

Contents

1	Introduction	1
1	General Picture	2
1.1	Technology Enhanced Learning	2
1.2	TEL and CS Technical Artifacts	3
1.3	Educational Software	4
1.4	Computer-Based Pedagogical Settings (CBPSs)	5
1.5	Non-definitional Character of Software	7
1.6	Summary	8
2	Examples	8
2.1	Examples of Computer-Based Pedagogical Settings (CBPSs)	8
2.2	Examples of Educational Software	11
3	Design of Educational Software: Different Realities	15
3.1	Designing and Implementing New Software	15
3.2	Articulating and/or Customizing Software Components	16
3.3	Education and CS Interplays	17
4	Addressing Educational Software Design	18
4.1	Considering Software Properties	18
4.2	Making Explicit Matters of Concern and Perspectives	19
4.3	Importance of Explicitness	22
4.4	Difficulty and Limits of Explicitness	23
5	Content and Structure of the Book	25
5.1	Objective	25
5.2	Content Synthesis	26
5.3	Rationale for the Organization	26
5.4	General Comments	28
2	A General Conceptualization for Educational Software	31
1	Reference Educational Notions	32
1.1	Pedagogical Setting	32

1.2 Teaching Setting	33
1.3 Activity	35
1.4 Pedagogical Objective	37
1.5 Pedagogical Setting Design	39
2 Educational Software Notions	40
2.1 Computer-Based Pedagogical Setting (CBPS)	40
2.2 Educational Software	41
2.3 Pedagogical-Setting Support Software	45
2.4 Software Pedagogical Rationale (SPR)	45
2.5 The Technology Enhanced Learning Field (TEL)	47
3 Important Dimensions and Issues Put to the Fore	48
3.1 Disentangling Pedagogical Objectives	48
3.2 Analyzing the SPR/CBPS Relationships	52
3.3 Synthesis of the Introduced Dissociations	55
3 Understanding Differences in Perspectives	57
1 Notions and Definitions	57
1.1 Examples	58
1.2 Discussion	59
2 Nature of the Setting Analysis: An Example	60
2.1 A General and a Domain-Specific Analysis	61
2.2 Differences and Implications	62
2.3 Conclusions	64
3 Acknowledgement of Influential Factors: Examples	65
3.1 Impact of Software	65
3.2 Software Usage and Socio-technical Dimensions of the Field	67
3.3 Diagnosing Learners' Activity	69
4 Acknowledgement of Activity-Related Uncertainties	70
4.1 Uncertainties Related to the Effectively Considered Task	71
4.2 Uncertainties Related to the Effective Use of Technology	72
4.3 Acknowledgement and Possible Implications for Design	74
4.4 Conclusions	76
5 Disciplinary Dimensions	76
5.1 Mono-disciplinary Work	77
5.2 Clarifying the X-Disciplinary Dimensions of Projects	78
5.3 Conclusions	79
6 Conclusions	79
4 Review of Prototypical Examples	83
1 <i>GeLMS</i> , the Generic Learning Management System	83
2 <i>Phys-edit</i> , the Physics Modeling Editor	86
3 <i>Argue-chat</i> , the Argumentation Chat Tool	87

4	<i>Colab-edit</i> , the Collaborative Editor Environment	88
5	<i>Bio-sim</i> , the Inquiry Setting Environment	90
6	<i>JavIT</i> , the Java Programming Intelligent Tutoring System	91
7	<i>Scen-play</i> , the Generic Scenario Player	93
8	<i>Colab-solver</i> , the Collaborative Problem-Solving Environment	94
9	<i>Geo-world</i> , the Mathematics Graphical Microworld	95
10	Discussion	96
5	CS Perspectives and TEL	99
1	Roles of Computer Science in TEL	100
1.1	Creating Novel Possibilities for Supporting Human Activities	100
1.2	Elaborating Powerful Abstractions	101
1.3	Implementing Specified Models and Processes on Computers	103
2	Engagement of Computer Scientists	103
2.1	TEL as a Place for Clever CS Applications	104
2.2	TEL as a Field Where Some CS Problems Arise	107
2.3	TEL as a Proper Field	108
3	Conclusions	109
6	Educational Software Engineering	111
1	Engineering and Research	111
2	Educational Software Engineering as a Scientific Field	112
2.1	Educational Software as Complex Artificial Objects	112
2.2	Definition and Matters of Concern	113
2.3	Transversal Efforts to Clarify Issues	114
2.4	Specific Efforts to Build Engineering Methodologies	115
2.5	Conducting Projects as Vectors for Knowledge Development	117
3	Reconsidering the CS-TEL Relationship	118
3.1	Educational Software Engineering and Research	118
3.2	Educational Software Engineering and CS Research	119
4	Conclusions	122
7	Characterizing the Design Context and the Software Artifact	123
1	Introduction	124
2	Characterizing the Design Context	126
2.1	Research/Development Nature of the Work	126
2.2	Theoretical Background	128
2.3	Nature of the Targeted Outcome	130
2.4	Rationale for Designing Software	132
2.5	How Software Is Considered Within the CBPS	133
2.6	Design Approach	135

2.7 Actors Concerned	137
2.8 Context and Historical Dimensions of the Project	138
3 Characterizing the Software Artifact	139
3.1 Level of Analysis of Software Properties	139
3.2 Actions Considered at the Level of Software	139
3.3 Reification of the Pedagogical Intention in Software	140
3.4 Nature of the CS Treatments	141
3.5 Level of Achievement	143
4 Examples	143
4.1 <i>GeLMS-4</i> , the Generic Learning Management System	144
4.2 <i>JavIT</i> , the Java Programming Intelligent Tutoring System	146
5 Conclusions	148
 8 Methodological Considerations	 151
1 Clarifying Concerns	151
2 Dealing with Complexity and Models	152
2.1 Multiplicity of Models	152
2.2 Foundations of Models	153
2.3 Traceability of Models	154
3 Making the SPR Explicit	155
4 Considering Activity and Indirect Design	158
5 Developing Knowledge	160
5.1 Definition of Issues	160
5.2 Definition of Results	162
5.3 Evaluation of Results	163
 9 Conclusions	 165
1 Educational Software Design and Evolution of Technologies	165
2 Lack of Knowledge Capitalization	167
3 Pushing Forward Educational Software Engineering	170
3.1 Conditions for Capitalizing Knowledge	171
3.2 Review of Possible Focuses	173
3.3 Analysis of the Different Perspectives	176
3.4 Conclusions	177
 Index	 179