

Contents

1	Historical Background	1
1.1	The 1970s	1
1.1.1	The Experimental Paradigm	1
1.1.2	Methodological Criticism	3
1.1.3	Absence of a Theoretical Framework	3
1.1.4	Badly Used Theoretical Borrowings	5
1.1.5	General Criticisms	6
1.2	Second Period	6
1.2.1	Theoretical Framework	7
1.2.2	Theoretical Changes	8
1.2.3	Paradigmatic Changes	8
1.3	Recent Thematic Developments	9
2	What Is a Computer Program?	13
2.1	Definition of a Program	13
2.1.1	From a Computer Point of View.	13
2.1.2	From a Psychological Point of View	14
2.2	Program vs Text	18
2.2.1	A Program Seen as a Narrative Text	18
2.2.2	A Program Seen as a Procedural Text	18
2.3	Programming Languages vs Natural Languages	19
2.3.1	Unambiguous Nature of Programming Languages	19
2.3.2	Use of Natural Language or Pseudo-Code	19
2.4	Toward a Broader Definition	19
3	Software Design: Theoretical Approaches	21
3.1	Features of the Problems of Program Design	22
3.1.1	Ill-defined Problems	22
3.1.2	Problems of Program Production	22
3.2	Knowledge-centred Approaches	23
3.2.1	Theory of Schemas	23
3.2.2	Programming Schemas	24
3.2.3	Other Types of Schema	25
3.2.4	Rules of Discourse	25
3.2.5	Limitations of Schemas	26
3.3	Strategy-centred Approach	26

3.3.1	Classification of Strategies	26
3.3.2	Triggering Conditions	28
3.4	Organization-centred Approach	31
3.4.1	Hierarchical Models	31
3.4.2	Opportunistic Models	31
3.4.3	The Iterative Nature of Design	32
3.5	Modelling the Expert	34
3.5.1	What Distinguishes an Expert from a Novice?	34
3.5.2	Can Different Levels of Expertise Be Distinguished?	35
3.5.3	What Are the Stages in Acquiring Expertise?	36
3.6	Making Tools More Suitable for Programmers	36
3.7	Future Research	39
4	Software Reuse	43
4.1	Analogical Reasoning Models	44
4.1.1	The Syntactic Model and the Pragmatic Model	44
4.1.2	Clement's Model	46
4.2	Cognitive Mechanisms Employed in Reuse	47
4.2.1	Generation of a Source	47
4.2.2	Use of the Source	49
4.2.3	Implications	49
4.3	Cognitive Classification of Reuse Situations	51
4.3.1	Reuse of New Code vs Reuse of Old Code	52
4.3.2	Reuse During the Different Design Phases	52
4.3.3	Implications	53
4.4	Future Research	54
5	Design and Reuse of Object-Oriented Software: the Effect of a Programming Paradigm	57
5.1	Cognitive Implications of OO: Hypotheses	57
5.1.1	The Object-Oriented Approach	57
5.1.2	The Naturalness of OO Design	58
5.1.3	Better Reusability of OO Components	59
5.2	Object-Oriented Design	59
5.2.1	Mapping Between the Problem Space and the Solution Space	60
5.2.2	OO Programming Schemas	60
5.2.3	Design Strategies	61
5.2.4	Organization of the Design Process	63
5.2.5	Development of Expertise in OO	64
5.2.6	Cognitive Implications of OO: Naturalness	66
5.2.7	Practical Implications	67
5.3	Reuse in the OO Paradigm	68
5.3.1	Reuse in a Procedural Paradigm vs Reuse in OO	68

5.3.2	Potential Reuse in OO	69
5.3.3	Cognitive Mechanisms Deployed in OO Reuse	70
5.3.4	Cognitive Implications of OO: Summary of Experimental Findings on Reuse	72
5.3.5	Practical Implications	72
5.4	Future Research	73
6	Understanding Software	75
6.1	Models of Text Understanding	75
6.1.1	Functional Models	76
6.1.2	Structural Models	77
6.1.3	The Mental Model Approach	78
6.2	Program Comprehension Seen as Text Understanding	79
6.2.1	To Understand a Program Is to Apply Knowledge Schemas	79
6.2.2	To Understand a Program Is to Construct a Network of Relations	87
6.2.3	To Understand a Program Is to Construct a Representation of the Situation	93
6.3	Program Comprehension Seen as Problem Solving	100
6.4	Conclusions and Practical Implications	102
7	Understanding Software: Effects of the Task and the Textual Structure	105
7.1	Influence of the Task	105
7.1.1	Effect of the Purpose for Reading on Text Comprehension	105
7.1.2	Effect of the Task on Program Comprehension	106
7.1.3	Research Prospects	109
7.2	Effect of the Textual Structure	110
7.2.1	Surface Structure vs Deep Structure	110
7.2.2	Organizers	110
7.2.3	Discontinuities and Delocalized Plans	113
7.2.4	Research Prospects	114
7.3	Practical Implications	115
8	The Future for Programming Psychology	117
8.1	Prospects for the Software Community	117
8.1.1	Obstacles	117
8.1.2	Removing the Obstacles to Effective Transfer	118
8.2	Contributions to Cognitive Psychology	119
	References	123
	Index	135



<http://www.springer.com/978-1-85233-253-2>

Software Design - Cognitive Aspect

Detienne, F.

2002, XIV, 146 p. 10 illus., Softcover

ISBN: 978-1-85233-253-2