

**A SEMI-AUTOMATED APPROACH TO SUPPORT
LOGICAL FORMALISM
FOR
REQUIREMENTS ANALYSIS AND VALIDATION**

RICHA SHARMA



**AMAR NATH AND SHASHI KHOSLA SCHOOL OF
INFORMATION TECHNOLOGY
INDIAN INSTITUTE OF TECHNOLOGY, DELHI
AUGUST 2016**

© INDIAN INSTITUTE OF TECHNOLOGY, HAUZ KHAS, NEW DELHI

**A SEMI-AUTOMATED APPROACH TO SUPPORT
LOGICAL FORMALISM
FOR
REQUIREMENTS ANALYSIS AND VALIDATION**

by

Richa Sharma

**Submitted in fulfilment of the
Thesis requirements for the degree of
Doctor of Philosophy**

to the



**Amar Nath and Shashi Khosla School of Information Technology
Indian Institute of Technology, Delhi
August 2016**

Certificate

This is to certify that the thesis titled “**A Semi-Automated Approach to support Logical Formalism for Requirements Analysis and Validation**” being submitted by **Richa Sharma** to the Indian Institute of Technology Delhi, for the award of the degree of **Doctor of Philosophy** in Amar Nath and Shashi Khosla School of Information Technology, is a record of bona-fide research work carried out by her under my supervision. To the best of my knowledge, the work presented in this thesis has not been submitted to any other university or institute for the award of any other degree or diploma.

Prof. K.K. Biswas

Department of Computer Science and Engineering

Indian Institute of Technology Delhi

New Delhi 110 016

ACKNOWLEDGEMENTS

This thesis work would not have been possible without the support and cooperation of many people. I would like to take this opportunity to thank them all.

First of all, I would like to express my sincere and profound gratitude to my supervisor, Prof. K.K. Biswas for his invaluable guidance, continuous encouragement and wholehearted support in every stage of this research. His constant encouragement in exploring new things immensely increased my knowledge. His critical comments on my technical writing and eye for details strongly motivated me to strive hard towards excellence in completion of this thesis.

I am extremely grateful to the members of my Research Committee – Prof. Niladri Chatterjee, Prof. Saroj Kaushik and Prof. Parag Singla, who have been very helpful by offering suggestions and advice. A special mention goes to Prof. Niladri Chatterjee whose constructive reviews enabled me to give final shape to my work and thesis. I would like to thank Prof. V. Gervasi, who I happened to meet at the time of my first international presentation and who gave me the directions for future work. My cordial thanks to two professors from Computer Science and Engineering department, Prof. S. Arun Kumar and Prof. S.K. Gupta, constant moral support and encouragement enabled me to overcome disappointing phases.

I am also obliged to those who have agreed to participate in the empirical studies and annotation tasks carried out as part of this research work with a special mention for Jaspreet Bhatia. I thank her for the fruitful and interesting discussions we had while working on some parts of this thesis together. I would like to thank students from Indian Institute of Technology - Delhi, Shiv Nadar University and Bharti Vidyapeeth's College of Engineering, Delhi for enthusiastically participating in empirical studies and annotations. I would also like to thank my colleagues from Shiv Nadar University and participants from Tech Mahindra without whose support, value-addition to empirical studies would not have been possible.

Most importantly, I would like to thank my parents for instilling me with a life-long love for learning and education. My mother Raj Rani Sharma and father K.C. Sharma showed immense patience and provided me great moral support during the course of my work. I would like to make special word of thanks for my mother who stood as a pillar of strength during all the ups and downs of the research. This thesis is dedicated to my parents.

Richa Sharma

Abstract

The growing complexity and size of software systems emphasize the need for capturing the requirements in a way that is amenable to automatic requirements analysis and validation so that the defects in the delivered software can be reduced considerably. Natural Language (NL) is the most preferred form of representing requirements in software industry. However, NL cannot be subjected to automated reasoning, analysis, and validation, thereby leaving inconsistencies unnoticed. NL requirements are usually analysed through manual reviews, inspections. Requirements analysis models like structured analysis diagrams (data-flow diagrams and structure charts) or object-oriented analysis diagrams (UML models) are also often drawn manually to assist the analyst while analysing the requirements. However, being manual in nature, these industry practices of requirements analysis are effort-consuming, time intensive, and dependent on analysts' expertise and experience. Moreover, NL is inherently ambiguous allowing the possibility of different interpretation of the requirements statements other than the intent of business users.

In the absence of tool support for automated reasoning, analysis, and validation, ambiguities and inconsistencies are often overlooked in requirements specification documents expressed in the form of NL. Formal Knowledge Representation techniques have been widely researched for representing requirements, each with varying degree of success. Such Knowledge Representation (KR) techniques include mathematical representations and logical representations. These representation techniques help in identifying and resolving the ambiguity and inconsistency issues in software requirements. These defects, if not controlled during requirements analysis, may percolate down to subsequent phases of software development like design and development. However, the earlier explored requirements representations are limited by their inability to maintain consistency between conflicting requirements. Secondly, they suffer from practicality and applicability concern from software industry point of view. The business users and stakeholders providing requirements are not comfortable with the logical representation of requirements.

This thesis work is an attempt to address ambiguity and inconsistency concern in the NL requirements documents as well as to address the practicality and applicability issues in the proposed solution approach. We propose Courteous Logic as a suitable form for representing requirements that not only supports automated reasoning and inferencing but also preserves consistency between conflicting requirements. Our key contribution lies in proposing a semi-automated approach that first addresses ambiguity concern in the requirements and then, translates unambiguous NL representation of requirements to Courteous Logic representations. The proposed semi-automated approach for translation has also been found effective for generating UML (class, activity, and sequence diagrams) models automatically. The work presented in this thesis is an attempt to bridge the gap between industry practice of capturing requirements, and the formal KR techniques. We make use of the expressive strength of NL and the reasoning capabilities of formal KR techniques.

Table of Contents

List of Figures	xv
List of Tables.....	xvii
List of Abbreviations.....	xix
Chapter 1. Introduction.....	1
1.1 Motivation	3
1.2 Research Methodology	6
1.3 Research Contributions	10
1.3.1 Inconsistency Concern.....	10
1.3.2 Ambiguity Concern	11
1.3.3 From NL to CL representation	12
1.3.4 From NL to UML diagrams	14
1.4 Thesis Organization.....	14
Chapter 2. Literature Review	17
2.1 Requirements Engineering – General Work.....	17
2.2 Representation Formalism for Requirements	19
2.3 Inconsistency Concern.....	26
2.4 Ambiguity Concern	28
2.5 UML Diagrams.....	33
2.5.1 Introduction	33
2.5.2 Class Diagrams	34
2.5.3 Activity Diagrams	36
2.5.4 Sequence Diagrams	37
2.6 Summary	37

Chapter 3. Inconsistency Concern – Logical Formalism	39
3.1 Inconsistency Concern.....	40
3.2 Desirable Characteristics: Requirements Specifications	42
3.3 Logical Formalism.....	47
3.3.1 First Order Logic	48
3.3.2 Description Logic and Ontology	50
3.3.3 Non-monotonic Logic	51
3.4 Courteous Logic	55
3.5 Case-Studies	57
3.6 Observations from Case-studies	74
3.7 Discussion	77
Chapter 4. Ambiguity Concern.....	79
4.1 Types of Ambiguity.....	80
4.2 Solution Approach – Semi-supervised Learning.....	83
4.2.1 Introduction	83
4.2.2 Semi-Supervised Learning	85
4.2.3 Evaluation Metrics.....	89
4.3 Coordination Ambiguity.....	90
4.3.1 Our Approach.....	91
4.3.2 Semantic Similarity Heuristics	93
4.3.3 Evaluation Study	98
4.3.4 Results and Observations	102
4.4 Pronominal Anaphora Ambiguity	107
4.4.1 Our Approach.....	109
4.4.2 Noun-Phrase Co-reference Heuristics	110
4.4.3 Evaluation Study	112
4.4.4 Results and Observations	116

4.5	Discussion	118
Chapter 5. NL to Frame-based Structured Representation		121
5.1	Introduction	122
5.2	Classification of Functional Requirements.....	124
5.2.1	Grounded Theory.....	125
5.2.2	Study Conducted	126
5.2.3	Functional Requirements Categories.....	130
5.2.4	Discussion.....	134
5.3	Structured Representation Generation.....	136
5.3.1	Norm Analysis Patterns.....	136
5.3.2	Grammatical Knowledge Patterns	138
5.3.3	Frame-based Structured Representation.....	143
5.4	Implementing Frame-based Structured Representation Generation Module	146
5.5	Examples: GKP Identification and Frame Population.....	148
5.6	Discussion	167
Chapter 6. Frame-based Structured Representation to CL Representation		169
6.1	Structured Representation to CL Conversion.....	170
6.2	Implementing CL Expression Generation Module.....	173
6.3	Case Studies	180
6.3.1	Validation	199
6.3.2	Observations	200
6.3.3	Threats to Validity and Mitigations.....	203
6.4	Test-Case Generation	204
6.4.1	Background.....	205
6.4.2	Our Approach.....	206
6.5	Implementing Test cases Generation Module	208
6.6	Case-studies.....	210

6.6.1	Validation	214
6.6.2	Observations	216
6.6.3	Threats to Validity and Mitigations	217
6.7	Discussion	217
Chapter 7. UML Diagram Generation		221
7.1	UML Diagrams	222
7.2	Implementing UML Diagrams Generation Module	223
7.3	Class Diagrams	224
7.3.1	Automated Generation Approach	225
7.3.2	Case-Studies	227
7.3.3	Validation	233
7.4	Activity Diagrams	238
7.4.1	Automated Generation Approach	238
7.4.2	Case-Studies	240
7.4.3	Validation	244
7.5	Sequence Diagrams	245
7.5.1	Automated Generation Approach	245
7.5.2	Case-Studies	247
7.5.3	Validation	250
7.6	Discussion	252
Chapter 8. Conclusion and Future Work		255
8.1	Research Contributions	255
8.1.1	Courteous Logic Representation to address Inconsistency in Requirements	256
8.1.2	Machine Learning to address Ambiguity in Requirements	257
8.1.3	From NL to Frame-based Structured Representation of Requirements	258
8.1.4	From Structured to CL Representation of Requirements	259
8.1.5	From Structured Representation of Requirements to UML Diagrams	260

8.2	Limitations.....	262
8.3	Discussion	265
8.4	Future Work	268
	Appendix A: List of Publications	271
	Appendix B: Feature Vector – Ambiguity	273
	Appendix C: Lexical Ambiguity – ignored words	277
	Appendix D: Memos for Functional Requirements Core Categories.....	279
	Appendix E: Dependency Relations used and Frame Structures	281
E.1	Why Stanford Parser?.....	281
E.2	Dependency Relations Used.....	283
E.3	Frame Structures for Identified GKPs	286
E.4	Comparison of our Frames with Fillmore’s Case-Frames.....	289
	Appendix F: Scenarios for Generating CL Representations and Test Cases.....	291
F.1	Scenarios for Generating CL Representations.....	291
F.2	Scenarios for Generating Test Cases	298
	Appendix G: Scenarios for UML Diagrams Validation Study.....	301
	REFERENCES	311

List of Figures

Figure 1.1: RE Activities	2
Figure 1.2 - Architectural framework of our Approach	13
Figure 4.1: 10-Folds Cross-validation in Weka.....	87
Figure 4.2: Supplying Test dataset in Weka.....	88
Figure 4.3: Illustration: Word-based Similarity Heuristic Computations	95
Algorithm 4.1: Final Labels Decision for correspondence between NP and Pronoun	114
Figure 5.1 NL to CL Translation Approach	123
Figure 5.2 Steps in GT Study	127
Algorithm – 5.1: GKP identification in NL Requirement Statement	142
Figure 5.3: NL Statement-Categorization	144
Algorithm – 5.2: Generating Frame-based Structured Representation	145
Algorithm – 6.1: Frame-based Structured Representation to CL conversion	170
Algorithm – 6.2: Test-case Generation from CL Representation of Requirements	207
Figure 7.1: UML Diagrams – Hierarchical Representation.....	223
Algorithm 7.1: Class Diagram Generation	225
Figure 7.2: Class Diagram – Library Scenario	230
Figure 7.3: Class Diagram – Library Scenario by Callan.....	231
Figure 7.4: Class Diagram – Library Scenario by Harmain and Gaizauskas	232
Algorithm 7.2: Activity Diagram Generation.....	239
Figure 7.5: Activity Diagram – Placement Application Scenario	242
Figure 7.6: Editing sequence of activities for Activity Diagram.....	243
Figure 7.7: Activity Diagram – Academic Registration Process.....	243
Algorithm 7.3: Sequence Diagram Generation	246
Figure 7.8: Sequence Diagram – ATM Transaction Process	249
Figure 7.9: Sequence Diagram – Railway Booking Scenario	250

List of Tables

Table 4.1: Training and Test Datasets – Coordination Ambiguity.....	100
Table 4.2: Identifying nocuous and innocuous labels – Coordination Ambiguity	101
Table 4.3: Unsupervised Clustering - Coordination Ambiguity.....	102
Table 4.4: Classification Results using all Similarity Heuristics: Training Dataset.....	103
Table 4.5: Classification Results using combination of Similarity Heuristics: Training Dataset .	103
Table 4.6: Classification Results with Distributional Similarity Heuristics: Training Dataset ...	104
Table 4.7: Classification Results with Test Dataset	105
Table 4.8: Classification Results with combination of Similarity Heuristics : Test Dataset.....	105
Table 4.9: Classification Results with Distributional Similarity Heuristics : Test Dataset.....	105
Table 4.10: Naïve Bayes Classification Results with Domain-specific Test Dataset	106
Table 4.11: Classification with combination of Similarity Heuristics: Domain-specific Test Set	106
Table 4.12: Classification using Distributional Similarity Heuristics: Domain-specific Test Set	107
Table 4.13: Identifying potential antecedent labels – RS6.....	114
Table 4.14: Identifying potential antecedent labels – RS7	115
Table 4.15: Distribution of noun-phrase candidacy for pronominal anaphora.....	115
Table 4.16: Unsupervised Clustering on pronominal anaphora dataset	116
Table 4.17: Classification Results - Pronominal Anaphora Ambiguity without NP Coreference	117
Table 4.18: Classification Results - Pronominal Anaphora Ambiguity with NP Coreference.....	118
Table 5.1: Requirements Corpus Details.....	127
Table 5.2: Open coding and Emergent Core Categories	130
Table 5.3: Grammatical Knowledge Patterns.....	141
Table 5.4: Frame Structure – Active Voice.....	144
Table 5.5: Frame Structure – Preposition.....	144
Table 5.6: Frame Structure – Relative Clause.....	145
Table 5.7: Frame Structure – RS11	149
Table 5.8: Frame Structure – RS11a	150
Table 5.9: Frame Structure – RS11b	151
Table 5.10: Frame Structure – RS11d.....	152
Table 5.11: Frame Structure – RS11e	153
Table 5.12: Frame Structure – RS11f.....	154
Table 5.13: Frame Structure – RS12	155

Table 5.14: Frame Structure – RS12a	157
Table 5.15: Frame Structure – RS12d	159
Table 5.16: Frame Structure – RS12e	160
Table 5.17: Frame Structure – RS12f.....	161
Table 5.18: Frame Structure – RS12g.....	162
Table 5.19: Execution-time Summary for RS11 and RS12.....	163
Table 5.20: Frame Structure – RS11-v.....	165
Table 5.21: Frame Structure – SS11.....	166
Table 6.1: Frame Structure – RS1.1	183
Table 6.2: Frame Structure – RS1.2	184
Table 6.3: Frame Structure – RS1.3	185
Table 6.4: Frame Structure – RS1.4	186
Table 6.5: Frame Structure – RS1.7	188
Table 6.6: Frame Structure – IRC ₂	196
Table 6.7: Frame Structure – OM ₁	196
Table 6.8: Summary of Responses – First Phase of CL Validation	201
Table 6.9: Summary of Responses to Questionnaire: Second Phase of CL Validation	203
Table 6.10: Test Cases – RS11.....	209
Table 6.11: Test Cases – RS12.....	210
Table 6.12: Test Cases – Scenario - 1	211
Table 6.13: Test Cases – Scenario - 2	212
Table 6.14: Summary of System-Generated Test-cases.....	214
Table 6.15: Summary of Test-cases reported by Subjects.....	215
Table 6.16: Summary of Responses to survey (Q2).....	216
Table 7.1: Frame Representation for RS1 – Library Scenario	228
Table 7.2: Frame Representation for RS2 - Library Scenario.....	228
Table 7.3: Class-Attributes : Library Case-study	230
Table 7.4: Inheritance : Library Case-study	231
Table 7.5: Results – Class Diagram Case Studies	236
Table 7.6: Frame Representation for RS3 – Activity Diagram Scenario	241
Table 7.7: Frame Representation for RS4 - Activity Diagram Scenario.....	241
Table 7.8: Summary of Responses: Activity Diagrams Validation.....	245
Table 7.9: Frame Representation for RS5	248
Table 7.10: Frame Representation for RS6	248
Table 7.11: Summary of Responses: Sequence Diagrams Validation	251

List of Abbreviations

AI	-	Artificial Intelligence
BNC	-	Brown National Corpus
CL	-	Courteous Logic
DL	-	Description Logic
FOL	-	First Order Logic
GKP	-	Grammatical Knowledge Patterns
GT	-	Grounded Theory
KR	-	Knowledge Representation
ML	-	Machine Learning
NL	-	Natural Language
NLP	-	Natural Language Processing
NML	-	Non-monotonic Logic
OOA	-	Object oriented Analysis
RE	-	Requirements Engineering
SE	-	Software Engineering
SDLC	-	Software Development Life Cycle
SSL	-	Semi-supervised Learning
UML	-	Unified Modelling Language