standardizate;
- *Arhitectura bazată pe microservicii* – promovează modularitatea, scalabilitatea și independența componentelor;

- *Cloud computing* – resursele sunt accesate și gestionate prin inter-mediu internetului.

(c) **Întrebări teoretice esențiale**

- Cum se proiectează o arhitectură rezilientă la erori și atacuri ciber-netice?
- Care sunt compromisurile între performanță, costuri și flexibilitate?
- Cum poate fi realizată migrarea eficientă către o arhitectură bazată pe cloud?

2. Experiment:

(a) **Studii de caz și implementări reale**

Un exemplu notabil este migrarea companiei Netflix de la un sistem on-premise la o arhitectură complet bazată pe Amazon Web Services (AWS), care a permis scalarea globală a serviciului și disponibilitate ridicată [188].

(b) **Validarea arhitecturilor informatice**

Se realizează prin simulări, prototipuri și testări de performanță, reziliență și securitate. Instrumente precum Kubernetes, Docker sau AWS CloudFormation sunt utilizate pentru modelarea și testarea arhitecturilor moderne.

3. Design:

(a) **Instrumente și standarde utilizate**

Printre cele mai utilizate unele pentru proiectarea arhitecturală se numără: ArchiMate, Enterprise Architect, UML și BPMN. Standardele ISO/IEC 42010 și ITIL v4 sunt esențiale în definirea și evaluarea arhitecturilor informatice.

(b) **Principii de proiectare**

Arhitectura trebuie să fie scalabilă, modulară, sigură și ușor de întreținut. De asemenea, trebuie să permită integrarea cu sisteme externe și să sprijine digitalizarea proceselor organizaționale.

4. Extra:

(a) **Cunoștințe necesare pentru studiu**

Este necesară înțelegerea rețelelor, bazelor de date, sistemelor de operare, securității informatici și modelării proceselor organizaționale. Cunoștințele de DevOps, cloud computing și modelare UML sunt, de asemenea, esențiale.

### (b) Literatură de specialitate

- *Software Architecture in Practice* de Bass, Clements și Kazman (2012)[189]
- *Enterprise Architecture As Strategy* de Ross et al. (2006) [190]
- *Building Microservices* de Newman (2021) [191]

#### 12.3.2 Relațiile cu celelalte subdomenii

Arhitectura sistemelor este strâns legată de bazele de date, care determină structura de stocare a informației, și de rețele, care asigură comunicarea între componente. MIS se bazează pe arhitecturi solide pentru a livra informații fiabile. De asemenea, guvernanța IT oferă cadrul pentru controlul arhitecturii, iar securitatea cibernetică este integrată în proiectare. Interacțiunea om-calculator influențează proiectarea interfețelor în arhitecturi complexe.

#### 12.3.3 Probleme importante și probleme deschise

Problemele actuale includ:

- proiectarea de arhitecturi hibride eficiente;
- menținerea interoperabilității între microservicii;
- automatizarea monitorizării și scalării în cloud;
- echilibrarea securității cu performanța.

Printre provocările deschise se numără: definirea de modele arhitecturale „explainable”, integrarea edge computing și proiectarea de arhitecturi sustenabile energetic.

#### 12.3.4 Persoane importante

Contribuții importante aparțin lui Len Bass [192] (modelarea arhitecturii software), Martin Fowler [47] (arhitectura microserviciilor) și Gregor Hohpe (arhitecturi distribuite și enterprise integration patterns). De asemenea, Mark Richards este cunoscut pentru lucrările de popularizare în domeniul arhitecturii software moderne.

#### 12.3.5 Forumuri importante

Revistele *IEEE Software*, *Journal of Systems and Software*, precum și conferințele *ICSA – International Conference on Software Architecture* și *QRS – IEEE International Conference on Quality, Reliability and Security* sunt esențiale pentru diseminarea cercetărilor.

### **12.3.6 Dimensiunea locală și dimensiunea globală**

- **Local:** La UVT, tematica arhitecturii informaticice este prezentă în cursuri precum „Sisteme distribuite”, „Proiectarea aplicațiilor web” și „Arhitectura sistemelor informaticice”. Studenții dezvoltă proiecte care implică definirea și implementarea unor arhitecturi reale.
- **Global:** La nivel mondial, arhitectura IT este cheia digitalizării, fiind abordată de mari companii precum Google, Netflix și IBM. Programe de formare precum cele oferite de AWS, Microsoft sau Coursera contribuie la profesionalizarea domeniului.

## **12.4 Rețele și securitatea informațiilor în organizații**

Subdomeniul rețelelor și securității informațiilor în organizații se referă la ansamblul de tehnologii, protocole și politici utilizate pentru asigurarea comunicării sigure, eficiente și fiabile între sisteme informaticice. Rețelele facilitează fluxul de date între entități interne și externe, în timp ce securitatea informației se ocupă de protejarea confidențialității, integrității și disponibilității acestor date.

Evoluția acestui domeniu a început în anii 1970–1980, odată cu dezvoltarea primelor rețele de calculatoare și nevoia de protejare a resurselor informaticice. Creșterea utilizării internetului, apariția cloud computing-ului și extinderea spațiilor de lucru digitale au intensificat cerințele de securitate. În prezent, subdomeniul include atât infrastructura fizică (LAN, WAN, VPN), cât și aspecte critice ale securității cibernetice, precum criptarea, firewall-urile, autentificarea, detectia și răspunsul la incidente.

### **12.4.1 Activitățile principale**

1. Teorie:

(a) **Concepțe esențiale și terminologie**

Rețelele de calculatoare permit interconectarea dispozitivelor într-o organizație, folosind protocole precum TCP/IP. Securitatea informațională presupune implementarea unui set coerent de măsuri tehnice și organizaționale pentru a preveni accesul neautorizat, manipularea sau distrugerea informației (Stallings, 2017).

(b) **Clasificare și arhitecturi**

Rețelele pot fi clasificate în funcție de dimensiune (LAN, MAN, WAN), topologie (bus, star, mesh) și model de comunicare (client-server, peer-to-peer). Modelele de securitate urmează standarde precum CIA (Confidențialitate, Integritate, Disponibilitate) sau modelul Zero Trust.

### (c) Întrebări fundamentale

- Cum pot fi detectate atacurile în timp real într-o rețea corporativă?
- Ce strategii sunt eficiente pentru protecția infrastructurilor critice?
- Cum se pot echilibra ușurința în utilizare și securitatea?

## 2. Experiment:

### (a) Studii de caz

Un exemplu relevant este incidentul de securitate din 2013 de la Target, unde o breșă într-un sistem de HVAC conectat la rețea a dus la compromiterea datelor personale ale peste 40 de milioane de clienți. Analiza ulterioară a evidențiat importanța segmentării rețelei și monitorizării traficului neobișnuit (Krebs, 2014).

### (b) Testare și evaluare

Evaluarea securității unei rețele se face prin teste de penetrare, audituri de securitate, simulări de atacuri (red teaming) și analiza logurilor prin sisteme SIEM (Security Information and Event Management).

## 3. Design:

### (a) Protocole și tehnologii

Exemple includ: VPN pentru comunicații criptate, firewall-uri de ultimă generație, IDS/IPS (sisteme de detecție/prevenție a intruziunilor) și autentificare multifactor. Standardele ISO/IEC 27001 și NIST oferă cadre metodologice pentru implementare.

### (b) Instrumente software

Utilizate frecvent: Wireshark pentru analiză de rețea, pfSense pentru firewall, Cisco Packet Tracer pentru simulare, Splunk pentru analiză de securitate.

## 4. Extra:

### (a) Competențe necesare

Sunt necesare cunoștințe de protocole de rețea, criptografie, sisteme de operare, precum și familiaritate cu concepte precum VLAN, ACL, NAT, tunneling și ingerință socială.

### (b) Literatură de specialitate

- *Computer Networking* de Kurose și Ross (2021) [193]
- *Network Security Essentials* de Stallings (2017) [194]
- *Hacking: The Art of Exploitation* de Erickson (2008) [195]

#### **12.4.2 Relațiile cu celelalte subdomenii**

Securitatea informațiilor este un strat transversal care influențează toate componentele informaticе organizaționale. Sistemele MIS și bazele de date trebuie protejate împotriva accesului neautorizat. Arhitectura IT definește modul în care sunt implementate controalele de acces și rețelistică. Automatizarea proceselor implică expunerea unor servicii prin rețea, necesitând măsuri suplimentare de protecție. Interacțiunea om-calculator implică protecția datelor personale, iar guvernanța IT reglementează politicile de securitate.

#### **12.4.3 Probleme importante și probleme deschise**

Printre provocările actuale se numără:

- apariția atacurilor de tip ransomware targetat;
- dificultatea asigurării securității în rețele hibride (on-premise + cloud);
- lipsa specialistilor în domeniul cybersecurity;
- dificultăți în aplicarea politicilor Zero Trust în organizații mari.

Problemele deschise includ:

- dezvoltarea de algoritmi de criptare post-cuantici;
- protecția infrastructurii critice (smart grids, IoT industrial);
- inteligență artificială în detecția proactivă a atacurilor avansate;
- crearea de standarde globale de etică în cybersecurity.

#### **12.4.4 Persoane importante**

Printre contribuabilii importanți se numără Whitfield Diffie [50] și Martin Hellman (cruciali pentru criptarea cu cheie publică), Bruce Schneier [49] (expert în criptografie și politici de securitate), și Kevin Mitnick [196] (reformator și promotor al conștientizării risurilor umane în securitate).

#### **12.4.5 Forumuri importante**

- *IEEE Symposium on Security and Privacy*;
- *USENIX Security Symposium*;
- *ACM Conference on Computer and Communications Security (CCS)*;
- revistele *Journal of Computer Security, Computers & Security*.

#### **12.4.6 Dimensiunea locală și globală**

- **Local:** La UVT, subiectele legate de rețele și securitate sunt abordate în cursuri precum „Rețele de calculatoare”, „Securitatea informației” și în laboratoarele de rețele. Studenții sunt încurajați să participe la competiții precum ECSC (European Cyber Security Challenge) sau proiecte în colaborare cu CERT-RO.
- **Global:** La nivel internațional, domeniul este prioritar pentru guverne, corporații și universități. Inițiative precum ENISA, NIST Cybersecurity Framework și directivele UE privind NIS2 stabilesc standarde. Programe precum CyberPatriot, DEF CON, și cursurile de la Stanford, MIT, și SANS Institute oferă formare avansată.

### **12.5 Automatizarea și optimizarea proceselor organizaționale**

Automatizarea și optimizarea proceselor organizaționale reprezintă un subdomeniu multidisciplinar care vizează utilizarea tehnologiei pentru a eficientiza, standardiza și îmbunătăți operațiunile interne ale unei organizații. Aceasta implică înțelegerea, modelarea și transformarea fluxurilor de lucru cu ajutorul unor soluții informaticе, precum sisteme ERP, BPM (Business Process Management) și RPA (Robotic Process Automation).

Dezvoltarea acestui domeniu a fost alimentată de nevoia organizațiilor de a reduce costurile, de a crește viteza de execuție și de a elimina erorile umane. În ultimele decenii, s-au evidențiat tranzițiile de la automatizare rigidă la optimizare inteligentă, prin integrarea analizei de date și a inteligenței artificiale în procesele operaționale.

#### **12.5.1 Activitățile principale**

1. Teorie:

(a) **Conceptualizarea proceselor organizaționale**

Procesele sunt definite ca un set de activități corelate care contribuie la atingerea unui obiectiv de afaceri. Automatizarea presupune transformarea acestora în fluxuri digitale, în timp ce optimizarea urmărește îmbunătățirea performanței pe baza unor metriți (temp, cost, calitate).

(b) **Metodologii de modelare și analiză**

Printre tehniciile utilizate se numără modelarea cu BPMN (Business Process Model and Notation), analiza fluxurilor de lucru, Six Sigma, Lean Management și simularea proceselor.

### (c) Întrebări teoretice esențiale

- Ce activități pot fi automatizate fără a pierde valoare umană?
- Cum se poate măsura obiectiv performanța unui proces?
- Ce rol are feedback-ul continuu în optimizarea proceselor?

## 2. Experiment:

### (a) Studii de caz relevante

Compania Siemens a implementat o soluție RPA pentru facturare, reducând cu 75% timpul necesar procesării și cu 90% erorile operaționale. Studiul a evidențiat importanța mapării corecte a procesului înainte de automatizare (UiPath, 2021).

### (b) Evaluarea performanței proceselor automatizate

Se utilizează indicatori precum timpul mediu de execuție, rata de eroare, nivelul de reutilizare a componentelor și gradul de integrare între sisteme.

## 3. Design:

### (a) Tehnologii utilizate

Printre soluțiile populare se numără: UiPath, Blue Prism, Automation Anywhere (RPA); Camunda, Bizagi, IBM BPM (BPM); și SAP pentru procese integrate în ERP.

### (b) Principii de proiectare

Designul proceselor trebuie să fie flexibil, centrat pe utilizator, adaptabil la schimbări și sustenabil pe termen lung. Se urmărește standardizarea fără a compromite inovația.

## 4. Extra:

### (a) Cunoștințe necesare

Este necesară înțelegerea principiilor de inginerie a proceselor, algoritmi de optimizare, fluxuri de date, sisteme informaționale și noțiuni de management organizațional.

### (b) Literatură de specialitate

- *Business Process Change* de Paul Harmon (2019) [197]
- *Fundamentals of Business Process Management* de Dumas et al. (2018) [198]
- documentația oficială UiPath Academy.

### **12.5.2 Relațiile cu celelalte subdomenii**

Acest subdomeniu se sprijină pe MIS și bazele de date pentru colectarea și analizarea informațiilor operaționale. Arhitectura sistemelor determină modul de integrare al proceselor. Rețelele și securitatea asigură comunicarea sigură a fluxurilor automatizate, iar interacțiunea om-calculator influențează adoptarea noilor procese de către personal.

### **12.5.3 Probleme importante și probleme deschise**

- rezistența la schimbare organizațională în fața automatizării;
- integrarea datelor provenite din surse eterogene;
- monitorizarea în timp real a proceselor distribuite;
- echilibrarea eficienței cu flexibilitatea în optimizare.

Problemele deschise includ dezvoltarea de algoritmi adaptativi de optimizare, scalarea automatizării în ecosisteme digitale complexe și definirea unor standarde deschise pentru interoperabilitate între platforme BPM/RPA.

### **12.5.4 Persoane importante**

Contribuții semnificative aparțin lui Wil van der Aalst [199] (fondatorul Process Mining), Paul Harmon [197] (consultant și autor în BPM), și Jakob Freund [200] (co-fondator Camunda). De asemenea, Daniel Dines [201] este fondatorul UiPath, una dintre cele mai influente companii din domeniul RPA.

### **12.5.5 Forumuri importante**

- *International Conference on Business Process Management (BPM)*;
- *Journal of Business Process Management*;
- *IEEE Transactions on Automation Science and Engineering*;
- platformele academice și industriale UiPath, Camunda și Gartner.

### **12.5.6 Dimensiunea locală și globală**

- **Local:** La UVT, automatizarea proceselor este studiată în cadrul cursurilor de „Analiza și modelarea proceselor organizaționale” și „Tehnologii RPA”. Studenții sunt implicați în proiecte practice de optimizare cu instrumente reale.

- **Global:** Pe plan internațional, acest domeniu este promovat de organizații precum IEEE, WfMC și de companii inovatoare în RPA. Programe educaționale (ex. UiPath Academy, MIT Sloan) contribuie la formarea de specialiști în optimizare digitală.

## 12.6 Interacțiunea om-calculator în mediul organizațional

Interacțiunea om-calculator (HCI – Human-Computer Interaction) este subdomeniul care studiază proiectarea, evaluarea și implementarea sistemelor interactive utilizate de oameni în contexte organizaționale. Scopul HCI este de a asigura eficiență, accesibilitate și satisfacție în utilizarea aplicațiilor software și a infrastructurii digitale.

Acest domeniu a evoluat de la interfețe bazate pe linia de comandă spre aplicații grafice complexe, dispozitive mobile, realitate augmentată și interfețe vocale. În context organizațional, accentul se pune pe productivitate, ergonomie, formare și adaptabilitate la nevoile utilizatorilor non-tehnici.

### 12.6.1 Activitățile principale

1. Teorie:

(a) **Concepțe fundamentale**

Interacțiunea om-calculator presupune analiza comportamentului utilizatorului, proiectarea interfeței, ergonomie cognitivă, și evaluarea uzabilității. Principii precum consistență, vizibilitatea stării sistemului și controlul de către utilizator sunt esențiale (Norman, 2013).

(b) **Modele teoretice și metodologii**

Modele precum GOMS (Goals, Operators, Methods, Selection rules), modelul mental și paradigma Model-View-Controller (MVC) sunt folosite pentru a înțelege interacțiunea utilizatorului cu sistemul.

(c) **Întrebări fundamentale**

- Cum putem reduce erorile de utilizare fără a limita funcționalitatea?
- Ce modele susțin învățarea rapidă a unei aplicații complexe?
- Cum asigurăm incluziunea digitală pentru toți angajații?

2. Experiment:

(a) **Studii de caz**

Google Workspace a adoptat un design minimalist și integrat care reduce supraîncărcarea cognitivă a utilizatorilor. Testarea A/B și analiza

comportamentului prin heatmaps au fost folosite pentru optimizare iterativă.

(b) **Testarea interfeței și evaluarea UX**

Evaluarea uzabilității se face prin metode precum testing cu utilizatori, heuristics de evaluare (ex. Nielsen) și instrumente de eye-tracking sau analiza comportamentală (ex. Hotjar, Crazy Egg).

3. Design:

(a) **Principii de proiectare**

Se folosesc paradigme precum design centrat pe utilizator, design adaptiv și accesibilitate (WCAG). Interfețele trebuie să fie consistente, responsive și compatibile cross-platform.

(b) **Instrumente și tehnologii**

Figma, Adobe XD, Sketch pentru prototipare; React, Angular pentru interfețe moderne; standarde precum Material Design și Bootstrap pentru coerentă vizuală.

4. Extra:

(a) **Cunoștințe necesare**

Psihologie cognitivă, design grafic, prototipare, front-end development, evaluare UX, metodologii agile și cercetare orientată pe utilizator.

(b) **Literatură de specialitate**

- *The Design of Everyday Things* de Don Norman (2013) [96]
- *About Face* de Cooper et al. (2014) [154]
- *Don't Make Me Think* de Krug (2014) [202]

### 12.6.2 Relațiile cu celelalte subdomenii

Interacțiunea om-calculator influențează accesibilitatea sistemelor MIS, ușurința de utilizare a aplicațiilor de baze de date și interfețele pentru procese automatizate. Arhitectura sistemelor trebuie să susțină interfețe scalabile și rapide. Securitatea informației depinde de implementarea unor controale ușor de înțeles. Guvernanta IT stabilește standarde pentru calitatea interacțiunii.

### 12.6.3 Probleme importante și probleme deschise

- proiectarea pentru utilizatori cu diverse niveluri de pregătire;
- reducerea încărcării cognitive în aplicații complexe;

- adaptarea interfețelor la dispozitive multiple și contexte de utilizare;
- echilibrul între automatizare și controlul manual.

Probleme deschise includ: dezvoltarea de interfețe explicabile pentru sisteme AI, integrarea naturală a interfețelor vocale și tactile, și personalizarea etică a experienței utilizatorului.

#### **12.6.4 Persoane importante**

Don Norman este considerat fondatorul designului centrat pe utilizator. Alan Cooper a introdus conceptul de persona în proiectarea UI. Steve Krug este recunoscut pentru popularizarea uzabilității în designul web. Jakob Nielsen a definit cele mai folosite heuristică UX.

#### **12.6.5 Forumuri importante**

- *CHI – Conference on Human Factors in Computing Systems*;
- *ACM Transactions on Computer-Human Interaction*;
- *UXPA – User Experience Professionals Association*;
- platformele Nielsen Norman Group și Interaction Design Foundation.

#### **12.6.6 Dimensiunea locală și globală**

**Global:** Internațional, domeniul este promovat de instituții academice (MIT Media Lab, Stanford HCI), companii de tehnologie și standarde internaționale (WCAG, ISO 9241). Conferințele CHI și UXPA sunt puncte centrale de diseminare.

### **12.7 Guvernanța IT și managementul strategic al tehnologiilor**

Guvernanța IT și managementul strategic al tehnologiilor reprezintă un subdomeniu esențial care vizează alinierea tehnologiei informației la obiectivele organizaționale, maximizarea valorii aduse de IT și controlul asupra risurilor și investițiilor IT. Guvernanța IT stabilește cadrul prin care deciziile privind tehnologiile sunt formulate, implementate și monitorizate într-o manieră transparentă și eficientă.

Această componentă a apărut în urma conștientizării faptului că tehnologia are un rol critic în succesul strategic al organizațiilor. Dincolo de implementarea tehnică, este necesar un cadru de responsabilitate, măsurare și evaluare a performanței IT. În prezent, guvernanța IT este integrată în guvernanța corporativă și susținută de standarde precum COBIT, ITIL și ISO/IEC 38500.

## 12.7.1 Activitățile principale

1. Teorie:

(a) **Cadre de referință și principii**

Cele mai utilizate cadre sunt COBIT (pentru control și audit), ITIL (pentru managementul serviciilor IT) și ISO/IEC 38500 (pentru guvernanța corporativă a IT). Acestea definesc procese, roluri, indicatori de performanță și bune practici pentru maximizarea valorii IT.

(b) **Strategii de management al tehnologiilor**

Managementul strategic presupune planificarea investițiilor IT, evaluarea riscurilor, dezvoltarea competențelor digitale și monitorizarea rezultatelor tehnologice. Este strâns legat de conceptul de „enterprise architecture”.

(c) **Întrebări teoretice esențiale**

- Cum se poate măsura valoarea adusă de IT în organizație?
- Cine este responsabil pentru eșecurile proiectelor IT majore?
- Cum se asigură transparenta deciziilor IT?

2. Experiment:

(a) **Studii de caz**

Un exemplu relevant este adoptarea cadrului COBIT 5 de către guvernul Africii de Sud pentru îmbunătățirea eficienței agenților publici. Acest model a dus la standardizarea indicatorilor de performanță și la creșterea transparentei în achizițiile IT [203].

(b) **Instrumente de monitorizare și evaluare**

Balanced Scorecard, KPI-uri pentru IT, audituri de conformitate și maturity models sunt utilizate pentru a evalua implementarea strategică și eficiența proceselor IT.

3. Design:

(a) **Instrumente și platforme**

Printre cele mai utilizate se numără ServiceNow, Jira Align, Power BI pentru managementul performanței și platforme ERP cu module de guvernanță.

(b) **Principii de proiectare organizațională**

Se aplică structuri decizionale clare, separarea responsabilităților, cicluri continue de îmbunătățire și integrarea IT în comitetele de conducere.

4. Extra:

(a) **Cunoștințe necesare**

Management strategic, managementul proiectelor, audit IT, reglementări (GDPR, NIS2), indicatori de performanță, leadership digital.

(b) **Literatură de specialitate**

- *IT Governance* de Weill și Ross (2004) [204]
- *Digital to the Core* de Hunter și Westerman (2016) [205]
- Documentația oficială COBIT, ITIL și ISO/IEC 38500.

### 12.7.2 Relațiile cu celelalte subdomenii

Guvernanța IT oferă cadrul care reglementează toate celelalte subdomenii: MIS este evaluat prin indicatori de performanță, bazele de date sunt supuse reglementărilor privind protecția datelor, arhitectura IT este guvernată prin standarde, iar securitatea cibernetică și interfața om-calculator sunt integrate în politicile de conformitate și riscuri IT.

### 12.7.3 Probleme importante și probleme deschise

- lipsa aliniamentului între IT și strategia de afaceri;
- dificultatea de măsurare a ROI-ului pentru investiții IT;
- eșecul proiectelor IT complexe din cauza lipsei de guvernanță;
- provocarea coordonării între IT și alte departamente.

Probleme deschise:

- dezvoltarea de modele predictive pentru decizii IT;
- automatizarea guvernanței prin AI;
- crearea de indicatori de etică și sustenabilitate în tehnologie;
- digitalizarea completă a proceselor de audit și guvernanță.

### 12.7.4 Persoane importante

Peter Weill [204] este autorul unor lucrări fundamentale privind guvernanța IT. Jeanne Ross a contribuit la cercetări despre arhitectura organizațională și IT. Jeanne W. Ross și David Robertson sunt cunoscuți pentru studiile de aliniere între IT și business. Craig Symons (Forrester) [206] a influențat analiza valorii IT în companii.

### **12.7.5 Forumuri importante**

- *ISACA Conference Europe*;
- *IT Governance & Risk Management Summit*;
- *Harvard Business Review* – articole strategice IT;
- *Journal of Information Systems Management*.

### **12.7.6 Dimensiunea locală și globală**

- **Local:** La UVT, concepțele de guvernanță și strategie IT sunt predate în cursuri de „Managementul proiectelor informatici”, „Audit IT” și „Sisteme informaticice pentru management”. Studenții analizează studii de caz și lucrează cu KPI-uri și planuri IT reale.
- **Global:** La nivel internațional, guvernanța IT este esențială în companii globale, administrații publice și instituții financiare. Organizații precum ISACA, ITSMF și Gartner definesc tenduri și bune practici în acest domeniu.

## 13 Bioinformatică

Bioinformatica este o știință interdisciplinară care se ocupă cu dezvoltarea de metode și instrumente software care ajută la colectarea, stocarea, analizarea și diseminarea datelor și informațiilor biologice, pentru a ne îmbunătăți înțelegerea sănătății și a bolilor și, în anumite cazuri, ca parte a îngrijirii medicale. Aceasta combină genetica și biochimia cu informatica, dar și alte domenii, precum statistică, matematică și ingineria.

Subdomeniile bioinformaticii sunt:

1. Analiza secvențelor biologice
2. Genomica
3. Proteomica
4. Bioinformatica structurală
5. Bioinformatica evoluționistă
6. Bioinformatica medicală
7. Bioinformatica aplicată în farmacologie
8. Biotehnologie
9. Bioinformatica sistemelor

### 13.1 Forumuri importante

Principalele conferințe internaționale de bioinformatică sunt:

- BIBE (Bioinformatics and BioEngineering)
  - BIBE este o conferință IEEE internațională fondată în anul 2000, dedicată colaborării dintre bioinformatică, bioinginerie și biomedicină. Scopul său este de a aduce împreună specialiști din aceste domenii complementare pentru a dezvolta soluții inovatoare în medicină, inginerie biomedicală și științele vieții.
- RECOMB (Research in Computational Molecular Biology)

- RECOMB este o serie de conferințe internaționale inițiate în 1997 pentru a oferi un forum științific pentru progresele teoretice în biologia computațională și aplicațiile acestora în biologia moleculară și medicină. Temele principale ale RECOMB includ dezvoltarea de algoritmi, modele matematice și metode statistice pentru analiza datelor moleculare: alinierea secvențelor ADN/ARN/proteină, modelare evoluționistă și filogenetică, genetică computațională, modelare structurală, simulări moleculare, și altele.
- ISMB (Intelligent Systems for Molecular Biology)
  - ISMB a debutat în 1993 și a devenit cel mai mare eveniment mondial dedicat bioinformaticii și biologiei computaționale, atrăgând anual peste 1500 de participanți. Domeniile abordate de ISMB acoperă practic toate aspectele bioinformaticii. Conferința este cunoscută pentru varietatea formelor de contribuții: sesiuni de comunicări orale și postere, articole publicate de obicei în volumele conferinței și workshop-uri asociate.
- BCB (ACM Conference on Bioinformatics, Computational Biology, and Health Informatics)
  - BCB este conferința anuală organizată de ACM SIGBio și este concepută ca un forum interdisciplinar care leagă informatică, matematică, statistică, biologie, bioinformatică, informatică biomedicală și informatică de sănătate. Domeniile de interes includ cercetări fundamentale și aplicate în bioinformatică și biologie computațională, precum și componente orientate spre sănătate.
- ECCB (European Conference on Computational Biology)
  - ECCB este conferința principală europeană de bioinformatică. În anii pari, ECCB se desfășoară împreună cu ISMB, iar această combinare a conferințelor, denumită ISMB/ECCB, devine cel mai important eveniment anual în domeniul bioinformatici. Temele abordate în cadrul ECCB se concentrează pe metode computaționale avansate aplicate problemelor biologice și biomedicale, acoperind arii similare cu ISMB(algoritmi și modele pentru genomica, proteomică, biologie sistemică, biotecnologie și aplicații medicale).
  - Conferința reunește cercetători din informatică, biologie moleculară, matematică și statistică, promovând colaborarea interdisciplinară.
- PSB (Pacific Symposium on Biocomputing)

- PSB este o conferință anuală, fondată în 1996 și organizată tradițional în Hawaii. Scopul său principal este prezentarea și discutarea cercetărilor în teoria și aplicațiile metodelor computaționale pentru biologie. Conferința PSB reunește cercetători din Statele Unite și regiunea Asiatico-Pacifică, facilitând schimbul de rezultate în toate aspectele bioinformaticii și biologiei computaționale. Tematicile PSB sunt alese în fiecare an și includ baze de date biologice, algoritmi de procesare, interfețe de utilizator, vizualizare și modelare computațională, aplicate problemelor biologice, cu accent pe domenile big data.
- WABI (Workshop on Algorithms in Bioinformatics)
  - WABI este o conferință internațională orientată strict spre lucrări algoritmice în bioinformatică, biologie computațională și biologie sistemică. Accentul se pune în principal pe algoritmi discreți și metode de învățare automată care abordează probleme importante din biologia moleculară, care se bazează pe modele solide, care sunt eficiente din punct de vedere computațional și care oferă dovezi ale utilității lor potențiale în practică, de preferință prin testarea pe seturi de date simulate sau reale, alese în mod corespunzător. Scopul este de a prezenta rezultatele cercetărilor recente, inclusiv lucrările semnificative în curs de desfășurare, și de a identifica și explora direcțiile de cercetare viitoare.

Un aticol important este articolul “A brief history of bioinformatics”, scris de Jeff Gauthier, Antony T. Vincent, Steve J. Charette și Nicolas Derome. [207]

Articolul detaliază dezvoltarea istorică a domeniului bioinformaticii, iar autorii subliniază faptul că începuturile domeniului sunt mult mai vechi decât se crede, chiar din anii 1960, înainte de existența calculatoarelor desktop și a secvențierii ADN moderne.

## 13.2 Dimensiunea locală și dimensiunea globală

### • Dimensiunea globală

O listă a universităților internaționale ce oferă programe cu specializare în bioinformatică include:

- Universitatea din Milano i Politecnico di Milano, Italia – Master în Bioinformatică pentru Genomica Computațională
- Universitatea din Bologna, Italia – Master în Bioinformatică
- Universitatea din Verona, Italia – Master în Bioinformatică Medicală
- Universitatea din Skövde, Suedia – Master în Bioinformatică

- Universitatea Hasselt, Belgia – Master în Statistică și Știința Datelor (cu module în bioinformatică)
- Universitatea Konstanz, Germania – Mater în Bioinformatică și Information Mining
- Universitatea din Aveiro, Portugalia – Master în Bioinformatică Clinică
- Universitatea din Maine (UMaineOnline), SUA – Master Profesional în Bioinformatică
- Universitatea de Stat din Dakota de Nord, SUA – Master în Genomică, Fenomică și Bioinformatică
- Universidad Internacional de Valencia (VIU), Spania – Master în Bioinformatică
- Universitatea Johannes Kepler din Linz, Austria – Master în Bioinformatică

#### • Dimensiunea locală

Pe plan local, Universitatea de Vest din Timișoara (UVT) oferă un program de masterat în Bioinformatică, organizat de Facultatea de Matematică și Informatică, în colaborare cu Universitatea de Medicină și Farmacie „Victor Babeș” din Timișoara. Programul este acreditat și se desfășoară pe o durată de 2 ani (4 semestre), totalizând 120 de credite ECTS, cu predare în limba română.

Scopul principal al programului este formarea de specialiști cu competențe interdisciplinare în informatică, genetică, matematică, biologie și chimie. Absolvenții vor fi capabili să utilizeze și să dezvolte instrumente informative pentru prelucrarea datelor biomedicale și modelarea computațională a proceselor biologice.

Programul se adresează absolvenților de licență din domeniile: Informatică, Matematică, Calculatoare și Tehnologia Informației, Medicină, Farmacie, Biologie, Chimie, Fizică medicală.

### 13.3 Analiza secvențelor biologice

În bioinformatică , analiza secvențelor este procesul de supunere a unei secvențe de ADN , ARN sau peptide oricareia dintre o gamă largă de metode analitice pentru a înțelege caracteristicile, funcția, structura sau evoluția acestora. Aceasta poate fi efectuată pe întregul genom, transcriptom sau proteom al unui organism și poate implica, de asemenea, doar segmente sau regiuni selectate, cum ar fi repetițiile în tandem și elementele transpozabile. Metodologiile utilizate includ alinierea secvențelor , căutările în bazele de date biologice și altele.

### **13.3.1 Activitățile principale**

1. Compararea și alinierea secvențelor
  - (a) Evidențiază grade de omologie (înrudire evolutivă) între gene sau proteine diferite. De exemplu, alinierile pot arăta dacă două secvențe de gene provin dintr-un strămoș comun.
  - (b) Algoritmii bazați de principiile programării dinamice ai lui Needleman-Wunsch(1970) și Smith-Waterman(1981) stau la baza acestor comparații, realizând calculul optim al asemănării dintre secvențe.
  - (c) Alinierea secvențelor se realizează de obicei cu programe BLAST [208] sau FASTA (pentru aliniament local rapid).
2. Identificarea elementelor și a caracteristicilor intrinseci ale secvenței
  - (a) Detectarea în cadrul unei secvențe a componentelor funcționale sau structurale importante. Aceasta include localizarea genelor și determinarea structurii lor, identificarea structurilor de secvență conservate (fragmente scurte cu funcții specifice), a siturilor active sau modificate în proteine, precum și a altor elemente reglatoare
  - (b) Această activitate este esențială pentru notarea genomică, adică adăugarea de informații biologice (cum ar fi poziția genelor) pe secvențele brute ale unui genom.
3. Detectarea variantelor și a mutațiilor
  - (a) Compararea secvențelor pentru a găsi diferențe la nivel de nucleotide sau aminoacizi prin identificarea mutațiilor punctiforme (schimbări ale unei baze în ADN) și a polimorfismelor nucleotidice unice (SNP-uri) între secvențe genomice ale diferiților indivizi sau specii.
  - (b) Aceste variații pot acționa ca indicatori genetici ce ajută la maparea genomului și la asocierea variațiilor cu boli sau trăsături ereditare.
  - (c) Analiza variațiilor de secvență este fundamentală cercetărilor în genetică moleculară (de exemplu identificarea mutațiilor asociate cu cancerul sau alte boli genetice).
4. Reconstrucția secvențelor lungi din fragmente
  - (a) Reunirea fragmentelor scurte rezultate din experimente de secvențiere pentru a reconstituи o secvență mai lungă sau un genom întreg.

- (b) Asamblarea secvențelor este necesară pentru a recompune genomul original, fragmentele mici de ADN sunt aliniate și îmbinate în secvență completă.
- (c) Această etapă este esențială în proiecte genomice. De exemplu Proiectul Genomului Uman a necesitat asamblarea a milioane de fragmente pentru a obține secvența completă a ADN-ului uman.
- (d) Asamblarea genomurilor se realizează de obicei cu software de tip de Bruijn graph assemblers.

#### 5. Analiza evoluției și a relațiilor de înrudire

- (a) Prin compararea secvențelor provenite de la organisme diferite, se poate stabili gradul de rudenie și se pot construi arbori filogenetici moleculari (evolutivi). De exemplu, mici diferențe acumulate între secvențe de proteine omoloage din specii diferite pot indica timpul scurs de la un strămoș comun.
- (b) Identificarea genelor se realizează de obicei cu algoritmi de Hidden Markov Models

#### 6. Predicția structurii și funcției pe baza secvenței

- (a) Deși această problemă a fost extrem de dificilă, au fost dezvoltate de-a lungul timpului metodologii de analiză secvențială, cum ar fi modelarea omologă, identificarea unei proteine cu structură cunoscută și secvență similară, sau prezicerea structurilor secundare de ARN pe baza secvenței.
- (b) Predicția funcției genelor sau a proteinelor din secvență se realizează prin căutarea domeniilor conserve cunoscute în secvența proteică sau prin folosirea bazelor de date pentru a găsi gene similare cu funcție cunoscută.
- (c) În ultimii ani, predicția structurii proteinelor din secvență a progresat remarcabil prin inteligența artificială, constituind de asemenea o problemă deschisă în cadrul bioinformaticii.

#### **13.3.2 Relațiile cu celelalte subdomenii**

Analiza secvențelor este strâns interconectată cu celelalte subdomenii din bioinformatică și biologie moleculară, stând chiar la baza multora, precum:

- Genomica
  - Analiza secvențelor biologice este procesul care stă la baza ramurii genomicii, iar fără aceasta genomica nu ar putea exista ca disciplină.

- Prin secvențierea și alinierea genomică, se pot compara genomuri diferite pentru a identifica gene omoloage, regiuni conservate evolutiv sau variații specifice unei specii.
- De asemenea, aceasta depinde de analiza secvențelor pentru a caracteriza funcțiile genelor identificate în secvențele brute și pentru a găsi elementele de reglare genetică pe baza tiparelor de secvență.

- Proteomica

- Deși proteomica implică în principal identificarea experimentală a proteinelor și caracterizarea lor chimică, secvențele de aminoacizi din proteine sunt fundamentale pentru clasificarea și analiza acestora.
- Proteomica interacționează cu analiza secvențelor în special atunci când se prezic situri active enzimatice sau locașuri de legare pe baza secvenței proteice, sau când se caută omologi pentru o proteină de interes în baze de date internaționale.

- Bioinformatica structurală

- În bioinformatica structurală se folosește analiza secvenței pentru a prezice structura 3D a unei proteine (se folosesc metode precum fold recognition sau threading, care aliniază secvența analizată cu structuri cunoscute).
- De asemenea, comparând secvențele a două proteine se poate deduce dacă au structuri similare (dacă două proteine au secvențe foarte apropiate, este foarte probabil să aibă și structuri aproape identice).
- Un succes major al relației dintre aceste două subdomenii este rezolvarea recentă a problemei predicției de structuri proteice folosind inteligență artificială (AlphaFold). Astfel, analiza secvențelor leagă direct secvența liniară al aminoacidului de structura tridimensională, punând în context structural rezultatele secvențierii.

- Genetica moleculară și medicina genomică

- Analiza secvențelor genomice de la mai mulți indivizi ajută la identificarea variantelor genetice ale predispozițiilor la anumite boli. În genetica medicală, secvențierea genei suspectate și compararea cu secvența de referință dezvăluie mutațiile care cauzează boala.
- Bazele de date medicale categorizează aceste variații și folosesc algoritmi de analiză de secvență pentru a prezice efectele mutațiilor.

- Analiza secvențelor cromozomiale stă la baza testelor pentru identificarea de paternitate sau identificarea criminalistică.
- Bioinformatica evoluționistă
  - Prin analiza secvențelor se poate investiga evoluția la nivel molecular, comparând gene din specii diferite și alcătuind arbori filogenetici care arată relațiile de înrudire și ordinea ramificării speciilor.
  - Filogenia moleculară este un domeniu de sine stătător, dar depinde strâns de alinierea secvențelor și de compararea lor, creându-se o legătură între genetica moleculară și teoria evoluției.

### **13.3.3 Probleme Importante și Probleme Deschise**

- Creșterea explozivă a datelor și gestionarea lor
  - Volumul de date de secvență și informație genomică generat de tehnologii moderne de secvențiere se dublează o dată la aproximativ 18 luni, ceea ce duce la probleme de stocare, procesare și analiză eficientă. Decodarea și interpretarea acestor “big data” biologice este dificilă deoarece capacitatea de a analiza o cantitate uriașă de secvențe nu a tăinut întotdeauna pasul cu abilitatea de a le genera. Dezvoltarea de algoritmi mai rapizi și de calcul performanți reprezintă o preocupare permanentă.
  - O altă dimensiune a problemei este calitatea datelor; secvențierea de mare viteză poate introduce erori, iar filtrarea și corectarea erorilor din citiri este un pas esențial înainte de orice analiză.
- Proiectul Genomului Uman și secvențierea genomurilor complexe
  - O problemă importantă a analizei secvențelor a fost Proiectul Genomului Uman în 2003, când cercetătorii au obținut pentru prima dată secvența completă a genomului uman. Acest proiect a fost posibil de realizat datorită metodei de secvențiere Sanger și a progreselor în asamblare și analiză de secvență. De atunci, mii de alte genomuri au fost secvențiate, de la organisme model (de exemplu șoarece sau drozofilă) la plante de cultură și până la genomurile a numeroși indivizi umani, în proiecte precum 1000 Genomes sau inițiative de medicină genomică.
  - Secvențierea și asamblarea genomurilor foarte mari sau foarte complexe rămâne totuși o provocare deschisă. Genomurile cu multe secvențe repetitive (de exemplu genomul grâului, de 17 miliarde de baze) sunt dificil de asamblat integral cu acuratețe. Chiar dacă sunt folosite tehnologii noi, precum secvențierea cu citiri ultra-lungi (PacBio, Oxford Nanopore),

acestea au rata de eroare mai ridicată, și necesită combinarea citirilor scurte cu cele lungi.

- Algoritmii de asamblare trebuie mereu îmbunătățiți pentru a ține pasul cu noile tehnologii și cu mărimea tot mai mare a genomurilor studiate.

- Predicția structurii și funcției proteinelor din secvență

- Una dintre marile probleme deschise ale subdomeniului este deducere structuri 3D a unei proteine din secvența sa de aminoacizi, fiind considerată extrem de dificilă, însă în 2020 echipa DeepMind a construit programul AlphaFold2, bazat pe inteligență artificială, care prezice structurile proteice cu o acuratețe tot mai mare.
- AlphaFold2 a fost antrenat pe baze foarte mari de date de secvențe și structuri cunoscute, ajungând să rezolve cu succes structuri de proteine ce au rămas enigmatice chiar și zeci de ani. Acest succes demonstrează puterea combinării analizei clasice de secvență cu metode de învățare automată.
- Totuși, încă nu toate aspectele sunt rezolvate, de exemplu predicția structurii complexelor proteină-proteină sau a proteinelor într-o membrană celulară rămâne încă dificil de desprins. De asemenea, înțelegerea funcției unei proteine pe baza secvenței sau a structurii sale este încă o problemă deschisă, deoarece AlphaFold indică structura dar nu neapărat și rolul proteinei în celulă. Predicția de funcție este astfel următoarea mare provocare din biologia computațională.

- Algoritmi mai buni și utilizarea inteligenței artificiale

- Cu toate că algoritmii clasici sunt deja bine structurați, se caută în mod continuu îmbunătățirea metodelor de descifrare a secvențelor. De exemplu căutarea de secvențe omoloage în baze de date mari a fost revoluționată de BLAST în 1990, care a introdus o abordare mult mai rapidă decât alinierea optimă clasică, iar acum tendința este utilizarea învățării automate și a învățării profunde (deep learning) pentru probleme de secvență, de la rețele de tip recurrent neural network, până la algoritmi de reinforcement learning ce pot propune noi secvențe.
- Provocările acestor abordări AI includ necesitatea unor seturi masive de date de antrenament de bună calitate și interpretabilitatea redusă.
- Succesul AlphaFold sugerează că inteligența artificială va juca un rol tot mai mare în rezolvarea problemelor de analiză a secvențelor care erau până acum inabordabile.

### **13.3.4 Persoane Importante**

#### **1. Frederick Sanger**

- Este considerat unul dintre părinții analizei secvențelor, fiind primul care a determinat secvența completă a unei proteine (insulina, 1952) [209] și care a dezvoltat ulterior metoda de secvențiere a ADN-ului în 1977, ceea ce reprezintă un moment revoluționar care a permis citirea rapidă a secvențelor de ADN și a stat la baza marilor proiecte genomice de mai târziu, fiind folosită inclusiv în Proiectul Genomului Uman.

#### **2. Margaret Oakley Dayhoff**

- A fost prima care a realizat o bază de date de secvențe biomoleculare, "Atlasul Secvențelor și Structurilor de Proteine" [210], publicat în 1965, care cuprindea toate secvențele de proteine cunoscute la momentul respectiv (doar 65 de secvențe) și a introdus codul cu o literă pentru aminoacizi.
- Ea a fost, de asemenea, pionieră în alinierea secvențelor proteice cu ajutorul calculatorului și în reconstituirea filogeniilor moleculare.
- Prin munca sa, Margaret Dayhoff a pus bazele bioinformaticii ca disciplină, iar atlasul ei și sistemul de acces la date de la distanță au pus bazele marilor baze de date moderne, stând la originea Protein Information Resource (PIR) și împreună cu eforturile lui Walter Goad contribuind și la GenBank.

#### **3. Saul B. Needleman și Christian D. Wunsch**

- Sunt informaticieni care au realizat în 1970 primul algoritm de aliniere globală a secvențelor optim, numit "Algoritmul Needleman-Wunsch", care folosește programarea dinamică pentru a găsi alinierea cu potrivirea maximă între două secvențe.
- Publicarea acestui algoritm marchează de fapt nașterea analizei computaționale a secvențelor ca domeniu.

#### **4. Stephen F. Altschul**

- Este un bioinformatician american, cunoscut pentru rolul principal în dezvoltarea algoritmului BLAST (Basic Local Alignment Search Tool) [208] în 1990, un instrument care a revoluționat căutarea de secvențe omoloage în baze de date. [211]

- Acest algoritm a devenit probabil cel mai folosit instrument din bioinformatică, fiind implementat online la NCBI și în sute de laboratoare și marcând trecerea analizei de secvență în epoca big data.

## 13.4 Genomica

Genomica este studiul setului complet de gene (genomul) organismelor, al modului în care genele funcționează și interacționează între ele și cu mediul înconjurător. Acest subdomeniu încorporează elemente de genetică clasică, dar se concentrează caracterizarea colectivă a genelor unui organism, mai degrabă decât de genele individuale. Fiind o ramură a bioinformaticii, genomica implică utilizarea intensivă a metodelor informatici pentru stocarea și analiza volumelor uriașe de date biologice, în special secvențe ADN, și pentru a investiga structura, funcția genomurilor, modul în care au evoluat și cum pot fi editate.[212]

### 13.4.1 Activitățile principale

1. Secvențierea genomului
  - (a) Determinarea secvenței ADN a genomul unui organism a debutat cu metoda Sanger (prin care s-a efectuat secvențierea primului genom complet, al unui virus în 1977) și a progresat odată cu apariția tehnologiilor moderne de secvențiere.
  - (b) Secvențierea genomică produce cantități foarte mari de date ce necesită stocare și procesare computerizată. Costul și timpul necesar pentru secvențiere au scăzut considerabil, de la miliarde de dolari și mulți ani în cazul Proiectului Genomului Uman, în 1990, la câteva sute de dolari și câteva zile în prezent, făcând posibilă secvențierea de rutină a genomilor în cercetare și practica medicală. [209]
2. Asamblarea și adnotarea genomului
  - (a) După secvențierea ADN-ului bioinformaticienii realizează asamblarea genomului, care ulterior este adnotat, adică se identifică genele și alte elemente funcționale.
  - (b) Pentru genomurile complexe, anotarea este în mare parte automată, fiind folosiți algoritmi de predicție și efectuând comparații cu baze de date de secvențe cunoscute.
  - (c) Adnotarea genomică realizează o hartă a genomului, indicând ce segmente conțin informație proteică, ARN sau elemente reglatoare, contribuind la interpretarea biologică a secvenței brute.

### 3. Compararea genomică

- (a) Reprezintă compararea genomurilor diferitelor specii sau a diferiților indivizi, prin alinierea și compararea secvențelor genomice. Pot fi identificate asemănări și deosebiri între specii, evidențiind gene omoloage sau trasee evolutive comune. De exemplu, comparând genomul uman cu cel al primatelor se poate deduce evoluția și genele conservate.
- (b) Filogenomica folosește datele genomice pentru a construi arbori evolutivi mai precisi, iar genomica comparativă stă la baza identificării variațiilor/mutațiilor genetice asociate anumitor boli.

### 4. Genomica medicală

- (a) Este ramura practică, ce integrează descoperirile genomice în sfera sănătății. Identificarea genelor și a variațiilor genetice aferente bolilor ereditare sau în susceptibilitatea la boli complexe permite diagnosticul molecular și predictia riscului genetic.
- (b) De asemenea, genomica stă la baza mediciniei personalizate. În farmacogenomică se studiază modul în care profilul genetic al unui pacient influențează răspunsul la medicamente, permitând alegerea unor terapii propice individului.

### 5. Metagenomica

- (a) Se ocupă cu analiza materialului genetic colectat direct din medii naturale (ex.: sol, apă, microbiom uman, etc), fără a izola organismele într-o cultură. Prin secvențierea ADN-ului metagenomica permite studierea comunităților microbiene în ansamblu (ex.: flora microbiană intestinală a omului sau microbiomul unei probe de apă oceanică). Această abordare a dus la descoperirea unei diversități de microorganisme și gene noi.
- (b) Metagenomica se bazează pe bioinformatică, deoarece trebuie să amestece și să atribuie fragmentele extrase din secvențe diferitelor specii folosind metode computerizate. Subdomeniul are aplicații în ecologie, epidemiologie și biotecnologie.

### 6. Editarea genomului

- (a) Este o tehnică de manipulare intentionată a secvenței genomice prin tehnologiile de editare genetică. Cea mai cunoscută tehnologie folosită în acest proces este sistemul CRISPR/Cas9, denumit și „foarfecele genetice”, proiectat în 2012 de Emmanuelle Charpentier și Jennifer Doudna.

Această tehnologie oferă posibilitatea de a săia și rescrie cu precizie genomul unui organism, facilitând experimentele funcționale (de exemplu inactivarea unei gene pentru a observa efectul) și spre terapii genice.

- (b) Editarea genomică se bazează pe cunoașterea secvenței genomului său și generează date despre efectele mutațiilor introduse care sunt interpretate cu ajutorul instrumentelor bioinformatici.
- (c) Deși se află încă într-un stadiu începător în aplicarea clinică, tehnologia CRISPR și cele complementare au revoluționat deja cercetarea fundamentală și se numără printre cele mai importante evoluții rezultate din genomică.

#### **13.4.2 Relațiile cu celelalte subdomenii**

- Transcriptomica
  - Bioinformatica transcriptomică implică prelucrarea datelor din secvențele de ARN și maparea lor în funcție de genomul speciei cercetate. Genomica este strâns corelată cu transcriptomica deoarece genomul furnizează secvențele pe care se bazează genele, iar analiza transcriptomică validează și îmbogățește informația genomică (de exemplu ajută la identificarea genelor transcrise).
- Proteomica
  - Deoarece proteinele sunt produsele genelor, genomica și proteomica sunt legate în mod direct. Cunoașterea secvenței genomice ajută la predicția secvențelor de aminoacizi din proteine, iar datele proteomice nu pot fi interpretate eficient fără genom și baza de date de secvențe derivate din acesta.
- Bioinformatica structurală
  - Omologia din secvența genomică poate sugera o omologie structurală între proteine. Genomica furnizează informațiile, identificând ce proteine există, iar bioinformatica structurală încearcă să le asambleze în modele 3D, arătând cum sunt ele configurate și cum interacționează.

#### **13.4.3 Probleme Importante și Probleme Deschise**

- Calitatea și diversitatea eșantioanelor genomicice

- În prezent, există un dezechilibru demografic în bazele de date genetice, iar majoritatea genomurilor secvențiate și studiate provin de la populații cu strămoși europeni. Acest dezechilibru înseamnă că multe variante genetice specifice altor popoare pot rămâne necunoscute, iar rezultatele cercetării s-ar putea să nu fie universal valabile.
- De asemenea, calitatea ADN-ului și a datelor sunt cruciale. Mostrele degradate sau cantitatea insuficientă de ADN pot duce la secvențe incomplete sau erori, fiind necesară reluarea analizelor. Standardizarea procedurilor de colectare a probelor biologice, crearea bazelor de date mai diverse și partajarea acestora între țări și laboratoare sunt inițiative menite să rezolve problema.

- Volumul și gestionarea datelor genomice

- Tehnologiile moderne generează o cantitate uriașă de date, care duce la probleme de stocare, transfer și procesare. Infrastructura IT necesară (servere, cloud computing, baze de date eficiente) și costurile aferente acestiei devin factori limitanți pentru multe instituții de cercetare.
- Crearea unor depozite de date genomice bine organizate și dezvoltarea de algoritmi capabili să proceseze rapid și fiabil acestor baze de date masive reprezintă o provocare continuă.

- Analiza și interpretarea rezultatelor

- Deficitul de personal calificat reprezintă o problemă în analiza complexității datelor genomice, deoarece nu toate echipele de cercetare au bioinformaticieni suficient de experiență, iar acuratețea analizelor este esențială într-un domeniu unde concluziile se bazează pe prelucrări masive de date.
- Provocarea constă în dezvoltarea de instrumente bioinformaticе automatizate și eventual asistate de inteligență artificială, pentru a accelera interpretarea și să reducă dependența de specialiști.

- Aplicarea cercetărilor genomice în practica clinică

- Implementarea clinică a inovațiilor din domeniul genomicii este dificilă, deoarece nu toți medicii dețin cunoștințele necesare, noile protocoale complicate pot descuraja pacienții, iar unitățile clinice pot fi nevoie să investească sume mari în noi tehnologii și infrastructură pentru a face adoptarea posibilă.

#### **13.4.4 Persoane Importante**

##### **1. Frederick Sanger**

- În 1977 echipa sa a reușit să secvențieze primul genom complet din istorie, genomul unui virus bacteriofag [209]

##### **2. James Watson, Francis Crick și Rosalind Franklin**

- Descoperirea structurii duble elicoidale a ADN-ului în 1953 a pus bazele geneticii moleculare și implicit și genomicii. Modelul Watson-Crick al ADN-ului a elucidat modul în care informația genetică este stocată și transmisă, iar Rosalind Franklin a avut o contribuție esențială la decriptarea structuri prin imaginile de difracție cu raze X. [213], [214], [215]
- Fără această descoperire nu am fi putut înțelege structura genomului, concept ce, bineînțeles, stă la baza tuturor proiectelor genomice.

##### **3. Francis Collins**

- Sub coordonarea sa, proiectul Proiectului Genomului Uman și-a atins obiectivul în 2003, secvențiind 3 miliarde de perechi de baze și identificând 20.000-25.000 de gene umane.[216]
- Collins este recunoscut pentru modul în care a coordonat colaborarea între sute de cercetători la nivel internațional, asigurând totodată ca datele genomului uman să fie accesibile în mod public. [217]
- În iunie 2000 acesta a anunțat alături de Craig Venter finalizarea primei schițe a genomului uman, eveniment considerat “nașterea genomicii moderne”.

##### **4. Craig Venter**

- A fost o figură-cheie și controversată în genomica anilor '90-2000. Fiind nemulțumit de ritmul Proiectului Genomului Uman, Venter a fondat o companie privată (Celera Genomics) și a aplicat strategia de secvențiere shotgun la scară largă pe întregul genom, cu ambiența de a finaliza primul genom uman înaintea Proiectului. [218]
- Deși în final cele două echipe au publicat aproape concomitent secvența genomului uman în 2001, Venter a demonstrat eficiența abordărilor industriale și informaticе la scară mare în genomică.

## 13.5 Proteomica

Proteomica este studiul la scară largă al proteinelor (macromolecule vitale ale tuturor organismelor vii, cu numeroase funcții, cum ar fi formarea fibrelor structurale ale țesutului muscular, digestia enzimatică a alimentelor sau sinteza și replicarea ADN-ului). Proteomul este întregul set de proteine produse sau modificate de un organism sau sistem.

Proteomica este un domeniu interdisciplinar care a beneficiat enorm de informațiile genetice din diverse proiecte genomice, inclusiv Proiectul Genomului Uman. Aceasta acoperă explorarea proteomelor de la nivelul general al compozиiei, structurii și activității proteinelor și este o componentă importantă a genomicii funcționale.[219]

### 13.5.1 Activitățile principale

1. Identificarea proteinelor
  - (a) Identificarea proteinelor dintr-un eșantion biologic se realizează prin tehnici experimentale de separare, urmate de tehnici bazate pe spectrometria de masă. Această abordare permite descifrarea profilului proteic al unui specimen, analog modului în care secvențierea ADN identifică genele.
  - (b) Bioinformatica are un rol important, deoarece se utilizează algoritmi specializați (precum SEQUEST) pentru a corela spectrele de masă ale peptidelor cu secvențele din bazele de date genomice, automatizând procesul de secvențiere peptidică și identificare a proteinelor.
2. Cuantificarea expresiei proteice
  - (a) Proteomica cantitativă folosește metode precum marcajul izotopic sau analize label-free pentru a determina cantitatea de proteine dintr-un eșantion și modul în care aceasta se modifică (de exemplu se efectuează comparări între o celulă normală și una bolnavă).
3. Caracterizarea modificărilor post-translaționale
  - (a) Se analizează modificările chimice suferite de proteine după sinteză. Acestea pot altera funcția, activitatea sau stabilitatea proteinelor.
  - (b) Proteomica de modificare implică îmbogățirea peptidelor modificate și detecția lor prin spectrometrie de masă, identificând detalii ale modificării proteinelor. Studiul acestor modificări este crucial pentru înțelegerea mecanismelor de semnalizare celulară și reglaj al activității enzimatice.

#### 4. Studiul localizării și al dinamicii proteinelor

- (a) Pentru a determina structura celulară a proteomului și modul în care proteinele își modifică starea se folosesc procese precum purificarea organitelor urmată de identificarea proteinelor, sau tehnici de „imaging” proteomic (microscopie cu anticorpi marcați). Astfel se cartografiază distribuția proteinelor în celulă și se determină unde și când anume acționează proteinele, completând informația oferită de genom sau transcriptom.

#### 5. Analiza interacțiunilor proteice și a funcțiilor în rețea

- (a) Sunt folosite tehnici precum co-imunoprecipitarea sau afinitatea, urmate de identificare pe bază de spectrometrie de masă pentru a descoperi cu ce parteneri interacționează o anumită proteină studiată.
- (b) Bazat pe rezultatele experimentelor sunt realizate hărți ce determină modul în care proteinele colaborează și rolul acestora în procesele biologice. Aceste studii ajută la descifrarea mecanismelor celulare complexe, punând accentul pe faptul că proteinele acționează interconectat în rețele moleculare.

### 13.5.2 Relațiile cu celelalte subdomenii

- Genomica și transcriptomica

- Genomica furnizează genele unui organism, transcriptomica arată ce ARN mesager este produs, iar proteomica evidențiază proteinele rezultate, care execută majoritatea funcțiilor celulare.
- Proteomica se bazează pe informația genomică; de exemplu pentru identificarea proteinelor prin spectrometrie de masă trebuie comparate fragmentele de peptide cu secvențele teoretice derive din genom.
- Proteogenomica este un subdomeniu ce integrează date de secvență genomică și date proteomice pentru a confirma și adnota gene.

- Metabolomică și biologie sistemelor

- Proteinele mediază reacțiile chimice care compun metabolismul; astfel, proteomica și metabolomica sunt complementare.
- Integrarea datelor proteomice cu cele metabolomice, dar și cu informații genomicice, se realizează în cadrul biologiei sistemelor pentru a construi modele holistice ale celulelor sau organismelor. De exemplu, cunoașterea nivelurilor enzimelor dintr-o cale metabolică (prin proteomică) împreună

cu concentrațiile metaboliștilor (prin metabolomică) permite modelarea fluxurilor metabolice.

- Această integrare a datelor este dificilă, deoarece necesită standarde comune de date și instrumente bioinformatici avansate, dar este esențială pentru a surprinde complexitatea organismelor vii.

- Bioinformatică structurală

- Proteomica structurală are ca scop determinarea structurii tridimensionale a proteinelor și complexelor proteice. Datele proteomice (de exemplu identificarea interacțiunilor sau a regiunilor modificate) pot ghida experimente de cristalografie sau crio-microscopie electronică. Progresația în domeniu (precum algoritmul AlphaFold) au permis predicția structurii proteice la scară genomică, folosind secvențele aminoacizilor. Aceste predicții se bazează pe principii bioinformatici, iar proteomica furnizează date experimentale pentru validare.
- Proteomica oferă target-uri (proteine de interes pentru rezolvare structurală) și date experimentale, iar modelele structurale obținute ajută la înțelegerea modului în care mutațiile sau modificările afectează funcția proteinei.

### 13.5.3 Probleme Importante și Probleme Deschise

- Complexitatea și dimensiunea proteomului

- Proteomul este mult mai complex decât genomul, deoarece un singur genom poate produce sute de proteine diferite. Numărul total de proteiforme este imens, depășind cu mult numărul de gene. În plus, proteinele au o mulțime de variații, ceea ce face dificilă detectarea celor foarte rare.
- Un obstacol major este caracterul spars al datelor, din cauza căruia orice experiment proteomic tinde să eșantioneze doar o parte din proteom, iar proteinele cu abundență scăzută rămân adesea nedetectate. Acest fapt reprezintă o problemă din cauza posibilității ca tocmai proteinele lipsă să fie proteinele cheie în reglarea proceselor.
- Dezvoltarea continuă a instrumentației (spectrometre de masă tot mai sensibile) și a metodologiilor de preparare a probelor vizează ameliorarea acestei probleme, permitând analiza proteomului la nivel de singură celulă și detecția componentelor rare.

- Lipsa unei imagini de ansamblu

- În genomică a existat un proiect major pentru secvențierea întregului genom uman, “Proiectul Genomului Uman”, însă proteomică nu are un asemenea proiect, deoarece proteomul uman nu este fix și unic, ci diferă de la un tip celular la altul și în funcție de timp.
- Chiar dacă au existat inițiative precum Human Proteome Project, scopul acestora este mai degrabă cartografarea treptată a proteinelor codificate de genom decât obținerea unei liste complete valabile universal. Identificarea acestor proteine rămase și elucidarea rolurilor lor biologice reprezintă o problemă deschisă majoră.

- Reproductibilitate și aspecte cantitative

- Fiind un domeniu experimental complex, proteomică s-a confruntat cu probleme de reproducibilitate a rezultatelor între laboratoare diferite. Variabilitatea poate proveni din diferențe în pregătirea probelor, în calibrul instrumentelor sau în analiza datelor. Comunitatea proteomică pune accent tot mai mare pe practicile experimentale, includerea de controale, partajarea seturilor de date brute în repozitorii publice (ex. PRIDE) și publicarea protocoalelor detaliate.
- De asemenea, cuantificarea absolută a proteinelor rămâne dificilă, majoritatea studiilor fiind de natură relativă. Dezvoltarea de standarde proteice și utilizarea isotop-tracerilor interni încearcă să ofere puncte de referință cantitative. Asigurarea reproducibilității și acurateței cantitative este o condiție esențială pentru ca proteomică să fie folosită pe scară largă în aplicații clinice.

#### **13.5.4 Persoane Importante**

1. Marc Wilkins

- A introdus conceptul de proteom în 1994, pe când era doctorand la Universitatea Macquarie. [220]
- A fost printre primii care au propus realizarea hărților tuturor proteinelor unui organism și a contribuit la fondarea primului laborator dedicat proteomicii, Australian Proteome Analysis Facility (APAF), în 1995, fiind considerat un pionier al domeniului.

2. Norman L. Anderson și N. Leigh Anderson

- Sunt un duo de biochimiști (tată și fiu) care în anii '70-'80 au dezvoltat tehnici de electroforeză bidimensională pentru separarea proteică și au pus bazele conceptului de analiză globală a proteomului. Ei au fost

printre primii care au folosit explicit termenul de proteomică într-o publicație științifică, subliniind apariția unor noi concepe și termeni legați de studiul proteomului.

- Articolul lor din 1998 (Electrophoresis) este considerat unul fondator, definind proteomică și subliniind potențialul tehnologiei în descoperirile noi.
- Duo-ul Anderson au realizat primele hărți proteice ale plasmei sanguine și au pus bazele identificării de biomarkeri serici, contribuind astfel la proteomică biomedicală.

### 3. John R. Yates

- A dezvoltat în 1994 algoritmul SEQUEST, prima metodă de corelare automată a spectrelor de masă ale peptidelor cu secvențele din bazele de date, reprezentând practic puntea dintre datele experimentale brute și identificarea proteinei. Acest algoritm a stat la baza multor altor instrumente și este încă folosit pe scară largă. [221], [222]
- De asemenea, Yates este pionierul tehnicii MudPIT (Multidimensional Protein Identification Technology), o abordare de tip shotgun proteomics ce combină cromatografia bidimensională cu tehnologia spectrală de masă pentru a analiza amestecuri proteice complexe integral, fără a necesita separarea pe gel.

## 13.6 Bioinformatica structurală

Bioinformatica structurală este ramura bioinformaticii care se ocupă de analiză și predicția structurii tridimensionale a macromoleculelor biologice (de exemplu proteinele, ARN-ul și ADN-ul). Investigația folding-ului (plierea) molecular, compară foldurile generale și motivele locale ale structurilor, studiază interacțiunile și relația structură-funcție, utilizând atât structuri experimentale rezolvate, cât și modele computaționale.

Obiectivul principal al bioinformaticii structurale este crearea de noi metode de analiză și manipulare a datelor macromoleculare biologice pentru a rezolva probleme în biologie și a genera noi cunoștințe.[223]

### 13.6.1 Activitățile principale

#### 1. Predicția structurii proteinelor

- (a) Această activitate presupune dezvoltarea de algoritmi și programe pentru a prezice structura 3D a proteinelor, plecând de la secvența lor de

aminoacizi. Aceasta include modelarea comparativă (pe baza omologiei cu proteine cunoscute), modelarea de novo fără şablon experimentale și predictia structurii secundare.

## 2. Alinierea structurală și clasificarea structurilor

- (a) Se compară structurile 3D ale proteinelor pentru a evidenția similitudini și diferențe de pliere. Astfel se pot descoperi relații evolutive dintre proteine chiar și atunci când asemănarea de secvență este scăzută.
- (b) Rezultatele se concretizează în baze de date de clasificare structurală precum SCOP și CATH, care grupează structurile cunoscute în funcție de pliuri și evoluție.

## 3. Andocarea moleculară (docking)

- (a) Se simulează interacțiunea dintre două molecule, de obicei un ligand (o moleculă mică, de exemplu un medicament) și o proteină țintă, cu scopul de a prezice poziția de legare și afinitatea dintre ele.
- (b) Această tehnică este esențială în descoperirea de medicamente, ajutând la identificarea modului optim în care o moleculă se poate atașa de situsul activ al unei proteine pentru a-i putea modula funcția.
- (c) Se folosesc algoritmi care generează numeroase poziții posibile ale ligandului în situsul proteinei și evaluează toate cazurile pentru a găsi complexul cel mai stabil.

## 4. Simulații de dinamică moleculară

- (a) Prin dinamica moleculară se pot observa oscilațiile, modificările conformatiionale și interacțiunile moleculare într-un sistem, ceea ce oferă informații despre stabilitatea structurală, flexibilitatea proteinelor și folding.
- (b) Se folosesc metode computaționale pentru a simula mișcarea atomilor și moleculelor în timp conform legilor fizicii (ecuațiile lui Newton și câmpuri de forță moleculare), pentru a putea observa cum oscilează proteinele în soluție, cum interacționează cu liganții sau cum trec prin tranziții conformatiionale și funcționale.

## 5. Dezvoltarea și menținerea bazelor de date structurale

- (a) Cea mai importantă bază de date de structuri biomoleculare este Protein Data Bank (PDB), arhiva internațională care stochează coordonatele 3D ale proteinelor, acizilor nucleici și altor macromolecule obținute experimental. Gestionarea PDB presupune validarea datelor și asigurarea formatelor standard.

- (b) Alte baze de date asociate includ Nucleic Acid Database (NDB) pentru structuri de acizi nucleici și arhive specializate cum ar fi Banca de Date pentru RMN Biomolecular (BMRB), SCOP și CATH (clasificări structurale), sau AlphaFold DB, baza de date de structuri descoperite cu ajutorul prezicerii AI, care conține modele pentru aproape toate proteinele cunoscute.
6. Vizualizarea și analiza structurală
- (a) Se dezvoltă instrumente software și metode pentru vizualizarea interactivă a structurilor (ex.: PyMOL, UCSF Chimera) și analiza caracteristicilor acestora. Reprezentările grafice permit observarea mai clară a elementelor structurale, a interacțiunilor slabe și identificarea motivelor funcționale.
  - (b) Vizualizarea eficientă este esențială pentru interpretarea datelor structurale și validarea rezultatelor.

### **13.6.2 Relațiile cu celealte subdomenii**

- Analiza secvențelor biologice
  - Există o legătură directă între secvența liniară a unei biomolecule (ADN, ARN sau proteină) și structura sa tridimensională. Analiza secvențelor se ocupă cu alinierea și compararea secvențelor, de unde pot fi identificate motive conservate și relații evolutive, informații care sunt punctul de plecare în bioinformatică structurală. De exemplu, modelarea omologă a unei proteine necunoscute se bazează pe aliniamentul secvenței sale cu secvența unei proteine de structură cunoscută.
  - De asemenea, dacă două proteine au secvențe similare, este probabil ca ele să aibă structuri pliate similar, iar cunoașterea structurii poate confirma aliniamentele de secvență, idei care leagă analiza secvențelor de predicția structurală și analiza structurală de predicția secvențială.
- Genomică, proteomică și bioinformatică integrativă
  - După secvențierea completă a genomilor una din marile provocări este adnotarea funcțională a genelor și proteinelor codificate, în cadrul căreia bioinformatică structurală face eforturi sistematice de a determina sau prezice structura fiecărei proteine codificate de un genom. Acest lucru a dus la formarea unor colaborări internaționale în domeniul genomicii structurale, care produc în mod intensiv date de structură, completând datele secvențiale.

- Proteomica de bazează de asemenea pe modele structurale. Cunoașterea structurii unei proteine ajută prezicerea și eventual identificarea funcției sale, la înțelegerea interacțiunilor proteină-proteină și la descifrarea rețelelor celulare complexe.
- În cadrul platformelor integrative, datele structurale sunt combinate cu date de expresie genică, modificări post-translaționale și interacțiuni, pentru a oferi o imagine holistică a celulei.
- Bioinformatica clinică
  - Mutăriile genetice implicate în boli pot fi interpretate mai bine dacă se știe structura proteinei afectate. O mutație poate destabiliza un domeniu structural sau poate afecta un situs de legare critic. Prin modelare structurală, se pot prezice efectele mutațiilor asupra funcției proteinei, contribuind la înțelegerea patogenezei unor boli ereditare sau a mecanismelor de rezistență la anumite medicamente.
- Bioinformatica farmacologică
  - Cunoașterea structurii targetelor biologice permite folosirea tehnicielor de andocare moleculară și screening virtual pentru a identifica molecule mici care se leagă cu afinitate ridicată de acestea, căutându-le în bibliotecile de compuși pentru a propune candidați medicamentoși potriviti, fiind o abordare mult mai eficientă și ieftină decât screening-ul experimental high-throughput.
  - Bioinformatica structurală furnizează platformele software pentru aceste analize (de ex. AutoDock, DOCK, etc.) și algoritmii de drug design asistat de calculator, iar datele structurale permit proiectarea rațională a medicamentelor, prin modificarea deliberată a structurii chimice a compușilor activi pentru a îmbunătăți legarea la țintă, selectivitatea și proprietățile farmacologice.
  - Colaborarea dintre experții în bioinformatică structurală și chimicii duce la optimizarea candidațiilor medicamentoși și scurtarea timpului de dezvoltare a medicamentelor.

### **13.6.3 Probleme Importante și Probleme Deschise**

- Problema plierii proteinelor
  - Deși s-au înregistrat progrese notabile (metode de modelare comparativă, utilizarea inteligenței artificiale ex. AlphaFold), predicția de novo a structurilor cu precizie atomică rămâne dificilă în anumite cazuri.

- Provocările actuale includ predicția structurilor de proteine membranare, a complexelor proteice multimerice și a proteinelor dezordonate.
- Chiar și cu AlphaFold, problema plierii nu este considerată pe deplin închisă și încă sunt necesare îmbunătățiri pentru a prevedea nu doar conformația statică cea mai stabilă, ci și stările alternative și dinamica de pliere.

- Interacțiunile macromolecularare și andocarea

- Predicția fiabilă a interacțiunilor dintre proteine sau dintre proteine și liganzi este o problemă deschisă. În cazul docking-ului proteina-proteina, acuratețea este limitată de flexibilitatea acestora (proteinele pot suferi schimbări conformatiionale semnificative la legare).
- De asemenea, scorarea interacțiunilor reprezintă o altă provocare, deoarece funcțiile folosite de algoritmi nu disting întotdeauna corect complexele native de cele artefact. Bioinformaticienii abordează aceste probleme prin evaluări periodice de tip benchmark, cum este exercițiul CAPRI (Critical Assessment of Predicted Interactions), similar cu CASP dar pentru docking, care stimulează îmbunătățirea metodelor. În mod asemănător, pentru screening-ul virtual de medicamente, provocările includ precizia limitată a funcțiilor de scoring.

- Dinamica și flexibilitatea moleculară

- Majoritatea metodelor de predicție structurală furnizează un model static, dar moleculele biologice sunt sisteme dinamice. Simulațiile de dinamică moleculară permit explorarea mișcărilor, însă sunt costisitoare din punct de vedere computațional și al timpului de execuție. O problemă deschisă este dezvoltarea acestor simulații fie prin algoritmi mai eficienți, fie prin tehnologii hardware.
- O altă problemă este mostrarea conformatiională. Se dorește ca simularea să exploreze suficient de bine toate stările relevante ale moleculelor, iar dezvoltarea de metode enhanced sampling și integrarea datelor experimentale ajută, dar rămâne loc de progres.

- Designul de novo de proteine și biomolecule

- Un domeniu derivat din bioinformatica structurală este proiectarea de proteine noi cu funcții dorite (proteine artificiale care nu există în natură). Deși progresele sunt impresionante, provocări precum proiectarea secvenței optime care poate fi pliate într-o structură stabilă,

sau introducerea în mod controlat de noi funcționalități rămân încă deschise. Aceste probleme combină cunoștințe din chimie, fizică și învățare automată.

- Grupul condus de David Baker a pionierat acest domeniu (software-ul Rosetta și numeroase design-uri reușite de proteine), însă designul de novo este departe de a fi un proces de rutină și necesită algoritmi mai buni și înțelegere fundamentală a principiilor de pliere. Inteligența artificială generativă începe să fie aplicată și în design-ul proteic, ceea ce sporește progresele domeniului, dar încă se află în dezvoltare.

#### 13.6.4 Persoane Importante

##### 1. David Baker

- Biochimist american renomunit pentru predictia și designul de proteine, care a dezvoltat în anii '2000 pachetul software Rosetta, care a introdus metode stocastice de folding proteic și a obținut succese repetitive la competiția CASP de predicție a structurilor. Ulterior, grupul lui Baker a trecut la design de novo, iar în 2010-2020 a creat primele enzime artificiale, proteine care se auto-asamblează și alte nano-constructe proteice, demonstrând că se pot proiecta proteine care nu există în natură. [224]
- În 2021 Baker a fost co-laureat al Premiului Breakthrough în Științe ale Vieții pentru munca sa inovatoare, iar în 2024 i s-a acordat (alături de colegii de la DeepMind) Premiul Nobel pentru Chimie, recunoscând impactul revoluționar al metodelor de predicție și design structural bazate pe AI.
- Contribuțiile lui ilustrează tranziția domeniului de la predicție pasivă la creație activă de structuri, deschizând calea către o biotecnologie bazată pe design rațional.

##### 2. Janet Thornton și Christine Orengo

- Janet Thornton este considerată o pionieră a bioinformaticii structurale, având numeroase lucrări scrise despre motive structurale, enzime și relația secvență-structură, iar împreună cu Christine Orengo, a creat în 1995-1997 baza de date CATH (Class, Architecture, Topology, Homology), care grupează domeniile proteice cunoscute ierarhic, pe criterii de structură și evoluție. [225], [226]

- CATH, alături de SCOP, a devenit un instrument esențial pentru biologi, ajutând la identificarea foldurilor comune și la înțelegerea diversității structurale. [227]
- Janet Thornton a condus ulterior Institutul European de Bioinformatică (2001-2015), promovând integrarea datelor structurale cu cele genomice și educând o nouă generație de bioinformaticieni.
- Christine Orengo este în continuare un lider activ, coordonând inițiative precum rețeaua europeană 3D-BioInfo (în cadrul ELIXIR) menită să integreze eforturile de bioinformatică structurală la nivel european.

### 3. Martin Karplus, Michael Levitt și Arieh Warshel

- Sunt considerați pionieri ai modelării moleculare și ai biologiei structurale computaționale, care au dezvoltat primele metode de simulare pe computer a proteinelor în anii '70 și au demonstrat că un model combinat cu mecanică cuantică și clasica poate descrie reacții enzimatiche complexe. Munca lor a schimbat modul în care gândim despre proteine, punând bazele unei noi discipline.
- Au fost recunoscuți pentru eforturile lor în 2013, când cei trei cercetători au primit Premiul Nobel pentru Chimie „pentru dezvoltarea modелelor multiscalare pentru sisteme chimice complexe”, evidențiind astfel importanța simulațiilor computaționale în înțelegerea moleculelor biologice.

### 4. Helen M. Berman

- Încă de la lansarea Protein Data Bank (PDB) în 1971, Helen Berman s-a implicat activ în acest proiect, fiind co-fondatoarea lui și a devenit ulterior chiar director al PDB, transformându-l într-o resursă globală. [228], [229]
- Berman a inițiat și colaborarea wwPDB (Worldwide Protein Data Bank) în 2003, asigurând unificarea arhivei de structuri la nivel mondial. Datorită viziunii și perseverenței sale, astăzi PDB este recunoscut drept “arhiva unică globală pentru datele structurale 3D ale macromoleculelor” și este fundamentală pentru toată comunitatea științifică.

## 13.7 Bioinformatica evoluționistă

Bioinformatica evoluționistă este un subdomeniu al bioinformaticii care se folosește de metode computaționale, statistice și algoritmice pentru a studia procesele evolutive la nivel molecular. Aceasta se ocupă în principal cu analiza comparativă

a secvențelor ADN, ARN și proteice cu scopul de a reconstrui relațiile filogenetice dintre organisme, de a înțelege dinamica evoluției genelor și a genomurilor și de a detecta semnături ale selecției naturale. Prin utilizarea modelelor matematice și a algoritmilor de aliniere, bioinformatică evoluționistă descoperă istoria evolutivă a diferitelor specii și a genelor.

### 13.7.1 Activitățile principale

1. Analiza aliniamentelor multiple de secvență
  - (a) Este un pas esențial pentru compararea secvențelor omoloage din diferite organisme, cu scopul de a identifica regiuni conservate, a mutațiilor evolutive și variabile între specii și este esențială pentru a înțelege conservarea funcțiilor genelor sau evoluției accelerate.
  - (b) Se folosesc instrumente precum Clustal Omega, MAFFT sau MUSCLE pentru a genera alinieri de înaltă calitate pe baza cărora se vor desfășura cercetările.
2. Reconstrucția filogenetică
  - (a) Se utilizează metode de inferență filogenetică, cum ar fi metoda maximum likelihood și Bayesian inference, folosind programe precum PhyML, RAxML și MrBayes, pentru a construi arbori filogenetici care reflectă relațiile evolutive dintre taxoni sau gene.
  - (b) Astfel se pot stabili relații de rudenie dintre organisme sau gene, se pot estima strămoși comuni, și pe baza metodelor de aliniere și comparare a secvențelor se pot studia mecanismele de evoluție, cum ar fi mutațiile, duplicările genice sau evenimentele de transfer orizontal de gene, sau clasificarea unei gene necunoscute, pornind de la poziția sa într-un arbore filogenetic.
3. Detectarea selecției naturale
  - (a) Analizele comparative de secvență permit identificarea regiunilor genomice supuse selecției pozitive, purificatoare sau derivării genetice, prin modele codonice (de ex., PAML, HyPhy) care estimatează raportul nonsinonimic/sinonimic ( $dN/dS$ ).
4. Analiza genomică comparativ
  - (a) Se compară structurile și conținutul genetic al diferiților genomi pentru a identifica evenimente precum duplicările genice, transferul orizontal

de gene și rearanjamente cromozomiale, care duc la deducerea originii și evoluției acestora.

### 5. Analiza populatională moleculară

- (a) Prin integrarea datelor de secvență în modele demografice (de ex., rețele haplotipice, analiza structurii populationale prin ADMIXTURE sau STRUCTURE) se facilitează descoperirea proceselor evolutive la scară intra-specifică (migrație, drift genetic, selecție locală).

#### 13.7.2 Relațiile cu celelalte subdomenii

- Genomica comparativă
  - Compararea secvențelor genomilor are un rol foarte important, deoarece majoritatea analizelor de filogenie și selecție se bazează pe date genomice la scară largă, iar bioinformatică evoluționistă oferă instrumentele analitice necesare interpretării acestor date, cu scopul de a descoperi gene conservate, inovații genetice (gene noi, gene duplicate sau pierderi de gene care explică adaptări specifice unei specii), transferuri orizontale de gene, sau chiar ritmul evoluției.
- Biologia structurală
  - Cunoașterea evoluției unei proteine este utilă în modelarea structurii sale și în predicția conservării funcționale a situsurilor active.
- Biologia sistemelor și rețelelor
  - Interacțiunile evolutive pot fi modelate și la nivel de rețele moleculare (interacțiuni proteină-proteină, căi metabolice), unde bioinformatică evoluționistă contribuie la înțelegerea co-evoluției componentelor biologice complexe.
- Ecologia moleculară și evoluția adaptativă
  - Datele genomice și metagenomice sunt utilizate în investigațiile interacțiunii dintre gene și mediu, pentru a studia adaptările locale, radiațiile adaptative și ecologia populațiilor naturale.

#### 13.7.3 Probleme Importante și Probleme Deschise

- Calitatea alinierilor și erorile de aliniere

- Descoperirile evolutive depind extrem de mult de calitatea aliniamentelor, deoarece acesta influențează direct calitatea arborelui filogenetic. Erorile de aliniere pot duce la estimări incorecte ale relațiilor filogenetice sau ale presunii de selecție.
- Modele de substituție limitate
  - Majoritatea metodelor de deducere evolutivă se bazează pe presupunerii simpliste despre rata și tipul mutațiilor. Pentru a capta diversitatea proceselor evolutive reale este nevoie de modele mai realiste, dar și mai complexe, iar ramura aflându-se încă în dezvoltare, această problemă rămâne deschisă.
- Genele orfane și evoluția de-novo
  - O altă problemă deschisă din domeniu, care rămâne una dintre întrebările fundamentale în evoluția moleculară, este identificarea originii genelor noi care nu prezintă omologi detectabili. Instrumentele actuale oferă doar o imagine parțială asupra acestui fenomen.
- Reconstruirea arborelui universal al vieții
  - Reprezintă o provocare majoră în bioinformatică evoluționistă, deoarece urmărește identificarea relațiilor filogenetice dintre toate organismele vii, pornind de la un strămoș comun universal (LUCA - Last Universal Common Ancestor).
  - Această reconstrucție se bazează pe analiza comparativă a genelor conservate, precum ARN-ul ribozomal sau proteine esențiale, și utilizează metode avansate de aliniere, inferență filogenetică și modelare evolutivă. Una dintre cele mai mari dificultăți este transferul orizontal de gene (HGT), frecvent întâlnit la organismele procariote, care poate distorsiona structura reală a arborelui, sugerând înrudiri false.
  - Cercetătorii explorează abordări alternative, cum ar fi rețelele filogenetice și metodele bayesiene multilocus, integrate cu algoritmi de învățare automată, pentru a depăși aceste limitări și a contura o imagine cât mai fidelă a evoluției vieții pe Pământ.

#### **13.7.4 Persoane Importante**

1. Walter M. Fitch

- A fost un pionier al filogeniei moleculare, având o contribuție esențială la dezvoltarea metodelor de construire a arborilor evolutivi folosind secvențe de ADN și proteine.
- Este co-autorul lucrării în care a fost folosit pentru prima dată termenul de „filogenie moleculară”, introducând astfel studiul relațiilor evolutive la nivel molecular. Prin munca sa, a pus bazele unor tehnici fundamentale utilizate astăzi în bioinformatică și biologie evoluționistă.

## 2. Joseph Felsenstein

- A avut o contribuție fundamentală la dezvoltarea teoriei filogenetice statistice. Aceasta a proiectat unul dintre primele pachete software complete pentru inferență filogenetică, PHYLogeny Inference Package (PHYLIP). A dezvoltat de asemenea și concepte esențiale precum bootstrapping în filogenie prin introducerea metodelor pentru efectuarea de comparații statistic independente folosind filogenii.

## 3. Ziheng Yang

- Este cunoscut pentru dezvoltarea pachetului PAML (Phylogenetic Analysis by Maximum Likelihood), un instrument esențial în bioinformatică evoluționistă. PAML permite analiza filogenetică precisă, folosind modele statistice complexe și este folosit pentru a detecta selecția naturală, a estima timpii de divergență între specii și a înțelege modul în care genele au evoluat.
- Metodele dezvoltate de Yang sunt frecvent utilizate în cercetări despre evoluție moleculară și stau la baza multor descoperiri moderne în genomică comparativă.

## 4. David Haussler

- Este un bioinformatician american, cunoscut pentru munca sa de conducere a echipei care a asamblat prima secvență a genomului în cadrul Proiectului Genomului Uman și ulterior pentru analiza comparativă a genomului care aprofundează înțelegerea funcției moleculare și a evoluției genomului.
- De asemenea, a contribuit și la dezvoltarea Hidden Markov Models (HMM) în context bioinformatic și la utilizarea acestora pentru predicția genelor și analiza comparativă, inclusiv în context evolutiv.

## **13.8 Bioinformatica medicală**

Bioinformatica medicală este un subdomeniu al bioinformaticii care se ocupă cu aplicarea metodelor informatice în medicină și sănătate, combinând biologia, științele medicale și informatica pentru a colecta, stoca, analiza și interpreta date biologice complexe. Scopul său principal este de a traduce datele biomedicale (secvențe genomice, date clinice, rezultate de laborator) în cunoștințe și instrumente care să ofere o îngrijire tot mai bună pacienților și să sprijine descoperirile medicale. Se folosește de computere și tehnologia informației pentru a gestiona și analiza informația medicală, astfel încât deciziile clinice să fie mai informate, riscurile medicale reduse și costurile optimizate.[230]

### **13.8.1 Activitățile principale**

1. Analiza datelor genomice și genetice clinice
  - (a) Se prelucrează secvențele ADN ale pacienților folosind algoritmi de aliniere a secvențelor, detectarea variantelor genetice și interpretarea lor, pentru a identifica mutații asociate cu boli genetice sau predispoziții la afecțiuni, ca să se poată oferi apoi un tratament personalizat. Un bun exemplu este analiza genomică pentru prevenția sau tratarea cancerului.
2. Integrarea datelor multi-omice și clinice
  - (a) Se combină informații genomice, transcriptomice, proteomice cu date clinice despre pacienți (simptome, analize de laborator, imagistică) pentru a obține o viziune în perspectivă asupra unei boli.
  - (b) Această activitate se mai numește și bioinformatică translatională, și ajută la corelarea interacțiunilor genetice cu manifestările bolii clinice, facilitând înțelegerea bolilor și personalizarea tratamentelor.
3. Dezvoltarea de baze de date și resurse informaționale
  - (a) Bioinformaticienii medicali proiectează baze de date biomedicale (care conțin informații despre gene, boli, mutații, medicamente) și se ocupă de menținerea caestora astfel încât clinicienii și cercetătorii le pot accesa eficient. Un exemplu ar fi baza de date OMIM (Online Mendelian Inheritance in Man).
4. Algoritmi de inteligență artificială (AI) pentru diagnostic și prognostic
  - (a) Dezvoltarea de metode de învățare automată aplicate pe date medicale este o activitate importantă a subdomeniului bioinformaticii medicale, și are scopul de a asista cadrele medicale în luarea deciziilor.

- (b) Aceste modele pot, de exemplu, să analizeze imagini medicale (radiografii, RMN) pentru a detecta leziuni, sau să analizeze istoricul unui pacient și profilul său pentru a detecta riscul la o boală sau pentru a sugera tratament personalizat.

### 13.8.2 Relațiile cu celelalte subdomenii

- Informatica medicală
  - Există o suprapunere parțială între bioinformatică medicală și informatică medicală. Informatică medicală se concentrează pe gestionarea informațiilor din sistemul de sănătate (sisteme de dosare electronice ale pacienților, sisteme de arhivare a imaginilor). Bioinformatică medicală, pe de altă parte, analizează datele biologice la nivel molecular. Cele două domenii se intersectează în zona bioinformaticii translatională, care ”traduce” informațiile biomoleculare pentru a putea fi folosite în aplicații clinice.
- Biostatistica și epidemiologia
  - Epidemiologia se ocupă cu studierea distribuției și determinanților bolilor în populație folosind markeri moleculari (genomi, biomarkeri de sânge, microbiom), colectând date din experimente și furnizându-le prin tipul de date ”big data”. Bioinformatică medicală lucrează cu acest tip de date și folosește instrumente și algoritmi pentru a le analiza și interpreta, corelând markerii moleculari cu fenomene epidemiologice (cum ar fi răspândirea unei boli infecțioase sau predispoziția genetică la o boală complexă), astfel permitând identificarea factorilor de risc la nivel de populație.
- Genetica medicală și genomica
  - Bioinformatică medicală este strâns legată de ramura genomicii și genetica medicală, deoarece datele genomice sunt esențiale pentru a identifica genele implicate în boli și variații cu relevanță clinică.
  - Proiectul Genomului Uman a unit domeniile biologiei, medicinii cu cel al informaticii, deoarece după secvențierea genomului, a fost nevoie de programe care să se ocupe de gestionarea datelor genetice și extragerea informațiilor utile pentru diagnostic și tratament.
- Biologia sistemelor

- Biologia sistemelor analizează interacțiunile complexe din sistemele biologice ca pe un întreg, iar informațiile preluate sunt apoi utilizare în medicina sistemică, unde boala este privită drept rezultat al unei perturbări de rețea biologică complexă, nu doar al unei singure gene sau proteine.
- Bioinformatică medicală preia date din experimente ale biologiei sistemelor și le folosește pentru a construi modele computaționale ale patologiilor.

### **13.8.3 Probleme Importante și Probleme Deschise**

- Integrarea și interoperabilitatea datelor
  - Integrarea datelor este o problemă deschisă, deoarece pentru a obține o imagine completă a sănătății unui pacient este necesară îmbinarea a diferite tipuri de date (genetice, clinice, de stil de viață, mediu înconjurător), iar fiecare tip are propriul format, propria variabilitate și erori specifice.
  - Este necesar să se stabilească standarde comune și platforme capabile să comunice între ele. De exemplu, un algoritm de medicină personalizată ar trebui să poată lua în calcul atât mutațiile genomilor unui pacient dintr-o bază genomică, cât și istoricul său medical din dosarul electronic, dar aceste sisteme nu sunt compatibile direct.
- Translația descoperirilor în practică medicală
  - Trecerea de la stadiul de cercetare la implementarea efectivă într-un spital este dificilă, iar provocările includ validarea clinică riguroasă a noilor algoritmi (asigurarea că un algoritm de diagnostic are acuratețe și specificitate suficientă într-un mediu real, nu doar pe seturi de date de test) și obținerea aprobărilor de reglementare de la autorități.
- Probleme etice, de confidențialitate și aspecte legale
  - Asigurarea confidențialității și securității datelor medicale este o problemă foarte importantă, având în vedere că se lucrează cu genomul personal (care poate dezvăluia predispoziții la boli) sau cu dosarele medicale electronice. Există riscuri de utilizare abuzivă a informațiilor genetice (de exemplu discriminare la angajare).
  - De aceea, cercetătorii trebuie să implementeze măsuri stricte de criptare, anonimizare a datelor și respectare a reglementărilor (precum GDPR în UE sau HIPAA în SUA).

- Nevoia de specialiști și colaborare interdisciplinară
  - O provocare a domeniului este formarea unui număr suficient de specialiști capabili să înțeleagă atât limbajul medical, cât și cel al programării și statisticii. Clinicienii pot fi copleșiți de jargonul tehnic și de complexitatea algoritmilor, iar informaticienii pot să nu fie familiarizați cu constrângerile și nevoile practice dintr-un spital.
  - Ca răspuns la această problemă, se pune accent pe programe educaționale cum ar fi masterate, și doctorate în bioinformatică medicală și pe echipe mixte.

#### **13.8.4 Persoane Importante**

1. Margaret Oakley Dayhoff
  - Pionieră a domeniului, considerată adesea “fondatoarea bioinformaticii”. [210]
2. Francis S. Collins
  - Medic genetician american, renumit pentru descoperirea genelor asociate cu boli ereditare și pentru conducerea Proiectului Genomului Uman.[217], [216]
3. Victor A. McKusick
  - Medic genetician american supranumit “părintele geneticii medicale”. Contribuțiile sale au pus bazele pentru integrarea datelor genetice în medicină. El a inițiat încă din anii 1960 catalogul “Mendelian Inheritance in Man”, o compilație a tuturor trăsăturilor și bolilor umane cu model Mendelian de moștenire. [231]
  - Acest catalog a fost actualizat continuu și ulterior transpus online sub denumirea de OMIM (Online Mendelian Inheritance in Man), devenind o bază de date esențială în bioinformatică medicală.

#### **13.9 Bioinformatica aplicată în farmacologie**

Bioinformatica aplicată în farmacologie combină instrumente de bioinformatică cu date din farmacologie, chimie și biologie moleculară, cu scopul de a înțelege cum interacționează xenobioticele (substanțe străine organismului) cu organismul uman și de a optimiza procesul de descoperire și dezvoltare a medicamentelor.

### **13.9.1 Activitățile principale**

1. Identificarea și validarea țintelor medicamentoase
  - (a) Se folosesc algoritmi de analiză a genomului și a proteomului pentru a identifica proteine sau gene implicate în boli, ce pot fi ținte terapeutice, și se coreleză anumite mutații genetice sau dereglați de expresie genică cu anumite afecțiuni, pentru a descoperi mai eficient aceste ținte asupra cărora un medicament ar putea acționa.
2. Descoperirea și proiectarea asistată de calculator a medicamentelor
  - (a) Se folosesc tehnici de chemoinformatică și modelare moleculară pentru a găsi molecule cu potențial terapeutic. Două metode folosite des ar fi: screening-ul virtual (adică se testează virtual milioane de compuși folosind docking-ului molecular, pentru a vedea cum s-ar lega de ținta de interes) și design-ul de novo și optimizarea ligandului (sunt folosiți algoritmi care generează sau modifică structura chimică a compușilor pentru a îmbunătăți afinitatea și selectivitatea).
  - (b) Metodele precum QSAR (quantitative structure-activity relationship) permit cuantificarea relației dintre caracteristicile fizico-chimice ale moleculelor și activitatea lor biologică, facilitând proiectarea unor molecule mai eficiente.
3. Analiza și gestionarea bazelor de date farmaceutice
  - (a) Bioinformaticienii farmaceutici dezvoltă baze de date de specialitate ce conțin informații despre compuși chimici, proteine țintă, căi metabolice și date clinice (de ex.: ChEMBL, DrugBank, PharmGKB) și le analizează prin procese de data mining pentru a identifica pattern utile, cum ar fi relații între structura chimică și activitate, sau asocieri între variațiile genetice și răspunsul la medicamente.
4. Farmacogenomică și medicina personalizată
  - (a) Această activitate presupune analizarea datelor genomice ale pacienților pentru a înțelege posibile răspunsuri la medicamente.
  - (b) Instrumentele bioinformatici sunt folosite pentru a dezvolta profiluri farmacogenomice și algoritmi care să prezică ce pacienți vor răspunde mai bine la un anumit medicament sau care sunt predispuși la efecte adverse și are ca scop optimizarea tratamentelor personalizate.
5. Modelarea și simularea proceselor farmacologice

- (a) Prin bioinformatică, se pot crea modele matematice care descriu absorbția, distribuția, metabolizarea și excreția unui medicament și interacțiunea acestuia cu ținta biologică. Pe baza acestor date se efectuează simulări care ajută la prezicerea comportamentului unui medicament în organism înainte de testarea efectivă, estimând potențialele probleme de toxicitate sau interacțiuni medicamentoase astfel eliminând potențiale riscuri de reacții adverse ale pacienților.

### 13.9.2 Relațiile cu celelalte subdomenii

- Bioinformatica medicală
  - Bioinformatica farmacologică relatează cu bioinformatica medicală în zone precum analiza eficacității clinice a medicamentelor. De exemplu, ambele domenii folosesc datele genomice ale pacienților: bioinformatica medicală pentru diagnostic și prognostic, iar bioinformatica farmaceutică pentru a corela genotipul cu răspunsul la tratament.
- Bioinformatica structurală
  - Bioinformatica structurală furnizează date precum structurile tridimensionale ale biomoleculelor și instrumentele precum vizualizare moleculară și simulări de dinamică moleculară, care permit farmacologilor să proiecteze molecule ce se potrivesc eficace țintei.
  - Un exemplu celebru este dezvoltarea inhibitorilor enzimei HIV. Cunoscând structura enzimei cercetătorii au putut folosi simulări pe calculator pentru a crea inhibitori potenți.
- Biologia sistemelor și farmacologia sistemică
  - Biologia sistemelor încearcă să înțeleagă organismul ca un tot unitar, iar ideea pe care se bazează farmacologie sistemică este aceea că medicamentele pot perturba rețelele întregi, nu doar o țintă singulară, și că bolile sunt adesea rezultate ale deregării rețelelor biologice complexe.
  - Bioinformatica joacă un rol cheie prin analizarea rețelelor de interacțiuni și identificarea nodurilor critice ce pot fi țintite medicamentos.

### 13.9.3 Probleme Importante și Probleme Deschise

- Modelele computaționale și limitările lor

- Simulările pe calculator sunt doar aproximări ale realității biologice, având precizie limitată. De exemplu, scorurile de docking și algoritmii de predicție a legării ligand-receptor pot produce atât rezultate fals pozitive, cât și fals negative. Limitările de modelare fac ca uneori compușii promițători in silico să eșueze in vitro sau in vivo. Acest lucru este valabil și pentru modelele QSAR (care pot să nu extrapoleze bine în afara setului pe care au fost antrenate) sau pentru rețelele de inteligență artificială.
- Cercetătorii lucrează la îmbunătățirea acestor algoritmi, prin dezvoltarea de funcții de scoring mai precise pentru docking sau la includerea dinamicii moleculare direct în procesul de screening virtual. Un exemplu de progres este utilizarea structurilor 3D calculate de AlphaFold2 pentru înte până acum indisponibile, ceea ce a sporit acuratețea predicțiilor.

- Înțelegerea incompletă a mecanismelor bolilor

- Farmacologia se loveste adesea de cunoașterea limitată a biologiei și a fundamentelor bolilor. Multe eșecuri în dezvoltarea de medicamente se datorează faptului că ipotezele inițiale despre o țintă sau o cale biologică s-au dovedit a fi de fapt incorecte.
- O provocare pentru bioinformatică este deducerea din datele disponibile a rețelelor reale cauzale ale bolii. Modelele de învățare automată care descoperă tipare în date pot sugera ipoteze noi, dar acestea trebuie validate experimental.

- Costurile ridicate și eficiența scăzută în descoperirea de medicamente

- Dezvoltarea unui medicament nou este un proces extrem de costisitor și îndelungat, iar unele din motivele ratei mari de eșec sunt costurile crescute, limitările modelelor preclinice și înțelegerea insuficientă a bolilor.
- Se speră ca instrumentele bioinformaticice bazate pe inteligență artificială să poată atenua această problemă, prin identificarea din timp a compușilor neviabili și concentrarea resurselor pe cei mai promițători.

- Reproductibilitatea și validarea experimentală

- Predicțiile bioinformaticii trebuie confirmate în laborator sau clinic, însă domeniul s-a confruntat uneori cu problema reproducerei, unde algoritmi diferiți pot da rezultate diferite, iar experimentele trebuie să valideze aceste rezultate.

- Atenția este îndreptată spre standardizarea procedurilor pentru a crește încrederea în metodele *in silico*.
- Considerații etice și de reglementare
  - Cu cât bioinformatica preia un rol mai mare în orientarea deciziilor terapeutice (algoritmi care sugerează doza optimă pe baza genomului pacientului, sau programe de AI care propun molecule-candidat), cu atât apar întrebări legate de responsabilitate, transparentă și reglementare. O întrebare crucială, mai ales când miza este siguranța pacientului este ”Cum putem explica deciziile unui algoritm de tip deep learning?”
  - Instituțiile de reglementare (precum FDA) au început să elaboreze ghiduri privind utilizarea algoritmilor în diagnostic și în dezvoltarea medicamentelor.

#### 13.9.4 Persoane Importante

1. Corwin Hansch

- A fost primul care a formulat metoda de corelare cantitativă a proprietăților fizico-chimice ale moleculelor cu activitatea lor biologică în 1960, punând bazele chimiiinformaticii moderne. [232]
- Analiza Hansch (denumirea inițială a conceptului ”QSAR”, Quantitative Structure–Activity Relationship) a oferit cercetătorilor farmaceutici un instrument matematic pentru a prezice potența unor compuși noi pe baza structurii lor, reducând necesitatea testărilor multiple.

2. Russ B. Altman

- A fost printre primii care au integrat date genetice cu informații despre medicamente pentru a înțelege variabilitatea răspunsului la tratament. [233]
- Este co-fondatorul PharmGKB (Pharmacogenomics Knowledge Base), o bază de date fundamentală ce cataloghează gene, variante genetice și răspunsuri medicamentoase și a pus-o la dispoziția comunității științifice.

3. Brian K. Shoichet

- Este cunoscut pentru contribuțiile sale în screening-ul virtual și designul bazat pe structură al medicamentelor. A dezvoltat noi metode de docking și le-a aplicat în primele screening-uri virtuale pe librării mari de molecule.[234]

- A creat resurse de referință pentru domeniu, cum ar fi ZINC database (o bibliotecă cu milioane de compuși disponibili comercial pentru screening virtual) și setul de date DUD (Directory of Useful Decoys) pentru validarea algoritmilor de docking.

4. Janet Thornton

## 13.10 Biotehnologie

Biotehnologia este un domeniu interdisciplinar ce combină biologia cu informatica pentru a aborda probleme complexe din biotehnologie. Bioinformatica în biotehnologie reprezintă utilizarea metodelor și algoritmilor computerizați pentru analiza, gestionarea și stocarea datelor biologice cu relevanță biotehnologică. [235]

### 13.10.1 Activitățile principale

1. Analiza secvențelor ADN/ARN
  - (a) Asamblare de genomi și aliniere de secvențe pentru identificarea genelor, mutațiilor și a relațiilor evolutive. De exemplu, algoritmi ca BLAST pot compara milioane de secvențe rapid, susținând studiile genomice.
2. Modelare și simulare moleculară
  - (a) Prezicerea structurii tridimensionale a proteinelor și modelarea interacțiilor dintre molecule (prin simulări de docking sau dinamica moleculară). Bioinformatica permite simularea in silico a activităților enzimaticе sau a interacțiilor proteină-ligand, esențială pentru dezvoltarea medicalmentelor.
3. Analiza omică integrată
  - (a) Studierea la scară largă a datelor de tip genom, transcriptom și proteom, pentru a înțelege rețelele moleculare complexe. Prin metode bioinformaticе se pot procesa date omice și extrage modele funcționale și biomarkeri pentru agricultură și sănătate.
4. Administrare bazelor de date
  - (a) Construirea și întreținerea de baze de date biologice (ex.: GenBank, UniProt) și de platforme software, care centralizează datele biologice și pun la dispoziție instrumente analitice pentru cercetători.
5. Dezvoltare de instrumente și algoritmi

- (a) Crearea de aplicații software și pachete algoritmice care să automatizeze pașii de analiză (ex. alinierea multiplă, reconstituirea filogenetică, identificarea motivelor proteice). Aceste instrumente se folosesc în proiectarea genetică, diagnosticare și optimizarea proceselor biotecnologice.

### 13.10.2 Relațiile cu celelalte subdomenii

- Ingineria genetică
  - Bioinformatica se ocupă cu identificarea secvențelor țintă și optimizarea constructelor genetice, prin programe de editare precum CRISPR/Cas9, se proiectează ARN-uri și la estimează evenimente off-target, îmbunătățind astfel precizia editării genetice.
- Biologia sintetică
  - Programele bioinformaticii ajută cu designul DBTL și permit proiectarea in silico a componentelor genetice, asamblarea de genomi sintetici și optimizarea producției biochimice. Bioinformatica facilitează proiectarea și testarea modelelor biologice pentru obținerea de noi molecule in vivo.
- Biologia moleculară computațională
  - Bioinformatica furnizează datele și bazele de date necesare, iar biologia moleculară computațională dezvoltă algoritmi pentru înțelegerea mecanicilor moleculare.

### 13.10.3 Probleme Importante și Probleme Deschise

- Big data și integrare de date
  - Volumul masiv de date omice (genome, proteome, metabolome, date fenotipice etc.) ridică probleme de stocare, preprocesare și integrare. Gestionarea eficientă a acestor big data, cu formate și dimensiuni diferite, rămâne o problemă deschisă, având în vedere că programele sunt în continuă dezvoltare și înbunătățire.
- Asigurarea reproductibilității rezultatelor
  - Pentru ca analizele bioinformaticice să fie valide, este necesară urmărirea atentă a parametrilor și a software-ului utilizat și standardizarea proceselor.
- Interpretabilitate și validare

- Pe măsură ce modelele *in silico* devin mai tot complexe, interpretarea biologică a rezultatelor devine tot dificilă. Noile metode bazate pe inteligență artificială oferă putere predictivă, dar interpretarea lor biologică poate fi limitată.

#### **13.10.4 Persoane Importante**

1. Margaret Dayhoff
2. Craig Venter
3. Michael S. Waterman

### **13.11 Bioinformatica sistemelor**

Bioinformatica sistemelor este un subdomeniu care combină biologia, informatică, matematica și știința sistemelor. Aceasta are ca scop analiza și simularea sistemelor biologice complexe prin metode computaționale avansate și privește datele ca parte a unui întreg, mai degrabă decât individual.

#### **13.11.1 Activitățile principale**

1. Modelarea computațională a rețelelor biologice
  - (a) Se concentrează în special pe rețelele complexe, cum ar fi cele genetice, proteice sau metabolice.
2. Simularea proceselor celulare și moleculare
  - (a) Se folosesc diverse metode computaționale pentru a face simulări ale diferitelor procese cu scopul de a le înțelege mai bine.
3. Predicția comportamentului sistemelor
  - (a) Se folosește inteligență artificială pentru a prezice eventuale dezvoltări și comportamente ale sistemelor în condiții patologice în contrast cu cele normale.
4. Analiza perturbațiilor
  - (a) Se analizează modul în care anumiți factori perturbatori, cum ar fi anumite mutații sau medicamente, influențează sistemele biologice complexe.

### **13.11.2 Relațiile cu celelalte subdomenii**

- Biologia computațională
  - Contribuie cu algoritmi specializați și metode de analiză
- Genomica și transcriptomică
  - Bioinformatică sistemelor se bazează foarte mult în cercetare și în proiectarea modelelor sistemic pe datele furnizate de aceste subdomenii.
- Biologia sistemelor
  - Acest subdomeniu al biologiei oferă fundamentele teoretice și ipotezele biologice pe care se bazează bioinformatică sistemelor.
- Inteligența artificială și învățarea automată
  - Acestea au o importantă legătură cu bioinformatică sistemelor și ajută la analiza predictivă și descoperirea de modele complexe.

### **13.11.3 Probleme Importante și Probleme Deschise**

- Complexitatea ridicată a sistemelor biologice
  - Modelarea completă a tuturor interacțiunilor este dificilă, iar golarile și lipsa datelor reprezintă de asemenea o provocare. Inteligența artificială poate prezice anumite părți lipsă, însă acestea trebuie mai întâi validate pentru a putea fi utilizate în cercetare.
- Scalabilitatea algoritmilor
  - Sistemele mari necesită resurse și algoritmi foarte complexi pentru a putea face față variației și volumului de date.
- Calitatea și omogenitatea datelor
  - Datele biologice pot fi incomplete sau incompatibile ceea ce ridică probleme în utilizarea lor.

#### **13.11.4 Persoane Importante**

1. Hiroaki Kitano
  - Pionier al biologiei sistemelor cunoscut pentru lucrările sale privind rețelele genetice și modelarea celulară.[236]
2. Trey Ideker
  - Cercetător cunoscut pentru dezvoltarea rețelelor de interacțiune proteică și aplicațiile acestora în medicină.[237]
3. Bernhard Ø. Palsson
  - Specialist în modelarea metabolică și autor al unor modele genomice de scară largă. [238]

### **14 Implementarea aplicației “Map Explorer”**

Aplicația dezvoltată în cadrul acestui proiect poartă numele “Map Explorer” și a fost realizată folosind limbajul de programare Python, împreună cu bibliotecile CustomTkinter și PIL pentru procesarea imaginilor. Scopul acesteia este de a permite navigarea interactivă printr-o hartă vizuală a domeniilor informaticii, de a ghida și a ajuta studentii, și nu numai, să găsească o direcție în domeniu vast al informaticii. Harta este structurată pe mai multe niveluri ierarhice: domenii (țări), subdomenii (regiuni) și concepte detaliate (clădiri).

#### **14.1 Structura aplicației**

Structura aplicației se bazează pe o detecție de culoare din imaginile de tip overlay, fiecare culoare fiind mapată la o zonă semantică din informatică (vezi Figura 1 și Figura 2).

Se definesc astfel trei niveluri de detecție:

- Nivelul 0 (harta mamă): culorile sunt asociate domeniilor mari, precum „AI și robotică” sau „Arhitectura calculatoarelor”. (Figura 1 și Figura 2)
- Nivelul 1 (regiuni): în cadrul fiecărui domeniu, se definesc subdomenii (ex. „Roboți de tip roi” în cadrul AI).
- Nivelul 2 (clădiri): pentru anumite subdomenii, se pot explora concepte specifice reprezentate vizual prin clădiri (ex. „parc”, „laborator”, „monument”).

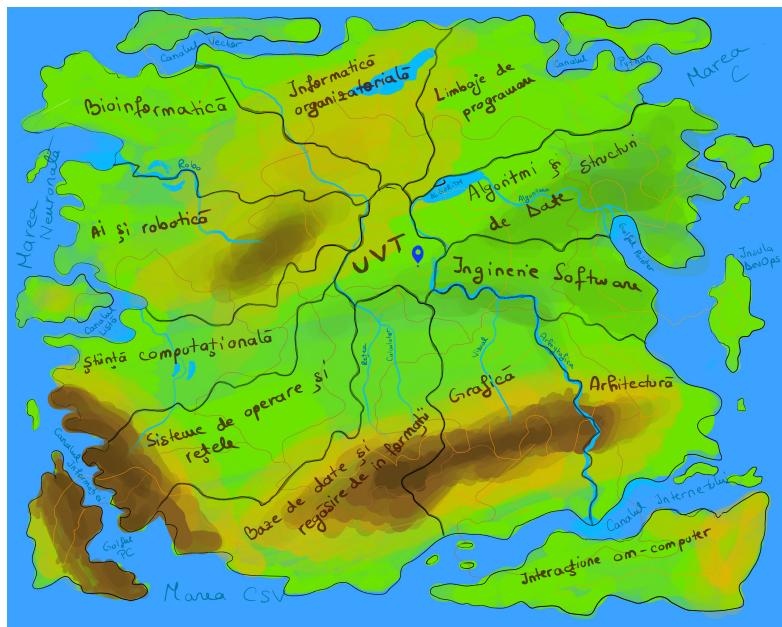


Figura 1: Harta principală

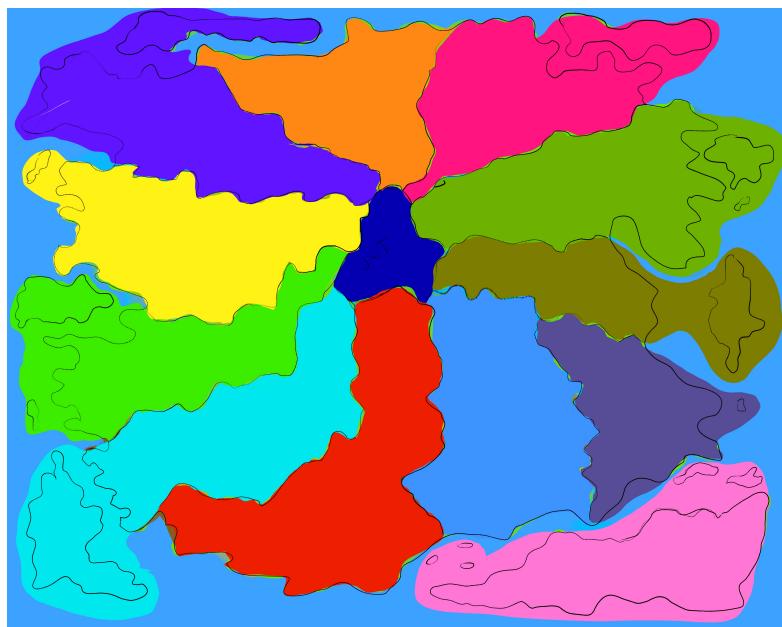


Figura 2: Overlay pentru harta principală

Hărțile și imaginile overlay sunt încărcate din fișiere externe în format .jpg sau .jpeg. Utilizatorul poate interacționa cu aplicația prin click-uri pe zonele colorate, care declanșează detectia culorii și încărcarea hărții următoare. Navigarea este

susținută de un sistem de istoric (`self.history`) care permite revenirea la stările anterioare, precum și de trei butoane funcționale:

- Back: revine la nivelul anterior.
- Home: revine la harta principală.
- Exit: închide aplicația.
- Help: deschide o fereastră ce conține instrucțiuni de utilizare.

## 14.2 Ce se află în spatele interfeței?

Programul, sau mai corect spus codul, este împărțit în șapte secțiuni, fiecare cu rolul și funcționalitatea ei.

Prima parte a programului reprezintă zona de importuri și căi către resurse. În această zonă includem biblioteci precum `customtkinter`, `PIL`, `os`, `webbrowser`, fiecare având rolul său. De exemplu, `webbrowser` - îl folosim pentru a deschide o pagină web pentru „țara” UVT. Tot în această parte de început avem căile către directoarele cu imagini și hărți:

```
BASE_PATH = os.path.join(os.path.dirname(__file__), "assets")
LANDS_PATH = os.path.join(BASE_PATH, "lands")
```

A doua parte a programului este formată din dicționarele de culori pentru a identifica hărțile. Dicționarul `country_map` conține triplete RGB drept chei și numele „țării” pentru valoare. Astfel se asociază culori RGB cu numele domeniilor. Pentru „țara” UVT, în loc de nume am folosit link-ul pentru pagina de internet a universității. Tot în această parte, se află încă două dicționare, cu funcționalități similare, unul pentru fiecare regiune/orăș și unul pentru clădirile din orașe/regiuni.

Aplicația este organizată în jurul unei clase principale, `class MapExplorer`. Această clasă initializează interfața grafică folosind `customtkinter` și `PIL`, gestioneză imaginea curentă și overlay-ul său, încarcă noua hartă, ține evidență nivelului curent și istoricul navigării. Totodată, în această clasă se găsesc metode pentru încărcarea hărților, detectarea culorilor și navigare.

Încărcarea imaginilor are loc folosind metoda `load_map(folder, image_file, overlay_file)` care verifică dacă există imaginea cu extensia `.png`, `.jpg` sau `.jpeg` în folderul specificat și încarcă imaginea vizibilă pentru utilizator. Dacă imaginea are overlay, îl încarcă și pe acesta (pentru hărțile orașelor/regiunilor, țărilor și harta „mamă”). De asemenea, această metodă se ocupă de redimensionarea imaginilor.

O altă componentă importantă a programului este funcția `on_click(event)` care obține coordonatele click-ului, citește pixelul din overlay la poziția respectivă,

obținând o culoare în format RGB. În funcție de nivelul la care se află utilizatorul se execută:

- Nivel 0: verifică în dicționarul `country_map` pentru a deschide o țară
- Nivel 1: verifică în dicționarul `region_color_maps` pentru a deschide un oraș/regiune
- Nivel 2: verifică în dicționarul `city_color_maps` pentru a deschide imaginea unei clădiri

De fiecare dată când trecem la alt nivel starea anterioară se salvează.

Pentru navigarea prin aplicație există o serie de butoane care au implementate următoarele funcționalități:

- Back: reîncarcă ultima hartă salvată în `self.history` și scade cu 1 nivelul la care se află utilizatorul
- Home: resetează întreaga aplicație și revine la harta principală (“harta mamă”)
- Exit: închide aplicația și se afișează doar când e activă harta principală.
- Help: deschide o fereastră ce conține instrucțiuni de utilizare.

Pentru control, pe lângă butoanele enumerate anterior, mai există două funcții foarte importante:

- `update_navigation_buttons()` - afișează și ascunde butoanele Home și Exit în funcție de nivelul curent
- `debounced_update_image(event)` - evită redimensionarea excesivă printr-un sistem de "debounce" (cu întârziere de 150 ms)

Pentru noii utilizatori, am creat o fereastră și un buton de help. La pornirea aplicației se va deschide și fereastra “Help” cu un scurt manual de utilizare. Butonul “Help” care este prezent tot timpul, odată apăsat va deschide această fereastră.

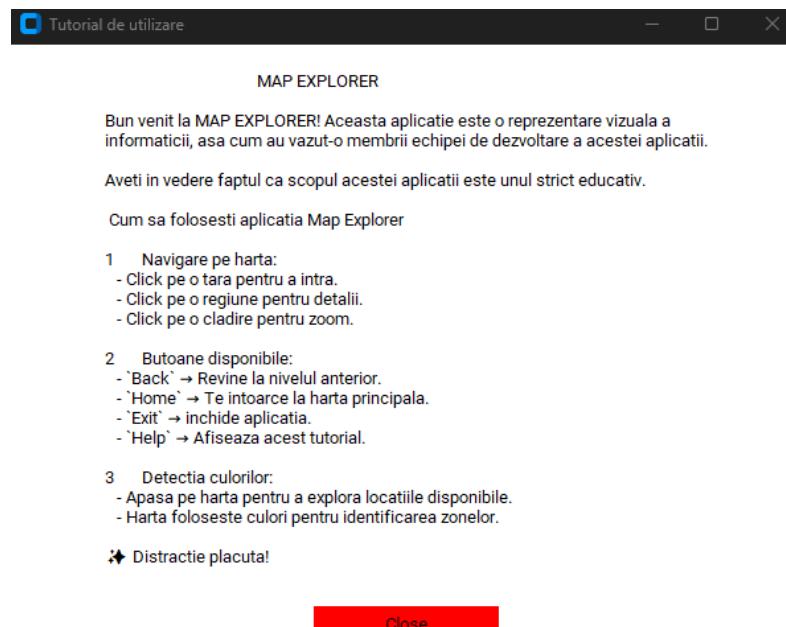


Figura 3: Fereastra de “Help”

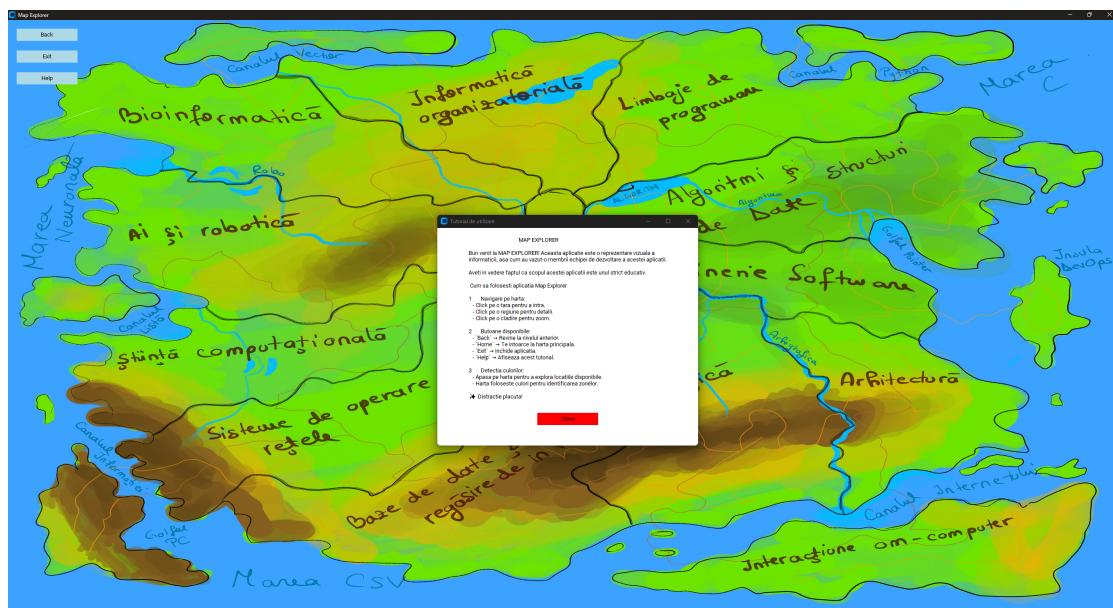


Figura 4: Aplicatia la pornire

## 15 Concluzii și direcții viitoare

Am strâns aceste informații pentru a produce un ghid pentru inițierea în domeniul informaticii și dorim astfel să ușurăm luarea unei alegeri informate asupra viitorului pe termen scurt și mediu. Aici se găsește tot conținutul scris și o aplicație menită să îmbunătățească formarea legăturilor prin reprezentarea grafică a informației pe zone specifice.

S-a dorit reprezentarea scheletului informaticii prin informațiile esențiale, la care, desigur se adaugă eventualele completări din revistele și cărțile de specialitate. Pe viitor se pot reactualiza unele detalii, totuși am creat un instrument util și cu informații actuale. Nu putem garanta veșnicia pentru acest ghid, ci doar încurajăm audiența să continue informarea pe drumul găsit aici. Am încercat să prezintăm intuitiv informațiile și să avem o structură clară pentru fiecare subdomeniu, astfel încât să facilităm transmiterea informației.

Reamintim că aplicația interactivă disponibilă pe GitHub este open-source și poate fi extinsă sau personalizată în funcție de nevoile utilizatorilor. Vă încurajăm să o adaptați, să o folosiți în contexte educaționale sau să propuneți îmbunătățiri.

Sperând că ați avut o călătorie lungă și plăcută pe meleagurile informaticii, aici se termină scurta prezentare a noastră. Vă dorim să investigați cu atenție subdomeniul ales și sperăm să vă dăm un pic de lumină asupra conexiunilor dintre părțile ce definesc informatica.

## Bibliografie

- [1] Peter Denning et al. “Computing as a discipline”. In: *Computer* 22 (Mar. 1989), pp. 63–70. DOI: 10.1109/2.19833.
- [2] Peter J. Denning et al. “Computing as a discipline”. In: *Computer* 22.2 (1989), pp. 63–70.
- [3] Alan M. Turing. “On computable numbers, with an application to the Entscheidungsproblem”. In: *Proceedings of the London Mathematical Society* s2-42.1 (1936), pp. 230–265.
- [4] Kurt Gödel. “Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I”. In: *Monatshefte für Mathematik und Physik* (1931).
- [5] Stephen A. Cook. “The complexity of theorem-proving procedures”. In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC)*. 1971, pp. 151–158.
- [6] Richard M. Karp. “Reducibility among combinatorial problems”. In: *Complexity of Computer Computations*. Springer, 1972, pp. 85–103.
- [7] Edsger W. Dijkstra. “A note on two problems in connexion with graphs”. In: *Numerische Mathematik* 1.1 (1959), pp. 269–271.
- [8] OpenStreetMap Contributors. *OpenStreetMap Project*. 2024. URL: <https://www.openstreetmap.org>.
- [9] Jeremy G. Siek, Lie-Quan Lee, and Andrew Lumsdaine. *The Boost Graph Library: User Guide and Reference Manual*. Addison-Wesley, 2002.
- [10] Chris Lattner and Vikram Adve. “LLVM: A compilation framework for lifelong program analysis and transformation”. In: *Proceedings of the International Symposium on Code Generation and Optimization*. IEEE, 2004, pp. 75–86.
- [11] Volker Strassen. “Gaussian Elimination is Not Optimal”. In: *Numerische Mathematik* 13.4 (1969), pp. 354–356.
- [12] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
- [13] Thomas H. Cormen et al. *Introduction to Algorithms*. 3rd. MIT Press, 2009.
- [14] Donald E. Knuth. *The Art of Computer Programming*. Vol. 1-4. Addison-Wesley, 1997-2011.
- [15] Michael Sipser. *Introduction to the Theory of Computation*. 3rd. Cengage Learning, 2012.

- [16] Stephen A. Cook. “The Complexity of Theorem-Proving Procedures”. In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing*. 1971, pp. 151–158.
- [17] Jon Kleinberg and Éva Tardos. *Algorithm Design*. Pearson Education, 2005.
- [18] Robert Sedgewick and Kevin Wayne. *Algorithms*. 4th. Addison-Wesley, 2011.
- [19] Mark Allen Weiss. *Data Structures and Algorithm Analysis in C++*. 4th. Pearson, 2013.
- [20] Steven S. Skiena. *The Algorithm Design Manual*. 2nd. Springer, 2008.
- [21] Peter Brass. *Advanced Data Structures*. Cambridge University Press, 2008.
- [22] Chris Okasaki. *Purely Functional Data Structures*. Cambridge University Press, 1999.
- [23] Shimon Even. *Graph Algorithms*. 2nd. Cambridge University Press, 2011.
- [24] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
- [25] Mark de Berg et al. *Computational Geometry: Algorithms and Applications*. 3rd. Springer, 2008.
- [26] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. 2nd. Springer, 1993.
- [27] CGAL Project. *Computational Geometry Algorithms Library (CGAL)*. 2025. URL: <https://www.cgal.org>.
- [28] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
- [29] Massachusetts Institute of Technology. *MIT OpenCourseWare – Probability and Computing*. 2025. URL: <https://ocw.mit.edu>.
- [30] Ananth Grama et al. *Introduction to Parallel Computing*. 2nd. Addison-Wesley, 2003.
- [31] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.
- [32] Martin Kleppmann. *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. O'Reilly Media, 2017.
- [33] T. Avacheva and A. Prutzkow. “The Evolution of Imperative Programming”. In: (2019). URL: <https://iopscience.iop.org/article/10.1088/1757-899X/714/1/012001/meta>.
- [34] “Quality and Paradigm”. In: (2021). URL: <https://concerningquality.com/quality-and-paradigm/>.

- [35] K. Khan. “Lecture Notes on Assembly Language Programming”. In: (2024). URL: [https://www.researchgate.net/publication/387321062\\_Lecture\\_Notes\\_on\\_Assembly\\_Language\\_Programming](https://www.researchgate.net/publication/387321062_Lecture_Notes_on_Assembly_Language_Programming).
- [36] *Concepts of Programming Languages – Robert W. Sebesta*, Pearson (2015). URL: [https://www.cs.csubak.edu/~jyang/Robert%20W.%20Sebesta%20-%20Concepts%20of%20Programming%20Languages - Pearson%20\(2015\).pdf](https://www.cs.csubak.edu/~jyang/Robert%20W.%20Sebesta%20-%20Concepts%20of%20Programming%20Languages - Pearson%20(2015).pdf).
- [37] *C Programming Language – 2nd Edition (OCR)*.
- [38] *Programming Languages Book*. URL: <https://www.ime.usp.br/~alvaroma/ucsp/proglang/book.pdf>.
- [39] Richard Bird and Philip Wadler. *Introduction to Functional Programming*. Prentice Hall, 1988.
- [40] “A Survey on Reactive Programming”. In: (2012). URL: [https://www.researchgate.net/publication/233755674\\_A\\_Survey\\_on\\_Reactive\\_Programming](https://www.researchgate.net/publication/233755674_A_Survey_on_Reactive_Programming).
- [41] Joel Falcou. “Practical C++ Metaprogramming”. In: (2016). URL: [https://www.researchgate.net/publication/323994820\\_Practical\\_C\\_Metaprogramming](https://www.researchgate.net/publication/323994820_Practical_C_Metaprogramming).
- [42] P. Fent, G. Moerkotte, and T. Neumann. “Asymptotically Better Query Optimization Using Indexed Algebra”. In: *PVLDB* (2023).
- [43] John L. Hennessy and David A. Patterson. *Computer Architecture: A Quantitative Approach*. 6th. Morgan Kaufmann, 2017.
- [44] Andrew S. Tanenbaum. *Structured Computer Organization*. 6th. Pearson, 2013.
- [45] M. Stonebraker. *Object-Relational DBMS*. Morgan Kaufmann, 1994.
- [46] M. Stonebraker et al. “C-Store: A Column-Oriented DBMS”. In: *VLDB*. 2005.
- [47] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley Professional, 2002.
- [48] Gene Kim et al. *The DevOps Handbook*. IT Revolution Press, 2016.
- [49] Bruce Schneier. *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World*. W. W. Norton & Company, 2015.
- [50] Whitfield Diffie and Martin Hellman. “New Directions in Cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.
- [51] and others. *Operating systems: design and implementation*. Vol. 68.

- [52] William Stallings. *Operating Systems: Internals and Design Principles*, 9/e. Pearson IT Certification, 2018.
- [53] Oracle. *Atomic Access*. URL: <https://docs.oracle.com/javase/tutorial/essential/concurrency/atomic.html> (visited on 05/13/2025).
- [54] Allen B Downey. “The little book of semaphores”. In: (2016).
- [55] Andrew S Tanenbaum and Herbert Bos. *Modern operating systems*. Pearson Education, Inc., 2015.
- [56] Kedar S. Namjoshi and Lenore D. Zuck. “Program Correctness through Self-Certification”. In: *Commun. ACM* 68.2 (Jan. 2025), pp. 74–84. ISSN: 0001-0782. DOI: 10.1145/3689624. URL: <https://doi.org/10.1145/3689624>.
- [57] Michael Howard and Steve Lipner. *Software Security Engineering: A Guide for Project Managers*. Addison-Wesley, 2006.
- [58] Ross Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. 2nd. Wiley, 2008.
- [59] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice*. 3rd. Addison-Wesley, 2013.
- [60] Gary McGraw. *Software Security: Building Security In*. Addison-Wesley, 2006.
- [61] Glenford J. Myers, Corey Sandler, and Tom Badgett. *The Art of Software Testing*. 3rd. Wiley, 2011.
- [62] Roger S. Pressman. *Software Engineering: A Practitioner’s Approach*. 8th. McGraw-Hill, 2014.
- [63] Ian Sommerville. *Software Engineering*. 10th. Pearson, 2016.
- [64] IEEE Computer Society. *Software Engineering Body of Knowledge*. 2014.
- [65] Alexander Kossiakoff and William N. Sweet. *Systems Engineering: Principles and Practice*. 2nd. Wiley, 2011.
- [66] Nicole Forsgren, Jez Humble, and Gene Kim. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*. IT Revolution Press, 2018.
- [67] E. F. Codd. “A Relational Model of Data for Large Shared Data Banks”. In: *Communications of the ACM* 13.6 (1970), pp. 377–387.
- [68] Peter P. Chen. “The Entity-Relationship Model: Toward a Unified View of Data”. In: *ACM Transactions on Database Systems* 1.1 (1976), pp. 9–36.
- [69] Leonid Libkin. *Foundations of Data Exchange*. Morgan & Claypool, 2014.
- [70] A. Guttman. “R-Trees: A Dynamic Index Structure”. In: *SIGMOD*. 1984.

- [71] J. Dean and S. Ghemawat. “MapReduce: Simplified Data Processing on Large Clusters”. In: *OSDI*. 2004.
- [72] P. G. Selinger et al. “Access Path Selection in a Relational Database Management System”. In: *SIGMOD*. 1979.
- [73] G. Graefe. “The Volcano Optimizer Generator: Extensibility and Efficient Search”. In: *ICDE*. 1993.
- [74] J. Gray and A. Reuter. *Transaction Processing*. Morgan Kaufmann, 1993.
- [75] P. Bailis et al. “Highly Available Transactions: Virtues and Limitations”. In: *VLDB* (2014).
- [76] D. Abadi et al. “Column-Stores vs. Row-Stores: How Different Are They Really?” In: *SIGMOD*. 2008.
- [77] C. Mohan et al. “ARIES: A Transaction Recovery Method”. In: *ACM TODS* (1992).
- [78] R. Sandhu et al. “Role-Based Access Control Models”. In: *IEEE Computer* (1996).
- [79] R. A. Popa et al. “CryptDB: Processing Queries on an Encrypted Database”. In: *Communications of the ACM* 55.9 (2012), pp. 103–111.
- [80] N. Johnson, J. P. Near, and D. Song. “Towards Practical Differential Privacy for SQL Queries”. In: *Proceedings of the VLDB Endowment* 11.5 (2018), pp. 526–539.
- [81] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. 2nd. Chapter 24: Protectia criptografică a bazelor de date. Wiley, 1996.
- [82] H. Hacıgümüş et al. “CryptDB: Processing Queries on an Encrypted Database”. In: *VLDB*. 2012.
- [83] E. Brewer. *Towards robust distributed systems*. PODC Keynote. 2000.
- [84] A. Lakshman and P. Malik. “Cassandra: A Decentralized Structured Storage System”. In: *SIGOPS*. 2009.
- [85] J. Corbett et al. “Spanner: Google’s Globally-Distributed Database”. In: *OSDI*. 2012.
- [86] M. Kleppmann. *Designing Data-Intensive Applications*. O’Reilly, 2017.
- [87] R. Kimball and M. Ross. *The Data Warehouse Toolkit*. 3rd. Wiley, 2013.
- [88] M. Armbrust et al. “Lakehouse: A New Generation of Open Platforms”. In: *CIDR*. 2021.

- [89] P. Boncz, M. Zukowski, and N. Nes. “MonetDB/X100: Hyper-Pipelining Query Execution”. In: *CIDR*. 2005.
- [90] R. Angles and C. Gutierrez. “Survey of Graph Database Models”. In: *ACM Comput. Surv.* (2012).
- [91] H. Garcia-Molina et al. “The TSIMMIS Approach to Mediation: Data Models and Languages”. In: *Journal of Intelligent Information Systems* (1997).
- [92] A. Halevy and A. Doan. “Data Integration: The Teenage Years”. In: *VLDB*. 2006.
- [93] A. Doan, A. Halevy, and Z. Ives. *Principles of Data Integration*. Morgan Kaufmann, 2012.
- [94] M. S. Lew et al. “Content-Based Multimedia Information Retrieval: State of the Art and Challenges”. In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)* (2006).
- [95] R. Jain. “Content-Based Multimedia Information Retrieval”. In: *IEEE Computer* (1999).
- [96] Donald A. Norman. *The Design of Everyday Things*. Basic Books, 2013.
- [97] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.
- [98] M. Mijwil. “What is 3D animation?” In: (2018). URL: [https://www.researchgate.net/publication/323969183\\_What\\_is\\_3D\\_Animation\\_Create\\_Cartoons](https://www.researchgate.net/publication/323969183_What_is_3D_Animation_Create_Cartoons).
- [99] “Ivan Sutherland and Bob Sproull create the first Virtual Reality Head Mounted Display System”. In: (). URL: <https://www.historyofinformation.com/detail.php?id=861>.
- [100] Xuya Yan. “Beyond Toy Story”. In: (2024). URL: [https://www.researchgate.net/publication/385085741\\_Beyond\\_Toy\\_Story\\_Pixar's\\_Business\\_and\\_Marketing\\_Strategies\\_for\\_Enduring\\_Success\\_in\\_Entertainment](https://www.researchgate.net/publication/385085741_Beyond_Toy_Story_Pixar's_Business_and_Marketing_Strategies_for_Enduring_Success_in_Entertainment).
- [101] Dragos T. George G. “3D Model based process in automotive industry”. In: (2018). URL: [https://www.researchgate.net/publication/323878455\\_3D\\_Model\\_Based\\_Process\\_in\\_Automotive\\_Industry](https://www.researchgate.net/publication/323878455_3D_Model_Based_Process_in_Automotive_Industry).
- [102] 3Dsoft. URL: <https://www.re-thinkingthefuture.com/career-advice/a2944-a-timeline-of-3d-softwares/>.
- [103] IEEE3D. URL: <https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=2945>.
- [104] JCGT. URL: <https://jcgt.org/>.

- [105] Edwin Catmull. “The Design of RenderMan”. In: (2021). URL: [https://www.researchgate.net/publication/353296274\\_The\\_Design\\_of\\_RenderMan](https://www.researchgate.net/publication/353296274_The_Design_of_RenderMan).
- [106] *JimBlinn*. URL: [https://en.wikipedia.org/wiki/Jim\\_Blinn](https://en.wikipedia.org/wiki/Jim_Blinn).
- [107] Turner Whitted. “Origins of Global Illumination”. In: (2020). URL: [https://www.researchgate.net/publication/338440221\\_Origins\\_of\\_Global\\_Illumination](https://www.researchgate.net/publication/338440221_Origins_of_Global_Illumination).
- [108] *SigGraph*. URL: <https://s2024.siggraph.org/>.
- [109] *BlenderArtists*. URL: <https://blenderartists.org/>.
- [110] Morelia Maravilla et al. “Defining virtual reality: Insights from research and practice”. In: 2019. URL: [https://www.researchgate.net/publication/332201962\\_Defining\\_virtual\\_reality\\_Insights\\_from\\_research\\_and\\_practice](https://www.researchgate.net/publication/332201962_Defining_virtual_reality_Insights_from_research_and_practice).
- [111] Magdalena and Brendan Jackson Balcerak Jackson. “Immersive Experience and Virtual Reality”. In: (2024). URL: [https://www.researchgate.net/publication/378127999\\_Immersive\\_Experience\\_and\\_Virtual\\_Reality](https://www.researchgate.net/publication/378127999_Immersive_Experience_and_Virtual_Reality).
- [112] “What is presence in Virtual Reality (VR)?” In: (2023). URL: <https://www.interaction-design.org/literature/topics/presence?srsltid=AfmB0oqCZIrF08orZS0LtCt-F2X3Wk7akrTTcgiLvHyTFgQg9PROHxbS>.
- [113] Yiwen Liu. “Analysis of interaction methods in VR Virtual Reality”. In: (2023). URL: [https://www.researchgate.net/publication/369872730\\_Analysis\\_of\\_Interaction\\_Methods\\_in\\_VR\\_Virtual\\_Reality](https://www.researchgate.net/publication/369872730_Analysis_of_Interaction_Methods_in_VR_Virtual_Reality).
- [114] Andrew Pressman. “Utilizing VR game design to improve problem-solving and logical thinking skills”. In: (2024). URL: [https://www.researchgate.net/publication/359790641\\_Utilizing\\_virtual\\_reality\\_game\\_design\\_to\\_improve\\_problem\\_solving\\_and\\_logical\\_thinking\\_skills](https://www.researchgate.net/publication/359790641_Utilizing_virtual_reality_game_design_to_improve_problem_solving_and_logical_thinking_skills).
- [115] “The Sensorama: One of the First Functioning Efforts in Virtual Reality”. In: (). URL: <https://www.historyofinformation.com/detail.php?id=2785>.
- [116] Michael Gigante. “Virtual Reality: definitions, history and applications”. In: (1993). URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1200000/pdf>.
- [117] Michael Holly. “VRChances”. In: (2024).
- [118] Rakesh Margam. “VR in Healthcare”. In: (2024). URL: [https://www.researchgate.net/publication/377534519\\_Virtual\\_Reality\\_in\\_Healthcare\\_Transforming\\_Treatment](https://www.researchgate.net/publication/377534519_Virtual_Reality_in_Healthcare_Transforming_Treatment).

- [119] Amit Verma. “An examination of skill requirements for augmented reality and virtual reality job advertisements”. In: (2022). URL: <https://journals.sagepub.com/doi/full/10.1177/0950422221109104>.
- [120] Jer Herder. “The Journal of Virtual Reality and Broadcasting”. In: (2004). URL: <https://www.jvrb.org/jvrb>.
- [121] Robert Macredie. “Virtual Reality - Springer”. In: (1995). URL: <https://link.springer.com/journal/10055>.
- [122] Mel Slater. “A framework for immersive virtual environments”. In: (1997).
- [123] Jeremy Bailenson. “A framework for Immersive Virtual Environments”. In: (2015). URL: <https://www.tandfonline.com/doi/abs/10.1080/15213269.2015.1015740>.
- [124] *VRChat Ask Forum*. URL: <https://ask.vrchat.com/>.
- [125] *VR/AR Association*. URL: <https://www.thevrara.com/>.
- [126] *Optionale UVT FMI*. 2024. URL: <https://info.uvt.ro/optionale/>.
- [127] Serkan Cankaya Ibrahim Sunger. “Augmented Reality: historical Development and area of usage”. In: (2019). URL: [https://www.researchgate.net/publication/336537084\\_Augmented\\_Reality\\_Historical\\_Development\\_and\\_Area\\_of\\_Usage](https://www.researchgate.net/publication/336537084_Augmented_Reality_Historical_Development_and_Area_of_Usage).
- [128] Ronald Azuma. “A survey of Augmented Reality”. In: (1997). URL: <https://direct.mit.edu/pvar/article-abstract/6/4/355/18336/A-Survey-of-Augmented-Reality?redirectedFrom=fulltext>.
- [129] Hirozaku Kato. “Development of ARToolkit”. In: (2002). URL: [https://scholar.google.com/citations?view\\_op=view\\_citation&hl=ja&user=zlyuC60AAAAJ&citation\\_for\\_view=zlyuC60AAAAJ:584KSnlxRPIC](https://scholar.google.com/citations?view_op=view_citation&hl=ja&user=zlyuC60AAAAJ&citation_for_view=zlyuC60AAAAJ:584KSnlxRPIC).
- [130] Marcin Wichrowski. “Google Glass Development in Practice”. In: (2015). URL: [https://www.researchgate.net/publication/286926421\\_Google\\_Glass\\_Development\\_in\\_Practice\\_UX\\_Design\\_Sprint\\_Workshops](https://www.researchgate.net/publication/286926421_Google_Glass_Development_in_Practice_UX_Design_Sprint_Workshops).
- [131] Serhiy O. Semerikov. “Proceedings of the 7th International Workshop on Augmented Reality in Education (AREdu 2024)”. In: (2024). URL: <https://ceur-ws.org/Vol-3918/>.
- [132] Deokgi Jeung. “Proceedings of the 7th International Workshop on Augmented Reality in Education (AREdu 2024)”. In: (2023). URL: <https://www.sciencedirect.com/science/article/abs/pii/S0169260722007040>.
- [133] Selcen Ozturkcan. “Service innovation: using augmented reality in the IKEA Place app”. In: (2020). URL: <https://journals.sagepub.com/doi/10.1177/2043886920947110>.

- [134] N. Saforrudin. “Tehnical skills in developing Augmented Reality”. In: (2011). URL: [https://link.springer.com/chapter/10.1007/978-3-642-25200-6\\_34](https://link.springer.com/chapter/10.1007/978-3-642-25200-6_34).
- [135] Michele Domenico Todino. *Tehnical skills in developing Augmented Reality*. 2017. URL: <https://www.igi-global.com/journal/international-journal-virtual-augmented-reality/145080>.
- [136] Mark Bolas. “Augmented reality using personal projection and retroreflection”. In: (2011).
- [137] Steven Feiner. “Mobile augmented reality”. In: (2004).
- [138] Mark Billinghurst. “A survey of augmented reality”. In: (2015). URL: [https://scholar.google.com/citations?view\\_op=view\\_citation&hl=en&user=S-J\\_ItYAAAJ&citation\\_for\\_view=S-J\\_ItYAAAJ:65Yg0jNCQDAC](https://scholar.google.com/citations?view_op=view_citation&hl=en&user=S-J_ItYAAAJ&citation_for_view=S-J_ItYAAAJ:65Yg0jNCQDAC).
- [139] VR/AR. URL: <https://www.avforums.com/forums/vr-ar-virtual-reality-augmented-reality.270/>.
- [140] Ar global. URL: <https://www.awexr.com/>.
- [141] A. Unwin C. Chen W. Hrdle. “A brief history of Data Visualization”. In: (2008). URL: [https://link.springer.com/chapter/10.1007/978-3-540-33037-0\\_2](https://link.springer.com/chapter/10.1007/978-3-540-33037-0_2).
- [142] A. Shadare M. Sadiku. “Data visualization”. In: (2016).
- [143] K. Lawonn B. Preim. “Visual Analytics”. In: (2016). URL: <https://www.sciencedirect.com/topics/computer-science/visual-analytics>.
- [144] In: (). URL: <https://www.interaction-design.org/literature/topics/information-visualization?srsltid=AfmB0opwE2IIWcr01g4f-nNPsn1mJQF4-FEBJCwbqlAkHvMQCRiyh9Tc&utm>.
- [145] JD Fekete D. Keim. “Visual Analytics: Definition, Process and Challenges”. In: (2008). URL: [https://www.researchgate.net/publication/29637192\\_Visual\\_Analytics\\_Definition\\_Process\\_and\\_Challenges](https://www.researchgate.net/publication/29637192_Visual_Analytics_Definition_Process_and_Challenges).
- [146] Lavanya Addepalli. “A comprehensive review of data visualization tools”. In: (2023).
- [147] Xiaoru Yuan Min Tian. *LitVis*. 2023. URL: <https://link.springer.com/article/10.1007/s12650-023-00941-3>.
- [148] Jeffrey Heer. “A tour through the visualization zoo”. In: (2010). URL: <https://dl.acm.org/doi/fullHtml/10.1145/1743546.1743567>.
- [149] Tamara Munzner. *Visualization Analysis and Design*. 2014.
- [150] Hanspeter Pfister. “Visualization of Intersecting Sets”. In: (2014). URL: <https://ieeexplore.ieee.org/abstract/document/6876017>.

- [151] URL: <https://www.visualizing.org/>.
- [152] URL: <https://admitere.uvt.ro/program/big-data-data-science-analytics-and-technologies/>.
- [153] Interaction Design Foundation. *Human–Computer Interaction*. n.d.; Accessed: April 2025. URL: <https://www.interaction-design.org/literature/topics/human-computer-interaction>.
- [154] Alan Cooper et al. *About Face: The Essentials of Interaction Design*. 4th. Wiley, 2014.
- [155] Susan Weinschenk. *100 Things Every Designer Needs to Know About People*. New Riders, 2011.
- [156] Jakob Nielsen. *10 Usability Heuristics for User Interface Design*. Retrieved 1995. 1995. URL: <https://www.nngroup.com/articles/ten-usability-heuristics/>.
- [157] Ben Shneiderman. “Human-Centered Artificial Intelligence: Reliable, Safe & Trustworthy”. In: *International Journal of Human-Computer Interaction* (2022).
- [158] Saleema Amershi, Daniel Weld, Mihaela Vorvoreanu, et al. “Guidelines for Human-AI Interaction”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2019.
- [159] Jonathan Grudin. “Groupware and Social Dynamics: Eight Challenges for Developers”. In: *Communications of the ACM* 37.1 (1994), pp. 92–105.
- [160] Jonathan Grudin and Sarah Poltrack. *Computer Supported Cooperative Work*. 2013.
- [161] Ben Shneiderman. *Designing the User Interface*. Lucrare seminală în domeniul HCI, autorul regulilor de design. Addison-Wesley, 1987.
- [162] Stuart K. Card, Thomas P. Moran, and Allen Newell. *The Psychology of Human-Computer Interaction*. Include discuții despre Model Human Processor și GOMS. L. Erlbaum Associates, 1983.
- [163] Paul M. Fitts. “The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement”. In: *Journal of Experimental Psychology* 47.6 (1954), pp. 381–391.
- [164] Helen Nissenbaum. *Privacy in Context*. Stanford University Press, 2009.
- [165] Ben Shneiderman. *The New ABCs of Research*. Oxford University Press, 2016.
- [166] Lorrie F. Cranor and Simson (eds.) Garfinkel. *Security and Usability*. O'Reilly Media, 2005.

- [167] Richard L. Burden and J. Douglas Faires. *Numerical Analysis*. Cengage Learning, 2010.
- [168] Josef Stoer and Roland Bulirsch. *Introduction to Numerical Analysis*. Springer, 2002.
- [169] J. Dongarra et al. *Sourcebook of Parallel Computing*. Morgan Kaufmann, 2020.
- [170] TOP500.org. *Lista celor mai puternice supercomputere din lume*. 2023. URL: <https://www.top500.org/>.
- [171] R. H. Landau, M. J. Páez, and C. C. Bordeianu. *Computational Physics*. Wiley, 2015.
- [172] J. M. Thijssen. *Computational Physics*. Cambridge University Press, 2013.
- [173] Richard Durbin et al. *Biological Sequence Analysis*. Cambridge University Press, 1998.
- [174] Arthur M. Lesk. *Introduction to Bioinformatics*. Oxford University Press, 2019.
- [175] Christopher J. Cramer. *Essentials of Computational Chemistry*. Wiley, 2004.
- [176] Frank Jensen. *Introduction to Computational Chemistry*. Wiley, 2017.
- [177] Leigh Tesfatsion and Kenneth L. Judd. *Handbook of Computational Economics*. Elsevier, 2006.
- [178] David A. Kendrick et al. *Computational Economics*. Princeton University Press, 2006.
- [179] O. C. Zienkiewicz. *The Finite Element Method*. Butterworth-Heinemann, 2005.
- [180] John D. Anderson. *Computational Fluid Dynamics*. McGraw-Hill, 1995.
- [181] Kenneth C. Laudon and Jane P. Laudon. *Management Information Systems: Managing the Digital Firm*. 16th. Pearson, 2020.
- [182] Thomas H. Davenport. “Putting the enterprise into the enterprise system”. In: *Harvard Business Review* 76.4 (1998), pp. 121–131.
- [183] Efraim Turban et al. *Decision Support and Business Intelligence Systems*. 9th. Pearson Education, 2010.
- [184] Luvai Motiwalla and Jeffrey Thompson. *Enterprise Systems for Management*. 2nd. Pearson Education, 2012.
- [185] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems*. Addison-Wesley, 2015.

- [186] C. J. Date. *An Introduction to Database Systems*. Addison Wesley, 2003.
- [187] Eric Redmond and Jim Wilson. *Seven Databases in Seven Weeks*. O'Reilly Media, 2012.
- [188] Adrian Cockcroft. *Netflix: Embracing the Cloud*. 2014. URL: <https://www.slideshare.net/adriancockcroft/netflix-embracing-the-cloud>.
- [189] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice*. Addison-Wesley Professional, 2012.
- [190] Jeanne W. Ross, Peter Weill, and David C. Robertson. *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution*. Harvard Business Press, 2006.
- [191] Sam Newman. *Building Microservices: Designing Fine-Grained Systems*. 2nd. O'Reilly Media, 2021.
- [192] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice*. 3rd. Addison-Wesley, 2012.
- [193] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach*. 8th. Pearson, 2021.
- [194] William Stallings. *Network Security Essentials: Applications and Standards*. 6th. Pearson, 2017.
- [195] Jon Erickson. *Hacking: The Art of Exploitation*. 2nd. No Starch Press, 2008.
- [196] Kevin D. Mitnick and William L. Simon. *The Art of Deception: Controlling the Human Element of Security*. Wiley, 2002.
- [197] Paul Harmon. *Business Process Change: A Business Process Management Guide for Managers and Process Professionals*. 4th. Morgan Kaufmann, 2019.
- [198] Marlon Dumas et al. *Fundamentals of Business Process Management*. 2nd. Springer, 2018.
- [199] Wil M. P. van der Aalst. *Process Mining: Data Science in Action*. 2nd. Springer, 2016.
- [200] Jan Freund and Bernd Rücker. *Real-Life BPMN: Using BPMN 2.0 to Analyze, Improve, and Automate Processes in Your Company*. 4th. Camunda Services GmbH, 2020.
- [201] Daniel Dines. *The UiPath Way: Leading the Automation First Era*. UiPath Publications, 2023.
- [202] Steve Krug. *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. 3rd. New Riders, 2014.

- [203] ISACA. *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*. ISACA, 2014.
- [204] Peter Weill and Jeanne W. Ross. *IT Governance: How Top Performers Manage IT Decision Rights for Superior Results*. Harvard Business Press, 2004.
- [205] Rick Hunter and George Westerman. *Digital to the Core: Remastering Leadership for Your Industry, Your Enterprise, and Yourself*. Bibliomotion, 2016.
- [206] Carl Symons. *IT Governance Framework: Structures, Processes and Responsibilities*. Forrester Research, 2005.
- [207] S. Y. Emmert-Streib and M. Dehmer. *A Brief History of Bioinformatics*. [Online; accessed May 2025]. 2018. URL: [https://www.researchgate.net/publication/326839021\\_A\\_Brief\\_History\\_of\\_Bioinformatics](https://www.researchgate.net/publication/326839021_A_Brief_History_of_Bioinformatics).
- [208] Nature Education. *Basic Local Alignment Search Tool (BLAST)*. [Online; accessed May 2025]. URL: <https://www.nature.com/scitable/topicpage/basic-local-alignment-search-tool-blast-29096/>.
- [209] Wikipedia contributors. *Frederick Sanger*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Frederick\\_Sanger](https://en.wikipedia.org/wiki/Frederick_Sanger).
- [210] Wikipedia contributors. *Margaret Oakley Dayhoff*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Margaret\\_Oakley\\_Dayhoff](https://en.wikipedia.org/wiki/Margaret_Oakley_Dayhoff).
- [211] Wikipedia contributors. *Stephen Altschul*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Stephen\\_Altschul](https://en.wikipedia.org/wiki/Stephen_Altschul).
- [212] World Health Organization. *Genomics*. [Online; accessed May 2025]. URL: [https://www.who.int/health-topics/genomics#tab=tab\\_1](https://www.who.int/health-topics/genomics#tab=tab_1).
- [213] Wikipedia contributors. *Francis Crick*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Francis\\_Crick](https://en.wikipedia.org/wiki/Francis_Crick).
- [214] Wikipedia contributors. *Rosalind Franklin*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Rosalind\\_Franklin](https://en.wikipedia.org/wiki/Rosalind_Franklin).
- [215] Wikipedia contributors. *James Watson*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/James\\_Watson](https://en.wikipedia.org/wiki/James_Watson).
- [216] National Human Genome Research Institute. *Francis S. Collins, M.D., Ph.D.* [Online; accessed May 2025]. URL: <https://www.genome.gov/staff/Francis-S-Collins-MD-PhD>.
- [217] Wikipedia contributors. *Francis Collins*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Francis\\_Collins](https://en.wikipedia.org/wiki/Francis_Collins).

- [218] Wikipedia contributors. *Craig Venter*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Craig\\_Venter](https://en.wikipedia.org/wiki/Craig_Venter).
- [219] Wikipedia contributors. *Proteomics*. [Online; accessed May 2025]. URL: <https://en.wikipedia.org/wiki/Proteomics>.
- [220] Wikipedia contributors. *Marc Wilkins (geneticist)*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Marc\\_Wilkins\\_\(geneticist\)](https://en.wikipedia.org/wiki/Marc_Wilkins_(geneticist)).
- [221] Wikipedia contributors. *John R. Yates*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/John\\_R.\\_Yates](https://en.wikipedia.org/wiki/John_R._Yates).
- [222] J. R. Yates III. “Recent Technical Advances in Proteomics”. In: *Molecular & Cellular Proteomics* 21 (2022). [Online; accessed May 2025], p. 100217. URL: [https://www.mcponline.org/article/S1535-9476\(20\)32756-0/fulltext](https://www.mcponline.org/article/S1535-9476(20)32756-0/fulltext).
- [223] AzoLifeSciences. *Structural Bioinformatics: An Overview*. [Online; accessed May 2025]. URL: <https://www.azolifesciences.com/article/Structural-Bioinformatics-An-Overview.aspx>.
- [224] Wikipedia contributors. *David Baker (biochemist)*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/David\\_Baker\\_\(biochemist\)](https://en.wikipedia.org/wiki/David_Baker_(biochemist)).
- [225] EMBL. *Janet Thornton retires: a pioneer in structural bioinformatics*. [Online; accessed May 2025]. URL: <https://www.embl.org/news/lab-matters/janet-thornton-retires-a-pioneer-in-structural-bioinformatics/>.
- [226] Wikipedia contributors. *Christine Orengo*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Christine\\_Orengo](https://en.wikipedia.org/wiki/Christine_Orengo).
- [227] Wikipedia contributors. *Janet Thornton*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Janet\\_Thornton](https://en.wikipedia.org/wiki/Janet_Thornton).
- [228] RCSB Protein Data Bank. *About Us: Helen Berman*. [Online; accessed May 2025]. URL: <https://www.rcsb.org/pages/about-us/helen-berman>.
- [229] Wikipedia contributors. *Helen M. Berman*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Helen\\_M.\\_Berman](https://en.wikipedia.org/wiki/Helen_M._Berman).
- [230] Wikilectures. *Medical Informatics*. [Online; accessed May 2025]. URL: [https://www.wikilectures.eu/w/Medical\\_Informatics](https://www.wikilectures.eu/w/Medical_Informatics).
- [231] Wikipedia contributors. *Victor A. McKusick*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Victor\\_A.\\_McKusick](https://en.wikipedia.org/wiki/Victor_A._McKusick).
- [232] Wikipedia contributors. *Corwin Hansch*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Corwin\\_Hansch](https://en.wikipedia.org/wiki/Corwin_Hansch).
- [233] Wikipedia contributors. *Russ Altman*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Russ\\_Altman](https://en.wikipedia.org/wiki/Russ_Altman).

- [234] University of California San Francisco. *Brian Shoichet*. [Online; accessed May 2025]. URL: <https://pharmacy.ucsf.edu/brian-shoichet>.
- [235] A. Kumar and N. Chordia. “Role of Bioinformatics in Biotechnology”. In: *Research and Reviews: Biosciences* 12.1 (2017), p. 116.
- [236] H. Kitano. “Computational systems biology”. In: *Nature* 420.6912 (2002), pp. 206–210.
- [237] T. Ideker and N. J. Krogan. “Differential network biology”. In: *Molecular Systems Biology* 8.1 (2012), p. 565.
- [238] B. Ø. Palsson. *Systems Biology: Constraint-Based Reconstruction and Analysis*. Cambridge University Press, 2015.
- [239] Peter Van Roy and Seif Haridi. *Concepts, Techniques and Models of Computer Programming*. 2003.
- [240] *The Art of Multiprocessor Programming*.
- [241] *Parallel Programming in C with MPI and OpenMP*.
- [242] *Taxonomy of The Fundamental Concepts of Meta-Programming*. Accessed: April 2025. URL: [https://www.researchgate.net/publication/267704068\\_Taxonomy\\_of\\_The\\_Fundamental\\_Concepts\\_of\\_Meta-Programming](https://www.researchgate.net/publication/267704068_Taxonomy_of_The_Fundamental_Concepts_of_Meta-Programming).
- [243] *ITC Article*. Accessed: April 2025. URL: <https://www.itc.ktu.lt/index.php/ITC/article/view/11931>.
- [244] *Metaprogramming Lecture Notes*. Accessed: April 2025. URL: <https://namin.seas.harvard.edu/files/namin/files/metaprogramming-lecture-notes.pdf>.
- [245] Alonzo Church. “An unsolvable problem of elementary number theory”. In: *American Journal of Mathematics* 58.2 (1936), pp. 345–363.
- [246] Kurt Mehlhorn and Stefan Näher. *LEDA: A platform for combinatorial and geometric computing*. Cambridge University Press, 1999.
- [247] Massachusetts Institute of Technology. *MIT OpenCourseWare – Introduction to Algorithms*. 2025. URL: <https://ocw.mit.edu>.
- [248] Stanford University. *CS Theory at Stanford*. 2025. URL: <https://theory.stanford.edu>.
- [249] Jon L. Bentley. *Programming Pearls*. Addison-Wesley, 1986.
- [250] Andrew S. Tanenbaum and Maarten van Steen. *Distributed Systems: Principles and Paradigms*. Prentice Hall, 2016.
- [251] George Coulouris et al. *Distributed Systems: Concepts and Design*. Addison-Wesley, 2011.

- [252] Andrew S. Tanenbaum. *Distributed Systems*. Pearson, 2007.
- [253] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach*. Pearson, 2017.
- [254] Andrew S. Tanenbaum. *Computer Networks*. Pearson, 2011.
- [255] Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill, 2010.
- [256] Jez Humble and David Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley, 2010.
- [257] Thomas Erl et al. *Cloud Computing: Concepts, Technology, & Architecture*. Prentice Hall, 2013.
- [258] Gene Kim, Kevin Behr, and George Spafford. *The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win*. IT Revolution Press, 2018.
- [259] Ross Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, 2020.
- [260] William Stallings. *Cryptography and Network Security: Principles and Practice*. Pearson, 2017.
- [261] *NIST Special Publication 800 Series*. Available online. Various publications on computer security. 2000.
- [262] W. Inmon. *Building the Data Warehouse*. Wiley, 1990.
- [263] AXELOS. *ITIL Foundation: ITIL 4 Edition*. TSO (The Stationery Office), 2019.
- [264] European Union Agency for Cybersecurity (ENISA). *ENISA Threat Landscape Report*. 2023. URL: <https://www.enisa.europa.eu/topics/cyber-threats>.
- [265] Gartner. *Magic Quadrant for Robotic Process Automation*. 2023. URL: <https://www.gartner.com/en/documents/4005676>.
- [266] Google. *Material Design Guidelines*. 2025. URL: <https://m3.material.io/>.
- [267] National Institute of Standards and Technology. *Framework for Improving Critical Infrastructure Cybersecurity*. 2018. URL: <https://www.nist.gov/cyberframework>.
- [268] Steve Weyhrich. *Human-Computer Interaction (HCI)*. 2013.
- [269] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.

- [270] Gary M. Olson and Judith S. Olson. “Distance Matters”. In: *Human-Computer Interaction* 15.2 (2000), pp. 139–178.
- [271] Gloria Mark. “Collaboration and Privacy in Multimedia Messaging”. In: *Proceedings of CSCW*. ACM, 2002.
- [272] Alan Dix et al. *Human-Computer Interaction*. 3rd. Pearson, 2004.
- [273] ISO 9241-210:2010 - *Ergonomics of Human-System Interaction*. 2010.
- [274] M. Prasad. “HCI in Mobile and Wearable Computing”. In: *International Journal of Engineering Research & Technology (IJERT)* (2022).
- [275] Harry Hochheiser, R. S. Valdez, J. J. Cimino, et al. “HCI, Ethics, and Biomedical Informatics”. In: *Journal of the American Medical Informatics Association* (2020).
- [276] Harvard Business Review. *Articles on IT Strategy and Digital Transformation*. Retrieved April 2025. 2025. URL: <https://hbr.org>.
- [277] Interaction Design Foundation. *Human-Computer Interaction*. Retrieved April 2025. 2025. URL: <https://www.interaction-design.org>.
- [278] ISO/IEC. *ISO/IEC 38500:2015 – Governance of IT for the Organization*. 2015.
- [279] Brian Krebs. *The Target Breach, By the Numbers*. Retrieved 2014. 2014. URL: <https://krebsonsecurity.com/2014/05/the-target-breath-by-the-numbers/>.
- [280] The Open Group. *ISO/IEC/IEEE 42010: Systems and Software Engineering — Architecture Description*. 2021. URL: <https://www.iso.org/standard/50508.html>.
- [281] UiPath. *UiPath Customer Stories: Siemens Finance Automation*. 2021. URL: <https://www.uipath.com/resources/automation-case-studies/siemens-finance-automation>.
- [282] User Experience Professionals Association (UXPA). *User Experience Professionals Association*. 2025. URL: <https://uxpa.org>.
- [283] Journal of Information Systems Management. *Special Issue on IT Governance and Performance*. 2025. URL: <https://www.tandfonline.com/loi/uism20>.
- [284] Peter Denning. “Komputer Science: the Discipline”. In: *Encyclopedia of Komputer Science (A. Ralston and D. Hemmendinger, Eds)* (1999).
- [285] Larry L Peterson and Bruce S Davie. *Computer networks: a systems approach*. Elsevier, 2007.

- [286] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [287] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.
- [288] Rodney Brooks. “Elephants Don’t Play Chess”. In: *Robotics and Autonomous Systems* (1991).
- [289] Q. Li et al. “Differential Privacy for SQL Queries”. In: *VLDB*. 2019.
- [290] Wikipedia contributors. *Bioinformatică*. [n.d.; Online; accessed May 2025]. URL: <https://ro.wikipedia.org/wiki/Bioinformatic%C4%83>.
- [291] National Human Genome Research Institute. *Bioinformatics - Genetics Glossary*. [n.d.; Online; accessed May 2025]. URL: <https://www.genome.gov/genetics-glossary/Bioinformatics>.
- [292] Wikipedia contributors. *Sequence analysis*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Sequence\\_analysis](https://en.wikipedia.org/wiki/Sequence_analysis).
- [293] Institutul Național de Cercetare-Dezvoltare în Informatică. *Căutarea în bazele de date de nucleotide*. [Online; accessed May 2025]. URL: [https://www.cnic.ro/bioinfo/cautarea\\_in\\_bazele\\_de\\_date\\_de\\_nu.htm](https://www.cnic.ro/bioinfo/cautarea_in_bazele_de_date_de_nu.htm).
- [294] J. Lin et al. “Deep learning enables accurate clustering with batch effect removal in single-cell RNA-seq analysis”. In: *Nature Communications* 13 (2022). [Online; accessed May 2025], p. 1244. URL: <https://www.nature.com/articles/s41467-022-29268-7>.
- [295] M. Asgari et al. “Deep learning models for biological sequence analysis”. In: *Frontiers in Bioinformatics* 1 (2021). [Online; accessed May 2025], p. 8190442. URL: <https://pmc.ncbi.nlm.nih.gov/articles/PMC8190442/>.
- [296] Wikipedia contributors. *BLAST (biotechnology)*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/BLAST\\_\(biotechnology\)](https://en.wikipedia.org/wiki/BLAST_(biotechnology)).
- [297] Wikipedia contributors. *Genomics*. [Online; accessed May 2025]. URL: <https://en.wikipedia.org/wiki/Genomics>.
- [298] Wellcome Genome Campus. *Timeline: History of Genomics*. [Online; accessed May 2025]. URL: <https://www.yourgenome.org/theme/timeline-history-of-genomics/>.
- [299] Wikipedia contributors. *Bioinformatics*. [Online; accessed May 2025]. URL: <https://en.wikipedia.org/wiki/Bioinformatics>.
- [300] ELGA LabWater. *4 Challenges Facing Genomics and Genetics Research*. [Online; accessed May 2025]. URL: <https://www.elgalabwater.com/blog/4-challenges-facing-genomics-and-genetics-research>.

- [301] Y. Smith. *History of Genomics*. [Online; accessed May 2025]. URL: <https://www.news-medical.net/life-sciences/History-of-Genomics.aspx>.
- [302] M. Covic. “Ce este medicina genomică? (1)”. In: *Viată Medicală* (2012). [Online; accessed May 2025]. URL: <https://www.viata-medicala.ro/ce-este-medicina-genomica-1-4886>.
- [303] Wikipedia contributors. *Celera Corporation*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Celera\\_Corporation](https://en.wikipedia.org/wiki/Celera_Corporation).
- [304] R. Aebersold and M. Mann. “Mass spectrometry-based proteomics”. In: *Nature* 422 (2003). [Online; accessed May 2025], pp. 198–207. URL: <https://pubmed.ncbi.nlm.nih.gov/12634793/>.
- [305] Utrecht University. *Proteomics pioneers Ruedi Aebersold and Matthias Mann receive the Dr. H.P. Heineken Prize for Biochemistry and Biophysics*. [Online; accessed May 2025]. 2022. URL: <https://www.uu.nl/en/news/proteomics-pioneers-ruedi-aebersold-and-matthias-mann-receive-the-dr-hp-heineken-prize-for>.
- [306] M. Mann et al. “The coming age of complete, accurate, and ubiquitous proteomes”. In: *Molecular Cell* 61.5 (2016). [Online; accessed May 2025], pp. 637–650. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5204337/>.
- [307] F. Zhong. “Development and challenges of proteomics in China”. In: *Genomics, Proteomics & Bioinformatics* 5.3–4 (2007). [Online; accessed May 2025], pp. 185–190. URL: <https://www.sciencedirect.com/science/article/pii/S1672022907600187>.
- [308] Technology Networks. *5 Key Challenges in Proteomics as Told by the Experts*. [Online; accessed May 2025]. URL: <https://www.technologynetworks.com/proteomics/lists/5-key-challenges-in-proteomics-as-told-by-the-experts-321774>.
- [309] Wikipedia contributors. *Structural bioinformatics*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Structural\\_bioinformatics](https://en.wikipedia.org/wiki/Structural_bioinformatics).
- [310] ELIXIR Europe. *3D-Bioinfo Community*. [Online; accessed May 2025]. URL: <https://elixir-europe.org/communities/3d-bioinfo>.
- [311] S. M. Velankar et al. “3D-BioInfo: a community building infrastructure for structural bioinformatics”. In: *F1000Research* 13 (2024). [Online; accessed May 2025], p. 1002. URL: <https://f1000research.com/articles/13-1002>.
- [312] Wikipedia contributors. *CATH database*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/CATH\\_database](https://en.wikipedia.org/wiki/CATH_database).

- [313] T. E. Ferrin et al. “The role of structural bioinformatics in drug discovery”. In: *Current Opinion in Structural Biology* 23 (2013). [Online; accessed May 2025], pp. 707–714. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3856823/>.
- [314] National Science Foundation. *Computational Science Takes Nobel Stage*. [Online; accessed May 2025]. URL: <https://www.nsf.gov/news/computational-science-takes-nobel-stage>.
- [315] Oregon Health & Science University. *What is Biomedical Informatics?* [Online; accessed May 2025]. URL: <https://www.ohsu.edu/school-of-medicine/medical-informatics-and-clinical-epidemiology/what-biomedical-informatics>.
- [316] E. H. Shortliffe and J. J. Cimino. “Biomedical informatics: The science and the pragmatics”. In: *Journal of the American Medical Informatics Association* 17.2 (2010). [Online; accessed May 2025], pp. 169–170. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2814957/>.
- [317] J. Wang. “Biomedical informatics and its role in translational medicine”. In: *Science China Life Sciences* 56 (2013). [Online; accessed May 2025], pp. 1–3. URL: <https://link.springer.com/article/10.1007/s11427-013-4439-7>.
- [318] P. D. Karp et al. “Pathway Tools version 13.0: integrated software for pathway/genome informatics and systems biology”. In: *Bioinformatics* 27.13 (2011). [Online; accessed May 2025], pp. 1741–1742. URL: <https://academic.oup.com/bioinformatics/article/27/13/1741/186256>.
- [319] D. Gurwitz et al. “Personal genomics: Where are we now and where are we heading?” In: *Investigative Genetics* 4 (2013). [Online; accessed May 2025], p. 15. URL: <https://investigativegenetics.biomedcentral.com/articles/10.1186/2041-2223-4-15>.
- [320] H. J. van der Lei. “Closing the loop between clinical practice, research, and education: the informatics infrastructure”. In: *Yearbook of Medical Informatics* 21 (2012). [Online; accessed May 2025], pp. 20–26. URL: <https://pubmed.ncbi.nlm.nih.gov/22450380/>.
- [321] Wikipedia contributors. *Epidemiologie*. [Online; accessed May 2025]. URL: <https://ro.wikipedia.org/wiki/Epidemiologie>.
- [322] Wikipedia contributors. *Systems biology*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Systems\\_biology](https://en.wikipedia.org/wiki/Systems_biology).
- [323] Wikipedia contributors. *Epidemiology*. [Online; accessed May 2025]. URL: <https://en.wikipedia.org/wiki/Epidemiology>.

- [324] Wikipedia contributors. *Translational bioinformatics*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Translational\\_bioinformatics](https://en.wikipedia.org/wiki/Translational_bioinformatics).
- [325] A. N. Doulamis. “Translational bioinformatics: Past achievements and future challenges”. In: *BioMedInformatics* 4.1 (2024). [Online; accessed May 2025], p. 6. URL: <https://www.mdpi.com/2673-7426/4/1/6>.
- [326] OMIM. *About OMIM*. [Online; accessed May 2025]. URL: <https://omim.org/about>.
- [327] S. A. Basharat et al. “The Role of Bioinformatics in Drug Discovery and Development”. In: *Drug Discovery - Challenges and Perspectives*. [Online; accessed May 2025]. IntechOpen, 2023. URL: <https://www.intechopen.com/chapters/88596>.
- [328] Wikipedia contributors. *Pharmaceutical bioinformatics*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Pharmaceutical\\_bioinformatics](https://en.wikipedia.org/wiki/Pharmaceutical_bioinformatics).
- [329] H. K. Sharma and S. Kumar. *The Role of Bioinformatics in Drug Discovery: A Comprehensive Overview*. [Online; accessed May 2025]. 2023. URL: [https://www.researchgate.net/publication/376053303\\_The\\_Role\\_of\\_Bioinformatics\\_in\\_Drug\\_Discovery\\_A\\_Comprehensive\\_Overview](https://www.researchgate.net/publication/376053303_The_Role_of_Bioinformatics_in_Drug_Discovery_A_Comprehensive_Overview).
- [330] D. Sliwoski et al. “Computational methods in drug discovery”. In: *Pharmacological Reviews* 66.1 (2014). [Online; accessed May 2025], pp. 334–395. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5421137/>.
- [331] CAS Insights. *Dealing With the Challenges of Drug Discovery*. [Online; accessed May 2025]. URL: <https://www.cas.org/resources/cas-insights/dealing-challenges-drug-discovery>.
- [332] S. Wang, Y. Sun, and J. Song. “Drug repositioning by integrating target information through a heterogeneous network model”. In: *Scientific Reports* 6 (2016). [Online; accessed May 2025]. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4865415/>.
- [333] A. Bender and J. L. Jenkins. “In silico target prediction: an overview”. In: *Drug Discovery Today: Technologies* 8.4 (2011). [Online; accessed May 2025], e277–e283. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3098752/>.
- [334] J. Davis-Turak et al. “Genomics pipelines and data integration: challenges and opportunities in the research setting”. In: *Expert Review of Molecular Diagnostics* 17.3 (2017), pp. 225–237.

- [335] P. Carbonell et al. “Bioinformatics for the synthetic biology of natural products: integrating across the Design–Build–Test cycle”. In: *Natural Product Reports* 33.8 (2016), pp. 917–937.
- [336] S. M. Vidanagamachchi and K. M. G. T. R. Waidyaratna. “Opportunities, challenges and future perspectives of using bioinformatics and artificial intelligence techniques on tropical disease identification using omics data”. In: *Frontiers in Digital Health* 6 (2024), Art. nr. 1471200.
- [337] EMBL-EBI. *People: Ewan Birney*. [Online; accessed May 2025]. 2025. URL: <https://www.ebi.ac.uk/people/person/ewan-birney>.
- [338] University of Kragujevac. *IEEE BIBE 2024: 24th IEEE International Conference on Bioinformatics and Bioengineering*. [Online; accessed May 2025]. 2024. URL: <https://www.bibe2024.kg.ac.rs/>.
- [339] RECOMB. *Conference on Research in Computational Molecular Biology*. [Online; accessed May 2025]. URL: <https://recomb.org/>.
- [340] International Society for Computational Biology. *About ISMB*. [Online; accessed May 2025]. URL: <https://www.iscb.org/about-ismb>.
- [341] Association for Computing Machinery. *ACM-BCB: ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*. [Online; accessed May 2025]. URL: <https://acm-bcb.org/>.
- [342] International Society for Computational Biology. *ISMB/ECCB 2025*. [Online; accessed May 2025]. 2025. URL: <https://www.iscb.org/ismbeccb2025/home>.
- [343] Wikipedia contributors. *Pacific Symposium on Biocomputing*. [Online; accessed May 2025]. URL: [https://en.wikipedia.org/wiki/Pacific\\_Symposium\\_on\\_Biocomputing](https://en.wikipedia.org/wiki/Pacific_Symposium_on_Biocomputing).
- [344] University of Maryland. *WABI Conference – Workshop on Algorithms in Bioinformatics*. [Online; accessed May 2025]. URL: <https://www.umiacs.umd.edu/events/wabi-conference>.
- [345] Masterstudies.ro. *Masterat în Bioinformatică*. [Online; accessed May 2025]. URL: <https://www.masterstudies.ro/masterat/bioinformatic%C4%83>.
- [346] Universitatea de Vest din Timișoara. *Bioinformatica – Masterat*. [Online; accessed May 2025]. URL: <https://info.uvt.ro/programe-master/bioinformatica/>.
- [347] L. Hood and M. Flores. “A personal view on systems medicine and the emergence of proactive P4 medicine: predictive, preventive, personalized and participatory”. In: *New Biotechnology* 29.6 (2012), pp. 613–624.

- [348] U. Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman & Hall/CRC, 2006.
- [349] A. Grama et al. *Introducere în Calculul Paralel*. Pearson, 2003.
- [350] Wikipedia contributors. *2D*. Accesat: 17 mai 2025. 2022. URL: <https://ro.wikipedia.org/wiki/2D>.
- [351] Wikipedia contributors. *2D computer graphics*. Accesat: 17 mai 2025. 2025. URL: [https://en.wikipedia.org/wiki/2D\\_computer\\_graphics#References](https://en.wikipedia.org/wiki/2D_computer_graphics#References).
- [352] Wikipedia contributors. *Ivan Sutherland*. Accesat: 17 mai 2025. 2025. URL: [https://en.wikipedia.org/wiki/Ivan\\_Sutherland](https://en.wikipedia.org/wiki/Ivan_Sutherland).
- [353] ACM SIGGRAPH. *Home - ACM SIGGRAPH*. Accesat: 17 mai 2025. 2025. URL: <https://www.siggraph.org/>.
- [354] Wikipedia contributors. *SIGGRAPH*. Accesat: 17 mai 2025. 2025. URL: <https://ro.wikipedia.org/wiki/SIGGRAPH>.
- [355] Adobe Inc. *Adobe MAX*. Accesat: 17 mai 2025. 2025. URL: <https://www.adobe.com/max.html>.
- [356] Wikipedia contributors. *Adobe MAX*. Accesat: 17 mai 2025. 2025. URL: [https://en.wikipedia.org/wiki/Adobe\\_MAX](https://en.wikipedia.org/wiki/Adobe_MAX).
- [357] OFFF Barcelona. *OFFF Barcelona - About*. Accesat: 17 mai 2025. 2025. URL: <https://www.offf.barcelona/about>.
- [358] Barbara Guttman and Edward Roback. *An Introduction to Computer Security: The NIST Handbook*. Tech. rep. NISTIR 6521. Accesat: 17 mai 2025. National Institute of Standards and Technology, 1995. URL: <https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir6521.pdf>.
- [359] TechTarget. *CGI (computer-generated imagery)*. Accessed: 17 May 2025. 2025. URL: <https://www.techtarget.com/whatis/definition/CGI-computer-generated-imagery>.
- [360] Wikipedia contributors. *Computer-generated imagery*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Computer-generated\\_imagery](https://en.wikipedia.org/wiki/Computer-generated_imagery).
- [361] Wikipedia contributors. *Grafică digitală*. Accessed: 17 May 2025. 2025. URL: [https://ro.wikipedia.org/wiki/Grafic%C4%83\\_digital%C4%83](https://ro.wikipedia.org/wiki/Grafic%C4%83_digital%C4%83).
- [362] Wikipedia contributors. *Rendering equation*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Rendering\\_equation](https://en.wikipedia.org/wiki/Rendering_equation).

- [363] Wikipedia contributors. *Bidirectional reflectance distribution function*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Bidirectional\\_reflectance\\_distribution\\_function](https://en.wikipedia.org/wiki/Bidirectional_reflectance_distribution_function).
- [364] Chaos Group. *What is PBR (Physically Based Rendering)? A Complete Guide*. Accessed: 17 May 2025. 2025. URL: <https://www.chaos.com/blog/what-is-pbr-physically-based-rendering-a-complete-guide>.
- [365] Scratchapixel 2.0. *Introduction to Global Illumination and Path Tracing*. Accessed: 17 May 2025. 2025. URL: <https://www.scratchapixel.com/lessons/3d-basic-rendering/global-illumination-path-tracing/introduction-global-illumination-path-tracing.html>.
- [366] Informatik Universität Freiburg. *GridFluids: Fluid Simulation with Euler and Particle Methods*. Accessed: 17 May 2025. 2025. URL: [https://cg.informatik.uni-freiburg.de/intern/seminar/gridFluids\\_fluid-EulerParticle.pdf](https://cg.informatik.uni-freiburg.de/intern/seminar/gridFluids_fluid-EulerParticle.pdf).
- [367] Max Planck Institute for Informatics. *Photorealistic avatars for VR: A new method advances accessibility*. Accessed: 17 May 2025. 2024. URL: <https://www.mpi-inf.mpg.de/news/detail/photorealistic-avatars-for-vr-a-new-method-advances-accessibility>.
- [368] Various authors. *arXiv paper 2402.00028v1*. Accessed: 17 May 2025. 2024. URL: <https://arxiv.org/html/2402.00028v1>.
- [369] NVIDIA. *Synthetic Data*. Accessed: 17 May 2025. 2025. URL: <https://www.nvidia.com/en-us/use-cases/synthetic-data/>.
- [370] Wikipedia contributors. *Subsurface scattering*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Subsurface\\_scattering](https://en.wikipedia.org/wiki/Subsurface_scattering).
- [371] Wikipedia contributors. *Edwin Catmull*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Edwin\\_Catmull](https://en.wikipedia.org/wiki/Edwin_Catmull).
- [372] Wikipedia contributors. *John Knoll*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/John\\_Knoll](https://en.wikipedia.org/wiki/John_Knoll).
- [373] Wikipedia contributors. *Graphical user interface*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Graphical\\_user\\_interface](https://en.wikipedia.org/wiki/Graphical_user_interface).
- [374] Narges Mahyar. *Mental and Conceptual Models*. Accessed: 17 May 2025. 2019. URL: <https://groups.cs.umass.edu/nmahyar/wp-content/uploads/sites/8/2019/10/325-10-ConceptualModels.pdf>.
- [375] Wikipedia contributors. *Human processor model*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Human\\_processor\\_model](https://en.wikipedia.org/wiki/Human_processor_model).
- [376] Jon Yablonski. *Fitts's Law*. Accessed: 17 May 2025. 2025. URL: <https://lawsofux.com/fittss-law/>.

- [377] Wikipedia contributors. *GOMS*. Accessed: 17 May 2025. 2025. URL: <https://en.wikipedia.org/wiki/GOMS>.
- [378] GeeksforGeeks. *User Interface Design*. Accessed: 17 May 2025. 2024. URL: <https://www.geeksforgeeks.org/software-engineering-user-interface-design/>.
- [379] Tanner Kohler. *Accessibility and Inclusivity: Study Guide*. Accessed: 17 May 2025. 2023. URL: <https://www.nngroup.com/articles/accessibility-inclusivity-study-guide/>.
- [380] Jakob Nielsen. *Response Times: The 3 Important Limits*. Accessed: 17 May 2025. 1993. URL: <https://www.nngroup.com/articles/response-times-3-important-limits/>.
- [381] Krzysztof Z. Gajos. *Consistency in User Interface Design*. Accessed: 17 May 2025. 2006. URL: <https://kgajos.seas.harvard.edu/papers/consistency06.pdf>.
- [382] Wikipedia contributors. *Computer graphics*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Computer\\_graphics](https://en.wikipedia.org/wiki/Computer_graphics).
- [383] Fatih Küçükkarakurt. *Introduction to Game Graphics*. Accessed: 17 May 2025. 2021. URL: <https://dev.to/fkkarakurt/introduction-to-game-graphics-3189>.
- [384] Wikipedia contributors. *Painter's algorithm*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Painter%27s\\_algorithm](https://en.wikipedia.org/wiki/Painter%27s_algorithm).
- [385] Brandon Hope. *Physically Based Rendering and Stylization*. Accessed: 17 May 2025. 2017. URL: <https://www.gamedeveloper.com/art/physically-based-rendering-and-stylization>.
- [386] Bart Wronski. *Why are video games graphics (still) a challenge? Productionizing rendering algorithms*. Accessed: 17 May 2025. 2020. URL: <https://bartwronski.com/2020/12/27/why-are-video-games-graphics-still-a-challenge-productionizing-rendering-algorithms/>.
- [387] Wikipedia contributors. *John Carmack*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/John\\_Carmack](https://en.wikipedia.org/wiki/John_Carmack).
- [388] Wikipedia contributors. *Tim Sweeney*. Accessed: 17 May 2025. 2025. URL: [https://en.wikipedia.org/wiki/Tim\\_Sweeney](https://en.wikipedia.org/wiki/Tim_Sweeney).
- [389] Romanian Game Developers Association. *Vrei să studiezi game development? Află unde poti face asta în România*. Accessed: 17 May 2025. 2025. URL: <https://rgda.ro/vrei-sa-studiezi-game-development-afla-unde-poti-face-asta-in-romania/>.

- [390] Tomas Akenine-Möller. *Tomas Akenine-Möller - NVIDIA Research*. Accessed: 17 May 2025. 2025. URL: <https://research.nvidia.com/person/tomas-akenine-moller>.
- [391] Bart Wronski. *Bart Wronski - NVIDIA Research*. Accessed: 17 May 2025. 2025. URL: <https://research.nvidia.com/person/bart-wronski>.