NON-FUNCTIONAL REQUIREMENTS IN SOFTWARE ENGINEERING

by

Lawrence Chung
Department of Computer Science
The University of Texas at Dallas

Brian A. Nixon
Department of Computer Science
University of Toronto

Eric Yu
Faculty of Information Science
University of Toronto

John Mylopoulos
Department of Computer Science
University of Toronto

SPRINGER SCIENCE+BUSINESS MEDIA, LLC

Library of Congress Cataloging-in-Publication Data

Non-functional requirements in software engineering / by Lawrence Chung...[et al.].
p. cm. -- (The Kluwer international series in software engineering)
Includes bibliographical references.
ISBN 978-1-4613-7403-9 ISBN 978-1-4615-5269-7 (eBook)
DOI 10.1007/978-1-4615-5269-7

1. Software engineering. 2. Computer software--Quality control. I. Series. II. Chung, Lawrence.

QA76.758 .N65 1999
005.1 21--dc21 99-046023

Copyright © 2000 by Springer Science+Business Media New York
Originally published by Kluwer Academic Publishers in 2000
Softcover reprint of the hardcover 1st edition 2000

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, mechanical, photo-copying, recording, or otherwise, without the prior written permission of the publisher, Springer Science+Business Media, LLC

Printed on acid-free paper.

THE KLUWER INTERNATIONAL SERIES IN SOFTWARE ENGINEERING

Series Editor

Victor R. Basili
University of Maryland
College Park, MD 20742

Also in the Series:

FORMAL SPECIFICATION TECHNIQUES FOR ENGINEERING MODULAR C PROGRAMS, *by TAN Yang Meng;* ISBN: 0-7923-9653-7

TOOLS AND ENVIRONMENTS FOR PARALLEL AND DISTRIBUTED SYSTEMS, *by Amr Zaky and Ted Lewis;* ISBN: 0-7923-9675-8

CONSTRAINT-BASED DESIGN RECOVERY FOR SOFTWARE REENGINEERING: Theory and Experiments, *by Steven G. Woods, Alexander E. Quilici and Qiang Yang;* ISBN: 0-7923-8067-3

SOFTWARE DEFECT MODELING, *by Kai-Yuan Cai;* ISBN: 0-7923-8259-5

The Kluwer International Series in Software Engineering addresses the following goals:

- To coherently and consistently present important research topics and their application(s).
- To present evolved concepts in one place as a coherent whole, updating early versions of the ideas and notations.
- To provide publications which will be used as the ultimate reference on the topic by experts in the area.

With the dynamic growth evident in this field and the need to communicate findings, this series provides a forum for information targeted toward Software Engineers.

NON-FUNCTIONAL REQUIREMENTS IN SOFTWARE ENGINEERING

Contents

List of Figures ix
List of Tables xvii
LEGEND FOR FIGURES xix
Preface xxix

1. INTRODUCTION 1
   1.1 Introduction 1
   1.2 The Nature of Non-Functional Requirements 6
   1.3 Literature Notes 9

Part I The NFR Framework

2. THE NFR FRAMEWORK IN ACTION 15
   2.1 Using the NFR Framework 16
   2.2 Acquiring Domain Knowledge 18
   2.3 Acquiring and Cataloguing NFR Knowledge 18
   2.4 Identifying NFRs 19
   2.5 Decomposing NFR Softgoals 21
   2.6 Dealing with Priorities 25
   2.7 Identifying Possible Operationalizations 27
   2.8 Dealing with Implicit Interdependencies among Softgoals 30
   2.9 Recording Design Rationale 33
   2.10 Selecting Among Alternatives 35
   2.11 Evaluating the Impact of Decisions 37
   2.12 Cataloguing Development Methods and Correlations 42
   2.13 Discussion 44

3. SOFTGOAL INTERDEPENDENCY GRAPHS 47
   3.1 Kinds of Softgoals 48
   3.2 Interdependencies 54
   3.3 The Evaluation Procedure 70

3.4 Coupling NFRs with Functional Requirements 80
3.5 Discussion 85

4. CATALOGUING REFINEMENT METHODS AND CORRELATIONS 89
4.1 Refinement Methods 90
4.2 NFR Decomposition Methods 90
4.3 Operationalization Methods 111
4.4 Argumentation Methods and Templates 119
4.5 Correlations 129
4.6 Putting Them All Together: The Goal-Driven Process 137
4.7 Discussion 141
4.8 Related Literature for the Framework 142

Part II Types of Non-Functional Requirements

5. TYPES OF NFRs 153
5.1 Categorizations of NFRs 155
5.2 Standards 158
5.3 A List of NFRs 159
5.4 Our Approach: The NFR Framework 159
5.5 Literature Notes 160

6. ACCURACY REQUIREMENTS 161
6.1 Accuracy Concepts 163
6.2 Decomposition Methods 167
6.3 Operationalization Methods 175
6.4 Argumentation Methods 180
6.5 Correlations 181
6.6 Illustration 184
6.7 Discussion 194

7. SECURITY REQUIREMENTS 197
7.1 Security Concepts 198
7.2 Decomposition Methods 201
7.3 Operationalization Methods 204
7.4 Argumentation Templates and Methods 207
7.5 Correlations 207
7.6 Illustration 208
7.7 Discussion 213

8. PERFORMANCE REQUIREMENTS 217
8.1 Performance Concepts 218
8.2 Factors for Dealing with Performance Requirements 223
8.3 Refinement Methods 225

8.4 Operationalization Methods from Software Performance Engineering 233
8.5 Argumentation Methods and Templates 236
8.6 Correlations 238
8.7 Illustration 239
8.8 Discussion 247

9. PERFORMANCE REQUIREMENTS FOR INFORMATION SYSTEMS 249
9.1 Language Features and Implementation Techniques for Information Systems 250
9.2 Example: A Research Management System 252
9.3 Extending the Performance Type 258
9.4 Organizing Issues via Language Layers 259
9.5 Decomposition Methods for Handling Data Management 264
9.6 Methods for Handling Inheritance Hierarchies 267
9.7 Methods for Handling Integrity Constraints and Long-Term Processes 273
9.8 Organizing Performance Methods 277
9.9 Organizing Correlations 280
9.10 Illustration 282
9.11 Discussion 283

Part III Case Studies and Applications

10. INTRODUCTION TO THE STUDIES AND APPLICATIONS 291
10.1 Introduction 292
10.2 Characteristics of Domains Studied 293
10.3 Our Approach to Conducting the Studies 297
10.4 Observations from Studies 300
10.5 Literature Notes 300

11. A CREDIT CARD SYSTEM 301
11.1 Domain Description and Functional Requirements 301
11.2 Non-Functional Requirements 304
11.3 Dealing with Performance Requirements 305
11.4 Dealing with Security and Accuracy Requirements 323
11.5 Discussion 328
11.6 Literature Notes 329

12. AN ADMINISTRATIVE SYSTEM 331
12.1 Introduction 331
12.2 Domain Description, Functional Requirements and Organizational Workload 331
12.3 Non-Functional Requirements 333
12.4 Recording Domain Information in a Design 334
12.5 Overview of SIGs 334
12.6 Time Softgoals for Managing Long-Term Tax Appeal Processes 336
12.7 Operationalization Methods for Integrity Constraints 340

12.8 Dealing with a Tradeoff 346
12.9 Discussion 350
12.10Literature Notes 350

13. APPLICATION TO SOFTWARE ARCHITECTURE 351
13.1 Introduction 352
13.2 Cataloguing Software Architecture Concepts using the NFR Framework 354
13.3 Illustration of the Architectural Design Process 358
13.4 Discussion 365
13.5 Literature Notes 366

14. ENTERPRISE MODELLING AND BUSINESS PROCESS REDESIGN 367
14.1 Introduction 367
14.2 The Strategic Dependency Model 370
14.3 The Strategic Rationale Model 374
14.4 Discussion 381
14.5 Literature Notes 382

15. ASSESSMENT OF STUDIES 383
15.1 Feedback from Domain Experts 383
15.2 Discussion: Lessons Learned for Conducting Studies 387
15.3 Literature Notes 389

POSTSCRIPT 391

BIBLIOGRAPHY 399

List of Figures

0.1	Logos for figures.	xx
0.2	Conventions for abbreviations and usage of fonts.	xxi
0.3	Legend for Softgoal Interdependency Graphs.	xxii
0.4	Legend for Softgoals and Interdependencies.	xxiii
0.5	Legend for Functional Requirements.	xxiv
0.6	Kinds of Refinements.	xxv
0.7	The "individual impact" of an offspring upon its parent for selected contribution types during the First Step of the evaluation procedure (Chapter 3). Parent labels are shown in the table entries.	xxvi
0.8	The "individual impact" of an offspring upon its parent during the First Step of the evaluation procedure (Chapter 3). An elaboration of Figure 0.7.	xxvi
0.9	Label propagation for selected contribution types and offspring labels during the First Step.	xxvii
0.10	Legend for Performance Requirements (Chapters 8 and 9).	xxviii
0.11	Legend for Business Process Redesign (Chapter 14).	xxviii
2.1	A catalogue of some NFR Types.	19
2.2	An initial Softgoal Interdependency Graph with NFR softgoals representing requirements for performance and security of customer accounts.	20
2.3	Decomposing NFR softgoals into more specific non-functional requirements.	22
2.4	Further decomposition of a security softgoal.	23
2.5	Considering a performance softgoal.	24
2.6	Decomposing a performance softgoal.	25
2.7	Identifying a softgoal as a priority.	26
2.8	Identifying possible operationalizations for NFR softgoals.	28
2.9	Detecting implicit interdependencies among existing softgoals.	31
2.10	Detecting implicit interdependencies among existing and other softgoals.	32

2.11	Recording design rationale.	34
2.12	Selecting among alternatives.	36
2.13	Evaluating the impact of decisions.	38
2.14	Relating decisions to Functional Requirements.	41
2.15	A catalogue of operationalization methods for achieving confidentiality.	43
2.16	A catalogue showing the impact of operationalizing softgoals upon NFR softgoals.	44
3.1	Representing sample non-functional requirements as NFR softgoals.	50
3.2	Portions of NFR type catalogues.	50
3.3	Sample operationalizing softgoals.	52
3.4	Representation of claims softgoals.	53
3.5	Decomposition and prioritization of NFR softgoals.	55
3.6	Kinds of Refinements.	56
3.7	Refining NFR softgoals into operationalizing softgoals.	57
3.8	Recording design rationale using claim softgoals.	59
3.9	A decomposition with an *AND* contribution.	61
3.10	Examples of several contribution types.	62
3.11	Intuitive distinction among contribution types.	63
3.12	Grouping positive and negative contribution types.	64
3.13	A conflict in contributions.	65
3.14	A softgoal interdependency graph with various contribution types.	69
3.15	Catalogue of label values.	72
3.16	Label propagation for selected contribution types and offspring labels during the First Step.	75
3.17	Examples of "automatic" label propagation during the Second Step.	76
3.18	Examples of developer-directed label propagation during the Second Step.	77
3.19	Meaning of symbols in softgoal interdependency graphs.	78
3.20	Recording developer's decisions to choose or reject softgoals.	80
3.21	Determining the impact of developer's decisions, using the evaluation procedure.	81
3.22	Relating functional requirements and the target system to a SIG.	82
3.23	Dealing with functional requirements guided by considerations of NFRs.	84
4.1	A catalogue of NFR decomposition methods.	91
4.2	A catalogue of NFR decomposition methods, including those for specific NFRs.	92
4.3	Definition and application of the AccountResponseTimeViaSubclass decomposition method.	93

4.4 Definition and application of ResponseTimeViaSubclass, a parameterized decomposition method. 95
4.5 Another application of the ResponseTimeViaSubclass method. 97
4.6 The StaffingAndAmountForOperatingCostViaSubclassAndDollarLimit method, parameterized on two topics. 98
4.7 A catalogue of NFR Types. 99
4.8 Definition and application of the AccountSecurityViaSubType method. 100
4.9 The AccountQualityViaSubType method, parameterized on NFR type. 101
4.10 The SubType method, parameterized on NFR Type and Topic, and applied to performance of accounts. 102
4.11 The SubType method, applied to adaptability of insurance claims processing. 103
4.12 The SubType type decomposition methods, with varying degrees of parameterization. 104
4.13 The Subclass method, applied to space performance of accounts. 106
4.14 The Subclass method, applied to modifiability of packages. 107
4.15 The Subclass topic decomposition methods, with varying degrees of parameterization. 108
4.16 The Attribute decomposition method and one of its applications. 109
4.17 A Softgoal Interdependency Graph with two method applications. 110
4.18 A catalogue of operationalization methods. 111
4.19 The PerformFirst method, applied to an operation on an attribute. 112
4.20 The CompressedFormat operationalization method and an application. 113
4.21 The Auditing method and one of its applications. 114
4.22 The Validation method and one of its applications. 115
4.23 Refining an operationalizing softgoal and adding a topic. 116
4.24 Applying the AuthorizationViaSubType method to accounts, resulting in several offspring of an operationalizing softgoal. 117
4.25 A Softgoal Interdependency Graph with applications of various decomposition and operationalization methods. 118
4.26 A catalogue of argumentation methods. 119
4.27 A template for claims about validation. 121
4.28 Claims about operationalization methods. 122
4.29 A template for the VitalFew argumentation method for prioritization, and an example of its usage in a claim. 123
4.30 A template for claims about prioritization. 125
4.31 Another template for claims about prioritization. 126
4.32 Modifying the offspring labels of an *AND* interdependency. 127
4.33 Modifying the parent label of an *AND* interdependency. 128
4.34 A SIG with operationalization and argumentation methods. 129

4.35 Sample correlations. 131
4.36 Different kinds of inferences which can be made by using a correlation rule. 133
4.37 A correlation catalogue. 135
4.38 Benefits of using a common component in refinements of an operationalization. 136
4.39 Selection of operationalizating softgoals and argumentation softgoals. 138
4.40 The impact of decisions on NFRs and Functional Requirements. 139
5.1 Software Quality Characteristics Tree (From [Boehm76]). 157
5.2 Types of non-functional requirements (From [Sommerville92]). 158
6.1 An example of information flow. 164
6.2 An accuracy type catalogue. 165
6.3 A catalogue of accuracy decomposition methods. 168
6.4 A description of information flow. 170
6.5 A catalogue of accuracy operationalization methods. 176
6.6 The ValidationResourceAvailability method. 179
6.7 A catalogue of accuracy argumentation methods. 181
6.8 Decompositions for accurate travel expenses. 185
6.9 Operationalizing the accuracy of reimbursement requests. 188
6.10 Detecting a negative impact on a security requirement. 190
6.11 Evaluation of selective certification of expense summaries. 191
6.12 Relating functional requirements to the target system. 193
7.1 A catalogue of security types. 198
7.2 A catalogue of security operationalization methods. 205
7.3 Refinement of a security softgoal by subtypes. 209
7.4 SIG for confidential accounts. 210
8.1 The Performance Type. 220
8.2 Characteristics of the Performance Type. 220
8.3 Catalogue of Decomposition Methods for Performance. 225
8.4 A SubType decomposition. 226
8.5 Decomposing a performance softgoal using the Subclass method. 227
8.6 Using the IndividualAttributes method. 228
8.7 A decomposition based on implementation components. 228
8.8 Using the FlowThrough method to link layers. 230
8.9 A template for inter-layer refinement. 230
8.10 A prioritization argument. 231
8.11 Using the EarlyFixing method. 233
8.12 Refining an EarlyFixing operationalizing softgoal. 234
8.13 Positive and negative impacts of an operationalizing softgoal. 236
8.14 Catalogue of Operationalization Methods for Performance. 237
8.15 A correlation catalogue for Performance. 238
8.16 A decomposition on attributes. 240
8.17 A prioritization argument based on workload. 241
8.18 Consideration of an operationalizing softgoal. 242

8.19 An inter-layer interdependency link. 242
8.20 Initial operationalizing softgoals. 243
8.21 Refined operationalizing softgoals. 245
8.22 Evaluation of the Softgoal Interdependency Graph. 246
9.1 Definitions of the Employee and Researcher classes. 253
9.2 Defining ComputerResearcher as a specialization of Researcher. 254
9.3 Entity (data) classes in the functional requirements for the research administration example. 254
9.4 Transaction classes in the functional requirements. 255
9.5 A long-term research administration process represented as a script. 256
9.6 Adding ManagementTime to the performance type. 258
9.7 Adding ManagementTime and characteristics to the performance type. 259
9.8 Performance issues arranged in a grid. 260
9.9 Layered organization of performance knowledge. 261
9.10 Space of implementation alternatives arranged by layer. 263
9.11 Arrangement of attribute values in the presence of inheritance hierarchies. 268
9.12 Horizontal splitting of attributes. 268
9.13 Vertical splitting of attributes. 268
9.14 Attributes stored as "triples." 269
9.15 Organization of Performance decomposition methods. 278
9.16 Some Performance decomposition methods for Layers 6 and 5. 279
9.17 Organization of Performance operationalization methods. 280
9.18 A performance correlation catalogue for information system development. 281
9.19 Dealing with inheritance hierarchies. 282
9.20 Linking another layer to the graph. 284
10.1 Overview of studies presented in Part III. 294
10.2 Overview of studies presented elsewhere. 295
11.1 Attributes of credit cardholders. 302
11.2 Classes in the credit card system. 302
11.3 Information Flow for the credit card system. 303
11.4 Some main NFRs for the credit card system. 304
11.5 Initial softgoal for fast cancellation of credit cards. 306
11.6 Softgoal decomposition and prioritization with argumentation. 307
11.7 Further softgoal decomposition and prioritization. 308
11.8 Selection of Operationalizing Softgoals. 309
11.9 Using inter-layer interdependency links. 310
11.10 Decomposition based on implementation components. 311
11.11 Evaluating the impact of decisions after selecting operationalizing softgoals. 312
11.12 Dealing with transaction hierarchies at Layer 4. 314
11.13 Dealing with priority operations at Layer 3. 315

11.14 Dealing with credit card attributes at Layer 2. 317
11.15 Evaluating the impact of decisions. 318
11.16 Considering inheritance hierarchies at Layer 4. 319
11.17 Selecting attribute storage methods at Layer 2. 321
11.18 Evaluating the impact of decisions. 322
11.19 SIG for storage of sales information. 323
11.20 Main Security Requirements for the credit card system. 324
11.21 Accuracy and Confidentiality Requirements for the credit card system. 325
11.22 Interactions among softgoals. 325
11.23 Evaluating the use of an operationalization. 327
11.24 Refining an Internal Confidentiality softgoal. 328
11.25 Evaluating the impact of alarms on Internal Confidentiality. 329
12.1 The tax appeals process represented as a Script. 335
12.2 Initial softgoal of good time performance for managing transitions of appeal process. 336
12.3 Decomposition into softgoals for individual transitions. 337
12.4 Further decompositions at Layer 6. 338
12.5 An inter-layer refinement. 339
12.6 Considering some operationalizing softgoals. 340
12.7 An Argument about an Operationalization. 341
12.8 Refining an Operationalizing Softgoal. 342
12.9 Evaluation of the SIG. 343
12.10 Time and Space softgoals and their refinements. 346
12.11 Identifying priorities using workload-based arguments. 347
12.12 Developing an hybrid method to deal with tradeoffs. 348
13.1 Catalogue of some NFR Types considered for software architecture. 355
13.2 A method for refining a modifiability softgoal for the system. 356
13.3 A method for refining a modifiability softgoal for the process. 356
13.4 A generic Correlation Catalogue, based on [Garlan93]. 357
13.5 Initial NFR Softgoals for a KWIC system. 358
13.6 Refining softgoals using methods. 359
13.7 Tradeoffs among operationalizations. 361
13.8 Domain-Specific Correlation Catalogue for KWIC Example. 362
13.9 Evaluating the impact of the chosen alternative on NFRs. 364
14.1 Strategic Dependency Model for existing auto insurance claims handling. 371
14.2 Strategic Dependency Model for letting the insurance agent handle claims. 373
14.3 Strategic Dependency Model for letting the body shop handle claims. 374
14.4 Strategic Rationale Model relating softgoals and tasks for insurance claims handling. 375
14.5 Strategic Rationale Model for handling claims centrally. 377

14.6 Strategic Rationale Model for handling small claims by the insurance agent. 378
14.7 Strategic Rationale Model for handling small claims by the body shop. 379
14.8 Overall Strategic Rationale Model for Claims Handling. 380

List of Tables

1.1	RADC software quality consumer-oriented attributes [Keller90].	2
3.1	The "individual impact" of an offspring upon its parent for selected contribution types during the First Step. Parent labels are shown in the table entries.	74
3.2	The "individual impact" of an offspring upon its parent during the First Step. Parent labels are shown in the table entries.	74
5.1	Software quality factors and criteria (From [Keller90]).	156
5.2	A list of non-functional requirements.	160
6.1	An accuracy correlation catalogue.	183

LEGEND FOR FIGURES

We present a collected legend for the figures of this book.

Figures in this book have a "logo" in the top left corner indicating the type of the figure. Logos for some figures types are given in Figure 0.1.

Some types of figures contain sub-figures. In Figure 0.1, their names are indented and preceded by a hyphen.

There are also *Informal* figures in Chapter 2, which use an approximate syntax.

Legend

Logo	Explanation
Claim Template	
Contribution Catalogue	
Contribution Type Examples	
Correlation Catalogue	
Evaluation Catalogue	
Evaluation Examples	
FRs	Functional Requirements
Informal Correlation Catalogue	(Chapter 2)
Informal Legend	(Chapter 2)
Informal SIG	(Chapter 2)
Information Flow	
Label Catalogue	
Layered Structuring	(Chapters 8 & 9)
Legend	
Method Application — Initial SIG — Resultant SIG	
Method Catalogue	
Method Definition, Parameterized on Topic	
Method Definition, Parameterized on NFR Type	
Method Definition, Parameterized on NFR Type and Topic	
Method Definition, Unparameterized	
Method Hierarchy	
NFR Type Catalogue	
Refinement Catalogue	
Rule Definition, Parameterized on Topic	
Rule Application — Initial SIG — Resultant SIG	
SIG	Softgoal Interdependency Graph
Softgoal Examples	
Strategic Dependency Model	(Chapter 14)
Strategic Rationale Model	(Chapter 14)
Template — Initial SIG — Resultant SIG	
Template Usage — Initial SIG — Resultant SIG	

Figure 0.1. Logos for figures.

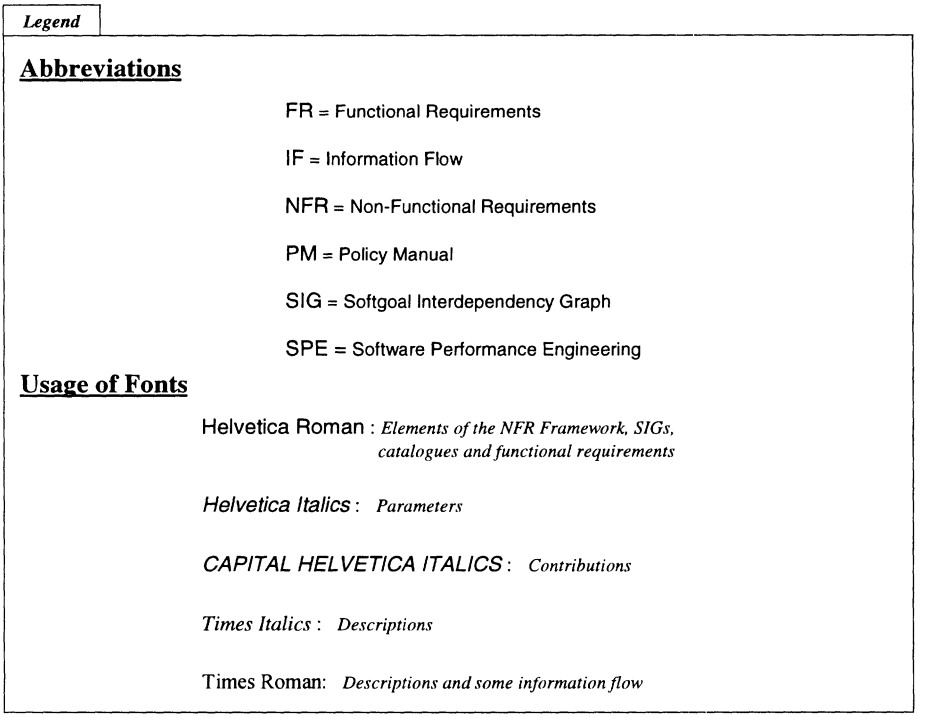

Figure 0.2. Conventions for abbreviations and usage of fonts.

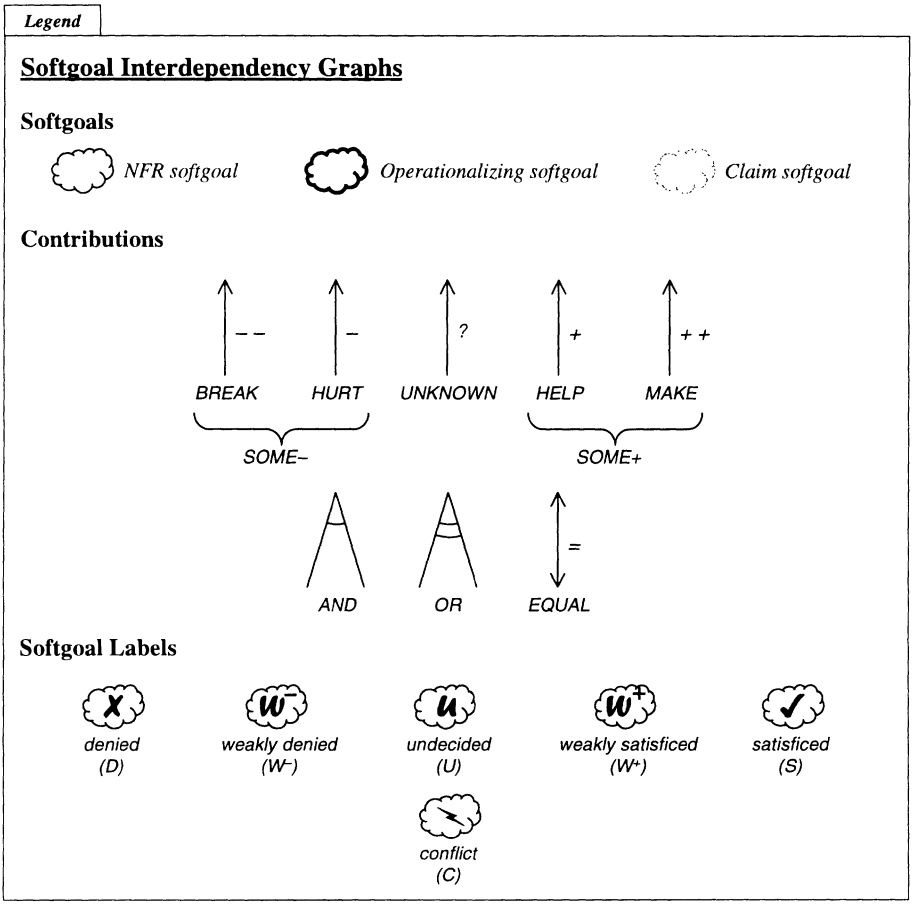

Figure 0.3. Legend for Softgoal Interdependency Graphs.

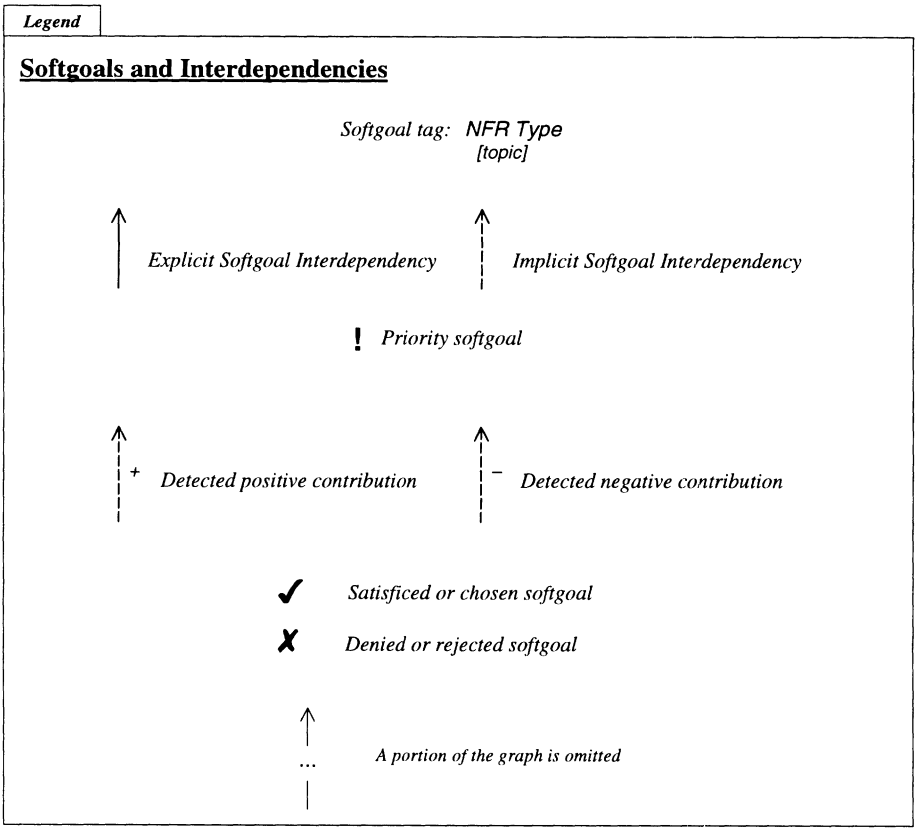

Figure 0.4. Legend for Softgoals and Interdependencies.

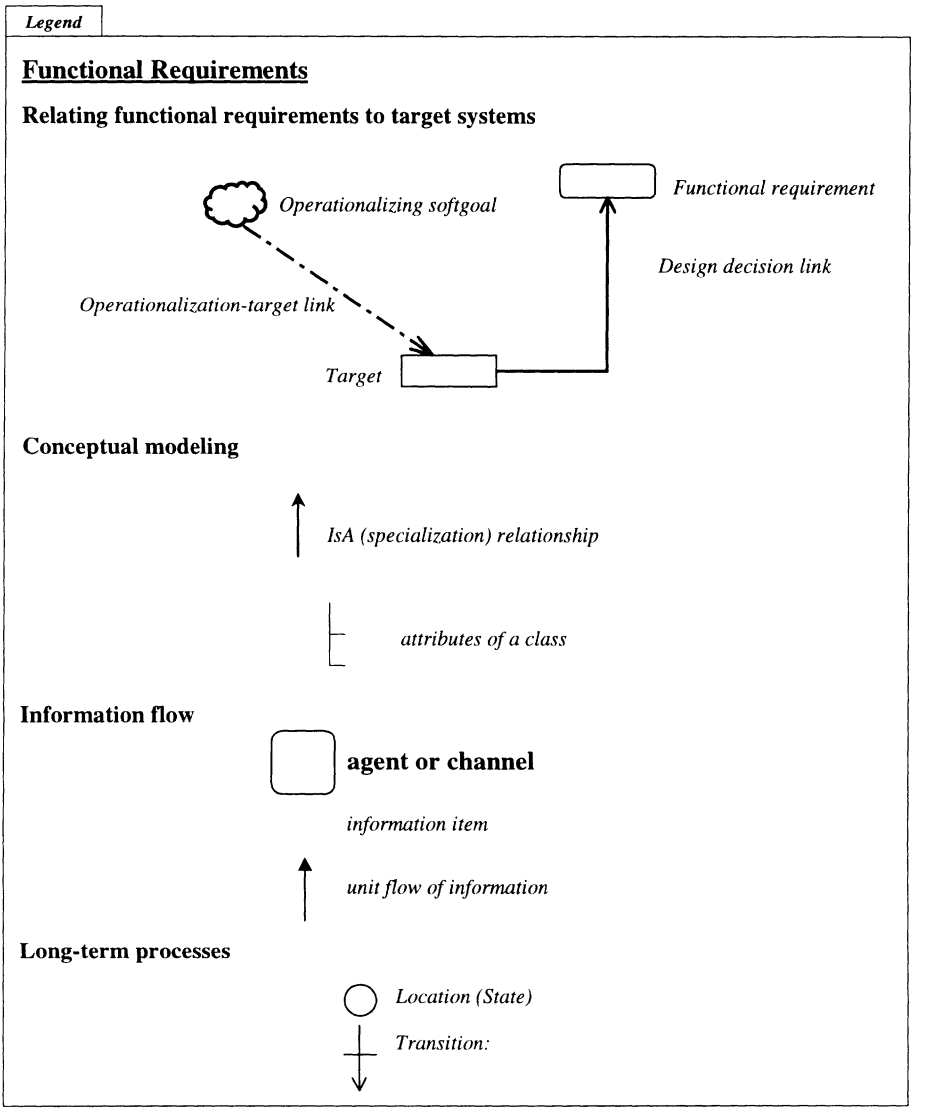

Figure 0.5. Legend for Functional Requirements.

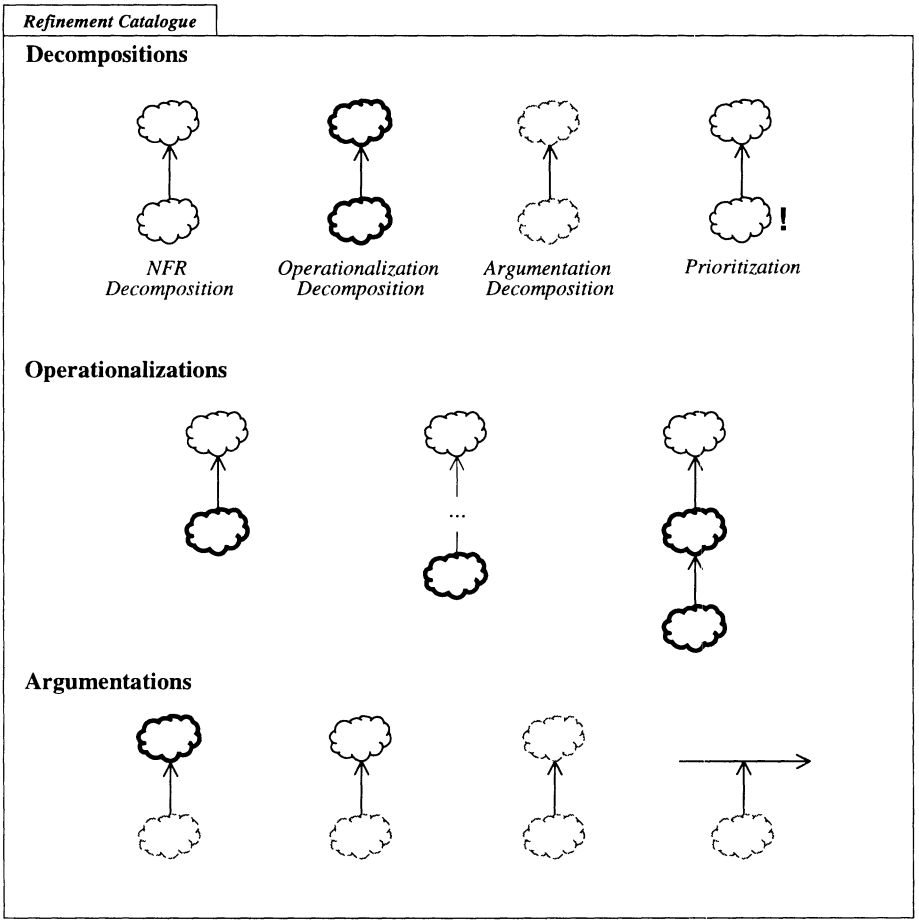

Figure 0.6. Kinds of Refinements.

Evaluation Catalogue

Individual Impact of **offspring** *with* **label:**	*upon* **parent label,** *given offspring-parent* **contribution type:**			
	SOME-	? (UNKNOWN)	*SOME+*	=
× *(Denied)*	W+	U	W−	×
↯ *(Conflict)*	↯	U	↯	↯
U *(Undetermined)*	U	U	U	U
√ *(Satisficed)*	W−	U	W+	√

Figure 0.7. The "individual impact" of an offspring upon its parent for selected contribution types during the First Step of the evaluation procedure (Chapter 3). Parent labels are shown in the table entries.

Evaluation Catalogue

Individual Impact of **offspring** *with* **label:**	*upon* **parent label,** *given offspring-parent* **contribution type:**							
	BREAK	*SOME-*	*HURT*	?	*HELP*	*SOME+*	*MAKE*	=
×	W+	W+	W+	U	W−	W−	×	×
↯	↯	↯	↯	U	↯	↯	↯	↯
U	U	U	U	U	U	U	U	U
√	×	W−	W−	U	W+	W+	√	√

Figure 0.8. The "individual impact" of an offspring upon its parent during the First Step of the evaluation procedure (Chapter 3). An elaboration of Figure 0.7.

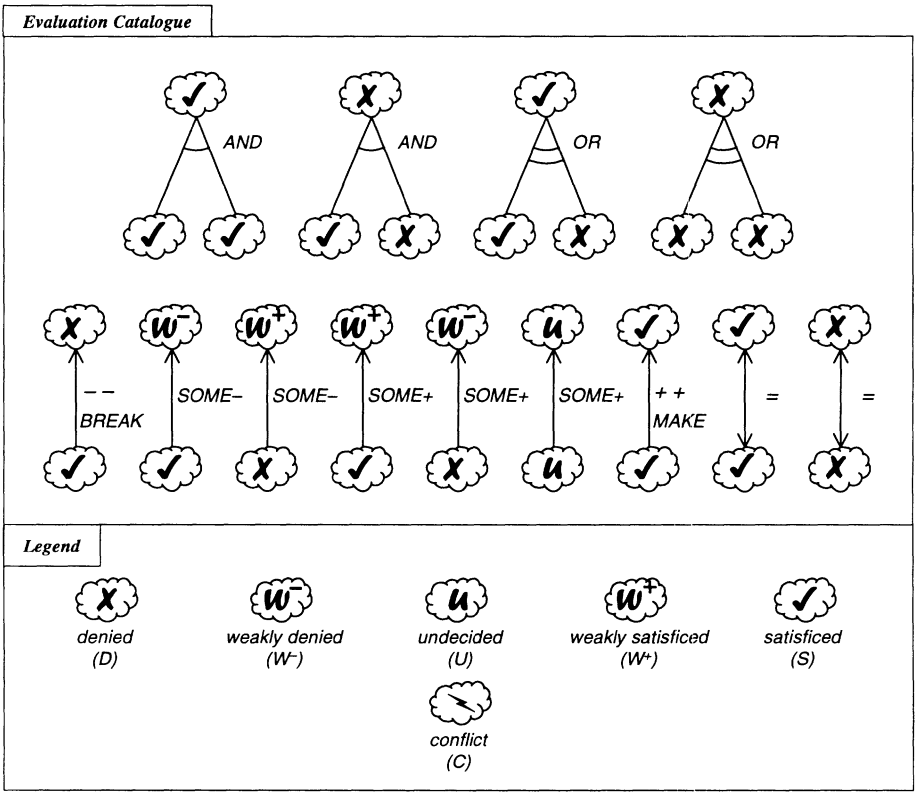

Figure 0.9. Label propagation for selected contribution types and offspring labels during the First Step.

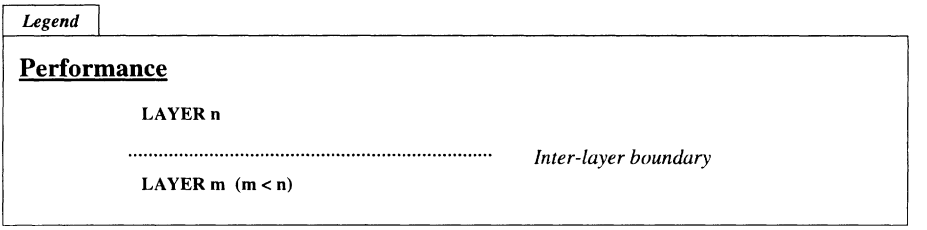

Figure 0.10. Legend for Performance Requirements (Chapters 8 and 9).

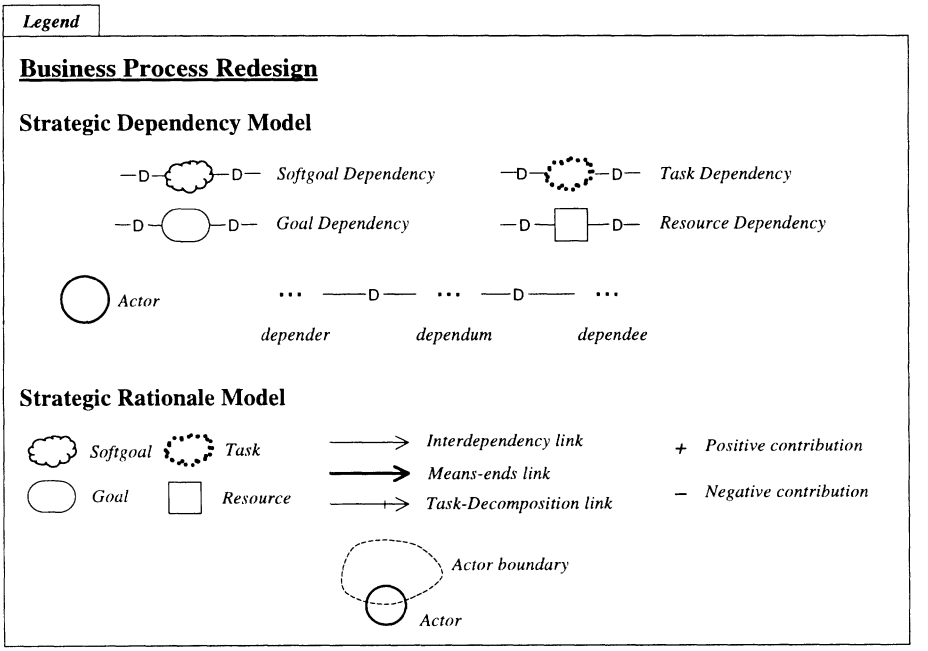

Figure 0.11. Legend for Business Process Redesign (Chapter 14).

The material in this book is based on the following publications, which are overviewed in Chapter 1: [Chung93a] [Nixon97a] [Yu94b] [Mylopoulos92a] [Chung94a,b] [Chung91a, 93b] [Nixon91, 93, 94a] [Chung95b] [Chung95c,d] [Yu94c] [Mylopoulos97] [Chung91b] [Nixon90]. The "Literature Notes" and "Discussions" at the end of each chapter describe the source publications for the particular chapter.

In writing this book, we have benefitted from the work of many people.

The following diagrams are taken from, or in some cases adapted from, the source publications listed below.

Table or Figure in this Book	*Source Publication*
Table 1.1	[Keller90]
Figure 5.1	[Boehm76]
Figure 5.1	[Keller90]
Figure 5.2	[Sommerville92]
Figure 13.4	[Garlan93]

The following trademarks have been used: SADT, a trademark of Softech Inc., and REFINE, a trademark of Reasoning Systems Inc.

We express our sincere gratitude to Scott Delman, Senior Publishing Editor of Kluwer Academic Publishers for first suggesting this book, and for his ongoing encouragement to complete it. We thank Scott for having the patience of Job during the writing and editing of this book. Scott, ably aided by Sharon Palleschi and Melissa Fearon (Editorial Assistants) and Suzanne St. Clair (Electronic Production Manager), provided help in many ways.

We thank Professor Victor R. Basili, Series Editor, The Kluwer International Series in Software Engineering, for reviewing the book proposal and arranging a review of a draft manuscript. We thank the anonymous reviewer for helpful comments.

Our deep appreciation for her contribution to our research goes to Elizabeth D'Angelo. Her excellent work in preparing the figures, proofreading the manuscript and suggesting helpful improvements has made the presentation more consistent and understandable. With the authors she designed the graphical notation used in this book. She has cheerfully dealt with numerous revisions. We also thank Ricardo D'Angelo for his help with the figures.

We thank our colleagues, Sol Greenspan, Alex Borgida, Matthias Jarke and Joachim Schmidt for sharing their expertise and ideas over the years. We have truly benefitted from collaborating with them.

We also thank Alex Borgida for coauthoring a source paper for Chapter 14.

The thesis committee members of the first three authors have also played an important role in the development of the material presented herein. We also appreciate the helpful comments received from the reviewers (often anonymous) of the source papers of this book. Many colleagues have read drafts of our earlier papers and offered helpful literature references, and we thank them, including Kostas Kontogianis, Ken Sevcik, Stewart Green, Vinay Chaudhri, Isabel Cruz, David Lauzon, Sheila McIlraith and Will Hyslop.

We thank colleagues who have provided us with references to identify issues and examples, organized workshops, and encouraged us to write up our work. They include Stephen Fickas, Anthony Finkelstein, Martin Feather, Peri Loucopoulos, Barry Boehm, John Callahan, David Garlan and Dewayne Perry.

A number of people with domain expertise aided our case studies, by providing domain documents, reviewing our studies, offering feedback, and referring us to others. We thank Lou Melli, Dominic Covvey, Bill Polimenakos, Brian Brett, David Braidwood, Honey Robb and Patrick Daly.

Many thanks to Belinda Lobo for excellent office support, as well as preparing the legend of logos for figures, and cheerfully maintaining our files.

We also thank Niloo Hodjati for carefully proofreading several chapters, Joseph Makuch for valuable systems support, and Daniel Gross for helpful discussions.

Many people have helped us develop tools for the NFR Framework, including Ian Maione, Thomas Rose, David Lauzon, Martin Staudt, Bryan Kramer and Martin Stanley. Although tools are not detailed in this book, the help of these people permitted us to try out some of the ideas of the Framework.

We acknowledge with gratitude the financial support received from the Institute for Robotics and Intelligent Systems (IRIS), the Consortium for Software Engineering Research (CSER) and the Natural Sciences and Engineering Research Council of Canada (NSERC).

Many colleagues and friends have encouraged us to complete this book.

Finally, we thank our families for their love and support throughout the development of this book.