SEMI-AUTOMATED DEVELOPMENT OF CONCEPTUAL MODELS FROM NATURAL LANGUAGE TEXT

MUSSA AHMED MOHAMMED OMAR

A thesis submitted to the University of Huddersfield in partial fulfilment of the requirements for the degree of Doctor of Philosophy

The University of Huddersfield
May 2018

Abstract

The process of converting natural language specifications into conceptual models requires detailed analysis of natural language text, and designers frequently make mistakes when undertaking this transformation manually. Although many approaches have been used to help designers translate natural language text into conceptual models, each approach has its limitations. One of the main limitations is the lack of a domain-independent ontology that can be used as a repository for entities and relationships, thus guiding the transition from natural language processing into a conceptual model. Such an ontology is not currently available because it would be very difficult and time consuming to produce. In this thesis, a semi-automated system for mapping natural language text into conceptual models is proposed. The model, which is called SACMES, combines a linguistic approach with an ontological approach and human intervention to achieve the task. The model learns from the natural language specifications that it processes, and stores the information that is learnt in a conceptual model ontology and a user history knowledge database. It then uses the stored information to improve performance and reduce the need for human intervention. The evaluation conducted on SACMES demonstrates that (1) designers' creation of conceptual models is improved when using the system comparing with not using any system, and that (2) the performance of the system is improved by processing more natural language requirements, and thus, the need for human intervention has decreased. However, these advantages may be improved further through development of the learning and retrieval techniques used by the system.

Table of Contents

Abstract 3
Table of Contents 4
List of Tables 7
List of Figures 9
Acknowledgements 14
List of Abbreviations 15
List of Publications 16
Chapter 1: Introduction and Motivation 17
1.1 Motivation and Statement of Problem 17
1.2 Research Aim 21
1.3 Research Objectives 21
1.4 Methodology 22
1.5 Bibliographical Preparation 22
1.6 Research Contribution 23
1.7 Thesis Contents 24
Chapter 2: Background and Literature Review 26
2.1 Conceptual Models 26
2.1.1 Problems in Natural Language Text 27
2.1.2 Problems Facing Designers during Conceptual Model Creation 30
2.2 Approaches for Extracting Conceptual Models from Natural Language Text 32
2.2.1 Linguistics-based Approach 32
2.2.1.1 Tools and systems based on a linguistic approach 33
2.2.1.2 Advantages and disadvantages of a linguistic approach 35
2.2.2 Pattern-based Approach 35
2.2.2.1 Tools and systems based on patterns approach 36
2.2.2.2 Advantages and disadvantages of a patterns approach 37
2.2.3 Case-based Approaches 37
2.2.4 Ontology-based Approach 37
2.2.4.1 Tools for using ontologies in conceptual models 40

2.2.4.2 Advantages and disadvantages of an ontology-based approach 41
2.2.5 Multiple Approaches 42
2.3 Natural Language Processing (NLP) 45
2.3.1 NLP Toolkits 49
2.4 Ontologies Overview 51
2.4.1 Ontology Types (Lightweight Ontologies and Formal Ontologies) 51
2.4.2 Methods for Creating Ontologies 52
2.4.2.1 Manual ontology creation 52
2.4.2.2 Ontology learning from text (semi-automated ontology creation) 55
2.4.2.2.1. Examples of ontology learning systems 57
2.4.2.2.2. Techniques Used for Ontology Learning from Text 57
2.4.3 Data Set Ontologies 61
2.4.4 Ontology Languages 70
2.5 Chapter Summary 72
Chapter 3: Rules to Derive a Conceptual Model from Natural Language Text 75
3.1 Rules to Determine Entities 75
3.2 Approach Applied for Entity Extraction 79
3.3 Rules to Determine Relationships between Entities 79
3.4 Approach Applied for Relationship Extraction 80
3.5 Rules to Determine Attributes 81
3.6 Chapter Summary 82
Chapter 4: Implementation of Semi-Automated Conceptual Model Extraction System (SACMES) 84
4.1 System Architecture 84
4.1.1 Pre-Processing Stage 88
4.1.2 Entities Identification Stage 90
4.1.3 Relationships Identification Stage 92
4.1.3.1 Identifying relationships from requirement specification text using Stanford typed dependencies 93
4.1.3.2 Identification of relationships from entities 95
4.2.3.3 Human intervention 97
4.2 Step-by-Step Case Study 97

4.3 Chapter Summary 112
Chapter 5: Empirical Evaluation of SACMES 113
5.1 Experimental Design One 113
5.1.1 First Group Results 116
5.1.1.1 Entities extraction 116
5.1.1.2 Relationships extraction 122
5.1.1.3 Cardinalities extraction 125
5.1.2 Second Group Results 131
5.1.2.1 Entities extraction 132
5.1.2.2 Relationships extraction 134
5.1.2.3 Cardinalities extraction 137
5.2 Experimental Design Two 139
5.2.1 Results 142
Chapter 6: Conclusion and Future Work 158
6.1 Conclusion 158
6.2 Limitations and Future Work 162
List of Appendices 165
Appendix 1: 165
Appendix 2: 170
Appendix 3: 194
Appendix 4: 196
Appendix 5: 215
References 224

List of Tables

Table 2.1 Comparison between Approaches Used for Extracting Conceptual Models from Natural Language Specifications 44

Table 2.2 Tokens of a Sentence 46

Table 2.3 Sentence Splitter Divides Text into Sentences 46

Table 2.4 PoS Tags in the Penn Treebank Project (Santorini, 1990) 47

Table 2.5 Example of NER 48

Table 2.6 Sentence Dependency Example 48

Table 2.7 Tasks and Languages Supported by Stanford CoreNLP (Manning et al., 2014) 50

Table 3.1 PoS Tagging for a Sentence 76

Table 3.2 Stanford Parser Defines Common Nouns and Proper Nouns 77

Table 4.1 Noun Phrases Defined by Stanford PoS from Company Database Scenario 100

Table 4.2 Entities List Defined from Company Database Scenario after Filtration 101

Table 4.3 Binary Relationships between Entities 104

Table 5.1 Subjects' Activities in the Experiment 115

Table 5.2 Comparison between System Answer and Manual Answer based on Model Answer for Company Database in Harder Problems Set 118

Table 5.3 Comparasion between System Answers and Manual Answers for Entities Extraction based on Model Answers 121

Table 5.4 Comparing Relationships Found in System Answer and Handcrafted Answer based on Model Answer for Company Database Case Study 122

Table 5.5 Comparison between System Answers and Manual Answers for Relationship Extraction based on Model Answers 124

Table 5.6 Comparing Relationship Cardinalities Found in System Answer and Manual Answer based on Relationship Cardinalities Found in Model Answer for Company Database Scenario 126

Table 5.7 Comparison between System Answers and Manual Answers for Cardinalities of Relationships Extraction based on Model Answers 128

Table 5.8 Comparison between System Answers and Handcrafted Answers for Entities Extraction based on Key Answers 134

Table 5.9 Comparison between System Answers and Handcrafted Answers for Relationships Extraction based on Model Answers 136

Table 5.10 Comparison between System Answers and Handcrafted Answers for Cardinalities Extraction based on Model Answers 139

Table 5.11 Summary of Results Obtained for Test Set from KBCMES before Training........... 143

Table 5.12 Summary of Results Obtained for Test Set from KBCMES after Training 146

Table 5.13 Comparison of Results for Extraction of Unrecognised Entities, Unrecognised Relationships, Entities, Relationships and Cardinalities from Test Set by KBCMES before and after Training 148

Table 5.14 Relationship between Unrecognised Entities Extraction and Count of Case Studies on which System is Trained 149

Table 5.15 Relationship between Unrecognised Relationships Extraction and Count of Case Studies on which System is Trained 150

Table 5.16 Relationship between Entities Extraction and Count of Case Studies on which System is Trained... 151

Table 5.17 Relationship between Relationships Extraction and Count of Case Studies on which System is Trained 153

Table 5.18 Relationship between Cardinalities of Relationships Extraction and Count of Case Studies on which System is Trained 155

List of Figures

Figure 2.1 A Type Hierarchy for a Physical Object (Meziane, 1994, p. 66) 28

Figure 2.2 Proportional Relationship between Entities Count and Relationships Count (Thonggoom, 2011, p. 20)..30

Figure 2.3 Lightweight and Heavyweight Ontologies (Giunchiglia & Zaihrayeu, 2009) 52

Figure 2.4 Ontology Learning: Output, Tasks and Techniques (Wong, 2009, p.15)....................55

Figure 2.5 Hypernym Chain for ‘Doctor’ in WordNet 62

Figure 2.6 Hypernym Chain for ‘Size’ in WordNet...63

Figure 2.7 Hypernym Chain for ‘Treatment’ in WordNet 64

Figure 2.8 SUMO Upper Level Hierarchy (Niles & Pease, 2001) 65

Figure 2.9 Relations between Doctor and Patient in TextRunner Ontology.................................68

Figure 2.10 Relations between Programmer and Programming Language in TextRunner Ontology...69

Figure 2.11 Web-Based Ontology Languages (Corcho et al., 2003) 71

Figure 4.1 SACMES Architecture 84

Figure 4.2 OCM Hierarchy and UHKB Database..86

Figure 4.3 Flow Chart of Pre-Processing Stage 88

Figure 4.4 Flow Chart of Entities Identification Stage 90

Figure 4.5 Flow Chart of Relationships Identification Stage...92

Figure 4.6 A Company Database (Du, 2008, p. 170)...97

Figure 4.7 Attachment of Requirement Specification Text into SACMES 98

Figure 4.8 SACMES Displays the RST to the User...99

Figure 4.9 Human Intervention for Entities Identification Stage 102

Figure 4.10 Human Intervention for Defining Relationships 106

Figure 4.11 Defining Names and Cardinality for Relationships...107

Figure 4.12 Review and Revision Form..108

Figure 4.13 Report Displaying Information for the Conceptual Model.....................................109

Figure 4.14 Ontology Hierarchy before and after Processing the..110

Figure 4.15 Entities and Relationships History before Processing Company Database 110

Figure 4.16 Entities and Relationships History after Processing Company Database 111

Figure 5.1 Company Database (Du, 2008, p. 170) 116

Figure 5.2 Model Answer for Company Database 117

Figure 5.3 Handcrafted Answer for Company Database 117

Figure 5.4 System Answer for Company Database 118

Figure 5.5 Screenshot of Entities Hierarchy and Relationships Hierarchy before the Experiment 129

Figure 5.6 Screenshot of Part of Entities Hierarchy and Relationships Hierarchy after the Experiment 130

Figure 5.7 Screenshot of the UHKB Database before the Experiment 131

Figure 5.8 Screenshot of UHKB Database Relationships Table after the Experiment 131

Figure 5.9 System Architecture for KBCMES 141

Figure 5.10 Relationship between Count of Case Studies on which the System is Trained and Average Recall and Precision in Defining Unrecognised Entities 150

Figure 5.11 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Unrecognised Relationships 151

Figure 5.12 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Entities 153

Figure 5.13 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Relationships 154

Figure 5.14 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Cardinalities of Relationships 156

Appendix Figure 1 Problem One in Easy Set (Du, 2008, p.169) 170

Appendix Figure 2 Model Answer for Problem One in Easy Set provided by Database Designer 170

Appendix Figure 3 Problem Two in Easy Set (Du, 2008, p.172) 171

Appendix Figure 4 Solution for Problem Two in Easy Set provided by Database Designer 171

Appendix Figure 5 Problem Three in Easy Set (Du, 2008, p. 167) 172

Appendix Figure 6 Solution for Problem Three in Easy Set provided by Database Designer 172

Appendix Figure 7 Problem Four in Easy Set (Du, 2008, p. 167) 173

Appendix Figure 8 Solution for Problem Four in Easy Set provided by Database Designer 173

Appendix Figure 9 Problem Five in Easy Set (Du, 2008, p. 167) 174

Appendix Figure 10 Solution for Problem Five in Easy Set provided by Database Designer 174

Appendix Figure 11 Problem Six in Easy Set (Du, 2008, p. 168) 175

Appendix Figure 12 Solution for Problem Six in Easy Set provided by Database Designer 175
Appendix Figure 13 Problem Seven in Easy Set (Zhang, 2012, p. 34) 176
Appendix Figure 14 Solution for Problem Seven in Easy Set (Zhang, 2012) 176
Appendix Figure 15 Problem Eight in Easy Set (Du, 2008, p. 168) 177
Appendix Figure 16 Solution for Problem Eight in Easy Set provided by Database Designer .. 177
Appendix Figure 17 Problem Nine in Easy Set (Du, 2008, p. 169) 178
Appendix Figure 18 Solution for Problem Nine in Easy Set provided by Database Designer ... 178
Appendix Figure 19 Problem Ten in Easy Set (Connolly & Begg, 2015, p. 431) 179
Appendix Figure 20 Solution for Problem Ten in Easy Set 179
Appendix Figure 21 Problem One in Harder Set (Du, 2008, p. 170) 180
Appendix Figure 22 Solution for Problem One in Harder Set provided by Database Designer. 180
Appendix Figure 23 Problem Two in Harder Set (Du, 2008, p. 98) 181
Appendix Figure 24 Solution for Problem Two in Harder Set provided by Database Designer 181
Appendix Figure 25 Problem Three in Harder Set (Du, 2008, p.172) 182
Appendix Figure 26 Solution for Problem Three in Easy Set 182
Appendix Figure 27 Problem Four in Harder Set (Atzeni, 1999, p. 213) 183
Appendix Figure 28 Solution for Problem Four in Harder Set provided by Database Designer 183
Appendix Figure 29 Problem Five in Harder Set (Gehrke, 2002, p. 8) 184
Appendix Figure 30 Solution for Problem Five in Harder Set 184
Appendix Figure 31 Problem Six in Harder Set (Teorey, Lightstone, Nadeau, & Jagadish, 2005, p. 131) 185
Appendix Figure 32 Solution for Problem Six in Harder Set (Teorey et al., 2005, p. 133) 185
Appendix Figure 33 Problem Seven in Harder Set (Connolly & Begg, 2015, p. B-6) 186
Appendix Figure 34 Solution for Problem Seven in Harder Set 187
Appendix Figure 35 Problem Eight in Harder Set (Zhang, 2012, p. 8) 188
Appendix Figure 36 Solution for Problem Eight in Harder Set (Zhang, 2012, p. 10) 189
Appendix Figure 37 Problem Nine in Harder Set (Zhang, 2012, p. 34) 190
Appendix Figure 38 Solution for Problem Nine in Harder Set (Zhang, 2012, p. 35) 191
Appendix Figure 39 Problem Ten in Harder Set (Thonggoom, 2011, p. 132) 192
Appendix Figure 40 Solution for Problem Ten in Harder Set provided by Database Designer . 193

Appendix Figure 41 Electronic Commerce Case Study (Pol & Ahuja, 2007, p. 73)..................196

Appendix Figure 42 Intercollegiate Football Championship Case Study (Pol & Ahuja, 2007, p. 74)...197

Appendix Figure 43 JobSearch Case Study (Pol & Ahuja, 2007, p. 75) 197

Appendix Figure 44 Course Timetable Case study (Pol & Ahuja, 2007, p. 74).........................198

Appendix Figure 45 Ford Distribution Centres Case study (Pol & Ahuja, 2007, p. 73) 198

Appendix Figure 46 Miami Hotel Case Study (Pol & Ahuja, 2007, p. 73) 199

Appendix Figure 47 Newark Divisional Office Case Study (Pol & Ahuja, 2007, p. 73) 199

Appendix Figure 48 Savannah's Family Farms Case Study (Pol & Ahuja, 2007, p. 71) 199

Appendix Figure 49 Florida Bus Traveling Agency Case Study (Pol & Ahuja, 2007, p.75).....200

Appendix Figure 50 GERU Company Case Study (Pol & Ahuja, 2007, p. 76) 200

Appendix Figure 51 SunRise Hotel Case study (Pol & Ahuja, 2007, p. 76) 201

Appendix Figure 52 University Housing Office Case Study (Pol & Ahuja, 2007, p. 74)..........201

Appendix Figure 53 Bookstore Case Study (Pol & Ahuja, 2007, p. 77) 202

Appendix Figure 54 Medicare Case study (Pol & Ahuja, 2007, p. 77) 202

Appendix Figure 55 Memorabilia Company Case Study (Pol & Ahuja, 2007, p. 76) 203

Appendix Figure 56 Wood Paneling Manufacturers Case study (Pol & Ahuja, 2007, p.78).....203

Appendix Figure 57 AACSB Case Study (Pol & Ahuja, 2007, p. 79) 204

Appendix Figure 58 University Database Case Study (Pol & Ahuja, 2007, p. 81) 204

Appendix Figure 59 National Car Rental Case Study (Pol & Ahuja, 2007, p. 81).....................204

Appendix Figure 60 USTA Case Study (Pol & Ahuja, 2007, p. 79) 205

Appendix Figure 61 Blood Bank Case Study (Pol & Ahuja, 2007, p. 82) 205

Appendix Figure 62 Company Wide Database Case Study (Teorey et al., 2005, p. 64)...........206

Appendix Figure 63 Medical School Case Study (Pol & Ahuja, 2007, p. 81) 206

Appendix Figure 64 YXZ Company Case Study (Pol & Ahuja, 2007, p. 82)...........................207

Appendix Figure 65 ABC Ltd Case Study Needs Page (Carter, 2003, p. 39) 207

Appendix Figure 66 Company Database Case Study (Rob & Coronel, 2009, p. 142)..............207

Appendix Figure 67 Publishers Database Case Study (Teorey, 1999, p. 76) 208

Appendix Figure 68 Wellmeadows Hospital Case Study Part One (Connolly & Begg, 2015, p. B-5)...209

Appendix Figure 69 Wellmeadows Hospital Case Study Part 2 (Connolly & Begg, 2015, p. B-5) 210

Appendix Figure 70 Wellmeadows Hospital Case Study Part 3 (Connolly & Begg, 2015, p. B-5) 211

Appendix Figure 71 Conference Review Database Case Study (Elmasri & Navathe, 2017, p. 134)....... 211

Appendix Figure 72 DVD Database Case Study (Connolly & Begg, 2015, p. 431) 212

Appendix Figure 73 Movie Database Case Study (Elmasri & Navathe, 2017, p. 132) 212

Appendix Figure 74 University Accommodation Office Case Study Part One (Connolly & Begg, 2015, p. B-1)....... 213

Appendix Figure 75 University Accommodation Office Case Study Part 2 (Connolly & Begg, 2015, p. B-1)....... 214

Appendix Figure 76 Votes Database Case Study (Elmasri & Navathe, 2017, p. 127) 214

Appendix Figure 77 Veterinary Hospital Case Study (Pol & Ahuja, 2007, p. 76) 215

Appendix Figure 78 Model Answer for Veterinary Hospital provided by Database Designer .. 215

Appendix Figure 79 DreamHome Case Study (Connolly & Begg, 2015, p. A-1)....... 216

Appendix Figure 80 Model Answer for DreamHome Case Study 217

Appendix Figure 81 Airline Case Study (Pol & Ahuja, 2007, p. 74) 218

Appendix Figure 82 A Model Answer for Airlines Case Study Provided by Database Designer 219

Appendix Figure 83 Florida Mall Case Study (Bagui & Earp, 2012, pp. 96-99) 220

Appendix Figure 84 Model Answer for Florida Mall Case Study....... 221

Appendix Figure 85 Coca Cola Case Study (Pol & Ahuja, 2007, p. 71)....... 222

Appendix Figure 86 Model Answer for Coca Cola Case Study provided by Database Designer 223

Acknowledgements

First, thanks to Allah, who has helped me to accomplish the work. I would like to thank my main supervisor, Dr David Wilson, without whose support this work could not have been completed. I acknowledge my mother, father and grandfather for their supplications to Allah for me to succeed. I am also grateful to my wife Hanan for her patience, and for creating an appropriate atmosphere during the study period. Last but not least, the most sincere thanks are due to the Ministry of Higher Education in my home country of Libya for their sponsorship of my studies by covering my tuition fees and living expenses.

List of Abbreviations

API	Application Programming Interface
CM	Conceptual Model
CMO	Conceptual Model Ontology
EHKB	Entities History Knowledge Base
ERD	Entity Relationship Diagram
KBSACMES	Knowledge-Based Semi-Automated Conceptual Model Extraction System
NER	Name Entities Recognition
NLP	Natural Language Processing
PoS	Part-of-Speech
RHKB	Relationships History Knowledge Base
RST	Requirement Specification Text
SACMES	Semi-Automated Conceptual Model Extraction System
UHKB	User History Knowledge Base
UML	Unified Modelling Language

List of Publications

1. Omer, M & Wilson, D (2015). Implementing a Database from a Requirement Specification. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 9(1), 33-41.

2. Omer, M. & Wilson, D. (2016). New Rules for Deriving Formal Models from Text. In *International Conference for Students on Applied Engineering, Newcastle, UK*. IEEE Xplore Digital Library, 328-333.

3. Omer, M. & Wilson. D. (2016). Deriving a Relational Database from Plain Text Using Predefined Patterns of Text and a Knowledge Environment Background Database. In *the International Conference on Human Computer Interaction and Artificial Intelligence (ICHCIAI), Manchester, UK.*

Chapter 1: Introduction and Motivation

1.1 Motivation and Statement of Problem

Conceptual model development is the most important stage in the design of a system and database. The conceptual model provides a blueprint for the system and database, explaining the system's functions and structure (Thalheim, 2000). To be considered a qualified conceptual model, it must have the ability to reflect the real world environment (Dullea, Song, & Lamprou, 2003). Furthermore, any errors in the conceptual model will be costly to fix during implementation (Thonggoom, 2011), so correcting errors during the early stages of developing the model is considerably cheaper than correcting them at a later stage (Boehm, 1981).

Natural language is used as the main tool to describe the requirement specifications of systems. People usually use natural language text to describe things in the real world and therefore, most requirement specifications in industry are written in natural language (Neill & Laplante, 2003; Luisa, Mariangela, & Pierluigi, 2004). However, there are as many as eighty different conceptual model notations that can be used to describe requirement specifications (Thalheim, 2000). Among these, the Entity Relationship Diagram (ERD) and Unified Modelling Language (UML) are the most commonly used in practice (Neill & Laplante, 2003). The ERD, proposed by Chen in 1976, is widely used to describe conceptual models for database design because it is easy to understand and capable of modelling real world problems (Chen, 1976).

Despite its importance, however, it is very difficult to design a well-made conceptual model (Thonggoom, 2011) as the process can face many problems, as described below.

1. Complex relationships between concepts: A conceptual model should represent all relationships between concepts in a specific domain. Novice designers frequently make errors in producing complex relationships between concepts (Topi, 2002), and even in producing simple binary relationships (Batra, 2007). An increasing number of entities leads to an increasing number of relationships (Batra, 2007), and as the number of relationships increases, the possibility of missing these relationships can also increase for both expert and novice designers.

2. **Incomplete natural language rules for conceptual model extraction**: Linguistic rules for mapping natural language text into a conceptual model are not complete, and applying such rules in an inappropriate way can lead to errors (Parsons & Saunders, 2004). There may also be conflicts between rules. For example, a noun can represent an entity but may also represent an attribute. Furthermore, applying many of these rules together within a tool is a very complex task.

3. Complex semantic relationships in natural language text: Mapping each relationship from a natural language description into a relationship in a database may lead to problems (Batra, 2007). One such problem is that incorrect relationships are added. For example, in the sentence 'The company is divided into departments', which is part of a problem description for a company database, if the relationship mentioned in the sentence is mapped into a relationship in the conceptual model, a relationship of one-to-many between ‘company’ and ‘department’ is created. However, from the scenario it is clear that there is just one company, and there is no need to add the company as an entity in the conceptual model. Equally, there may be relationships required by the database that have not been explained in the natural language description.

4. Novice designers’ lack of domain knowledge and experience: Expert designers are clearly more capable and skilled than novice designers at translating natural language specifications into conceptual models, as they can use knowledge from previous experience they have gained (Kim, Lee, & Moon, 2008). However, even expert designers may fail to build a good conceptual model if they have an incomplete requirement specifications text.

5. Different solutions for the same problem: One of the main issues in translating natural language specifications into conceptual models is the availability of more than one solution (Moody & Shanks, 1994). Various alternative solutions may be correct. For example, in a sentence such as ‘A student has a department, a name and an address’, one solution would be to consider ‘a student’ and ‘a department’ as entities, with a relationship of one to many between them, in addition to considering ‘a name’ and ‘an address’ as attributes of the student entity. Another solution, however, would be to consider that ‘a student’, ‘a department’ and ‘an address’ are entities, with a relationship of one to many between ‘a department’ and ‘a student’ and a relationship of one to many between ‘a student’ and ‘an address’.

6. Natural language specification problems: The fact that requirement specifications are written in natural language text can lead to many issues. These issues include noise, silence, overspecification, contradiction, forward reference and wishful thinking. In addition, the greatest problem linked with the use of natural language text to describe requirement specifications is ambiguity. Ambiguity is the occurrence in the text of an element that allows a feature of the problem to be understood in at least two different ways. Ambiguity in natural language text is divided into three types. These types are (1) lexicographic ambiguity, which occurs when a word in English has more than one meaning; (2) grammatical ambiguity, which occurs when a sentence can be parsed in several different ways; and (3), textual cohesion, which means all parts of the text should be linked properly with a smooth transition from one idea to another (Meziane, 1994).

Because of the difficulties faced by designers, especially novice designers, in the creation of conceptual models, technologies have become involved in conceptual model creation, as well as in mapping from conceptual models to logical or physical models. There are many commercial graphical CASE tools which can be used to automatically convert a conceptual model into a logical or physical model (Thonggoom, 2011). However, there is no commercial or non-commercial tool which can automatically convert natural language text into a conceptual model (Song, Zhu, Ceong, & Thonggoom, 2015; Šuman, Jakupović, & Kuljanac, 2016; Thonggoom, 2011). Instead, various semi-automated approaches are used for this purpose, which include the following (Thonggoom, 2011).

1. Linguistics-based approach: The linguistics-based approach uses natural language techniques and rules to translate natural language descriptions into conceptual models. Chen (1983), suggested eleven rules for mapping requirement specification text into an Entity Relationship Diagram (ERD). Chen's work was followed by other studies, such as those by Hartmann and Link (2007), Omar, Hanna and McKevitt (2004) and Overmyer, Lavoie and Rambow (2001), to use, enhance and extend Chen's rules, but the rules are still incomplete, inaccurate and overlapped. These rules can service only the basic requirements of the process of translating natural language into a conceptual model. The strength of the linguistic approach is that it is domain independent; the disadvantage is that it does not have a knowledge base (Song et al., 2015).

2. Pattern-based approach: In his book on architecture and urban planning, Christopher (1979) explained the importance of using patterns in designing. His idea was that designers should use patterns instead of trying to solve design problems from scratch. In the same way, patterns are suggested as a means of reusing solutions to recurrent problems in software development, and reuse of patterns can bring many benefits to this context, including improvement of quality and saving of time and money (Hoffer, Prescott, & McFadden, 2004). The approach takes advantage of previous conceptual model designs and reuses them. A repository of case studies is stored and used as a knowledge base to help in creating conceptual models from requirement specification text. Choobineh and Lo (2004), Paek, Seo and Kim (1996) and Storey, Chiang, Dey, Goldstein and Sudaresan (1997) all provide examples of using this technique. However, the practice of using patterns in the creation of conceptual models is a challenge, since creating a pattern repository is difficult and requires extensive time and effort. In addition, most of the proposed tools for using patterns in conceptual model development are built manually, and the manual building of such tools requires time and domain knowledge (Song et al., 2015).

3. Case-based approach: For developing knowledge-based systems, a technology called case-based reasoning can be used. Case-based reasoning works by finding a solution for a new problem by retrieving a solution to a similar problem and adapting it into a suitable solution for the new problem. However, only a limited number of researchers have used the case-based approach. Although the approach benefits from reusing previous designs, its main disadvantage is that developing conceptual model libraries is extremely costly (Thonggoom, 2011).

4. Ontology-based approach: The use of ontologies has become widespread in fields such as information systems, databases and natural language processing. Artificial intelligence researchers have taken the word ontology from philosophy, and the term has come to be used in various different scientific domains (Roussey, Pinet, Kang, & Corcho, 2011). An ontology can be used in solving problems of semantic relationships in information systems (El-Ghalayini, Odeh, & McClatchey, 2006). The main gain of using ontologies in the creation of conceptual models is the possibility of reusing real-world relationships in the upper level or domain level. Sugumaran and Storey (2006) offer an example of using an ontology in the extraction of entity relationship models from natural language descriptions. The main disadvantage is the difficulty of the approach, in that extensive time and effort are needed for ontology development (Song et al., 2015).

5. Multiple approaches: As there is no perfect approach for extracting conceptual models from requirement specifications, Song, Yano, Trujillo and Luján-Mora (2004), and Thonggoom (2011) suggest using more than one approach to tackle the limitations of each individual approach. However, in the author's view it is necessary to integrate different approaches in a specific way in order to tackle these limitations.

The linguistics-based approach services the basic requirements for extracting conceptual models from natural language text, but it cannot stand by itself because it is not capable of solving ambiguity issues in natural language text and because it does not include a knowledge base. Therefore, using the linguistics-based approach in combination with other approaches may give a better result. The pattern-based, case-based and ontology-based approaches are all applied to take advantage of reusing information from previous designs. In particular, ontologies are widely employed in reusing data, and this approach also provides a good set of components which can represent information about the knowledge base in an appropriate way. These components include terms, concepts, relationships and axioms. A combination of linguistics and ontology-based approaches should be able to produce a powerful application for extracting conceptual models from natural language text. However, because of the likelihood of ambiguity in natural language, this combination will also need a minimum degree of human intervention to resolve such issues in requirement specification texts.

This thesis therefore suggests the use of a multiple approach to build a semi-automated model for extracting conceptual models from natural language specifications. The proposed model will integrate natural language processing tools and ontologies to produce conceptual models from natural language text. The model will learn from the natural language texts that it processes and store what has been learnt in its knowledge base in order to update it. The information stored in the knowledge base will help the model to minimise human intervention and to improve its performance.

1.2 Research Aim

The aim of the research is to improve the creation of conceptual models from natural language text by developing a tool that can help designers in this process.

1.3 Research Objectives

The objectives of the research are as follows:

1. To explore and analyse the approaches that are currently used for extracting conceptual models from natural language text, to examine their strengths and weaknesses, and to identify the features that could be integrated in a new tool (see Chapter Two).

2. To examine the natural language rules that are used in mapping natural language requirements into a conceptual model, to identify their strengths and weaknesses, and to determine which rules will be suitable for use (see Chapter Three).

3. To design a semi-automated, domain-independent methodology that attempts to tackle the limitations of current methodologies (see Chapter Four).

4. To implement a prototype for the methodology (see Chapter Four).

5. To conduct an empirical evaluation of the methodology using the prototype to ascertain the effectiveness of the implemented tool (see Chapter Five).

1.4 Methodology

1. To achieve objective number one, a literature review is conducted to identify, examine and analyse approaches used to map natural language specifications into conceptual models. Having identified the knowledge gap, the author then proposes a model to fill this gap, which combines natural language processing, ontology, linguistics rules and human intervention. The author reviews natural language processing tools in order to select those that will be suitable for incorporation into the model. Ontologies are also reviewed in order to identify (1) methods for developing the ontology that will be included in the model; (2) techniques to be used in training the ontology; and (3) which existing ontologies can be incorporated into the model.

2. To achieve objective number two, the author reviews linguistic rules, identifies their weaknesses, and selects some of the rules to be incorporated into the model.

3. To achieve objectives three and four, a model is implemented. The model integrates natural language processing tools with an ontology and linguistics rules to help designers produce conceptual models from natural language text. The model learns from the natural language requirements that it processes and uses the learnt information to update its ontology and improve its performance.

4. To achieve objective number five, the model is evaluated. The author demonstrates that the performance of novice designers is improved when they use the system. The author provides a test set of case studies with model answers and requests subjects to provide answers for these case studies, once by using the model and once without using the model. The model answers for the case studies are employed to evaluate the subjects' performance when using the model and when not using it. The author also shows that the information stored by the model can help the system to produce conceptual models and minimise human intervention. The model is trained and the evaluation shows the performance of the model is improved by the training.

1.5 Bibliographical Preparation

In order to start the bibliographical aspect of this study, the author conducted a review of the literature regarding the conversion of natural language text into conceptual models and possible solutions to tackle the limitations of this process. The relevant literature was identified using Google Scholar, as it is a free open search engine providing access to a variety of sources including academic publishers and universities. The author searched using several keywords to identify relevant literature, the most productive of these being 'From text to entity relationship model' and 'From English to entity relationship model'. The first ten results retrieved from Google Scholar for each search term were selected. During analysis of these documents, one particular paper caught the author's attention. This paper was entitled 'English Sentence Structure and Entity Relationship Diagrams'. The paper was published in 1983, and since then has been reproduced in nine versions and cited three hundred and four times. The author believed this paper to be significant for this research, not only because of the huge number of citations it has received, but also because it was the first to propose rules for mapping natural language text into ERDs. The author also looked at all the documents that cited this paper, which revealed what researchers have added since the rules for mapping natural language text into conceptual models were defined. This would allow the author to be more confident about determining what could be added and more aware of any possible limitations. Google Scholar was able to retrieve three hundred of the three hundred and four documents that cited the paper. These documents include books, book sections, journal articles, conference papers and reports. Forty of the three hundred documents are written in different languages, such as Spanish, German and French, but only the documents written in English were considered. The author looked at the title and read the abstract of each document in order to decide whether it would be relevant to the research. In this manner, sixty-eight documents were identified to be read in more depth and detail. Appendix 1 provides a list of these documents, which include conference papers, journal articles, book sections, PhD theses and Masters theses. These documents were used to start the bibliographical aspect of this study. In addition to these documents, the author undertook further reading about natural language processing tools, ontologies and linguistic rules for mapping natural language to conceptual models to understand how these techniques could be integrated in an appropriate way to achieve the research aim.

1.6 Research Contribution

The thesis will make a contribution to knowledge by developing a framework and ontology for extracting conceptual models from natural language text for an independent domain. The developed tool that supports the framework learns from the natural language texts that it processes and stores what has been learnt in its knowledge base to update it. The information that is stored in the knowledge base helps the tool to minimise human intervention and to improve its performance.

1.7 Thesis Contents

In addition to this Introduction and Motivation chapter, the thesis comprises five chapters and a series of appendices. The following is a short description of each part of the thesis contents.

Chapter 2: Background and Literature Review

This chapter introduces the main problems involved in the creation of conceptual models, reviews approaches used to map natural language text into conceptual models and discusses topics related to this mapping, such as natural language processing and ontologies. Section 2.1 introduces conceptual models and the main problems involved in their creation. Section 2.2 discusses the approaches used for mapping natural language text into a conceptual model and identifies the advantages and disadvantages of each approach. This section also introduces the proposed model. Section 2.3 discusses the natural language tasks that will be included in the proposed model and selects a natural language toolkit to perform such tasks. Section 2.4 discusses ontologies. This section considers ontology types, ontology creation methods, data set ontologies and ontology languages.

Chapter 3: Rules to Drive a Conceptual Model from Natural Language Text

In this chapter, the author reviews rules that may help in extracting conceptual model components such as entities, relationships and attributes from natural language text. The chapter is divided into six main sections. Rules for determining entities are discussed in Section 3.1. In Section 3.2, the author selects which rules will be applied to determine entities in the proposed tool. Rules for determining relationships between entities are discussed in Section 3.3. In Section 3.4, the author selects which rules will be used to determine relationships in the proposed tool. Rules for determining the attributes of entities are discussed in Section 3.5. The findings from this review and a summary of the chapter are given in Section 3.6.

Chapter 4: Implementation of Semi-Automated Conceptual Model Extraction System (SACMES)

In this chapter, the Semi-Automated Conceptual Model Extraction System (SACMES) is introduced. The chapter is divided into three sections. Section 4.1 demonstrates the SACMES architecture and Section 4.2 presents a demonstration of how SACMES is used to process requirement specifications. The chapter summary is given in Section 4.3.

Chapter 5: Empirical Evaluation of SACMES

This chapter shows how SACMES has been evaluated. The author aims to demonstrate that designers' performance in conceptual model extraction will improve when using the system. This hypothesis is explained in Section 5.1. The author also shows that the information learnt by

SACMES can help designers to produce conceptual models and minimise human intervention. This second hypothesis is explained in Section 5.2.

Chapter 6: Conclusion and Future Work

This chapter summarises the research findings and offers suggestions for future work.

Appendices

The appendices are used to include extra data and detail which it is not possible to include in the body of the thesis.

Chapter 2: Background and Literature Review

This chapter introduces the main problems involved in the creation of conceptual models, reviews approaches used to map natural language text into conceptual models and discusses topics related to this mapping, such as natural language processing and ontologies. Section 2.1 introduces conceptual models and the main problems involved in their creation. Section 2.2 discusses the approaches used for mapping natural language text into a conceptual model and identifies the advantages and disadvantages of each approach. This section also introduces the proposed model. Section 2.3 discusses the natural language tasks that will be included in the proposed model and selects a natural language toolkit to perform such tasks. Section 2.4 discusses ontologies. This section considers ontology types, ontology creation methods, data set ontologies and ontology languages.

2.1 Conceptual Models

The development of a conceptual model is the most important stage in the design of a system and database. This is because the conceptual model provides a blueprint of the system and database. Furthermore, the conceptual model can explain the structure of the system and its functions (Thalheim, 2000). In order to qualify as such, a conceptual model must have the ability to reflect the real-world environment (Dullea et al., 2003). A good model must be able to represent the concepts of the real-world situation effectively, as any errors that are made in the conceptual model will be costly to fix during implementation (Thonggoom, 2011).

Natural language is used as the main tool to describe requirement specifications. People usually use natural language text to describe things in the world and, in the same way, most requirement specifications in industry are written in natural language (Neill & Laplante, 2003; Luisa et al., 2004).

There are many formal notations which can be used to describe the requirement specifications for a conceptual model written in natural language text; indeed, the total number of such notations can reach eighty (Thalheim, 2000). Among these notations, the Entity Relationship Diagram (ERD) and Unified Modelling Language (UML) are the most common formalisms used in practice (Luisa et al., 2004). The ERD, proposed by Chen (1976), is widely used to describe conceptual models for database design because it is easy to understand and capable of modelling real world problems. Therefore, in this research, the ERD is chosen as a formalism for database design to be translated from requirements in natural language.

The ERD is a collection of entities, attributes and relationships, and this collection is powerful enough to describe real world problems. UML is another conceptual data model formalism commonly used in object-oriented software design. The UML is a data modelling language that has many different notations; for example, UML 2.2 has fourteen model diagrams (Thonggoom, 2011). However, class model diagrams are most widely used in practice to describe software engineering.

Despite its importance, it is very difficult to design a conceptual model (Simsion, 2007). Conceptual models are difficult to design because of (1) problems in the natural language text used to describe a problem domain and (2) other problems facing designers when they create conceptual models. Many researchers have studied problems with natural language text, while others have studied the problems facing designers. Section 2.2.1 discusses in more detail the weaknesses in natural language text and Section 2.2.2 discusses the problems faced by designers during conceptual model creation.

2.1.1 Problems in Natural Language Text

The main issue in using natural language to write specifications is the problem of ambiguity. It is recommended that any ambiguity in natural language specification documents is detected and removed prior to further analysis (Jackson, 1982; Meziane, 1994). Meyer (1985) and Pohl (1993) have studied the definition of problems in natural language text. There are seven classes of insufficiency in natural language specifications as shown by Meyer (1985), and these are:

1. Noise:

Noise is the existence of an element within the text that does not carry any information relevant to the problem.

2. Silence

Silence is the existence of a feature of the problem which is not covered in the natural language specification text.

3. Overspecification

This is the occurrence in the text of an element that links not to features of the problem, but to features of a possible solution.

4. Contradiction

The existence in the text of elements that describe a feature of the system in a mismatched way.

5. Ambiguity

The occurrence in the text of an element that allows a feature of the problem to be understood in at least two different ways.

6. Forward reference

The occurrence in the text of an element that introduces features of the problem that are not explained until later in the text.

7. Wishful thinking

The occurrence in the text of an element explaining a feature of the problem in such a way that a named solution will not in reality be effective in the context of this feature.

Natural language ambiguities can be divided into three categories, namely, lexicographic ambiguities, grammatical ambiguities, and ambiguities due to textual cohesion (Meziane, 1994).

1. Lexicographic ambiguities

Lexicographic ambiguities are usually words in English that have more than one meaning. To resolve this problem, a word should only be attached to one specific meaning. There are two categories of lexicographic ambiguity, namely, object-type lexicographic ambiguities and syntactic lexicographic ambiguities. Objects in the world are classified into groups and each group has its own features/attributes. One of the most important features of any object is its type, and the use of types can sometimes unambiguously identify these objects. An example of a type hierarchy for a physical object is illustrated in Figure 2.1.

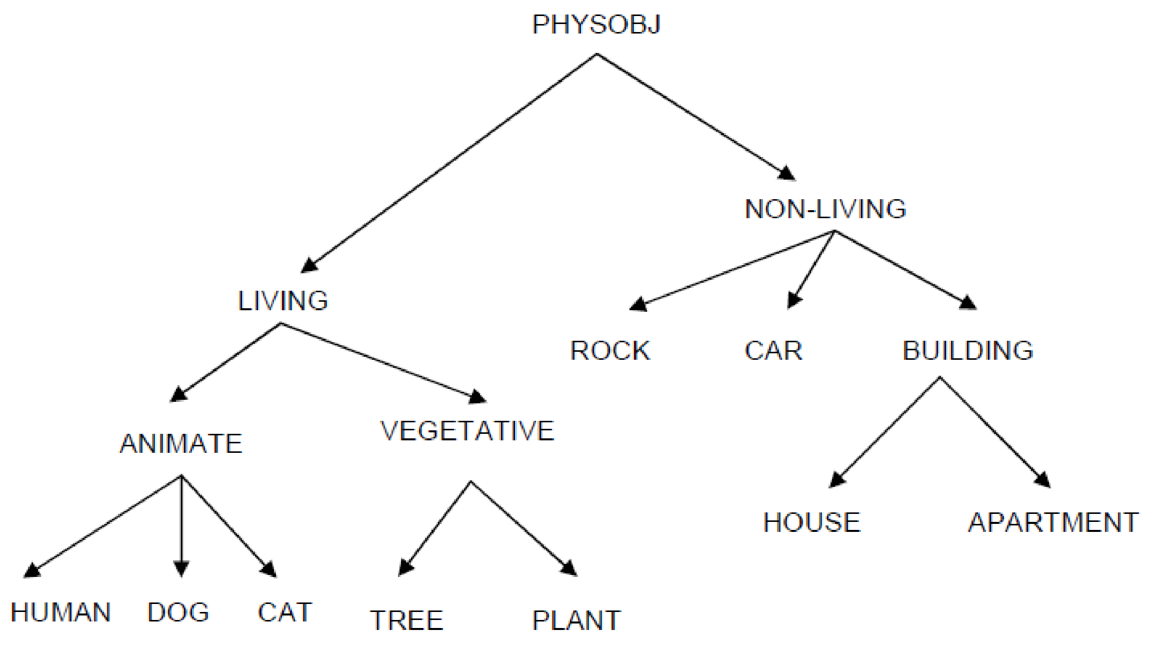


Figure 2.1 A Type Hierarchy for a Physical Object (Meziane, 1994, p. 66)

The deconstruction shown in Figure 2.1 is exhaustive for some entities. The physical objects are divided into two types, living and non-living. In the same way, the living objects are subdivided into human, dog and cat, though there are clearly other animate things that are not included in this hierarchy. Depending on the context and the kind of objects deployed, each group or institution has its own classification and its own hierarchy (Meziane, 1994).

The other category of lexicographic ambiguity is that a word may belong to more than one syntactic category. For instance, the word ‘books’ can be either the plural form of the noun book or the present simple form of the verb book. It is only when the correct syntax is given that such syntactical ambiguities are resolved (Meziane, 1994).

2. Grammatical ambiguities

Grammatical ambiguities occur when there is more than one way of parsing a sentence or part of a sentence. Each parser has its own interpretation (Meziane, 1994).

3. Textual cohesion

In the process of writing texts, many methods are used to guarantee that all parts of the text are linked properly and that there is a smooth transition from one idea to another. These techniques provide textual cohesion (Meziane, 1994). There are many types of textual cohesion, namely, references, substitution, conjunctions and lexical cohesion (Jackson, 1982).

1. References: references include things that cannot have their own interpretation but make a reference to something else. For example, in the sentence ‘When a student works on modules, he must pass all registered modules’, the pronoun ‘he’ is a reference for the noun phrase ‘a student’. To remove any textual ambiguity from such a reference, the pronoun must be replaced with the noun phrase ‘a student’.

2. Substitution: a substitution is defined as “A grammatical relation, where one linguistic item substitutes for a longer one”. For example, in the sentence “The program reads all client records and checks each record to determine if a premium notice is due or a cancellation (i.e., past due) notice should be issued and if so, prints the appropriate notice” (Presland, 1986, p. 193), the word ‘so’ is substituted for the clause ‘a premium notice or a cancellation notice should be issued’.

3. Conjunctions: a conjunction is a part of speech used to connect a word, a phrase or a sentence with another word, phrase or sentence. For example, in order to remove conjunction ambiguities in the sentence ‘A student learns English and French’, the sentence should be divided into two small sentences, ‘A student learns English’ and ‘A student learns French’.

4. Lexical cohesion: lexical cohesion means the replacement of a word by a synonym or related word in consecutive sentences. For example, in the sentences ‘A teacher teaches students. Each instructor can teach many students’, the noun ‘instructor’ is a synonym of the noun phrase ‘a teacher’. In this case, only one of the two synonyms should be used as an entity, as clearly it is undesirable to create two entities rather than one.

2.1.2 Problems Facing Designers during Conceptual Model Creation

In previous studies, many researchers have reported difficulties which work against the creation of conceptual models, such as Antony and Batra (2002), Batra (2007), Currim (2008), Dey, Storey and Barron (1999), Liao and Palvia (2000), Moody (2004) and Shoval and Shiran (1997). Although conceptual models are highly significant and important, researchers report that they are often not designed well (Simsion, 2007). Furthermore, some researchers have studied errors made by novice designers during the creation of conceptual models. The results of such studies are important in building tools and developing techniques which can overcome these errors, thus leading to the creation of qualified conceptual models.

1. Combinatorial complexity

The findings of some studies show that novice designers have more difficulty in modelling relationships than in modelling entities (Topi, 2002). Other studies show that novice designers have difficulties in modelling different kinds of relationships, including unary, binary and ternary relationships (Batra, 2007; Batra & Antony, 1994). There is a proportional relationship between an entities count and relationships count, as explained in Figure 2.2.

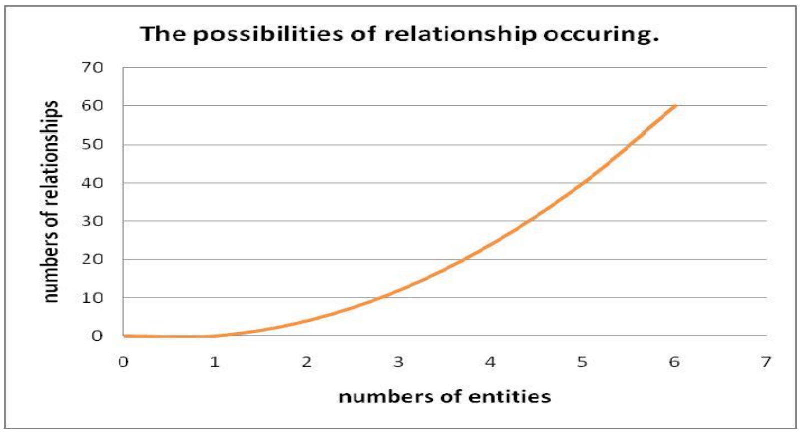


Figure 2.2 Proportional Relationship between Entities Count and Relationships Count (Thonggoom, 2011, p. 20)

When the entities count is increased, the relationships count is also increased. Therefore, in order for a designer to establish a good set of relationships, three criteria should be met: (1) semantic relationships in the application must not be missed; (2) the relationships between entities must not be redundant; and (3) the degree of relationship should be minimal (Thonggoom, 2011).

2. Scattered modelling rules

Rules created for extracting conceptual models from natural language text are usually incomplete; natural text will eventually throw up an example that defeats a set of rules. In overall terms, rules are useful, but they sometimes cause cognitive errors called biases (Batra & Antony, 1994; Parsons & Saunders, 2004). Rules can be in conflict and overlapped, and such overlapping and conflict can lead to a set of rules which cannot work together (Thonggoom, 2011). For example, entities in natural language specifications are usually extracted from nouns, but attributes can also be extracted from nouns.

3. Semantic mismatch

Literally mapping from natural language specifications into a database leads to 'literal translation errors' (Batra, 2007). For example, the sentence 'An order records a sale of products to customers' may contain an incorrect relationship between a customer and a product. This illustrates that not all actual-world relationships stated in the requirement specifications text are mapped to database relationships, while some actual-world relationships are determined at the database level. Furthermore, some relationships are derived indirectly from natural language specifications.

4. Inexperience of novice designers and incomplete knowledge

Expert designers have a wide range of knowledge and experience to draw on, whereas novice designers' limited knowledge means that they may struggle and make errors during the creation of conceptual models. Even skilled designers might fail to produce a valid conceptual model due to lack of domain knowledge, unless they have a clear awareness of the requirement specifications (Kim et al., 2008). Expertise in domain knowledge is required to recognise hidden entities. The most significant issue, therefore, is how trainee designers can be taught professionally and how domain knowledge can be transmitted to designers (Thonggoom, 2011).

Because of the difficulties that work against the creation of conceptual models, as explained in Sections 2.2.1 and 2.2.2, researchers have begun exploring the automated creation of conceptual models. Although a fully automated system for mapping natural language specifications into a conceptual model is not yet available, semi-automated systems do now exist and Section 2.2 discusses the approaches used to extract conceptual models from natural language text. At the end of the section, a comparison is made between these approaches and the author suggests a new semi-automated approach for mapping natural language specifications into conceptual models.

2.2 Approaches for Extracting Conceptual Models from Natural Language Text

2.2.1 Linguistics-based Approach

People use natural language to communicate and describe things and therefore, linguistic theories and Natural Language Processing (NLP) are used for designing many information systems (Castro, Baiao, & Guizzardi, 2009; Métais, 2002). Chen (1983) suggested eleven rules for mapping requirement specification text into an Entity Relationship Diagram (ERD). Chen's work was followed by other studies, such as those by Hartmann and Link (2007), Omar et al. (2004), and Overmyer et al. (2001), to use, enhance and extend Chen's rules, but these rules are still incomplete, inaccurate and overlapped. Therefore, a linguistic approach can provide only the basic requirements for either manual or semi-automated transformation from natural language text into a Conceptual Model (CM). In addition, rules for transformation from natural language text into CMs are based on particular syntaxes in natural language specifications, but these rules cannot solve all the ambiguity problems inherent in natural language processing and, because natural languages are different, the rules cannot be universal (Thonggoom, 2011).

In order to solve inherent ambiguities in natural language requirements, some studies have set constraints on the input. These constraints are based on the vocabularies and sentence structures of the input (Ambriola & Gervasi, 2006; Osborne & MacNish, 1996; Tjoa & Berger, 1994). Using these constraints, in addition to basic natural language processing techniques such as part-of-speech tagging and chunking, allows the process of mapping from natural language specifications into conceptual models to achieve a realistic result. However, the use of constraints alone is limited in solving such problems. Constraints (controlled language) place unrealistic restrictions on the writers of requirement specifications. Other studies have suggested using formal languages such as Z, Object-Z, OCL, VDM and B for the specification writing process. Formal languages are expressive but do not include supporting tools. Furthermore, the use of formal languages demands deep knowledge of the languages in order to write them professionally. In addition, formal language tools have often been designed for specific applications and their use in different applications can be problematic (Thonggoom, 2011). Dialogue tools have also been suggested as a means of dealing with natural language specifications (Buchholz, Cyriaks, Düsterhöft, Mehlan, & Thalheim, 1995; Kim et al., 2008) .

However, dialogue tools rely on human intervention and thus may not be useful for large-scale batch processing (Thonggoom, 2011).

Classification and categorisation theory has also been applied to conceptual data modelling (Larman, 2001; Song et al., 2004). Categorisation involves determining particular properties attached to a category's members, while attributes are used to classify the entities. Missing entities can be spotted by using class categories. Class categories for domain knowledge can thus be applied to discover hidden entities which are not mentioned in the requirement specification text (Song et al., 2004).

The linguistic approach is also supported by linguistic dictionaries and common-sense ontologies (Burg & Van de Riet, 1998; Miyoshi, Sugiyama, Kobayashi, & Ogino, 1996). Linguistic dictionaries deliver semantic links between concepts, which include synonyms, antonyms, hyponym/hypernym (is-a) and meronym/holonym (part-of). Linguistic dictionaries also deliver syntactical and morphological information. More detail about these types of relationships is found in Storey (1993). WordNet is a good example of a linguistic dictionary to be used in the development of conceptual models. It is available in English and other European languages, while WordNet++ includes more semantic relationships which are not found in the first version of WordNet (Dehne, Steuten, & van de Riet, 2001).

2.2.1.1 Tools and systems based on a linguistic approach

The majority of tools which map natural language specifications into CMs use a linguistic approach. This approach usually starts by applying natural language processing tools and Chen's rules, in addition to human intervention from designers. Examples of tools using a linguistic approach are given in Gomez, Segami and Delaune (1999), Buchholz et al. (1995), Burg and van de Riet (1998), Du (2008), Harmain and Gaizauskas (2003), Meziane and Vadera (2004), Mich and Garigliano (1999), Omar et al. (2004), Storey (1993), Tjoa and Berger (1994), Tseng, Chen and Yang (1992), Athenikos and Song (2013) and Ambriola and Gervasi (2006). Du (2008) provides a review of these systems and the following is a description of some of the tools which use a linguistic approach.

1. LIDA: Linguistic assistant for Domain Analysis

LIDA is a semi-automated tool for mapping natural language specifications into a class diagram (Overmyer et al., 2001). The tool uses Chen's rules for transforming a specification into a class diagram; it maps nouns into classes and verbs into relationships. However, this tool is limited because Chen's rules are incomplete and overlapped.

2. COLOR-X: Conceptual Linguistically-based Object-oriented Representation language for information and communcation systems

COLOR-X is a tool for converting natural language specifications into a CM based on WordNet and Chen’s rules (Burg & van de Riet, 1998). The tool practises linguistic concepts that are similar to Chen’s rules for generating models that reflect static and dynamic features of the system. Dehne et al. (2001) revised the tool by using WordNet++, but the tool remains limited because it is based on incomplete linguistic rules.

3. CM-Builder

Harmain and Gaizauskas (2003) designed a natural-language-based case tool called Class Model Builder (CM-Builder). It was intended to assist in extracting classes, attributes and relationships automatically from natural language specification text. In other words, it produces a class model representation, similar to that found in the Unified Modelling Language (UML). The CM-Builder works automatically but, like similar tools, it does require human intervention. There are two versions of CM-Builder: version 1 and version 2. Version 2 has a better performance profile and requires less human intervention than version 1. The purpose of this work was not to produce a class model automatically from text, without human intervention, but to show that Natural Language Processing (NLP) can assist in producing an initial diagram, which can then be reconsidered and refined by the software engineer to produce a final version of a class diagram. This tool is also limited, however, because it is based on Chen’s rules for analysing natural language specifications, and those rules cannot solve inherent ambiguity problems.

4. ER-Converter

Omar et al. (2004) used rules linked with weightings in designing a semi-automated tool known as an Entity Relationship Converter (ER-Converter). For instance, when a noun phrase is followed by a verb such as ‘has’ or ‘have’, then the noun phrase is given 0.7 as a weighting for being an entity. The ER-Converter assists in producing an Entity Relationship Diagram (ERD) from a requirement specification written in natural language. The process starts when a requirement specification is read by the system, which then uses rules and human intervention to build the ERD. Therefore, although the ER-Converter works better than CM-Builder, the tool still requires a degree of human intervention.

5. ACDM: Automated Conceptual Data Modelling

Du (2008) proposed ACDM as a system for identifying an entity relationship diagram from requirement specifications written in a controlled language. The ACDM is integrated with a parser, WordNet and search services. The controlled language requirements are parsed, and then converted into an entity relationship diagram using Chen’s rules. The use of controlled language is the main limitation of ACDM.

2.2.1.2 Advantages and disadvantages of a linguistic approach

The main advantage of the linguistic approach is that it is domain independent. However, domain independency can also be a disadvantage for a linguistic approach (Thonggoom, 2011). Linguistic tools do not include domain knowledge and therefore, this approach does not deliver a top solution for many natural language specifications because the approach is unable to solve natural language problems such as ambiguities.

2.2.2 Pattern-based Approach

The use of patterns in designing was introduced by Alexander in 1979, in his book entitled ‘On Architecture and Urban Planning’ (Alexander, 1979). Alexander explained that using patterns is a better way for designers to solve problems than solving them from first principles. Nowadays, the use of patterns is well established and is regularly used as an approach to solving problems in the software development process. Higher productivity, improvement in software quality and reduction in time and cost are all benefits obtained by using patterns in software development. In conceptual model design, however, pattern usage can be difficult. The works presented by North, Mayfield and Coad (1995), Hay (1996) and Fowler (1997) can be considered as recognition of the use of patterns in developing conceptual models, but from empirical research, it is obvious that specialists can use patterns whereas novices cannot (Chaiyasut & Shanks, 1994).

The patterns process includes three main tasks, namely, retrieval, adaptation and integration (Anthony & Mellarkod, 2009). Retrieval consists of selecting patterns that may be related to a certain problem. Once a pattern is selected, it must be adapted so that it is appropriate for the problem. Finally, the pattern is integrated with further patterns to produce a comprehensive model in the form of a conceptual data model.

Authors have suggested various types of patterns. Examples of these authors are North et al. (1995), Fayad (1997), Fowler (1997), Gamma (1995), Hay (1996), Johannesson and Wohed (1999), Johnson and Foote (1988), Pree (1994), Silverston, Inmon, and Graziano (2001) and Szyperski (1997). Blaha (2010) suggests several pattern types for modelling, including universal antipatterns, archetypes and canonical patterns. However, designers should avoid using universal antipatterns within applications. Archetypes are the most common modelling patterns and can be applied through a range of different applications, while canonical patterns are appropriate for meta models of modelling formalisms. Blaha offers approaches for mapping patterns into a relational schema for database design.

Silverston et al. (2001) and Kimball and Ross (2002) provide common patterns packaged for data models. The use of these packages decreases implementation time and cost, and provides quality models (Hoffer, Prescott, & Mcfadden, 2004), but packaged data models cannot be regarded as a substitute for good database analysis and design. Expert analysts and designers are still required to define the database requirements and to choose, adapt and integrate any packaged systems that are in use (Thonggoom, 2011).

Three measures, namely usability, reusefulness and efficiency, are used to evaluate patterns (Han, Purao, & Storey, 2008). First, usability specifies the ease with which an artefact can accomplish retrieval (search and adaptation of the artefact for the current design) and assembly (integration of the reusable artefact with other parts of the design). Domain independency is used to measure reusefulness, which refers to the extent to which a pattern of this kind could be deployed in a different but similar problem area. The amount of effort required to create the artefact is used as a measure of the efficiency of an artefact.

2.2.2.1 Tools and systems based on patterns approach

1. APSARA

Analysis pattern repositories are the most commonly utilised approach among conceptual modelling tools and systems (Thonggoom, 2011). An analysis pattern repository is a group of generic objects with stereotypical properties which display relationships in a domain-neutral manner (Batra, 2005). Purao (1998) proposed APSARA as a knowledge-based system which utilises natural language processing tools for mapping natural text requirements into objects. The objects are used to retrieve analysis patterns from a pattern repository, and then the analysis patterns are instantiated and synthesised into a CM. Thirty analysis patterns developed by (North et al., 1995) are included in APSARA, which is updated by including learning mechanisms. These learning mechanisms assist designers by signifying specific patterns that might relate (Purao, Storey, & Han, 2003). The limitations of APSARA are that the analysis patterns are so abstract that mismatches of patterns are fairly common (Thonggoom, 2011), and beginner designers are unable to reason with analogy (Anthony & Mellarkod, 2009).

2. Modelling Wizard tool

Wohed (2000) proposed the Modelling Wizard dialogue tool for choosing appropriate patterns. The tool stores numerous patterns, and an appropriate pattern is chosen in a stage-by-stage manner based on answers given to questions posed by users. The restriction of the tool is that extensive user intervention is needed for answering the questions, and thus it is very difficult to use the tool for large-scale batch processing.

2.2.2.2 Advantages and disadvantages of a patterns approach

Using patterns is beneficial in (1) speeding up the design via reuse and (2) improving software quality by using a design which has proved superior in numerous applications. However, designers wishing to build a patterns repository need to have domain knowledge regarding objects in the domain and the extent of abstraction of the objects. Thus, building a patterns repository is time consuming and the majority of pattern repositories used for CMs are built manually. Furthermore, the majority of tools in the patterns approach use analysis patterns which require manual matching (Thonggoom, 2011). Extracting pattern artefacts from existing designs is presented as a solution which can decrease experts' involvement in creating a pattern repository (Han et al., 2008), and if this can be achieved in different application domains, it will help to support the generation of practically reusable pattern artefacts. Reusable pattern artefacts can be understood and used easily because they are domain specific (Thonggoom, 2011).

2.2.3 Case-based Approaches

For developing knowledge-based systems, a technology called case-based reasoning is used. Case-based reasoning works by finding a solution for a new problem by retrieving a similar problem and adapting it into a suitable solution for the new problem. Retrieval mechanisms for reusable artefacts mainly involve natural language processing techniques, with an indexing technique used to speed up artefact retrieval (Thonggoom, 2011). However, only a limited number of researchers have used case-based techniques. A Common Sense Business Reasoner (CSBR) (Storey et al., 1997), a Design Expert System for Database Schema (DES-DS) (Paek, et al., 1996) and a Case-Based System for Database Design (CABSYDD) (Choobineh & Lo, 2004) are all examples of using a case-based approach, and a comparison between these three systems is found in Choobineh and Lo (2004). Although the approach benefits from reusing previous designs, the main disadvantage of this approach is that developing conceptual model libraries and indexing mechanisms is extremely costly (Thonggoom, 2011).

2.2.4 Ontology-based Approach

Many definitions of ontology are given in the literature, and these definitions vary according to their involvement in artificial intelligence and computing in general. The most frequently cited one is that ontology is a “specification of a conceptualisation” (Gruber, 1993). This is definitely the most concise definition. ‘Conceptualisation’ refers to an abstract and basic view of the world. It is used when a knowledge base within an intelligent system is needed to represent world knowledge for a particular purpose. Conceptualisation is based on objects, concepts, entities and relationships between them within an area of interest. The definition also refers to

‘specification’, which means that formal and declarative representation is required (Dermeval et al., 2016). The structure of the ontology, including the concepts, entities and constraints on how they are used, should be stated declaratively, explicitly and by using formal language. The ontology must be machine readable (Gaševic, Djuric, & Devedžic, 2006). Another definition of ontology is that it is “a set of knowledge terms, including the vocabulary, the semantic interconnections, and some simple rules of inference and logic for some particular topic” (Hendler, 2001). The use of ontologies in software development has been growing (Gašević, Kaviani, & Milanović, 2009; Pan, Staab, Aßmann, Ebert, & Zhao, 2012). From the literature, it can be seen that ontologies are used in (1) requirement engineering processes; (2) requirement modelling styles; (3) supporting functional and non-functional requirements; and (4) addressing requirement engineering problems.
According to Kotonya and Somerville (1998), there are five phases in the requirement engineering process, namely, elicitation, analysis and negotiation, specification, validation and management. According to a systematic literature review on using ontology in requirement engineering conducted by Dermeval et al. (2016), ontologies are used in all requirement engineering stages. Al Balushi, Sampaio and Loucopoulos (2013) and Anwer and Ikram (2008) provide examples of using ontology in the elicitation stage, while Assawamekin, Sunetnanta and Pluempitiwiriyawej (2010) and Bicchierai, Bucci, Nocentini and Vicario (2012) offer examples of its use in the analysis and negotiation stages. Cardei, Fonoage and Shankar (2008) and Castañeda, Ballejos and Caliusco (2012) exemplify the use of ontology in the specification stage, Kroha, Janetzko and Labra (2009) give an example of using ontology in the validation stage, and Ghaisas and Ajmeri (2013) provide an example of its use in the management stage.
Ontologies support many requirement modelling styles, including textual requirements such as in Chicaiza, López, Piedra, Martínez and Tovar (2010), Daramola, Sindre and Moser (2012) and Daramola, Stålhane, Omoronyia and Sindre (2013). Examples of ontology use with UML include Boukhari, Bellatreche and Jean (2012), Cardei et al. (2008) and Castañeda et al. (2012). Ontologies also support functional requirements, such as in Gandhi and Lee (2011), non-functional requirements, such as in López, Astudillo and Cysneiros (2008), and both functional and non-functional requirements as in Pires et al. (2011) and Polpinij (2009).
Some researchers have taken advantage of existing ontologies from previous studies and reused them, such as in Reinhartz-Berger, Sturm and Wand (2011), Riechert and Berger (2009), and Saeki, Hayashi and Kaiya (2013). On the other hand, other studies have developed their own ontologies, such as in Velasco, Valencia-García, Fernández-Breis and Toval (2009), Li, Jin, Xu

and Lu (2011) and Lima, Garcia, Amaral and Caran (2011). According to a systematic literature review conducted by Dermeval et al. (2016), 66% of studies chose to develop their own ontology rather than using existing ontologies developed by others, while 34% used existing ontologies.

La-Ongsri and Roddick (2015) argue that existing conceptual models are not sufficiently expressive to allow a combination of ontologies in one single conceptual model. Therefore, they investigated the incorporation of ontologies into three collective conceptual models, namely, the Ontological Entity Relationship (OntoER) model, Ontological Role Modelling (OntoORM) and Ontological Unified Modelling Language (OntoUML).

In general, using ontologies in requirement engineering offers three benefits, which are (1) decrease of ambiguity, inconsistency and/or incompleteness in requirements; (2) domain knowledge representation support to guide requirements elicitation; and (3) support in requirements management/ requirement evolution (Dermeval et al., 2016).

Many researchers utilise ontologies in evaluating, improving and developing conceptual modelling formalisms. The main benefit of utilising ontologies in conceptual modelling is the reusability of a knowledge repository. The reusable knowledge repository is divided into two parts, namely, a domain ontology and an upper level ontology (Thonggoom, 2011). A domain ontology indicates concepts, relationships between concepts and inference rules for a specific domain (Conesa, Storey, & Sugumaran, 2010). Protégé is an example of tools that support ontology development, while SPARQL is an example of tools used in enquiring into domain ontologies. A comparison between these tools is represented in Corcho, Fernández-López and Gómez-Pérez (2003). On the other hand, an upper level ontology represents concepts which can fit all domains. Cyc[1], and SUMO[2] are examples of upper domain ontologies. A review and comparison between upper ontologies is available in Mascardi, Cordì and Rosso (2007). Although upper level ontologies are domain independent, it is challenging to integrate them and make them really useful. A main problem with existing upper level ontologies is the lack of availability of a user interface or respectable API to facilitate their use (Thonggoom, 2011). Clearly, domain ontologies are more practical than large-scale ontology domains (Conesa et al., 2010).

[1] http://www.cyc.com/

[2] http://www.adampease.org/OP/

2.2.4.1 Tools for using ontologies in conceptual models

1. Ontology Management and Database Design Environment (OMDDE)

Sugumaran and Storey (2002) proposed a methodology to be used in creating ontologies and validating entity relationship models. Their argument was that a repository of ontologies is needed to support the database design and conceptual model database design processes. The repository should be divided into sub-ontologies and each ontology should cover specific domain knowledge. The methodology involves four steps, the first being identification of basic terms. This step involves identification of the most frequent terms in each domain, as well as definition of synonyms of the most frequent terms in each domain. The second step involves identification of relationships between basic terms. The authors covered the three most common relationships between terms, which are generalisation, association and synonyms. This stage also involves defining relationships between ontologies to confirm that the terms have consistent relationships across all ontologies; this helps in updating the ontologies easily into one ontology. The third step is identification of basic constraints. The authors paid attention to the four most common constraints between terms, which are prerequisite constraints, temporal constraints, mutually inclusive constraints and mutually exclusive constraints. The fourth step is identification of higher level constraints capturing domain knowledge. These constraints are domain dependent and capture business rules for each domain.

The OMDDE is a prototype for implementation of Sugumaran and Storey's methodology. Sugumaran and Storey selected an auction as a domain for the ontology. The system was tested on beginner designers, as well as on qualified designers who used case tools such as UML and other sources of information such as Wikipedia, to show that the use of ontologies is a good way to provide high quality information for building conceptual models from requirement specification text. The results show that beginner designers who used the OMDDE system produced qualified conceptual models better than those who did not use the system. They also show that qualified designers who used the system produced a higher quality of conceptual model than those who used a case tool such as the UML Case Tool[3] and information sources such as Wikipedia (Sugumaran & Storey, 2006). This work provides a good example of how ontologies can be used in extracting conceptual models. However, the authors used a lightweight, domain-dependent ontology for an auction, which means that the system is unlikely to work properly with other, different domains. Although the system allows more ontologies to

[3] http://gentleware.com

be added and existing ontologies to be updated, it will require considerable effort and expertise in the knowledge base to achieve this.

2. DC-Builder

Herchi, Abdessalem (2012) proposed a tool called DC-Builder. This tool integrates natural language processing with a domain ontology in order to produce a class diagram from natural language specifications. The DC-Builder includes three stages. The first stage is called the natural language analysis block. This stage employs General Architecture for Text Engineering (GATE[4]) as a natural language processing toolkit for achieving natural language processing tasks. The requirement specification text is the input for this stage; the text is divided into sentences via a sentence splitter, and then noun phrases within the requirement specifications are defined via a part-of-speech tagger. Parsing is also included in this stage, which helps in discovering important elements in the requirement specifications such as sentence subjects, sentence objects and verbs. The second stage of the DC-Builder is called extraction using heuristics. In this stage, rules for extracting class diagram elements from natural language are employed. The DC-Builder employs Chen's rules to define the main elements of the class diagram. The third stage is called refinement. The output from the second stage contains many elements that may not be entities, but are included because of applying Chen's rules. Using a domain entity can reduce the number of elements by keeping only nouns with potential for inclusion in the class diagram.

Recall, precision and overgeneration are used as factors to evaluate the DC-Builder's performance. The DC-Builder is evaluated using case studies from object-oriented analysis books. Its performance is also compared with other tools, such as the CM-builder and is shown to give a better performance than the CM-builder. The DC-Builder uses a domain-dependent ontology, though the authors do not mention which domain was used to provide domain knowledge for the DC-Builder. The reliance on a domain-dependent knowledge base may be considered a limitation of the DC-Builder.

2.2.4.2 Advantages and disadvantages of an ontology-based approach

The main benefit of utilising ontologies in building conceptual models is the reusability of a knowledge repository, but ontology development is challenging. Even for a particular domain, creating an exhaustive domain ontology is labour intensive and time consuming. Automatic ontology creation is also challenging work due to the lack of a structured knowledge base. Although there are many tools which support the creation of an ontology, such as OntoEdit,

[4] https://gate.ac.uk/

Ontolingua and Protégé, ontology development does require human effort. The majority of ontology development applications involve a manual process (Thonggoom, 2011).

2.2.5 Multiple Approaches

The majority of tools developed for mapping natural language text into conceptual models require human intervention during the transformation. Furthermore, no approach works perfectly all the time and each approach has its limitations. Ideally, therefore, many approaches should be incorporated into the design process in order to achieve a better output. The following are some examples of studies which have used multiple approaches for creating conceptual models from natural language specifications.

1. EIPW

Thonggoom, Song and An (2011a) developed an automated methodology for building Entity Instance Patterns (EIP) and Relationship Instance Patterns (RIP) from previously designed databases. EIP is a repository of entities and RIE is a repository of relationship patterns. These repositories are integrated with WordNet ontology (ontology approach), natural language processing techniques (a linguistic approach) and human intervention to develop the Entity Instance Pattern WordNet (EIPW). The EIPW is a semi-automated tool for extracting conceptual models from natural language text. The process is started by inserting natural language specifications into the EIPW, which then uses part-of-speech tagging as a natural language processing technique for defining a list of noun phrases as candidate entities. The EIPW then uses WordNet, human intervention and a knowledge base represented in EIP and RIP to extract entities and relationships as pre-requirements for the conceptual model. Extracted entities and relationships are inserted into the EIP and the RIP respectively to keep them updated. One of the limitations with the EIPW is that it is not clear how the EIP and RIP are structured and organised. It is also unclear to what extent the updated EIP and RIP will continue to capsulise and abstract properly.

2. HBT

Thonggoom (2011) developed the Heuristic Based Technique (HBT). The HBT is a semi-automated tool for extracting an ERD from natural language specification text. It uses linguistic rules integrated with WordNet ontology, a relationships instance repository and human intervention during the extraction process. The process is started by feeding natural language specifications into the HBT. Like the EIPW, the HBT uses part-of-speech tagging as a natural language technique for extracting candidate entities. The HBT then uses human intervention, WordNet and a relationships instance repository to guide the extraction of the entity relationship diagram. The extracted relationships are added into a relationship instance repository for updating. As with the EIPW, however, it is again not clear how the relationships instance repository is structured and organised, and it is unclear to what extent the updated relationships instance repository will still capsulise and abstract properly.

To summarise, the literature reveals that there are five approaches to extracting CMs from natural language text, namely, the linguistics-based approach, pattern-based approach, case-based approach, ontology-based approach and multiple approaches (Thonggoom, 2011). The main advantage of a linguistic approach is that it is domain independent, but linguistic tools do not include domain knowledge; therefore, this approach does not deliver a top solution for many natural language specifications and it is unable to solve natural language ambiguities. Using patterns is beneficial in speeding up the design via reuse, and in improving software quality by using a design which has proved superior in numerous applications. However, the majority of pattern repositories used for conceptual models are built manually. Furthermore, the majority of tools for the patterns approach use analysis patterns, which require manual matching. The case-based approach benefits from the reuse of previous designs, but the main disadvantage of this approach is that developing conceptual model libraries and indexing mechanisms is costly. The main benefit of utilising ontologies in conceptual modelling is the reusability of knowledge repositories, but ontology development is challenging. Even for a particular domain, creating an exhaustive domain ontology is labour intensive and time consuming. Table 2.1 illustrates a comparison between the different approaches used for extracting CMs from natural language text.

Approach Name	Examples	Advantages	Disadvantages
Linguistics-based approach	CM-Builder (Harmain & Gaizauskas, 2003) and ER-Converter (Omar et al., 2004)	Domain independent	Does not include domain knowledge and not capable for solving natural language ambiguity
Pattern-based approach	Modelling Wizard tool (Wohed, 2000) and APSARA (Purao, 1998)	Speeding up design via reuse and improving software quality by using designs, which have proved superior in numerous applications.	It is time consuming and very difficult to build a pattern library.

Approach Name	Examples	Advantages	Disadvantages
Case-based approach	CSBR (Storey et al., 1997) and DES-DS (Paek et al., 1996)	Cases-based approach benefits from reusing previous designs.	Developing conceptual model libraries and indexing mechanisms are costly.
Ontology-based approach	OMDDE (Sugumaran & Storey, 2006) DC-Builder (Herchi & Abdessalem, 2012)	The main benefit of utilising ontologies in conceptual modelling is the reusability of a knowledge repository	Development of both domain-dependent ontology and domain-independent ontologies is challenging.
Multiple approach	EIPW (Thonggoom et al., 2011a) HBT (Thonggoom, 2011)	Using more than one approach can help to avoid the limitations of each approach alone	The approaches cannot be integrated ideally to minimise the limitations of each individual approach.

Table 2.1 Comparison between Approaches Used for Extracting Conceptual Models from Natural Language Specifications

The author believes that integrating multiple approaches can help in solving the limitations which appear when each approach stands alone. For example, a linguistic approach is domain independent but it does not include a domain knowledge base. In addition, the approach faces difficulties in solving natural language ambiguities. Thus, it is a good idea if a linguistic approach is supported by adding a domain knowledge base. This can be achieved by incorporating an ontological approach with a linguistic approach. Conversely, knowledge-based approaches such as the pattern-based, case-based and ontology-based approaches do need a set of linguistic rules extracted from a linguistic approach to guide the process of extraction of CMs from natural language text, since there is no domain-independent knowledge designed to support the creation of conceptual models. Furthermore, because of natural language ambiguities, and the fact that fully-automated extraction of conceptual models from natural language is not possible (Song et al., 2015; Šuman et al., 2016; Thonggoom, 2011), integrated approaches need to be supported by a minimum level of human intervention to help in solving ambiguities in natural language text. An integrated approach supported by a minimum level of human intervention would therefore help in producing a semi-automated tool to guide the process of extracting and producing conceptual models from natural language specifications.

This research uses the integration of a linguistic approach with a knowledge-based approach. These approaches are supported by a minimum level of human intervention to resolve natural language ambiguities. The integrated approach uses an ontology as the knowledge-based approach. Moreover, because the ontology of a specific domain will not be sufficient to produce suitable reusable knowledge to support the creation of conceptual models, a domain-independent ontology is needed. However, building a domain-independent ontology or an upper domain ontology is challenging and time consuming, and requires domain knowledge expertise. Therefore, the author's intention is to fill this gap by building a domain-independent ontology which can be updated from the natural language specification text that is inserted into the proposed model. As the ontology is updated, it should be increasingly capable of providing useful knowledge to guide and support the process of conceptual model extraction from natural language text. More detail about the architecture of the model, and how the model's components are integrated, is given in Chapter Four.

2.3 Natural Language Processing (NLP)

NLP applications usually employ natural language processing toolkits to achieve natural language processing tasks. Another option is that some people may develop their own natural language toolkit to achieve their desired tasks. There are currently many natural NLP toolkits available for carrying out common tasks (Pinto, Gonçalo Oliveira, & Oliveira Alves, 2016). People who develop NLP applications will not start their applications from scratch, but use toolkits which are available without cost to perform tasks such as tokenisation, Part-of-Speech (PoS) tagging, and Name Entities Recognition (NER). In fact, the problem now is not how to develop NLP toolkits, but rather, which toolkit to choose from the many available in the literature. To answer this question, it is necessary for the author to define the tasks required by the proposed model, and then to look at different natural language toolkits in order to try to choose one of them. The following sections identify the natural language processing tasks which the proposed model needs to undertake during the pre-processing stage.

1. Tokenisation

Tokenisation divides a sentence into tokens. A token includes words, punctuation and numbers within the sentence (Grefenstette, 1999). Table 2.2 shows tokens for the sentence 'A Student takes a course'.

ID	Token
1	A
2	student
3	takes
4	a
5	course
6	.

Table 2.2 Tokens of a Sentence

2. Sentence splitter

A sentence splitter splits natural language text into sentences (Bontcheva et al., 2013). An example is given in Table 2.3.

A student takes a course. A teacher teaches a course. A student must pass a course; otherwise, he needs to retake it.	
ID	Sentence tokens
1	A student takes a course.
2	A teacher teaches a course.
3	A student must pass a course; otherwise, he needs to retake it.

Table 2.3 Sentence Splitter Divides Text into Sentences

In the proposed model, a sentence splitter will be required to divide natural language specifications into a set of sentences. Sentence splitting and tokenisation are prerequisites for part-of-speech tagging.

3. PoS tagger

PoS taggers identify the part of speech for a word. In general, there are four main PoS types, namely, nouns, verbs, adjectives and adverbs. Each type has sub-types. The Penn Treebank has thirty-six diverse identifiers for PoS (Santorini, 1990). Table 2.4 illustrates PoS tags in the Penn Treebank Project.

Number	Tag	Description
1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner

Number	Tag	Description
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	to
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb

Table 2.4 PoS Tags in the Penn Treebank Project (Santorini, 1990)

The proposed model requires a PoS tagger to distinguish nouns from other PoSs included in requirement specification text. The model assigns nouns as candidate entities, then organises some filtration to identify actual entities.

4. Name Entity Recognition (NER)

NER is used to classify noun phrases into different classes, such as a person, a location, an organisation, date, money, percentage and time (Tjong Kim Sang & De Meulder, 2003). An NER tool receives text as input and outputs a noun classification type, an example of which is given in Table 2.5. The NER tool helps the model to eliminate NER nouns from being entities. For example, a noun such as ‘Peter’ is classified as a person and the University of Huddersfield is classified as an organisation, so both can be eliminated from the list of candidate entities.

Input	David has been a student at Huddersfield University since 2015.
Output	David/**Person** has been/O a/O student/O at/O Huddersfield/**Organisation** University/**Organisation** since/O 2015/Date ./O
Table keys	O=not classified

Table 2.5 Example of NER

5. Sentence dependencies

Sentence dependencies tools provide grammatical information about a sentence (De Marneffe & Manning, 2008). An example is given in Table 2.6. These tools are easy to use without linguistic expertise.

Input	A student takes a course.
Output	root (ROOT-0, takes-3) det (student-2, A-1) nsubj (takes-3, student-2) det (course-5, a-4) dobj (takes-3, course-5)
Table keys	det: determiner. nsubj: nominal subject. dobj: direct object.

Table 2.6 Sentence Dependency Example

The model uses sentence dependencies to define subjects, objects and verbs for each sentence. Sentence subjects and objects may be mapped into entities and the verb may be mapped into a relationship.

2.3.1 NLP Toolkits

After defining a list of tasks which need to be performed by a natural language processing toolkit, the author needs to choose an NLP toolkit to perform these tasks. There are two types of NLP toolkit (Pinto et al., 2016), Standard NLP toolkits and Social NLP toolkits. Standard NLP toolkits are not designed for any specific task. GATE[5] (Cunningham, 2002), Stanford CoreNLP[6] (Manning et al., 2014), Apache OpenNLP[7] and NLTK[8] (Bird, 2006) are all examples of standard NLP toolkits. Social NLP toolkits are designed for use with short text in social networking. Alan Ritter's TwitterNLP[9], CMU's TweetNLP[10] and TwitIE[11] are examples of social NLP toolkits. The author believes that the natural language text that will be mapped into a conceptual model would be processed more successfully by a standard NLP toolkit than a social NLP toolkit, and therefore no further consideration will be given to social NLP toolkits.

Although many natural language toolkits are referred to in the literature, each of the tools considered by the author was trained for English, available as an open source, extensively used by the NLP community and implemented by Java, which is the most common programming language used in natural language processing applications. The following is a description of the most common NLP toolkits which use Java:

1. General Architecture for Text Engineering (GATE)

GATE is open source software developed at Sheffield University in the UK. It is a powerful tool for solving most text processing problems. The GATE community includes students, developers and scientists. It is active in different language applications, including Voice of the Customer (VOC), cancer research, drug research, decision support, information extraction and semantic annotation (Cunningham, 2002).

2. Apache OpenNLP

Apache OpenNLP is a Java library developed by volunteers and performs popular natural language tasks such as tokenisation, PoS tagging, chunking, NER and parsing by using machine-

[5] https://gate.ac.uk
[6] https://stanfordnlp.github.io/CoreNLP/
[7] https://opennlp.apache.org/
[8] http://www.nltk.org/
[9] https://github.com/lmucs/grapevine/wiki/Twitter-NLP
[10] http://www.cs.cmu.edu/~ark/TweetNLP/
[11] https://gate.ac.uk/wiki/twitie.html

learning techniques. People who use Apache OpenNLP rely on pre-trained models of former tasks (Kwartler, 2017).

3. Stanford CoreNLP

The Stanford CoreNLP is an open source pipeline library based on Java programming language. It was developed at Stanford University in the United States and delivers popular natural language processing tasks. English is the language most supported by the Stanford CoreNLP, but other languages such as Arabic, Chinese, French and German are also supported (Manning et al., 2014), as shown in Table 2.7. The Stanford CoreNLP is easy to download and run, and users are not required to understand complex procedures during the installation.

Annotator	Arabic	Chinese	English	French	German
Tokenise	Yes	Yes	Yes	Yes	Yes
Sentence Splitter	Yes	Yes	Yes	Yes	Yes
Truecase			Yes		
PoS	Yes	Yes	Yes	Yes	Yes
Lemma			Yes		
Gender			Yes		
NER		Yes	Yes		Yes
RegexNER	Yes	Yes	Yes	Yes	Yes
Parse	Yes	Yes	Yes	Yes	Yes
Dependency Parse		Yes	Yes		
Sentiment			Yes		
Coreference Resolution			Yes		

Table 2.7 Tasks and Languages Supported by Stanford CoreNLP (Manning et al., 2014)

Ease of use is another criterion to be taken into consideration by the author in choosing an NLP toolkit. Compared to GATE, the Stanford CoreNLP is easy to install and configure (Pinto et al., 2016), as is the Apache OpenNLP. As a result, the author stopped further considering GATE and started focusing on Stanford CoreNLP and Apache OpenNLP.

Which NLP toolkit performs best depends on the task itself (Al Omran & Treude, 2017), since no toolkit is superior to others for all tasks (Pinto et al., 2016). Each performs well at certain tasks and not at others. This suggests that more than one NLP toolkit could be used for the same application. Sentence segmentation, PoS tagging and NER can be achieved by both Stanford CoreNLP and Apache OpenNLP. Although Pinto et al. (2016) report that OpenNLP outperforms

Stanford CoreNLP in tasks such as PoS, sentence segmentation and NER in news text, Stanford CoreNLP also performs well on these tasks, as mentioned by Toutanova, Klein, Manning and Singer (2003) and Manning et al. (2014). However, sentence dependencies are only supported by Stanford CoreNLP. Therefore, the author is confident to choose Stanford CoreNLP as the toolkit to perform the NLP tasks required by the proposed model. Employing Stanford CoreNLP to achieve the proposed model's natural language tasks leads the author to select Java as the programming language to be used to implement the model. Furthermore, the model needs, at some points, to keep track of user history and to store users' behaviour. Therefore, the model will need to store information on user history in a relational database. There are many relational databases which could be used with the model to achieve this task, such as Microsoft Access, MySQL and Microsoft SQL Server. At the moment, however, the model uses the Microsoft SQL server to store user history.

2.4 Ontologies Overview

In this section, the author reviews ontology topics related to this research. This section is divided into four sub-sections as follows. In Section 2.4.1, the author discusses different types of ontology and decides which type is suitable for the proposed model. Section 2.4.2 explores different methods of ontology creation and the most suitable methods for the proposed model are selected. Section 2.4.3 discusses different data set ontologies. In this section, the author selects which ontology will be incorporated in the model. In Section 2.4.4, the different languages used in ontology creation are discussed and a language to be used in ontology creation within the proposed model is selected.

2.4.1 Ontology Types (Lightweight Ontologies and Formal Ontologies)

Ontologies are represented as a graph with nodes and edges. The concepts are represented by nodes, while relationships are represented by the edges. Concepts are represented by noun phrases in natural language text. For example, 'a person' is a noun phrase representing the concept of a person. The concept of a person can be further divided into sub-concepts which include different instances of persons, such as an employee, a doctor or an engineer (Wong, Liu, & Bennamoun, 2012). An ontology can also be a collection of specifications defined by a shared conceptualisation (Gruber, 1993). This definition emphasises that concepts and relationships between concepts should be defined in a formal language, such as Web Ontology Language (OWL). Formal languages are natural language independent and can allow constraints and axioms to be added into ontologies without including lexical knowledge (Hjelm & Volk, 2011).

Figure 2.3, which is taken from Giunchiglia and Zaihrayeu (2009), shows the ontologies spectrum. Ontologies which have no axioms are called lightweight ontologies and are represented to the left of a red line located in the middle of the figure. Ontologies which have axioms are called heavyweight ontologies (Fürst & Trichet, 2006) and are located on the right side of the line. Lightweight ontologies usually include concepts and terms taken from controlled languages, which include glossaries, data dictionaries and thesauri, whereas heavyweight ontologies contain term relationships with extensive use of axioms to put constraints and rules on the ontological terms. Therefore, these kinds of ontology require the use of formal and descriptive languages.

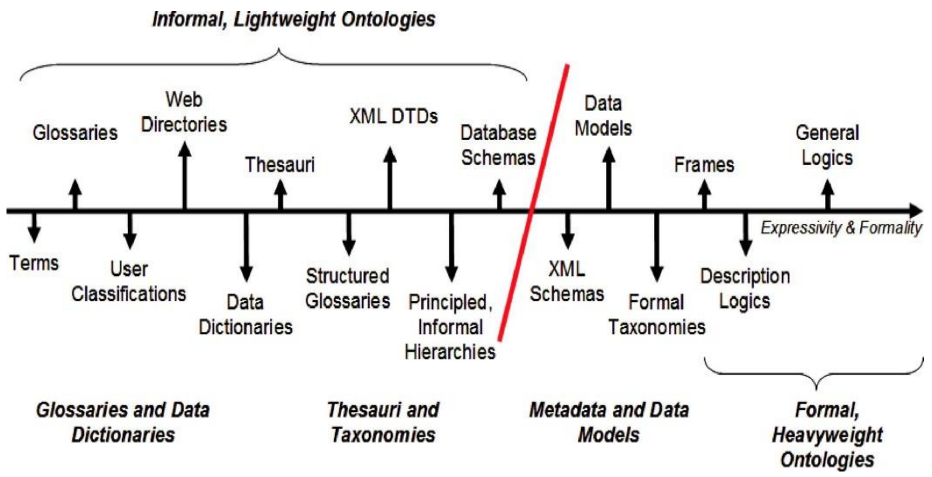


Figure 2.3 Lightweight and Heavyweight Ontologies (Giunchiglia & Zaihrayeu, 2009)

Because (1) more research into axiom extraction is required (Buitelaar, Cimiano, & Magnini, 2005), and because (2), although positively successful, many ontological learning systems are still struggling with the fundamentals of term and relations extraction (Fürst & Trichet, 2006), the author is more confident about selecting an informal, lightweight ontology to be included within the proposed model.

2.4.2 Methods for Creating Ontologies

2.4.2.1 Manual ontology creation

Manual ontology creation requires the expertise of an ontology developer. Sugumaran and Storey (2002) proposed a methodology for manual creation of an ontology to be used for database design automation. This methodology involves several steps and each step includes several heuristics, as follows.

Step 1: Identification of basic terms

In this step, domain terms are identified. Each term and its properties are given a definition. This step is fundamental in the creation of any ontology. For example, if the domain ontology is a hospital, terms such as a doctor, a patient, a clinic, a nurse and medicine should be defined within the set of terms. Since this methodology is proposed for designing ontologies suitable for database design, it is recommended that the ontological terms are linked in some kind of conceptual model, such as an ERD. The main concern in this step is the completeness of terms. 'Completeness' means ensuring that each potential term is included in the terms set. This is not a trivial task, especially when a designer is not knowledgeable about a domain. This completeness is addressed by defining the most frequent terms, along with their synonyms, and by ensuring the ontology can evolve. Ontology evolution is essential to allow the ontology to meet the demands of domain evolution. For example, 'online trading' is a new term in a retail ontology and must be considered in the ontology if it is not already included.

Heuristic 1.1 (identification of most-frequent terms)

The methodology suggests the creation of a use case diagram for the domain. A use case diagram combines the concepts and processes required to describe a domain scenario and is commonly used in system analysis (Jacobson, 1992). By analysing use case diagrams, designers can identify the most basic terms within a domain.

Heuristic 1.2 (identification of synonyms or related terms)

Synonyms of a term can be defined manually or by using an online thesaurus. For example, terms such as 'client' or 'consumer' can be synonyms for the term 'customer'. When there are several possible terms, the most used term should feature in the domain ontology. However, it may also be necessary to include more than one synonym for a term. For example, a 'passenger' and a 'traveller' both need to be included in a travel domain ontology because both are used interchangeably.

Step 2: Identification of relationships

A domain ontology includes complex relationships, and a developer who is not sufficiently familiar with the domain ontology for an application may not feel confident about setting all these relationships. Following heuristics support, however, the developer should be able to capture most types of relationships that occur between domain terms.

Heuristic 2.1 (relationships between basic terms)

There are three common relationships between terms. These are 'is-a' relationships, such as 'a trip' is a kind of 'travel product'; association relationships, for example 'students' are related to ‘departments’; and synonym relationships, for example ‘a customer’ and ‘a passenger’ are synonyms for ‘a traveller’. Capturing these three relationships helps ontology developers to consider most relationships in a domain.

Heuristic 2.2 (relationships between ontologies)

A domain ontology can be huge and wide. Therefore, it is not a trivial mission to keep track of all relationships within the domain. Dividing a domain ontology into sub-ontologies or sub-domains helps in this task of keeping track of relationships. For example, a travel domain can be subdivided into sub-domains of ‘aeroplane travel’, ‘train travel’ and ‘bus travel’. Each sub-ontology/sub-domain has its own terms and relationships. It is a developer’s job to maintain consistency between the domain terms and domain relationships of all sub-ontologies, and following this approach allows the domain ontology to evolve.

Step 3: Identification of term constraints

Constraints help a developer to capture business rules between terms within a domain. When one term depends upon another term, this is called a prerequisite constraint. For example, a ‘payment’ is a pre-requisite for a ‘ticket’. When one term/relationship must occur before another term, this is called a temporal constraint. For example, a ‘booking’ is a temporal constraint for a ‘ticket’. When a term/relationship needs another term/relationship in order to occur, this is called a mutually inclusive constraint. For example, to travel to a foreign country a visa may be required. When terms/relationships cannot occur together at the same time, this is called a mutually exclusive constraint. For example, a customer cannot pay for a trip by credit card and cash at same time. Identifying these four constraints will help in capturing most business rules in a domain.

Step 4: Identification of higher-level constraints capturing domain knowledge

In this stage, a developer should define constraints and facts upon a domain, but not between terms within the domain. There are two types of higher-level domain constraints, which are domain constraints and domain dependency constraints. When constraints are put on domain terms, they are called domain constraints. When constraints are placed on multiple terms and multiple relationships, they are called domain dependency constraints.

Although Sugumaran and Storey's (2002) methodology is systemic and suitable for creating a domain ontology, it is not appropriate to be used in this research. The proposed model within this research aims to create a domain-independent ontology which can support designers in the creation of conceptual models. Using Sugumaran and Storey’s methodology to design such an ontology would be time consuming. The methodology would require the author to define terms, relationships and constraints for domain-independent terms, and the stages involved would be difficult to prepare. The definition of a domain-independent ontology would require the inclusion of unlimited numbers of sub-ontologies, and this goal is unachievable.

2.4.2.2 Ontology learning from text (semi-automated ontology creation)

Ontology learning is a process of identifying terms, concepts, relationships and maybe axioms from text in an automated or semi-automated manner, and using them to evolve an ontology. Techniques from different fields including information retrieval, information extraction, data mining and machine learning are all methods used in this process (Wong et al., 2012). Brewster, Ciravegna and Wilks (2002) proposed a semi-automated methodology for building an ontology via a text corpus and existing ontologies. Liu, Weichselbraun, Scharl and Chang (2005) also proposed a semi-automated method for evolving seed ontologies by using online webpages. The ontology learning process includes a sequence of four outputs, namely, terms, concepts, relationships and axioms. The combination of these outputs creates an 'ontology layer cake' (Buitelaar et al., 2005). In order to deliver each output, certain tasks are undertaken and the techniques employed for each task are different from one system to another, as shown in Figure 2.4.

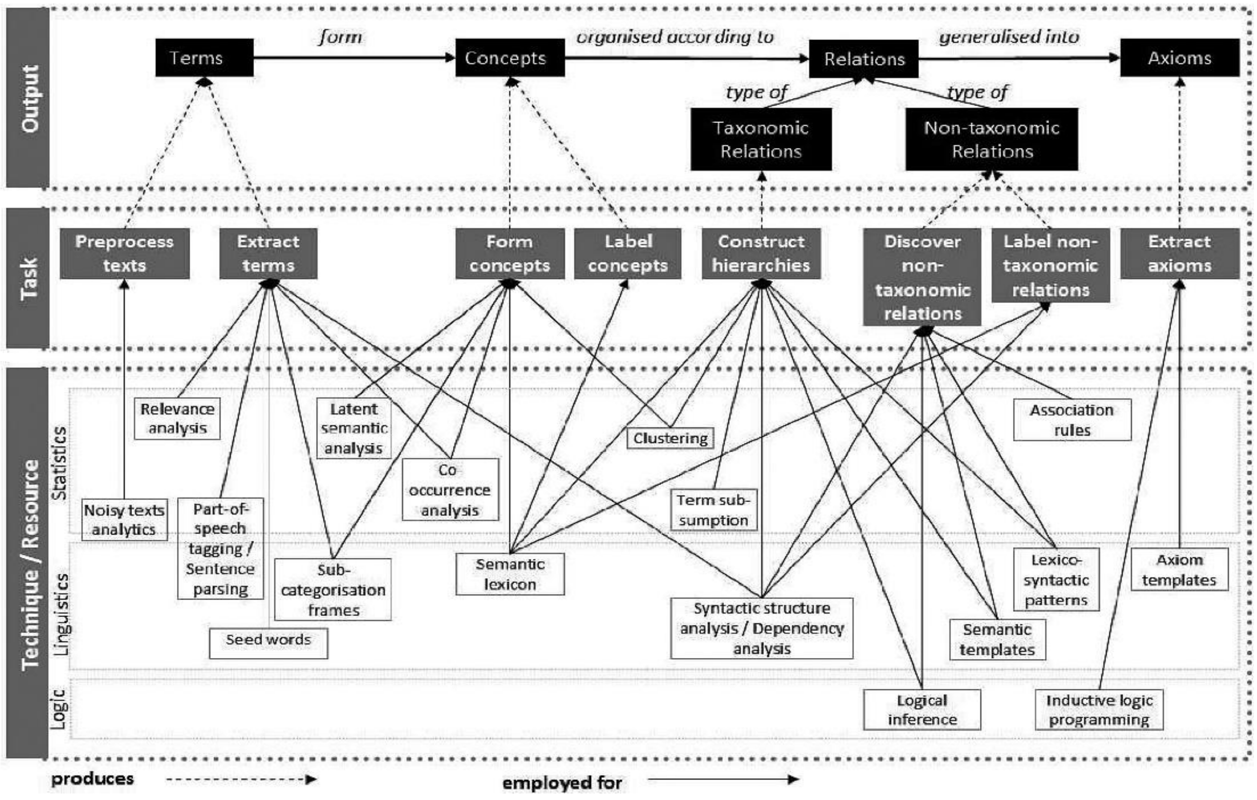


Figure 2.4 Ontology Learning: Output, Tasks and Techniques (Wong, 2009, p.15)

Terms are the fundamental elements of any ontology. Terms can be made of a single word or multiple words (complex terms). Everything important in an ontology is expressed by a term. Pre-processing of text and term extraction are key tasks associated with terms. Noisy text analytics is a technique associated with pre-processing text to ensure the text is ready for term extraction processing. Term extraction is also known as keyphrase extraction (Medelyan & Witten, 2005).
Concepts are created by linking similar terms together. For example, apple tart, egg tart, chocolate tart and French apple tart are linked into a 'tart' concept. Forming and labelling concepts are the key tasks associated with concepts (Wong et al., 2012).
Relationships create interaction between concepts and discovering relationships is not an easy task. A concepts hierarchy is achieved by discovering 'is-a' relationships, which are embedded in hypernym and hyponym relationships. These are called taxonomic relationships (Cimiano, Pivk, Schmidt-Thieme, & Staab, 2005), and the construction of hierarchies is a task for discovering this type of relationship. There are also non-taxonomic relationships. Meronymy and possession are both of this type, and discovering and labelling non-taxonomic relations are further tasks to be set. Identification of interaction between concepts using verbs also helps in discovering non-taxonomic relationships (Wong et al., 2012).

In any ontology, there are usually sentences which must be true all the time, and these kinds of sentences are called axioms. Discovering axioms is a task associated with discovering relationships that meet certain criteria (Wong et al., 2012).

In determining the methodology to be followed in building and evolving an ontology for this study, the following factors have been taken into consideration. (1) Fully-automated ontology learning does not yet exist, and ontology learning does need human intervention (Gómez-Pérez & Manzano-Macho, 2003). (2) The total automation of ontology learning may not be possible (Wong et al., 2012). (3) The majority of ontology learning systems are semi-automated and designed to assist domain experts in curating ontologies (Shamsfard & Barforoush, 2003). (4) Human involvement is therefore still obligatory and desirable (Zhou, 2007). (5) Fully manual ontology creation is time consuming and unlikely to be appropriate for the development of a domain-independent ontology. For these reasons, an ontology learning system (semi-automated ontology creation) has been chosen as the appropriate methodology for building and evolving the open, domain independent ontology that is one of the components of the proposed model.

2.4.2.2.1. Examples of ontology learning systems

1. OntoLearn

OntoLearn is an ontology learning system which implements ontology learning tasks. The system is divided into three phases. In the first phase, the system receives input text from different text sources. The system extracts a domain terminology by using a natural language processor and statistical techniques. Secondly, the system performs semantic interpretation with support from WordNet and Semcor (semantically tagged corpus) (Miller, Leacock, Tengi, & Bunker, 1993). Finally, the system discovers taxonomic relationships and concept similarities, and generates a 'concept forest'. The OntoLearn system was applied in the European 'Harmonise' project for building a tourism ontology and showed a good level of performance. Numerical evaluation shows precision ranging from 72.9% to about 80% and recall of 52.74% (Missikoff, Navigli, & Velardi, 2002).

2. CRCTOL

Concept-Relation-Concept Tuple-based Ontology Learning (CRCTOL) is another system for ontology learning. This system utilises a full parsing method to obtain a more comprehensive level of syntactic information. It also uses a distinguishing approach to concept extraction, which allows the system to extract a set of concepts more precisely. The use of a simple and effective unsupervised word sense disambiguation method to detect the intended meaning of each word helps the system to create correct relations between concepts. The system also has a rule-based technique for non-taxonomic relations extraction. CRCTOL was used to create a terrorism ontology and a sport event ontology, and the results were compared with the Text-to-Onto and Text2Ont systems (Völker, Fernandez Langa, & Sure, 2008). The findings showed that CRCTOL is capable of obtaining concepts and semantic relations with a sophisticated level of precision. The results also showed that the system can create ontologies with a respectable semantic level (Jiang & Tan, 2010).

2.4.2.2.2. Techniques Used for Ontology Learning from Text

1. Statistics-based techniques

Statistical techniques are extracted from fields such as information retrieval, data mining and machine learning. Such techniques are used in the early stages of ontology learning and are involved in terms extraction and concepts extraction (Wong et al., 2012). Common statistics-based techniques are clustering (Wong, Liu, & Bennamoun, 2007), co-occurrence analysis (Budanitsky, 1999), term subsumption (Njike-Fotzo & Gallinari, 2004) and association rule mining (Srikant & Agrawal, 1995).

Clustering technique

A clustering technique measures similarities between ontological terms and divides them into groups to construct an ontology hierarchy or to discover concepts (Lindén & Piitulainen, 2004). Paradigmatic similarity and syntagmatic similarity are two types of similarity. If a term can be substituted for another term, this is called paradigmatic similarity. If a term is related to another term because of the occurrence, this is called syntagmatic similarity. For instance, ‘a knife’ and ‘cut’ are related, but there is no similarity between them. Clustering can be done by attaching each individual term or concept to a group, which is known as agglomerative clustering. Clustering can also be achieved by starting with whole concepts or terms and dividing them into a set of groups, known as divisive clustering (Wong et al., 2012).

Co-occurrence analysis

Co-occurrence analysis is a statistical technique that relies on the occurrence of terms (terms occurring together within a corpus) to define the relations between terms or discover relations between concepts (Bordag, 2008). The occurrence of a group of words is called a collection (Wong et al., 2012). To define the extent to which a collection of words are related, co-occurrence measures are used (Bordag, 2008).

Term subsumption

Term subsumption is a statistical technique used to automatically define term hierarchies (Sanderson & Croft, 1999). Term subsumption defines the most frequent terms in a corpus. As the most frequent terms are those most related to the topic, by finding the relations between them, more information is known about the topic. Then, the hierarchy of terms can be defined by learning the generality and specificity of relations between the most frequent terms (Njike-Fotzo & Gallinari, 2004).

Association rule mining

By determining set pairs of concepts, association rule mining can be utilised to define the associations between the concepts at an appropriate level of abstraction (Jiang, Tan, & Wang, 2007). For example, if {chips, beer} and {peanuts, soda} are given as set pairs of concepts, the association rule is utilised to generalise the pairs and delivers {snack, drink} (Maedche & Staab, 2001).

2. Linguistics-based techniques

Linguistics-based techniques are suitable for most tasks associated with ontology learning from text and they rely on natural language processing tasks. Some linguistics-based techniques rely on PoS tagging, sentence parsing, syntactic analysis and dependencies analysis, while others depend on semantic lexicon, sub-categorisation frames and seed words (Wong et al., 2012).

PoS tagging and syntactic parsing

Part-of-speech tagging and syntactic parsing deliver syntactic structure and dependencies information, which are prerequisites for further text analysis to discover terms and relationships between terms. The Brill Tagger (Brill, 1992) and TreeTagger (Schmid, 1994) are examples of PoS taggers, while Principar (Lin, 1994) and Minipar (Lin, 2003) are examples of sentence parsers. GATE (Cunningham, 2002) and NLTK (Bird, 2006) are examples of natural language toolkits that can achieve most natural language tasks.

Semantic lexical resources

General semantic lexical resources such as WordNet (Miller, Beckwith, Fellbaum, Gross, & Miller, 1990), and domain-specific lexical resources such as the Unified Medical Language System (Lindberg, Humphreys, & McCray, 1993) are common resources used in ontology learning. Many tools and systems employ WordNet in (1) lexical acquisition (O'Hara, Mahesh, & Nirenburg, 1998); (2) word sense disambiguation (Ide & Véronis, 1998); and (3) similarity measurement (Pedersen, Patwardhan, & Michelizzi, 2004). Semantic lexical resources provide access to a huge collection of predefined concepts and relationships. Concepts in semantic lexicon resources are structured into sets of synonyms called synsets. The synsets are utilised for determining terms (Turcato et al., 2000) and for developing concepts. The associations found in semantic lexical resources such as hypernyms, hyponyms, meronyms and holonyms are useful for discovering taxonomic and non-taxonomic relations.

Subcategorisation frame

In the sentence ‘Dave writes an email’, the verb ‘writes’ takes ‘Dave’ as the subject and ‘email’ as an object. This is called a subcategorisation frame (Agustini, Gamallo, & Lopes, 2003). Clearly, Dave is an individual and an email is a written statement, and in overall, an individual and written statement are restrictions of selection for the subject and object of the verb ‘write’. Such restrictions are extracted from text parsers. The restrictions, in cooperation with clustering techniques, are used for concept extraction (Faure & Nédellec, 1998).

Seed words

Seed words and seed terms (Yangarber, Grishman, Tapanainen, & Huttunen, 2000) are used in many systems for many tasks in ontology learning. Seed words deliver good initial facts for the detection of extra terms related to a specific domain (Hwang, 1999) and can guide the automatic building of a text corpus from the web (Baroni & Bernardini, 2004).

3. Logic-based techniques and resources

Logic-based techniques are linked to knowledge representation and reasoning in machine learning (Wong et al., 2012). Inductive logic programming (Lavrac & Dzeroski, 1994) and logical inference (Shamsfard & Barforoush, 2004) are the most commonly utilised logic-based techniques (Wong et al., 2012).

Inductive logic programming

In inductive logic programming, rules are derived from concepts and relationships in the existing collection. These rules are separated into positive and negative examples (Wong et al., 2012). For instance, if training starts with the positive example 'tigers have fur', followed by another positive example 'tigers have fur', a generalisation can be derived, which is 'foxes have fur'. If this is followed by another positive example, 'dogs have fur', the generalisation 'mammals have fur' is obtained by the technique. Once a negative example is met, such as 'humans do not have fur', the generalisation is amended to 'canines and felines have fur' (Oliveira, Pereira, & Cardoso, 2001).

Logical inference

Logical inference extracts new relationships from existing relationships. For example, from existing relations such as 'Socrates is a man' and 'All men are mortal', a new relation can be obtained, which is 'Socrates is mortal'. However, despite the capabilities of inference for extracting new relationships, unacceptable relationships may be obtained if the rules are not complete. For example, the relationships 'human eats chicken' and 'chicken eats worm' can produce an invalid relationship because the intransitivity of eating relationships is not clearly identified in advance (Wong et al., 2012).

Statistics-based techniques are mostly used in the early stages of ontology learning, such as for term extraction and hierarchy construction, but these tasks are not required within the domain-independent ontology proposed in this research. The use of logic-based techniques is not popular in ontology learning and when such techniques are used, it is largely for more complex tasks like axiom extraction. However, axiom extraction is also not required within the ontology proposed for this research. Linguistics-based techniques are appropriate to nearly all tasks in ontology learning and mostly rely on natural language processing tasks (Wong et al., 2012). Therefore, linguistics-based techniques have been chosen for use in this research. PoS tagging is used as the prerequisite for a semantic lexical resource to guide conceptual model extraction from natural language text.

2.4.3 Data Set Ontologies

1. WordNet

WordNet[12] is a lexical ontology developed by Princeton University in 1985 and the latest version of WordNet is 3.1. WordNet includes nouns, verbs, adjectives and adverbs, but word functions such as determinations and prepositions are excluded from the ontology. The words in WordNet are linked by a set of synonyms called a synset, and the ontology includes semantic relationships between each synset. The semantic relationships include is-a, part-of, synonyms and antonyms. The is-a relationship is the basis for creating a synset taxonomic hierarchy. Due to ambiguity in natural language text, a word can have many different meanings and a word can have many synonyms. WordNet can therefore function as a combined dictionary and thesaurus which aims to automatically analyse text and thus help artificial intelligence applications to reduce ambiguity (Fellbaum, 1998).

WordNet can distinguish between entities and non-entities by using noun hierarchy (Du, 2008). It divides nouns into three categories, which are strong entities, mid-entities and non-entities:

- Strong entities: these are further divided into four sub-categories, which are Group, Physical Object, Physical Entity and Thing.
- Mid-entities: these are further divided into four sub-categories, namely, Substance, Event, Communication and Physical Process.
- Weak entities are further divided into five sub-categories, which are Cognition, Attribute, Measure, Constituent and Language unit.

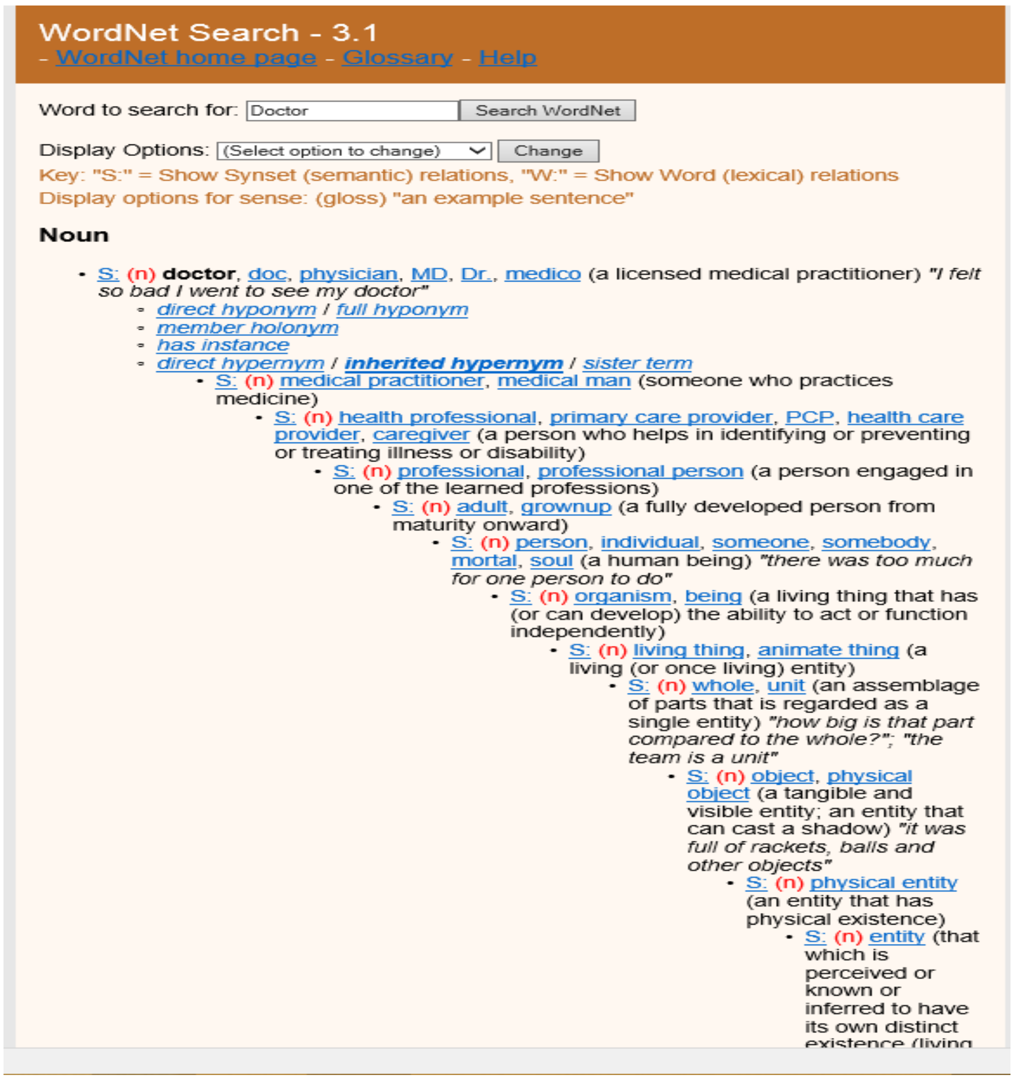
For each noun phrase, a noun hypernym tree is viewed. If the noun's hypernym tree matches one of the categories included in the strong entity group, then the noun phrase is categorised as a strong entity.

Figure 2.5 shows the hypernym chain for the noun phrase 'a doctor'. The noun phrase is sequenced from top to bottom as follows:

Health Professional>Professional>Adult>Person>Organism>Living thing>Whole>Object>**Physical Entity**>Entity.

The hypernym tree for the noun phrase matches 'Physical Entity'. A Physical Entity is categorised as a strong entity, so the noun is considered a strong entity.

[12] https://wordnet.princeton.edu


WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: Doctor [Search WordNet]

Display Options: (Select option to change) [Change]
Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: (gloss) "an example sentence"

Noun

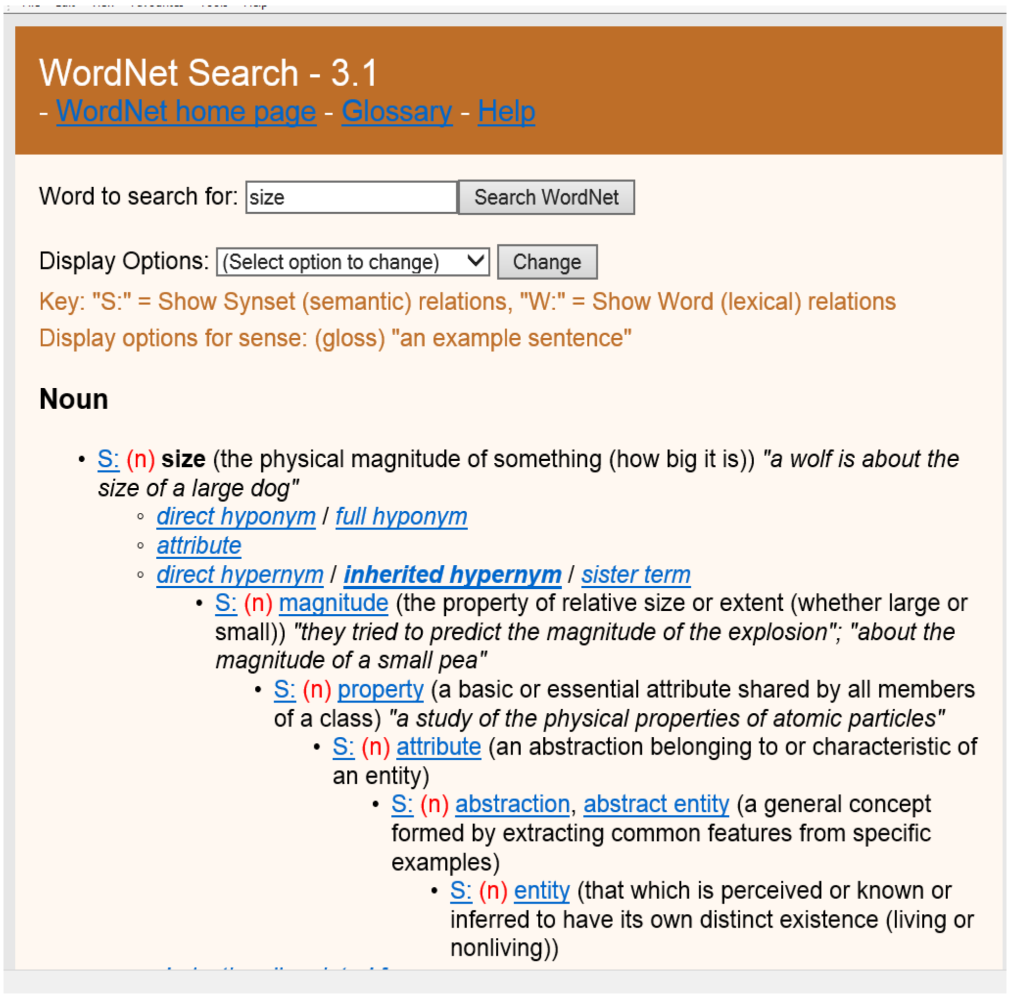
- S: (n) **doctor**, doc, physician, MD, Dr., medico (a licensed medical practitioner) *"I felt so bad I went to see my doctor"*
  - *direct hyponym* / *full hyponym*
  - *member holonym*
  - *has instance*
  - *direct hypernym* / ***inherited hypernym*** / *sister term*
    - S: (n) medical practitioner, medical man (someone who practices medicine)
      - S: (n) health professional, primary care provider, PCP, health care provider, caregiver (a person who helps in identifying or preventing or treating illness or disability)
        - S: (n) professional, professional person (a person engaged in one of the learned professions)
          - S: (n) adult, grownup (a fully developed person from maturity onward)
            - S: (n) person, individual, someone, somebody, mortal, soul (a human being) *"there was too much for one person to do"*
              - S: (n) organism, being (a living thing that has (or can develop) the ability to act or function independently)
                - S: (n) living thing, animate thing (a living (or once living) entity)
                  - S: (n) whole, unit (an assemblage of parts that is regarded as a single entity) *"how big is that part compared to the whole?"; "the team is a unit"*
                    - S: (n) object, physical object (a tangible and visible entity; an entity that can cast a shadow) *"it was full of rackets, balls and other objects"*
                      - S: (n) physical entity (an entity that has physical existence)
                        - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence (living

Figure 2.5 Hypernym Chain for 'Doctor' in WordNet

Figure 2.6 shows the hypernym chain for the noun phrase 'size'. The noun phrase is sequenced from top to bottom as follows:

Property>**Attribute**>Abstraction>Entity>

As the hypernym tree for the noun phrase matches 'Attribute', and the Attribute group is categorised as comprising weak entities, then the noun is eliminated from being an entity.


WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: size [Search WordNet]

Display Options: (Select option to change) [Change]

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

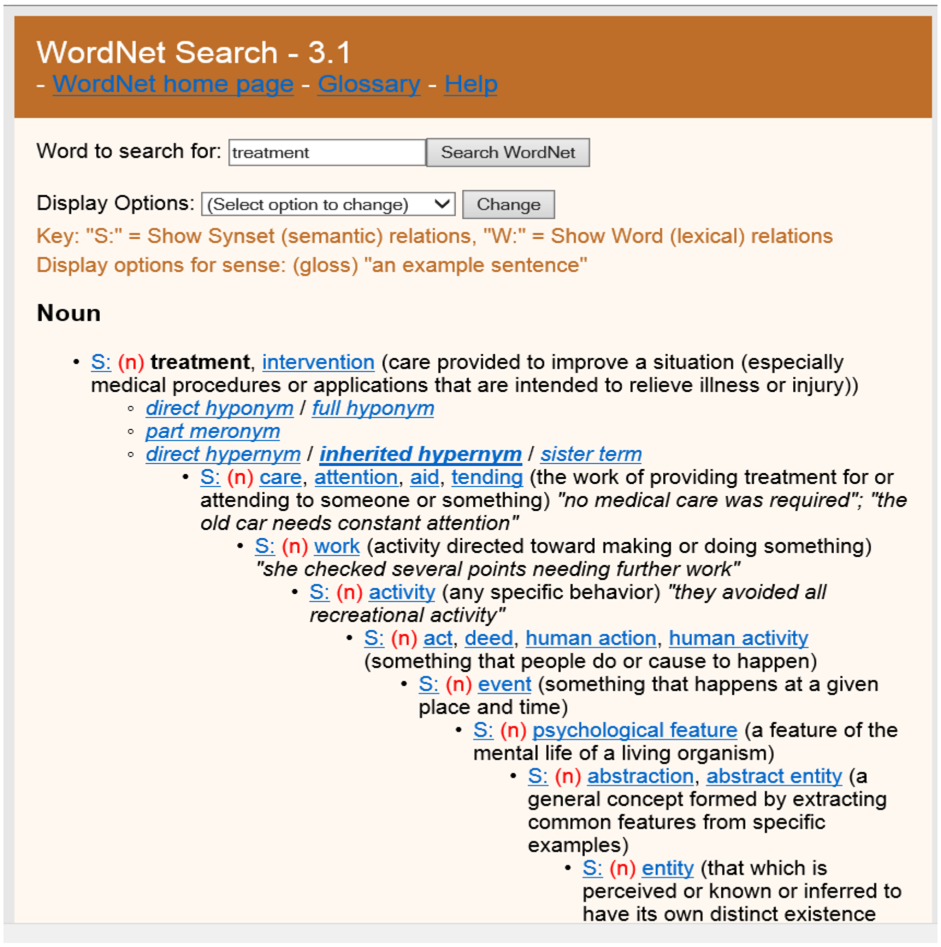
- S: (n) **size** (the physical magnitude of something (how big it is)) *"a wolf is about the size of a large dog"*
  - *direct hyponym* / *full hyponym*
  - *attribute*
  - *direct hypernym* / ***inherited hypernym*** / *sister term*
    - S: (n) magnitude (the property of relative size or extent (whether large or small)) *"they tried to predict the magnitude of the explosion"; "about the magnitude of a small pea"*
      - S: (n) property (a basic or essential attribute shared by all members of a class) *"a study of the physical properties of atomic particles"*
        - S: (n) attribute (an abstraction belonging to or characteristic of an entity)
          - S: (n) abstraction, abstract entity (a general concept formed by extracting common features from specific examples)
            - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

Figure 2.6 Hypernym Chain for 'Size' in WordNet

Figure 2.7 demonstrates the hypernym tree for the noun phrase 'treatment'. The noun phrase is sequenced from top to bottom as follows:

Care>Work>Activity>Act>**Event**>Psychological Feature>Abstraction>Entity

As the hypernym tree for the noun phrase matches 'Event', then the noun is considered a mid-entity. In this case, human intervention may be required to decide whether the noun phrase is kept or eliminated from being an entity.


WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: treatment [Search WordNet]

Display Options: (Select option to change) [Change]

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **treatment**, intervention (care provided to improve a situation (especially medical procedures or applications that are intended to relieve illness or injury))
  - *direct hyponym* / *full hyponym*
  - *part meronym*
  - *direct hypernym* / ***inherited hypernym*** / *sister term*
    - S: (n) care, attention, aid, tending (the work of providing treatment for or attending to someone or something) *"no medical care was required"; "the old car needs constant attention"*
      - S: (n) work (activity directed toward making or doing something) *"she checked several points needing further work"*
        - S: (n) activity (any specific behavior) *"they avoided all recreational activity"*
          - S: (n) act, deed, human action, human activity (something that people do or cause to happen)
            - S: (n) event (something that happens at a given place and time)
              - S: (n) psychological feature (a feature of the mental life of a living organism)
                - S: (n) abstraction, abstract entity (a general concept formed by extracting common features from specific examples)
                  - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence

Figure 2.7 Hypernym Chain for 'Treatment' in WordNet

2. Suggested Upper Merged Ontology (SUMO)

The Suggested Upper Merged Ontology is an high level ontology. SUMO was suggested "as a starter document for the Standard Upper Ontology Working Group, an IEEE-sanctioned group of collaborators from the fields of Engineering, Philosophy and Information Science" (Niles & Pease, 2001). SUMO delivers general definitions of terms and can serve as a basis for domain-dependent ontologies. It is divided into two main levels, which comprise upper level and mid-level ontologies. Figure 2.8 illustrates a snapshot of the upper level hierarchy for SUMO.

```
Physical
    Object
        SelfConnectedObject
            ContinuousObject
            CorpuscularObject
        Collection
    Process
Abstract
    SetClass
            Relation
    Proposition
    Quantity
        Number
        PhysicalQuantity
    Attribute
```

Figure 2.8 SUMO Upper Level Hierarchy (Niles & Pease, 2001)

The root node in SUMO, as in any ontology, is an entity. The entity is further divided into two main concept types, which are Physical and Abstract. Physical concepts include everything which physically exists in space and time, while Abstract concepts include all concepts that are not classified as physical. Physical concepts are further divided into Objects and Processes, while the Abstract class is also further divided into separate concepts, which are SetClass, Proposition, Quantity and Attribute. The mid-level ontologies are attached to upper-level ontologies according to the hierarchy of the upper level ontology. Examples of mid-level ontologies are communications, economy, finance, automobiles and engineering components, food, sports, shopping catalogues and hotels, geography, government and justice, language taxonomy, law, weapons of mass destruction and others[13]. All SUMO concepts are mapped into WordNet synsets. As all SUMO concepts are nouns, they are mapped to synsets of nouns. The relationships used to map WordNet synsets to SUMO concepts are synonyms, hypernyms and instantiation.

3. The DBpedia

Wikipedia is the sixth most widespread website and is used globally. There are Wikipedia versions in 287 different languages, though the sizes of these Wikipedia editions vary from one to another. Some editions contain a couple of hundred articles, while others can reach up to 3.8 million articles. Wikipedia articles are made up of free text (unstructured data), but also contain structured data such as infoboxes, images, lists, tables and categorisations. Wikipedia provides users with a free text search facility, but this search facility does not enable users to find answers to specific questions, such as all the routes to Manchester in the UK which are no lengthier than

[13] http://www.adampease.org/OP/

fifteen miles, or the names of all British singers born in the 18th century. The DBpedia[14] project is a multilanguage knowledge base which extracts structured data from Wikipedia in 111 languages and makes it freely accessible on the web. This structured knowledge can be queried to find answers to the above questions. The biggest DBpedia knowledge base is taken from the English edition and has more than 400 million facts. These facts define more than 3.7 million objects. DBpedia knowledge bases taken from languages other than English define 1.46 billion facts, describing 10 million objects. The DBpedia project maps infoboxes in different languages into a single united ontology. This ontology has 320 classes and includes 1,650 properties (Lehmann et al., 2015).

4. Cyc ontology

The main purpose of the Cyc project[15] is to build a large knowledge base which should be able to support reasoning for a variety of different domains. The project has involved 900 persons and years of effort. The Cyc knowledge base is divided into three ontology levels, which are the upper, middle and lower ontologies. The upper ontology level is the smallest, but is the most widely referenced area of Cyc knowledge base. The middle level is bigger than the upper but smaller than the lower ontology level, and is used to capture the kind of abstraction that is extensively used. Domain-specific ontologies are among the lowest level ontologies in Cyc. The Cyc knowledge base is browsed by using the OpenCyc KB browser, which is available for free download (Matuszek, Cabral, Witbrock, & DeOliveira, 2006). The Cyc ontology has provided a step forward in developing a knowledge base that can help natural language applications with reasoning in a variety of domains. However, it cannot provide comprehensive associations of relationships suitable for supporting the creation of conceptual models.

5. Yet Another Great Ontology (YAGO)

YAGO[16] is an ontology with high coverage and precision, which is automatically extracted from WordNet and Wikipedia. YAGO extracts information from infoboxes and category pages within Wikipedia and combines this with taxonomy relationships in WordNet. The YAGO knowledge base is a combination of entities, relationships and facts. This includes more than one million entities and five million facts, as well as taxonomic and semantic relationships (Suchanek, Kasneci, & Weikum, 2007). The purpose of YAGO is to build a large-scale knowledge base, which is domain independent and automatically extracted with high precision and accuracy. The following provides an example:

[14] http://wiki.dbpedia.org/
[15] http://www.opencyc.org/
[16] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/

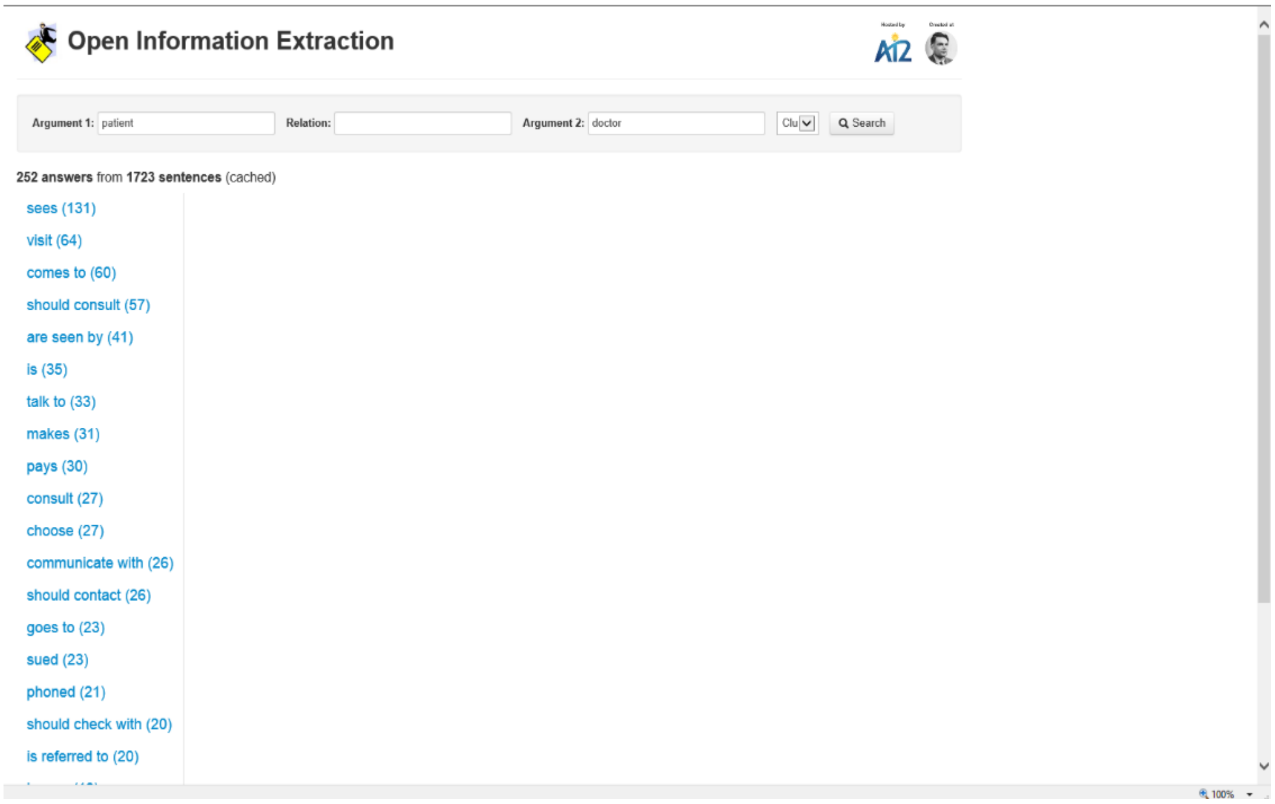
1. Elvis Presley isA singer
2. Singer subClassOf person
3. Elvis Presley bornOnDate 1935-01-08
4. Elvis Presley bornIn Tupelo
5. Tupelo locatedIn Mississippi(state)
6. Mississippi(state) locatedIn USA

All objects are considered as entities. For example, Elvis Presley and Tupelo are entities. Moreover, entities are involved in relationships, as in example number four. YAGO facts are represented in the triple form of entity, relation, entity, such as ‘Elvis Presley hasWonAward Grammy Award’. Each fact has a unique identifier. Numbers and dates are also entities, for example, ‘Elvis Presley BornInYear 1935’ (Suchanek, Kasneci, & Weikum, 2008). However, although YAGO has 1.7 million entities, the majority of them are not suitable for mapping into a conceptual model, as some of them are numbers, some are objects and others are words (text). Furthermore, from exploring the YAGO browser, the author cannot see how YAGO can cover association relationships between entities in a manner that would be suitable for conceptual model extraction.

6. TextRunner

TextRunner[17] provides open information extraction of objects and enables extraction of relationships in tuples (Banko, Cafarella, Soderland, Broadhead, & Etzioni, 2007; Yates et al., 2007). Figure 2.9 represents the result when TextRunner is asked to seek relationships between ‘patient’ and ‘doctor’.

[17] http://openie.allenai.org/

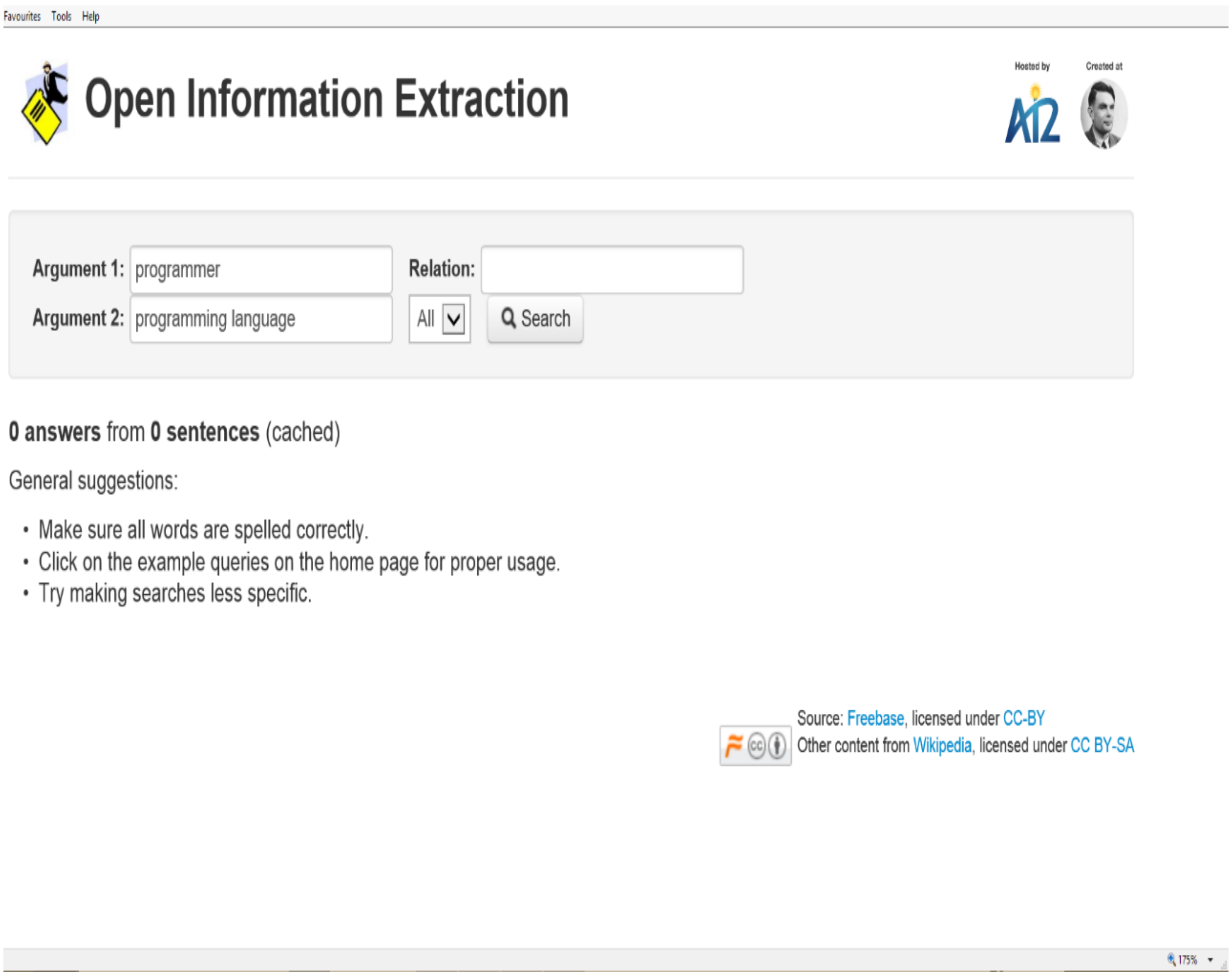


252 answers from 1723 sentences (cached)

sees (131)
visit (64)
comes to (60)
should consult (57)
are seen by (41)
is (35)
talk to (33)
makes (31)
pays (30)
consult (27)
choose (27)
communicate with (26)
should contact (26)
goes to (23)
sued (23)
phoned (21)
should check with (20)
is referred to (20)

Figure 2.9 Relations between Doctor and Patient in TextRunner Ontology

The result returns 252 relations between a patient and a doctor. Analysis of the outcome reveals that some of these answers could be suitable for matching association relationships between entities in conceptual models. For example, ‘Patient is Examined by Doctor’ is one of the 252 answers given by TextRunner when the relationship between patient and doctor is explored. This could match the association relationship between a patient and a doctor in a sentence such as ‘Doctors examine patients in order to prescribe proper treatment’. However, 252 is a huge number of relationships to be given to a user to choose from; this would require a good filtering system to eliminate answers, particularly as some of the 252 answers are synonyms for each other. It is also evident that some of the 252 answers are not suitable for creating association relationships for conceptual models. For example, relations such as ‘is in’, ‘should discuss with’, ‘comes to’, ‘talk to’, ‘choose’, ‘communicate with’, ‘phoned’, ‘leaves’, ‘goes to’, ‘see is in’, ‘rate’, ‘find’, ‘should be taken to’, ‘shook’ and ‘talk with’ are all found within the 252 relations between patient and doctor. Such relationships may not be suitable for the development of a conceptual model for database design, as they are likely to have been extracted from text which is not appropriate for the problem description. Therefore, it cannot be certain that TextRunner would be able to extract suitable relationships for all the expected entities. For example, when an enquiry was made about the relationships between a programmer and programming language, the result was Zero, as shown in Figure 2.10. However, from a sentence such as ‘a programmer uses a programing language and a programming language can be used by many programmers’, there is an important association which needs to be remembered.



Argument 1: programmer
Relation:
Argument 2: programming language
All
Search

0 answers from **0 sentences** (cached)

General suggestions:

- Make sure all words are spelled correctly.
- Click on the example queries on the home page for proper usage.
- Try making searches less specific.

Source: Freebase, licensed under CC-BY
Other content from Wikipedia, licensed under CC BY-SA

Figure 2.10 Relations between Programmer and Programming Language in TextRunner Ontology

After reviewing most of the existing open ontology datasets in the literature, it is clear that none of the existing ontologies would be able to provide full support for the creation of conceptual models. However, WordNet has been found to be the most relevant open-source ontology knowledge base, and it could be useful for the proposed model in this research. It could be employed to distinguish between nouns that represent entities and those which do not represent entities by using a hypernym tree chain for noun phrases. To conclude, existing ontologies are useful but do not provide full support for conceptual model creation. This is because they are designed to be used to provide domain-independent knowledge for different applications, rather than for a specific task. Therefore, in this research, the author will use WordNet as the existing ontology to be integrated with the proposed model in order to deliver a knowledge base for conceptual model creation.

2.4.4 Ontology Languages

Since 1990, many formal languages have been developed for ontology creation. Examples of these languages are KIF (Genesereth & Fikes, 1992), Loom (Brill, 1993), OCML (Motta, 1999), FLogic (Kifer, Lausen, & Wu, 1995) and web-based ontology languages. In this section, the author will describe some of these ontology languages before selecting the most appropriate language to be used for ontology development within the proposed model.

1. KIF

Knowledge Interchange Format (KIF) is a language built on first-order logic and was created by Genesereth and Fikes (1992). Ontolingua (Farquhar, Fikes, & Rice, 1997; Gruber, 1992), the first ontology development tool based on KIF, was established in 1992 by the Knowledge Systems Laboratory (KSL) at Stanford University. KIF can represent concepts, concept taxonomies, relationships and axioms. Because KIF has a high degree of expressiveness, it is challenging to construct reasoning mechanisms for it, and thus KIF does not include reasoning support.

2. Loom

Loom was developed concurrently with Ontolingua at the Information Science Institute (ISI) of the University of South California. Originally, it was not intended for employing ontologies, but for general knowledge bases. Loom, which is built on description logic and production rules, delivers automatic concept classification. Ontology components such as concepts, concept taxonomies, n-ary relations, functions, axioms and production rules can all be expressed by Loom.

3. OCML

OCML was developed in 1993 at the Knowledge Media Institute of the Open University in England. The majority of definitions that are represented by Ontolingua can be also represented by OCML. In addition, more features are defined by OCML. Deductive rules, production rules, functions and operational definitions are examples of additional features expressed by OCML.

4. Web-based ontology languages

Widespread use of the internet has led to the creation of a new generation of ontology languages which can use web characteristics. This group is known as web-based ontology languages, or ontology markup languages (Corcho et al., 2003). Figure 2.11 illustrates the web-based ontology languages and the relationships between them.

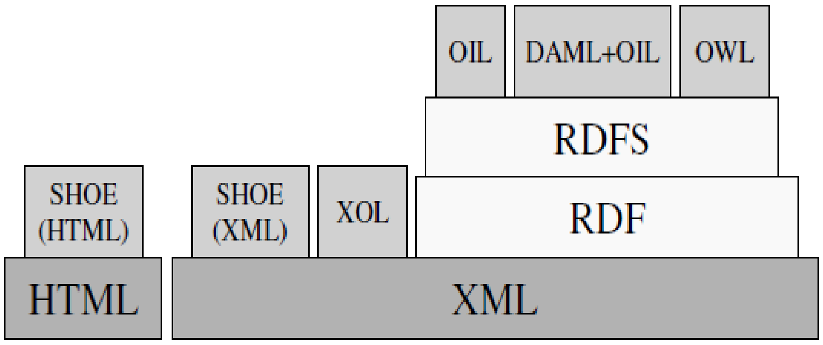


Figure 2.11 Web-Based Ontology Languages (Corcho et al., 2003)

In 1996, SHOE[18] (Luke & Heflin, 2000) was developed as an extension version of HTML at Maryland University. SHOE tags are different from HTML tags, but SHOE can insert ontologies in HTML documents. The language has rules and frames. It can represent concepts, n-ary relations, instances and deduction rules. A SHOE inference engine uses deduction rules to derive new knowledge.

The development of SHOE was followed by that of Extensible Markup Language (XML). XML (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 1997) is extensively accepted and used as a standard language for exchanging information on the web. Then, the SHOE syntax was reformed to be able to use XML syntax, and many other languages are built based on XML syntax.

In 1999, the XML-based Ontology Exchange Language (XOL) (Karp, Chaudhri, & Thomere, 1999) was developed for ontological exchange in the biomedical domain by the Artificial Intelligence Centre of SRI international. However, XOL is a very limited language. It can only represent concepts, concept taxonomies and binary relations. XOL does not include inference mechanisms.

The World Wide Web Consortium (W3C) developed the Resource Description Framework (RDF) (Lassila & Swick, 1999) as a language based on a semantic network for describing web resources. The RDF Schema (Brickley & Guha, 2004) was also developed by the W3C as an updated version of RDF. Both versions, RDF and RDF Schema, are called RDF(S). RDF(S) is not an expressive language. It can only represent concepts, concept taxonomies and binary relations, but constraint checking is included as an inference engine for the language. Three additional languages have been developed as extensions to RDF(S). These languages are OIL, DAML+OIL and OWL. Both OIL and DAML+OIL can represent concepts, taxonomies, binary relations, functions and instances. In 2001, a working group called Web-Ontology (WebOnt) was

[18] http://www.cs.umd.edu/projects/plus/SHOE/spec1.0.html

established by the W3C. The main objective of the group was the creation of a new ontology markup language called Web Ontology Language (OWL).

OWL resulted from the hard work achieved by experts in web semantics, and is now a standard ontology language for the semantic web. The language is compatible with early ontology languages such as RDF(S), SHOE, DAML+OIL and offers additional control to express semantics (Pulido et al., 2006). OWL includes three versions, namely, OWL Full, OWL DL and OWL Lite (Berendt et al., 2004). OWL Full is used when it is necessary to be fully compatible with RDF at both syntactic and semantic levels. OWL DL allows more efficient reasoning but lacks some compatibility with RDF. OWL Lite is an expressive language with decidable inference, and this version is most preferred by developers (Pulido et al., 2006).

If development of an ontology is required, it is important for the developer to consider what sort of expressiveness and inference the ontology will need. Not all ontology languages represent the same components and not all languages are reasoned in the same way (Corcho et al., 2003). The ontology to be included within the proposed model is a lightweight ontology. It will include concepts, entities and relations between entities. The ontology will not include either axioms or reasoning services. Such an ontology can be built by different ontology languages, such as KIF, Loom, XML, RDF(S) and OWL. In the future, however, the ontology component within the proposed model may need to be upgraded, whereby rules and axioms may be added to the ontology. Thus, it will be better to choose an expressive ontology language, even though currently, the ontology component within the proposed model is lightweight. Dermeval et al. (2016) conducted a study to determine which ontology languages are used in requirement engineering. Dermeval et al. found that OWL was employed to develop ontologies within the majority of the studies considered, and OWL was reported to be the most expressive and widely accepted ontology language. Thus, the author is confident to use OWL to define ontology components within the proposed model.

2.5 Chapter Summary

This chapter started by providing an introduction that included a review of the main problems working against the creation of conceptual models, followed by a review of approaches used for the extraction of conceptual models from natural language text. The problems facing the creation of conceptual models are (1) natural language text problems and (2) other difficulties working against conceptual model creation.

Researchers use different approaches for mapping conceptual models from natural language text. These include linguistics-based, pattern-based, cases-based and ontology-based approaches. None of these approaches works perfectly all the time and each approach has its advantages and disadvantages. Therefore, the author has decided to incorporate a linguistics-based approach with an ontology-based approach in order to produce a model which can support the creation of conceptual models. The model will include a domain-independent ontology that is capable of learning from natural language specifications provided by users, and which can update itself and retrieve information to support designers in creating of conceptual models from natural language text. To achieve such a mission, the author has reviewed natural language tasks, defined which tasks need to be incorporated, and selected Stanford CoreNLP as the toolkit that will be employed to achieve natural language tasks. As the application requires the development of an ontology, the author has needed to review ontology types and select a suitable type for the purpose, and to review methods used in ontology development and select a suitable approach to produce the best ontology component for the proposed model. In addition, the chapter has reviewed existing ontologies to determine how these ontologies may assist the proposed model, raising the question of whether any existing ontology could be incorporated within the proposed model to support conceptual model creation. Finally, the author has reviewed ontology languages and selected a suitable language for use in developing the ontology.

Ontology types are reviewed in Section 2.4.1. The author has chosen to create a lightweight ontology. The author believes that a lightweight, domain-independent ontology, which will include concepts, terms and relationships between real-world concepts, can improve the creation of conceptual models.

In Section 2.4.2, the author has reviewed ontology creation methods. There are two methods of ontology development, manual and semi-automated. Automated ontology development is not currently possible. Manual development is challenging and time consuming, and thus the author has selected a semi-automated approach to develop the ontology for the proposed model.

In this section, the author has also reviewed some examples of existing ontology learning systems, in order to introduce examples of such systems to readers. In addition, techniques used for ontology learning have been evaluated. These include statistics-based techniques, logic-based techniques and linguistics techniques. Linguistics-based techniques are appropriate for nearly all tasks in ontology learning and mostly rely on natural language processing tools; thus, the author will incorporate linguistics techniques within the proposed model. PoS tagging will be used as a prerequisite for the semantic lexical resource to guide conceptual model extraction from natural language text.

In Section 2.4.3, the author has reviewed a set of existing ontologies. To the best of the author's knowledge, no existing ontology can deliver full support for conceptual model creation. However, the author has selected WordNet as an existing ontology that provides some sort of support for the conceptual model creation process. WordNet will be employed to distinguish between nouns that represent entities and those that do not represent entities by using hypernym tree chains for noun phrases.

Finally, in Section 2.4.4, ontology languages have been reviewed, and OWL has been selected as a standard and expressive language for development of the ontology component within the proposed model.

Chapter 3: Rules to Derive a Conceptual Model from Natural Language Text

In this chapter, the author reviews rules that may help in extracting conceptual model components such as entities, relationships and attributes from natural language text. The chapter is divided into six main sections. Rules for determining entities are discussed in Section 3.1. In Section 3.2, the author selects which rules will be applied to determine entities in the proposed tool. Rules for determining relationships between entities are discussed in Section 3.3. In Section 3.4, the author selects which rules will be used to determine relationships in the proposed tool. Rules for determining the attributes of entities are discussed in Section 3.5. The findings from this review and a summary of the chapter are given in Section 3.6.

3.1 Rules to Determine Entities

1. Common Nouns Represent Entities

A common noun is a word in the English language that relates to either things or objects, for example, a person, doctor, school, chair, restaurant, etc. A person is a common noun, whereas Smith, Johan and William are not, as they relate to specific persons. These would be proper nouns. Proper nouns have two specific features: they refer to one-of-a-kind items and begin with a capital letter. Common nouns can represent entities in ERMs (Chen, 1983; Tjoa & Berger, 1994). For example, in the sentence, ‘A person owns a car and may belong to a political party’ (Chen, 1983), the common nouns ‘person’, ‘car’ and ‘party’ can be mapped into entities to form the ERD.

However, the current author asserts that this rule is not entirely accurate. Common nouns may represent entities, but in reality this is not always the case. Not all common nouns in a script are suitable for mapping into entities. This can be demonstrated by the following example sentence: ‘*The goal of this case study is to design a system for the university to keep the records of numerous departments, lecturers and students*’. In the Penn Treebank project ‘NN’ is a tag that represents singular common nouns, while ‘NNS’ represents plural common nouns (Santorini, 1990). Table 3.1 demonstrates the PoS tags for the above example.

Input	*The goal of this case study is to design a system for the university to keep the records of numerous lecturers, departments and students.*
Output	The/DT goal/NN of/IN this/DT case/NN study/NN is/VBZ to/TO design/VB a/DT system/NN for/IN the/DT university/NN to/TO keep/VB the/DT records/NNS of/IN numerous/JJ lecturers/NNS ,/, departments/NNS and/CC students/NNS./.
Table keys	DT: Determiner. NN: Common noun, singular. NNS: Common noun, plural. VBZ: Present tense verb. TO: to VB: Verb, base form. CC: Coordinating conjunction. IN: Preposition or subordinating conjunction. JJ: Adjective.

Table 3.1 PoS Tagging for a Sentence

Within the example sentence, there are several common nouns, which include a goal, case study, system, university, record, lecturer, department, and student. However, not all the common nouns in the sentence should be mapped into entities. In fact, only three nouns out of the eight are mapped into entities. A Requirement Specification Text (RST) is a collection of such sentences, and the sentence count differs from one RST to another. Consequently, there are many common nouns that will not be mapped into entities.

2. A Sentence Subject Represents an Entity

A sentence subject describes a part of speech that establishes an action. For example, in the sentence, ‘Students work on modules’, it can be determined that ‘students’ are the subject of the sentence. The sentence subject represents an entity (Sagar & Abirami, 2014). However, not all sentence subjects within requirement specification text are mapped into entities. For example, in the sentence ‘A company is distributed over several branches’, ‘company’ is the sentence subject, but ‘company’ does not represent an entity. From this sentence, a system designer can understand that either there are several branches in the company, or the company has several branches but there is only one company. Therefore, although ‘company’ is the subject for the above sentence, it should not be mapped into an entity. Mapping each sentence subject within a requirement specification text into an entity can result in incorrect entities.

3. A Sentence Object Represents an Entity

A sentence object describes a part of speech that receives an action, and a sentence object can be mapped into an entity (Sagar & Abirami, 2014). For example, in the sentence ‘A student works on modules’, the noun ‘modules’ is the object of the sentence and can be mapped into an entity. In this sentence there are two entities, namely, ‘student’ and ‘modules’, and the relationship between them is many-to-many. A student can work on many modules and a module can have many students working on it. However, not every sentence object in requirement specification text can be mapped into an entity. In the sentence, ‘A student has a name’, ‘name’ is a sentence object but it would be mapped into an attribute for a ‘student’ entity. Thus, mapping each sentence object within a requirement specification text can result in incorrect entities.

4. A Proper Noun Represents an Entity

A proper noun may incorrectly represent an entity (Omar et al., 2004). This rule may be overlooked because proper nouns represent people, countries and things. However, people, countries and things can also represent a record or an attribute, as shown in Table 3.2.

Input	There are five airlines in different countries: Libya, Egypt, UK, Tunisia and France. The customers could come from any state, not just the above, and from any city.
Output	There/EX are/VBP five/CD airlines/NNS in/IN different/JJ countries/NNS :/: Libya/NNP ,/, Egypt/NNP ,/, UK/NNP ,/, Tunisia/NNP and/CC France/NNP ./. The/DT customers/NNS could/MD come/VB from/IN any/DT state/NN ,/, not/RB just/RB the/DT above/JJ ,/, and/CC from/IN any/DT city/NN./.

Table 3.2 Stanford Parser Defines Common Nouns and Proper Nouns

In Table 3.2 there are five proper nouns. Within the script in Table 3.2, there are three candidate entities, which are ‘airline’, ‘country’ and ‘customer’, all of these being common nouns. If all proper nouns within the script were also mapped into entities, there would be eight entities. As a result, five out of the eight entities would be incorrectly classified as entities, which would lead to a dramatic reduction in the precision of extraction.

5. Noun Category Entities

Some people suggest that a noun phrase can be a class entity if it belongs to a specific class, such as people, places, physical things, organisations, events, transactions, interactions, policies and containers (Song et al., 2004). To the best of the author’s knowledge, definition of such classes relies on human intervention, as there is no tool that can achieve such a task. Some of these classes are explained below.

People: this class represents persons who are employed to achieve particular functions in requirement specification text. Examples are a student, doctor and nurse.

Places: this class represents places where business activities take place. Examples are a hospital, university and bank.

Physical things: this includes nouns that are important in a requirement specification text, such as a product, book or device.

Organisations: this class represents important units in a requirement specification text, such as a branch, department and team.

Events: these are sometimes called transactions. Examples are payment, booking and order.

Containers: this class is able to hold or carry things, such as a store or a bin.

6. WordNet Entities

WordNet divides nouns into three groups: strong entities, mid-entities and weak entities. A noun phrase hypernym chain is obtained and, if a noun phrase's hypernym matches the strong entities group, then the noun phrase is mapped into an entity. If a noun phrase's hypernym matches the weak entities group, then the noun phrase is eliminated from being an entity. If a noun phrase's hypernym matches the mid-entities, human intervention is employed to decide whether the noun phrase should be mapped into an entity or eliminated from being an entity (Thonggoom, 2011).

7. Domain Independent Rules

Thonggoom (2011) developed the Heuristic-Based Technique (HBT) for extracting a conceptual model from natural language text. The HBT is based on six domain-independent rules obtained as a result of twenty years' work on a teaching database. These rules can be used to teach novice designers how to develop conceptual models. Examples of these rules are given below.

Identifier rule

If a noun phrase needs to include a unique identifier, then it can be mapped into an entity. For example, an identification number is required for a student in a university, so a student can be an entity.

Multi-attributes

If a noun phrase can include multiple attributes, it can be mapped into an entity. For example, a student can have many attributes, such as an address, a telephone number etc., and therefore a student can be an entity.

Multi-value attributes

If a noun phrase can represent attributes with multiple values, then it can be mapped into an entity. For example, a telephone number for a person can have several values, as a person can have more than one phone number.

Domain-importance rule

If a noun phrase is important within a requirement specification text, then it can be mapped into an entity. For example, a doctor or a patient are important within a requirement specification text that describes a hospital.

3.2 Approach Applied for Entity Extraction

The rules that are used to map noun phrases into entities are not complete. There is no rule that is true all of the time. Common nouns, sentence subjects and sentence objects can all be mapped into entities, but not every common noun, sentence subject or sentence object within a requirement specification text should be mapped into an entity. Rules that use noun categories and domain-independent rules can give accurate results if applied in an appropriate way, but these rules cannot be automatically applied. There is no tool that is able to apply such rules to natural language text and extract entities. Therefore, the rules need human intervention. On the other hand, extraction of entities using WordNet can be fully automated, and therefore, the author has chosen to use WordNet for entity extraction. In the proposed model, the system will also use a conceptual model ontology to search for noun phrases found in natural language text. If a noun phrase is found in the ontology, then it will be mapped into an entity. Furthermore, the author plans to use human intervention to consider noun phrases that are not found in the ontology and not defined by WordNet as entities. The human intervention will apply the domain-importance rule and the Multi-attributes rule to either accept or reject a noun as an entity. Section 4.1.2 presents detail and a flowchart diagram of how entities will be extracted in the proposed model.

3.3 Rules to Determine Relationships between Entities

Identified below are the most common rules for extracting relationships from the text of a requirements specification.

1. A transitive verb determines a relationship between entities (Chen, 1983; Elbendak, 2011; Btoush & Hammad, 2015). A transitive verb is a verb that has a noun to receive an action and a noun to do the action. For example, in the sentence ‘A student takes a course’, ‘student’ and ‘course’ are entities. ‘Student’ is the subject of the sentence and does the action ‘take’, while ‘course’ is the object of the sentence and receives the action.

2. If a preposition such as ‘on’, ‘in’, ‘by’ or ‘to’ comes after a verb, this may indicate a relationship between entities (Btoush & Hammad, 2015; Omar et al., 2004; Sagar & Abirami, 2014). For example, in the sentence ‘An employee works on a project’, ‘employee’ and ‘project’ are entities and the relationship between them is ‘works on’.

3. “The adjective ‘many’ or ‘any’ may suggest a maximum cardinality” (Omar et al., 2004). For example, in the sentence ‘A doctor treats many patients’, the group of ‘patients’ that ‘a doctor’ treats consists of more than one and could be any number.

4. “A comparative adjective ‘more’ followed by the preposition ‘than’ and a cardinal number may indicate the degree of the cardinality between two entities” (Omar et al., 2004), for example, ‘A patient is treated by more than one doctor’.

5. The need-to-know rule specifies that if a verb represents a relationship between entities that need to be remembered in a problem specification, then there is a relationship between the entities (Thonggoom, 2011). For example, the sentence ‘Each plant is divided into departments’ indicates that there is a relationship between ‘plant’ and ‘departments’, the relationship being that each plant is divided into departments. It is therefore important to know and remember how many departments each plant is divided into and to which plant each department belongs.

3.4 Approach Applied for Relationship Extraction

In the view of the author, the rules used in relationship extraction are not sufficient to extract a good set of relationships for a conceptual model. This is because they are not enough to satisfy the syntax variables within requirement specification scripts. Syntax variables are the many different ways of inferring the same thing, for example, ‘John is employed by the company’ or ‘John is an employee of the company’. Furthermore, the relationship extraction rules would require human intervention to be applied to natural language text. There is no existing tool which can be used to extract relationships for conceptual models. Therefore, for binary relationship extraction, the author suggests the use of an integrated approach combining Stanford typed dependencies with a conceptual model ontology and human intervention. Stanford typed dependencies can be used to extract relationships between sentence subjects, sentence objects and verbs in a requirement specification text, while the conceptual model ontology is used to retrieve the relationships between entities that are stored in it. Human intervention will then be used to either accept or reject the relationships extracted by Stanford typed dependencies and the conceptual model ontology, and to apply the need-to-know rule to identify relationships between the extracted entities. Section 4.1.3 presents detail and a flowchart of relationship identification. Non-binary relationship extraction is outside the scope of this research.

3.5 Rules to Determine Attributes

Identified below are the most common rules for extracting attributes from the text of a requirements specification.

1. A possessive noun phrase might signify an attribute of a noun (Elbendak, 2011; Btoush & Hammad, 2015; Omar et al., 2004; Slankas, 2015; Sagar & Abirami, 2014). For example, in the sentence ‘An employee’s address is stored in the database’, ‘address’ is an attribute of the ‘employee’ table.

2. “The genitive case when referring to a relationship of the possessor using the ‘of’ construction signifies an attribute” (Sagar & Abirami, 2014). For example, in ‘The name of the student is stored’, the ‘name’ may represent an attribute of the ‘student’ table.

3. Noun phrases in two parts where the second part is an abbreviation may represent an attribute (Btoush & Hammad, 2015). Examples of this are ‘vehicle no.’ and ‘employee ID’.

4. A noun phrase that comes after the verb ‘has / have’ may represent an attribute or group of attributes (Btoush & Hammad, 2015). For example, in ‘Each dependent has a unique ID and name’, the ‘ID’ and the ‘name’ are attributes of a ‘dependent’ table.

5. A noun phrase that follows the verb phrase ‘identified by’ and ‘recognised by’ might represent attributes (Elbendak, 2011; Gomez et al.,1999; Sagar & Abirami, 2014). The attribute in this case might be a primary key for an entity. For instance, in ‘A patient is identified by ID’, the ‘ID’ is not only an attribute of a ‘patient’ entity, but also a primary key for the entity.

6. An adjective in English might represent an attribute (Chen, 1983; Elbendak, 2011; Tjoa & Berger, 1994; Sagar & Abirami, 2014). An adjective in English describes a noun. For example, in the sentence ‘A large item has extra charges on carriage and delivery’, ‘large’ is an adjective that describes an ‘item’. An ‘item’ has an attribute, ‘size’, which can have a value such as large, medium, standard and small.

7. If there is a relationship between entities, an adverb might represent an attribute of the relationship (Chen, 1983). For example, in ‘An employee works at a company for 20% of his time’, the ‘employee’ and ‘company’ are entities and there is a relationship, ‘works at’, between them. ‘20% of his time’ is an adverb that adapts the verb phrase ‘works at’, and consequently, time percentage can be an attribute of the relationship between an employee and a company.

8. If “a sentence has the form ‘X of Y is Z’, and Z is not a proper noun, we may treat X as an attribute of Y” (Chen, 1983). For example, in the sentence ‘The colour of the desk is blue’, since ‘blue’ is not a proper noun, we may infer that ‘colour’ is an attribute of the entity ‘desk’.
9. Numeric operations may represent attributes (Chen, 1983). For example, in ‘The average salary is £20,000, and the maximum credit limit is £500’, there are two algebraic operations, ‘average’ and ‘maximum’. As such, a ‘salary’ and ‘credit’ are attributes of an ‘employee’ entity.

There are many rules used to define attributes within natural language text. However, these rules are not sufficient to attach a good set of attributes for each entity within a conceptual model. Each individual describes requirement specifications in his own words with no idea about the rules used to extract entities, attributes and relationships. The individual may have no awareness of the meaning of entities, attributes and relationships.
Applying the above rules could therefore result in an unsatisfactory set of attributes, with incorrect attributes attached to incorrect entities. For example, when Rule number 1, which states that a noun phrase with possessive case might signify an attribute of an entity, is applied to a sentence containing the phrase ‘company’s hierarchy’, then ‘hierarchy’ could be attached as an attribute of the entity ‘company’. In reality, however, this is incorrect. Furthermore, in applying Rule number 2, ‘application’ may be interpreted as an attribute of ‘computer’ as a result of the noun phrase ‘applications of computers’, and ‘number’ could be regarded as an attribute of a ‘skill’ entity as a result of the noun phrase ‘number of skills’. This would result in both incorrect extraction of entities and incorrect attachment of attributes. Because the rules used for attributes extraction cannot be universal, and there is no existing tool that is capable of attributes extraction, the author has decided not to include attributes extraction in the proposed model. The author believes that details such as attributes can be included during the design of a logical model or a physical model. It is proposed that if attributes need to be included during the design of the conceptual model, human intervention should be employed to perform this task.

3.6 Chapter Summary

From the above review, the author has learned that rules cannot be sufficiently universal to satisfy the syntax variables within requirement specification scripts. Syntax variables are the many different ways of inferring the same thing. Furthermore, the linguistic rules which are used for mapping natural language text into conceptual models are incomplete. Such rules are sometimes valid and sometimes invalid; there is no rule that is true at all times. For example, common nouns can represent entities, but not every common noun within a requirements specification should be mapped into an entity. This raises the issue of how to differentiate between common nouns that represent entities and those that do not. To solve such problems, human intervention must be used.

Linguistic rules are overlapped and it is not possible for a large group of rules to work together. For example, nouns can be mapped into entities, but nouns can be mapped into attributes as well. If these two rules are used together, a problem occurs with regard to whether the nouns should be mapped into entities or attributes. Therefore, linguistic rules can only provide a basic service in developing a tool to map natural language text into a conceptual model. In order to achieve a better result, only a minimum number of these rules should be used in developing such a tool, and the group of rules must not be overlapped or conflicted. Furthermore, the use of rules should be integrated with human intervention to ensure that valid outputs are obtained.

Entities can be mapped from common nouns, sentence subjects, sentence objects, noun categories, WordNet and domain-independent rules. In this research, the author proposes to integrate WordNet with the domain-importance rule and multi-attributes rule (domain-independent rules) to extract entities from natural language text. For relationship extraction, the Stanford typed dependencies will be integrated with a conceptual model ontology and human intervention. Attributes extraction will not be included in the proposed model. Because of the unavailability of good rules suitable for attributes extraction from natural language text, the author recommends the use of human intervention for this purpose.

Chapter 4: Implementation of Semi-Automated Conceptual Model Extraction System (SACMES)

Rather than developing a domain-independent ontology to help designers map natural language text into conceptual models, which would be extremely time-consuming, the author's aim is to produce a model that can learn from natural language specifications and store what it has learnt in an ontology to update it. The model will also store designers' behaviours in a database and use these behaviours when it processes a new situation. Consequently, the model will improve its performance and reduce the need for human intervention. In this chapter, the Semi-Automated Conceptual Model Extraction System (SACMES) is introduced. The chapter is divided into three sections. Section 4.1 demonstrates the SACMES architecture and Section 4.2 presents a demonstration of how SACMES is used to process requirement specifications. The chapter summary is given in Section 4.3.

4.1 System Architecture

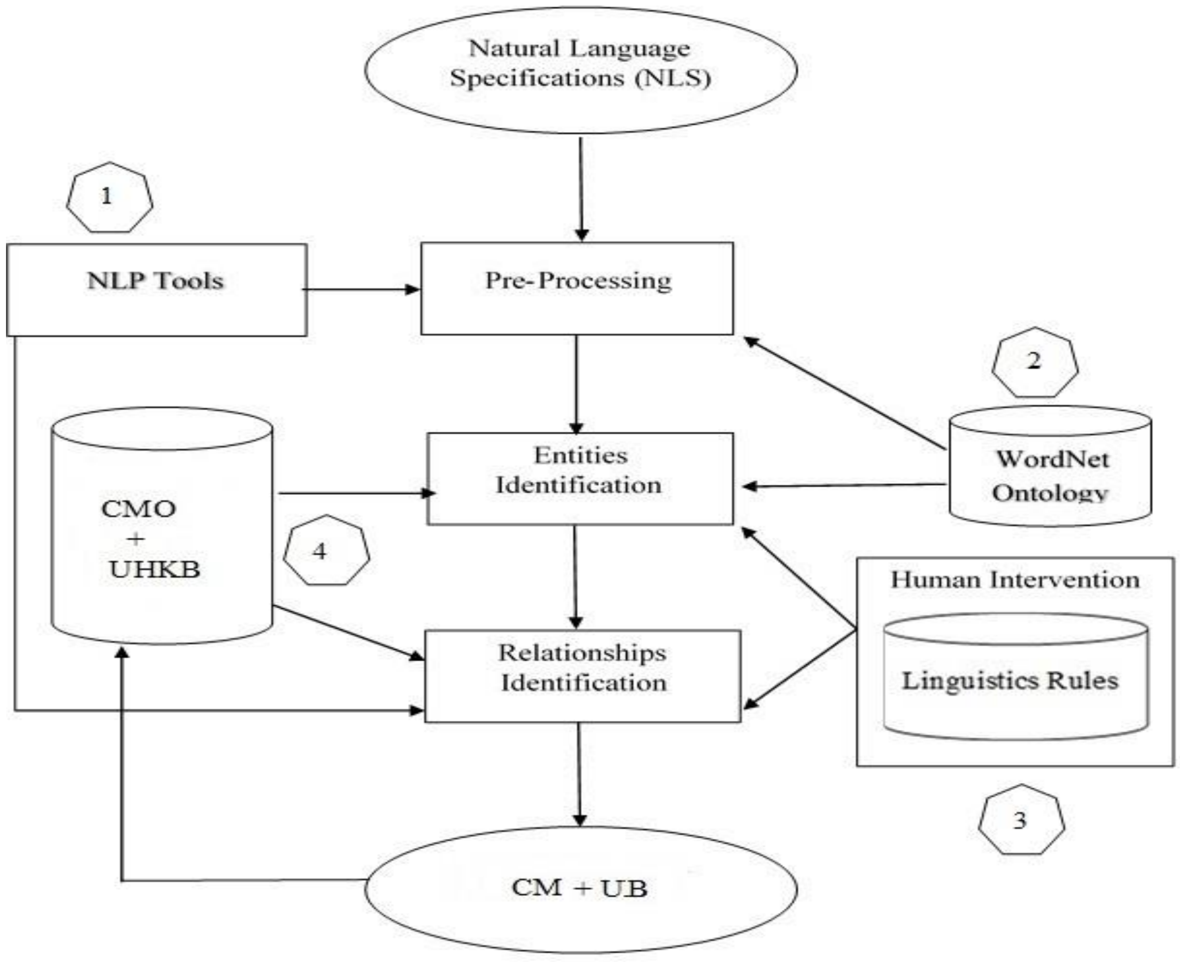


Figure 4.1 SACMES Architecture

Figure 4.1 illustrates the architecture of the model. The model integrates natural language processing tools, WordNet ontology, linguistic rules, a Conceptual Model Ontology (CMO), and a User History Knowledge Base (UHKB) to help designers produce conceptual models from natural language text. The model is implemented by Java programming language. The input for the system is natural language specifications that describe a specific problem.

1. Natural language Processing

The model employs a natural language processing component to perform the natural language tasks required by the model. At the pre-processing stage, the natural language processing component helps in identifying noun phrases that are included in the text. Natural language processing also helps also in eliminating nouns and noun phrases that are unlikely to be entities. Furthermore, natural language processing helps in identifying relationships between entities by using Stanford typed dependencies.

2. WordNet Ontology

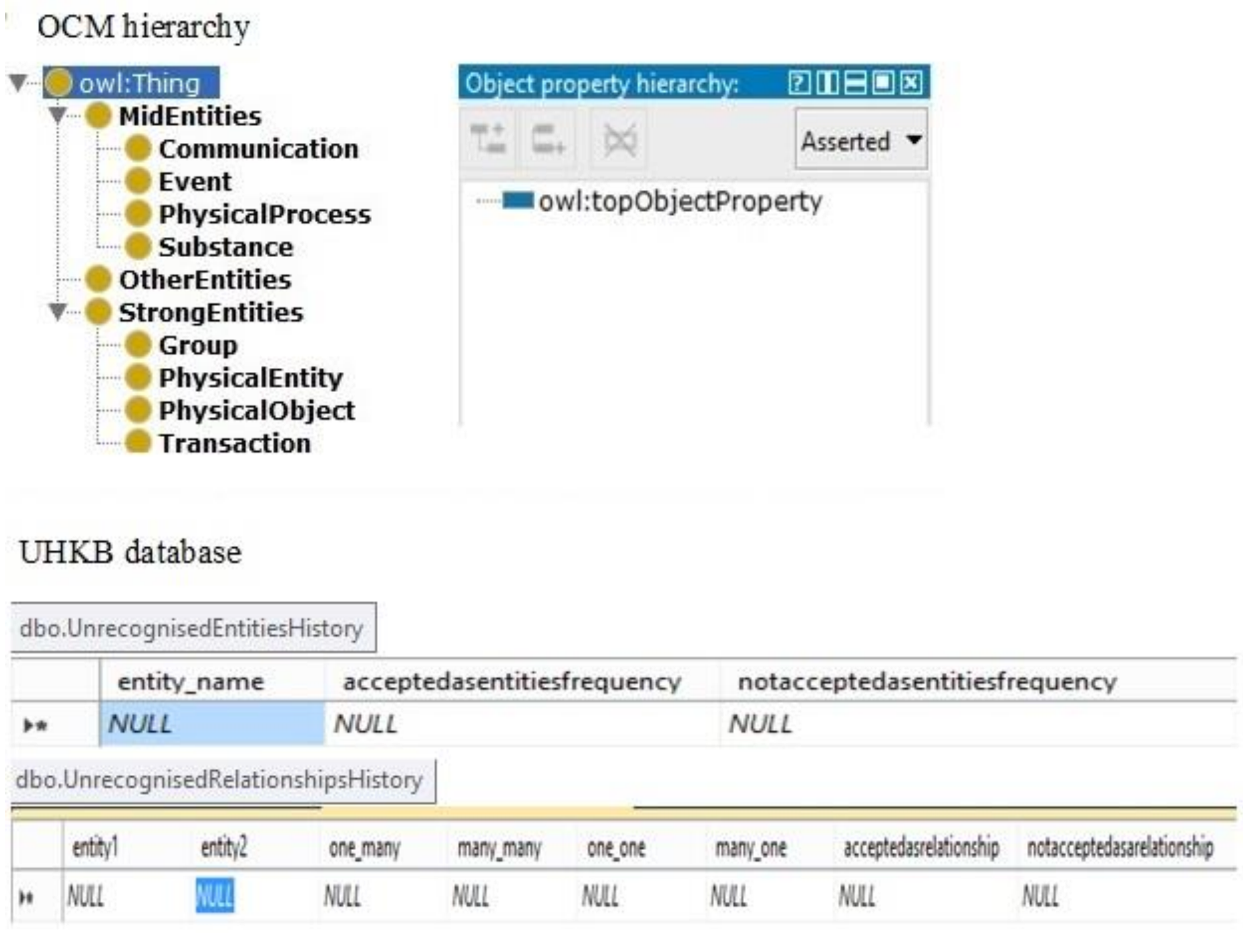
The model employs WordNet ontology to distinguish between nouns that can be mapped into entities and those that are unlikely to be mapped into entities.

3. Linguistic Rules

The model employs linguistic rules to help the user identify entities and relationships. The linguistic rules component requests human intervention to apply the domain-importance and Multi-attributes rules to identify entities. In addition, this component requests human intervention to apply the need-to-know rule to identify relationships.

4. Conceptual Model Ontology (CMO) and User History Knowledge Base (UHKB)

The CMO learns from natural language requirements and uses this information to support users when a similar scenario is processed. The UHKB database records users' behaviour when applying the SACMES. Figure 4.2 shows the CMO hierarchy and UHKB database.


OCM hierarchy

- owl:Thing
  - MidEntities
    - Communication
    - Event
    - PhysicalProcess
    - Substance
  - OtherEntities
  - StrongEntities
    - Group
    - PhysicalEntity
    - PhysicalObject
    - Transaction

Object property hierarchy:

Asserted

- owl:topObjectProperty

UHKB database

dbo.UnrecognisedEntitiesHistory

	entity_name	acceptedasentitiesfrequency	notacceptedasentitiesfrequency
▸*	NULL	NULL	NULL

dbo.UnrecognisedRelationshipsHistory

	entity1	entity2	one_many	many_many	one_one	many_one	acceptedasrelationship	notacceptedasarelationship
▸*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 4.2 OCM Hierarchy and UHKB Database

Entities in the CMO are divided into three groups, namely, strong entities, mid-entities and other entities (entities that have been defined by designers but which do not belong in the strong or mid-entities groups). Each group is further divided into subgroups. The entities added by the system are introduced into these subgroups, whereas relationships are added under the object properties hierarchy. The CMO hierarchy is adapted from WordNet, which divides nouns into strong entities, mid-entities and weak entities. Weak entities are not considered as entities and therefore are not added into the CMO hierarchy. Furthermore, the UHKB records users' behaviour in a relational database, and utilises this history to guide subsequent users in extracting conceptual models. The UHKB database includes two tables, namely, the Entities History Knowledge Base (EHKB) and Relationships History Knowledge Base (RHKB). The EHKB records users' behaviour with regard to entities, whilst the RHKB stores users' behaviour regarding relationships.

As depicted in Figure 4.1, the system is divided into three stages which comprise the pre-processing stage, entities identification stage and relationships identification stage.

1. Pre-processing stage

The input of this stage is the requirement specification text and the output is a list of candidate entities. Natural language tools and the WordNet ontology introduce appropriate support for performance of this stage.

2. Entities identification stage

The input of this stage consists of the candidate entities, while the output is the entities list. Support from the CMO, UHKB, WordNet ontology and linguistic rules applied by human intervention enable this stage to be appropriately performed.

3. Relationships identification stage

The input of this stage is the entities list and natural language specification text for a specific problem. The output is the entity relationship diagram and users' behaviour recorded during the process of creating a conceptual model using SACMES. Natural language tools, the CMO, UHKB and linguistic rules applied by human intervention provide support for appropriate performance of this stage.

The outputs of the system are a conceptual model and User Behaviour (UB). After the conceptual model has been viewed by the user, it is inserted into the CMO in order to update the ontology and increase its ability to release relevant information to guide future users in the creation of conceptual models. Users' behaviour is also inserted into the UHKB to update it. The system then copies its behaviour from this knowledge when a similar situation to that stored in the UHKB is processed by the system.

Each of the above stages is divided into sub-stages.

4.1.1 Pre-Processing Stage

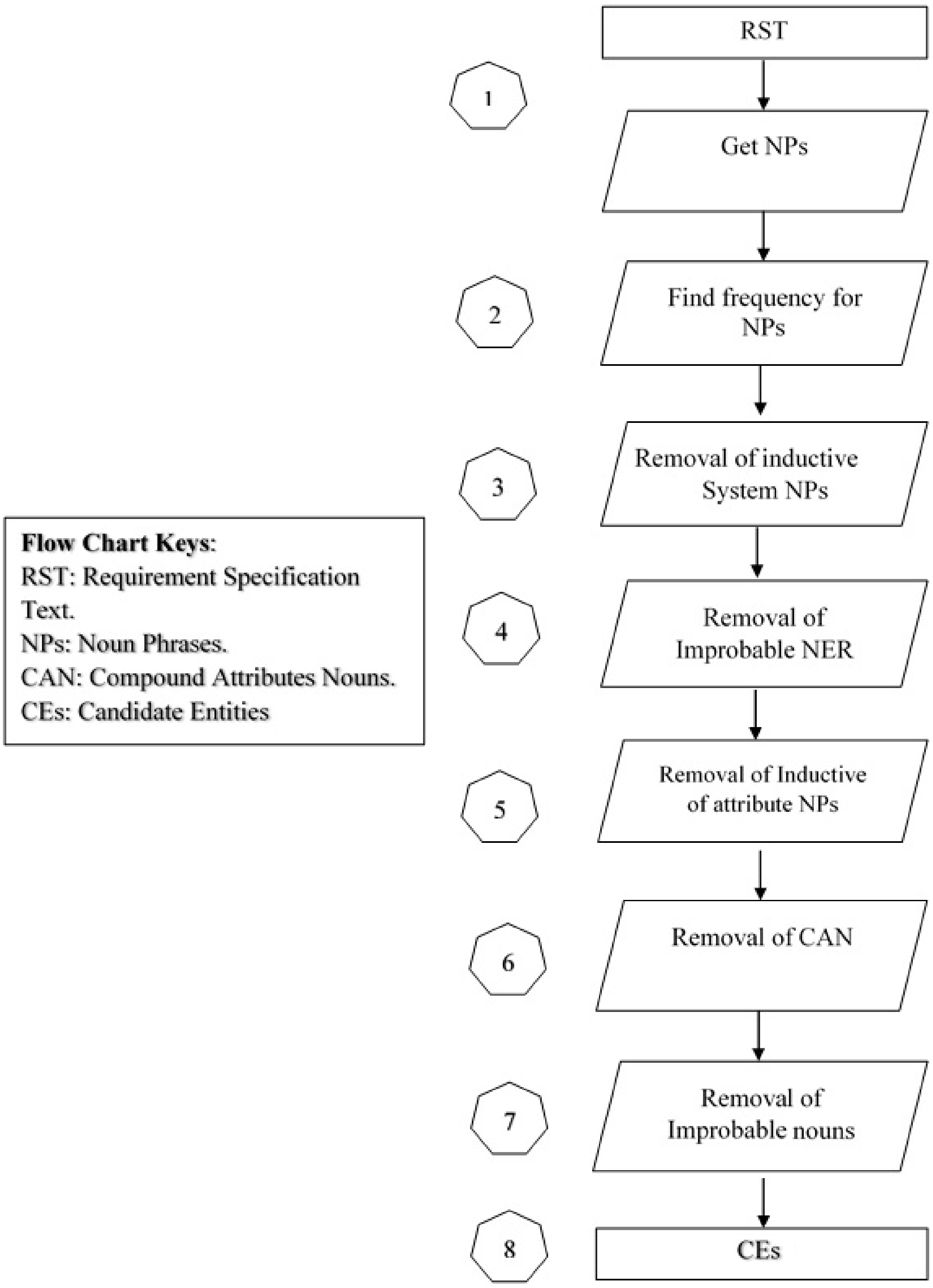

RST
1
Get NPs
2
Find frequency for NPs
3
Removal of inductive System NPs
Flow Chart Keys:
RST: Requirement Specification Text.
NPs: Noun Phrases.
CAN: Compound Attributes Nouns.
CEs: Candidate Entities
4
Removal of Improbable NER
5
Removal of Inductive of attribute NPs
6
Removal of CAN
7
Removal of Improbable nouns
8
CEs

Figure 4.3 Flow Chart of Pre-Processing Stage

Figure 4.3 illustrates the pre-processing stage. In the pre-processing stage, Stanford CoreNLP[6] is employed to achieve natural language tasks for SACMES. This stage has sub-stages as follows:

1. The system uses the Stanford PoS tagger[19] to define noun phrases (NPs) from the Requirement Specification Text (RST).

2. The system defines a frequency for each Noun Phrase (NP). The frequency refers to how many times a noun phrase is mentioned in the RST.

3. Removal of system indicative nouns. Some nouns, such as ‘system’, ‘database’, ‘record’ and ‘application’ are indicative of the system (Btoush & Hammad, 2015). These nouns are removed from the NPs list. The system uses string matching to eliminate such nouns.

4. Removal of improbable NER classes. The system uses the Stanford Name Entities Recogniser[20] to exclude nouns that are indicative of being an organisation, location, person, percentage or time from being tabled. Logically, these classes would not normally be mapped into entities.

5. Removal of NPs indicative of attributes. The system uses a predefined list of nouns which are indicative of attributes. This list includes name, birthdate, number, gender, size, colour, age, username, password, date and year, month and day. If a NP matches any of this list, then it is removed.

6. Removal of Compound Attribute Nouns (CAN). For example, the noun phrase ‘student number’ is a noun phrase made of two nouns. The second noun is indicative of an attribute, and therefore such nouns are removed from the noun phrase list.

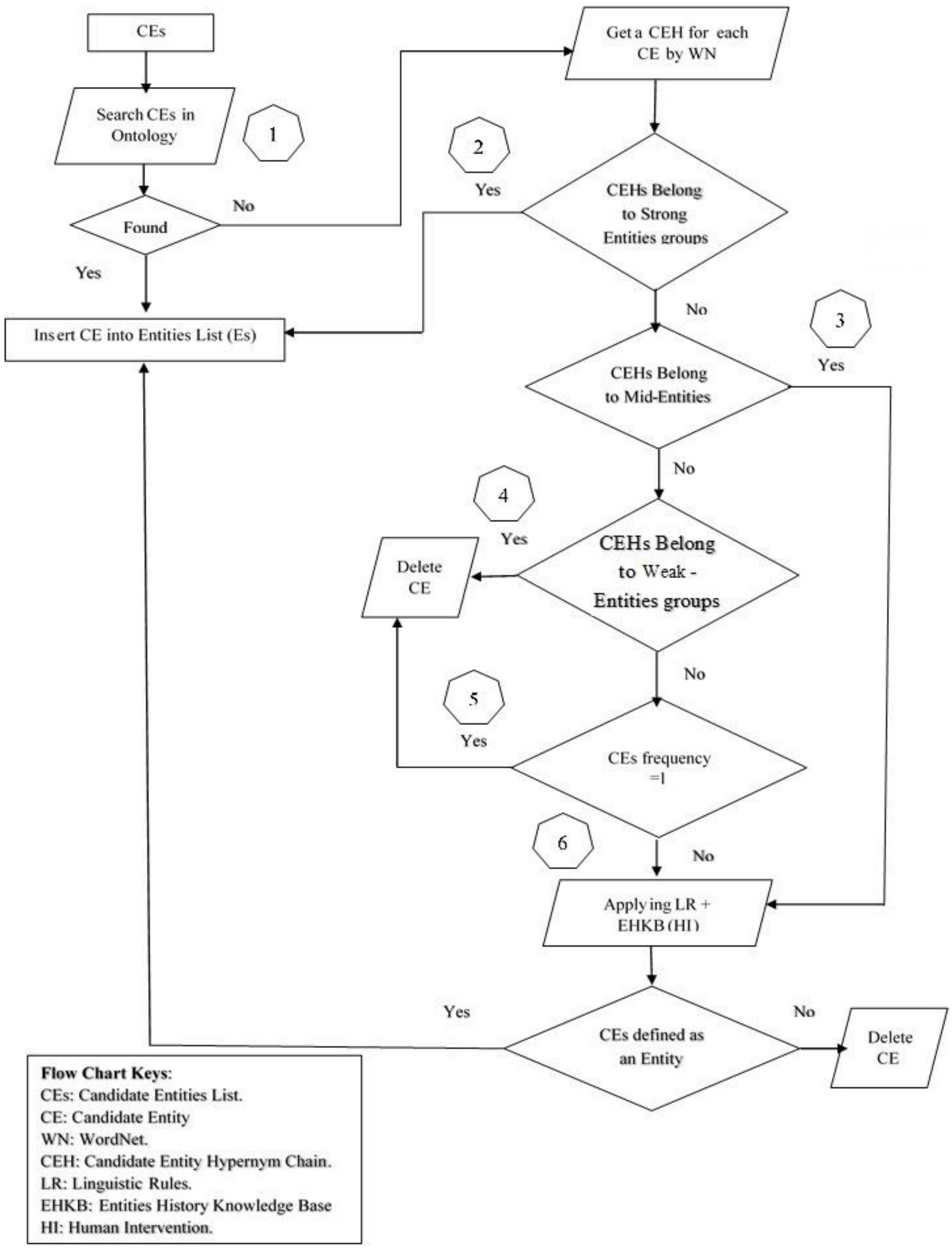
7. Removal of improbable nouns. In some cases, the PoS tagger defines improbable nouns. The system uses noun phrases found in WordNet 3.1 as standard for NPs. Each NP in the NPs list is matched with the WordNet nouns list. If it is not matched, then it is removed. Sometimes a NP is made of compound nouns. In this case, the system divides a compounded noun into separate nouns and matches each to WordNet nouns. All sub-nouns within the compound nouns must match with nouns in WordNet; otherwise, the NP is removed from being a candidate entity.

8. The NPs remaining after completion of these steps are considered Candidate Entities (CEs). The CEs are the output of the pre-processing stage and the input for the entities identification stage.

[19] https://nlp.stanford.edu/software/tagger.shtml

[20] https://nlp.stanford.edu/software/CRF-NER.shtml

4.1.2 Entities Identification Stage



CEs
Search CEs in Ontology
1
Found
No
Yes
Insert CE into Entities List (Es)
Get a CEH for each CE by WN
2
Yes
CEHs Belong to Strong Entities groups
No
3
Yes
CEHs Belong to Mid-Entities
No
4
Yes
Delete CE
CEHs Belong to Weak - Entities groups
No
5
Yes
CEs frequency =1
6
No
Applying LR + EHKB (HI)
Yes
CEs defined as an Entity
No
Delete CE
Flow Chart Keys:
CEs: Candidate Entities List.
CE: Candidate Entity
WN: WordNet.
CEH: Candidate Entity Hypernym Chain.
LR: Linguistic Rules.
EHKB: Entities History Knowledge Base
HI: Human Intervention.

Figure 4.4 Flow Chart of Entities Identification Stage

Figure 4.4 illustrates the entities identification stage. The entities identification stage is achieved by means of the following steps:

1. The process starts by searching for each item on the candidate entities list in the conceptual model ontology. If a candidate entity is found in the ontology, then it is considered as an entity.

2. WordNet is used to find a hypernym chain for each candidate entity that is not found in the ontology. If the hypernym chain of a candidate entity matches the strong entities group, then the candidate entity is considered an entity and inserted into the entities list.

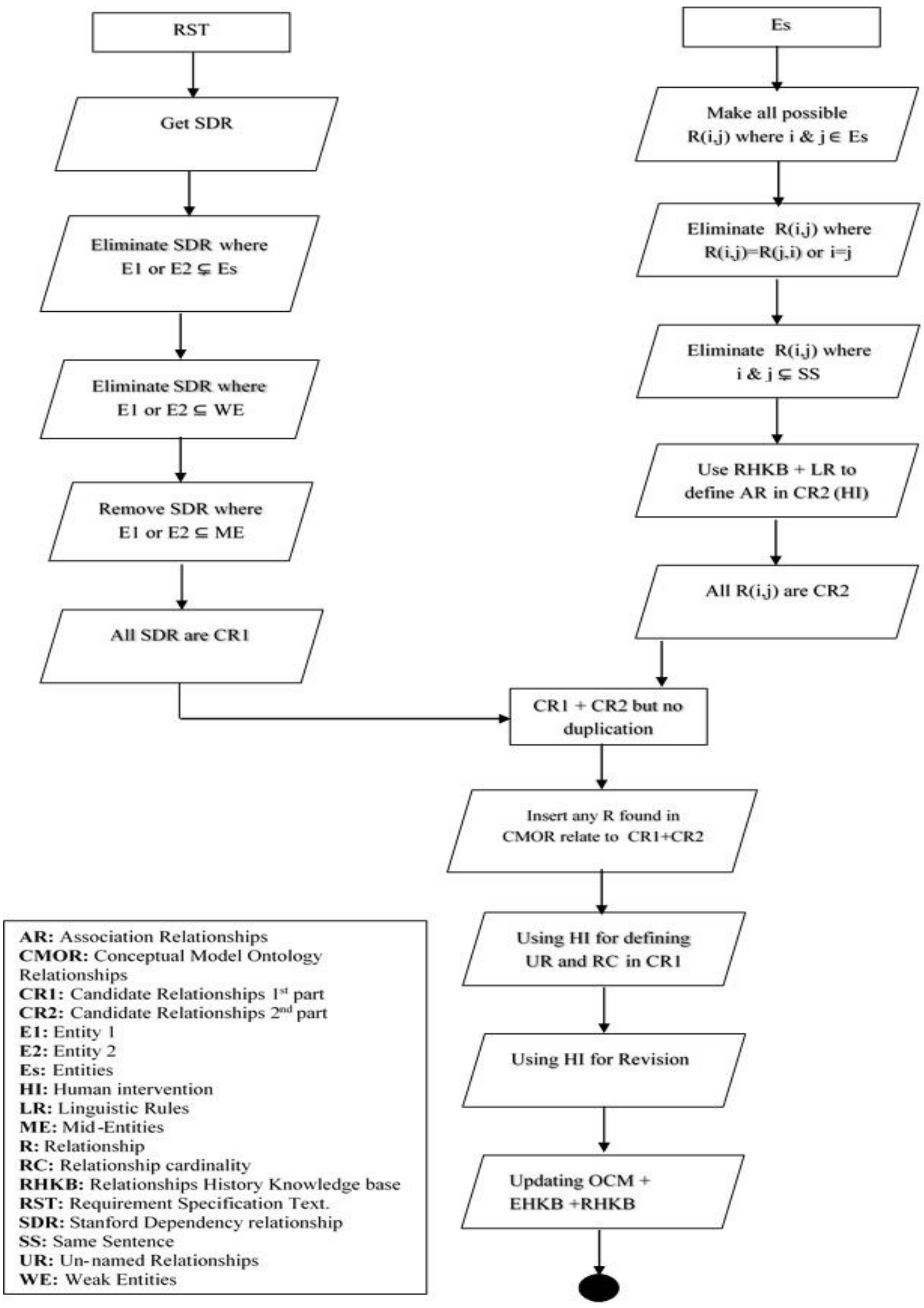
3. If the hypernym chain of a candidate entity does not match the strong entities group but matches the mid-entities group, then the system uses the EHKB and linguistic rules either to discard the candidate noun or to accept it as an entity (using human intervention).

4. If the hypernym chain of a candidate entity does not match the strong or mid-entities groups, but does match the weak entities group, then the candidate entity is removed from the candidate entities list.

5. If the hypernym chain of a candidate noun does not match the strong entities, mid-entities or weak entities groups, and its frequency is equal to one, then the candidate entity is removed from the candidate entities list.

6. If the hypernym chain of a candidate noun does not match the strong entities, mid-entities or weak entities groups, and its frequency is greater than one, then the system uses the EHKB and linguistic rules to either discard the candidate noun or accept it as an entity (human intervention).

4.1.3 Relationships Identification Stage



RST
Get SDR
Eliminate SDR where E1 or E2 ⊊ Es
Eliminate SDR where E1 or E2 ⊆ WE
Remove SDR where E1 or E2 ⊆ ME
All SDR are CR1
Es
Make all possible R(i,j) where i & j ∈ Es
Eliminate R(i,j) where R(i,j)=R(j,i) or i=j
Eliminate R(i,j) where i & j ⊊ SS
Use RHKB + LR to define AR in CR2 (HI)
All R(i,j) are CR2
CR1 + CR2 but no duplication
Insert any R found in CMOR relate to CR1+CR2
Using HI for defining UR and RC in CR1
Using HI for Revision
Updating OCM + EHKB +RHKB
AR: Association Relationships
CMOR: Conceptual Model Ontology Relationships
CR1: Candidate Relationships 1st part
CR2: Candidate Relationships 2nd part
E1: Entity 1
E2: Entity 2
Es: Entities
HI: Human intervention
LR: Linguistic Rules
ME: Mid-Entities
R: Relationship
RC: Relationship cardinality
RHKB: Relationships History Knowledge base
RST: Requirement Specification Text.
SDR: Stanford Dependency relationship
SS: Same Sentence
UR: Un-named Relationships
WE: Weak Entities

Figure 4.5 Flow Chart of Relationships Identification Stage

Figure 4.5 demonstrates a flow chart of the relationships identification stage. This stage is divided into three sub-stages. The first sub-stage defines relationships from the requirement specification text using Stanford typed dependencies. The second sub-stage defines relationships from the entities list identified in the first stage. The third sub-stage is human intervention.

4.1.3.1 Identifying relationships from requirement specification text using Stanford typed dependencies

The input for this sub-stage is the requirement specification text for a specific problem, and the outputs are candidate relationships defined by Stanford typed dependencies (De Marneffe & Manning, 2008). The author employed Stanford dependencies as part of Stanford CoreNLP to achieve this stage. Relationships are interactions between nouns and verbs. The nouns represent subjects and objects, the subject being a person or thing doing something and the object having something done to it. Stanford dependencies can deduce sentence subjects and sentence objects from the following list of relationships.

Nominal subject (nsubj)
Nominal subject passive (nsubjpass)
Clausal subject (csubj)
Passive clausal subject (csubjpass)
Direct object (Dobj)
Indirect object (iobj)
Preposition object (pobj)

Clausal subjects and passive clausal subjects represent sentence subjects in the form of a clause, and a clause cannot represent an entity. For example, in the sentence ‘What Ali said makes sense’, ‘What Ali said’ is a clausal subject. Therefore, clausal subjects are not mapped into entities. Similarly, an indirect object (iobj) is not useful because it represents a noun phrase stating to a person or a thing which is influenced by the acting out a transitive verb (typically as a recipient), but is not the primary object. For example, in ‘She gives me a raise’, the subject is ‘she’, the object is ‘raise’ and the action/verb is ‘gives’. The indirect object is ‘me’. A prepositional object (pobj) is equally unhelpful in defining a relationship because it modifies the noun rather than showing something done to the verb. For example, in ‘A patient sat on the chair’, the subject is ‘A patient’, the action/verb is ‘sat’ and ‘on the chair’ is a prepositional object.

A nominal subject (nsubj), however, is useful in defining a relationship because it shows who/what does the action. For example, in ‘A school offers courses’, ‘A school’ is nsubj for the action ‘offers’. A direct object (dobj) is also useful in defining a relationship because it shows what is acted on by the verb. In the previous example, ‘courses’ is dobj for the action ‘offers’. In addition, a nominal subject passive (nsubjpass) can be useful because it shows what/who does the action. A nominal subject passive (nsubjpass) is supported by an agent. An agent is the complement of a passive verb which is introduced by the preposition ‘by’ and does the action (De Marneffe & Manning, 2008). For example, in ‘An invoice is paid by a customer’, the nsubjpass is ‘An invoice’, while the agent is ‘a customer’.

The system uses nsubj, dobj and nsubjpass to extract relationships from requirement specification text. The following example considers text which may form part of a requirement specification for a mall database: ‘A customer buys many products. A customer is served by an employee.’ Here, the Stanford dependency relationship of the first sentence is:

```
root ( ROOT-0 , buys-3 )
det ( customer-2 , A-1 )
nsubj ( buys-3 , customer-2 )
amod ( product-5 , many-4 )
dobj ( buys-3 , product-5 )
```

From nsubj (buys-3, customer-2), the action/verb ‘buys’ and the subject of the sentence ‘customer’ can be defined. From dobj (buys-3, product-5), the action/verb ‘buys’ and sentence object ‘product’ can be defined. Thus, from nsubj (buys-3, customer-2) and dobj (buys-3, product-5), the system can define the relationship whereby a customer buys a product in the following format:

Buy (customer, product)

For the second sentence, the Stanford dependencies relationship is as follows:

```
root ( ROOT-0 , served-3 )
nsubjpass ( served-3 , Customer-1 )
auxpass ( served-3 , is-2 )
```

case (employee-6 , by-4)
det (employee-6 , an-5)
agent (served-3 , employee-6)

From the relationship nsubjpass (served-3, customer-1), the action/verb ‘served’ and the subject ‘customer’ can be defined. From the relationship agent (served-3, employee-6), the action/verb ‘served’ and agent ‘employee’ can be defined. Thus, from the relationship nsubjpass (served-3, customer-1) and the relationship agent (served-3, employee-6), the system can define the relationship whereby an employee serves a customer in the following format:

Served (employee, customer)

As shown in Figure 4.5, this stage has the following steps.

1. The system employs Stanford dependencies to extract relationships from requirement specification text.

2. The system eliminates any relationship involving a subject or object that is not included in the entities list defined in the entities identification stage.

3. The system eliminates any relationship that includes entities with a hypernym chain matching the weak or mid-entities groups. WordNet is incorporated to achieve this mission. The remaining relationships are called Candidate Relationships 1st part (CR1), and CR1 is the output of this stage.

4.1.3.2 Identification of relationships from entities

The input of this stage is the entities list defined in the entities identification stage. As shown in Figure 4.5, this stage is divided into the following sub-stages.

1. The system uses the entities list to determine all possible binary relationships between the entities. For example, if the three entities defined from the entities identification stage are ‘customer’, ‘product’ and ‘employee’, the possible binary relationships between these entities are:

(customer, customer)
(customer, product)
(customer, employee)
(product, product)

(product, employee)
(product, customer)
(employee, employee)
(employee, product)
(employee, customer)

2. Reversing the order of terms should not produce a distinct relationship. For example, the relationships (customer, product) and (product, customer) match this condition. After eliminating such redundancies in the list of relationships, the list is updated as follows.

(customer, customer)
(customer, product)
(customer, employee)
(product, product)
(product, employee)
(employee, employee)

3. Similarly, entities should not have relationships to themselves, as with (customer, customer), for example. After eliminating the relationships that meet this condition, the relationships list is updated as follows.

(customer, product)
(customer, employee)
(product, employee)

4. The process eliminates any relationship that does not have a first and second entity both mentioned in one sentence within a requirement specification text. The entities that are mentioned in the same sentence may have a relationship between them. However, the system then uses human intervention to revise the result by adding any missing relationships and removing any inappropriate relationships.

5. The system requires users to use the Relationships History Knowledge Base (RHKB) and need-to-know rule (Thonggoom, 2011) to define the associations within those relationships that remain after the above filtering steps.

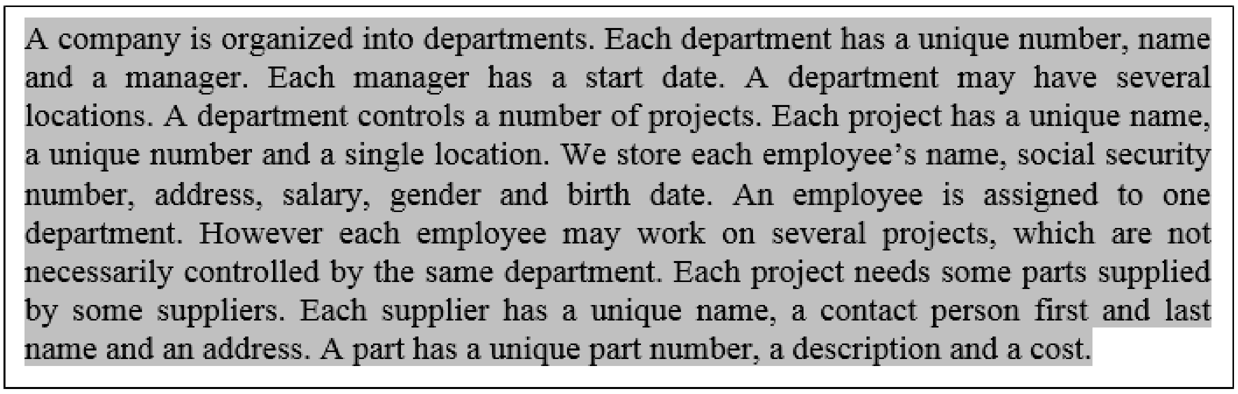
6. The remaining relationships are all considered 2nd part candidate relationships. Candidate Relationships 2nd part (CR2) is the output of this stage.

4.2.3.3 Human intervention

The inputs of this stage are candidate relationships 1st part and candidate relationships 2nd part. Before the stage is started, the system searches the OCM for relationships identified in CR1 and CR2, and adds them to the relationships list. The system then uses human intervention to define the cardinality of relationships, giving appropriate names to unnamed relationships. The user is also given an opportunity to review the whole process and to add or remove entities or relationships. The user can then print a report containing a list of entities and list of relationships defined for the problem. The system also updates the OCM by adding these entities and relationships into the ontology, as well as updating the UHKB by saving the user's behaviour into the database.

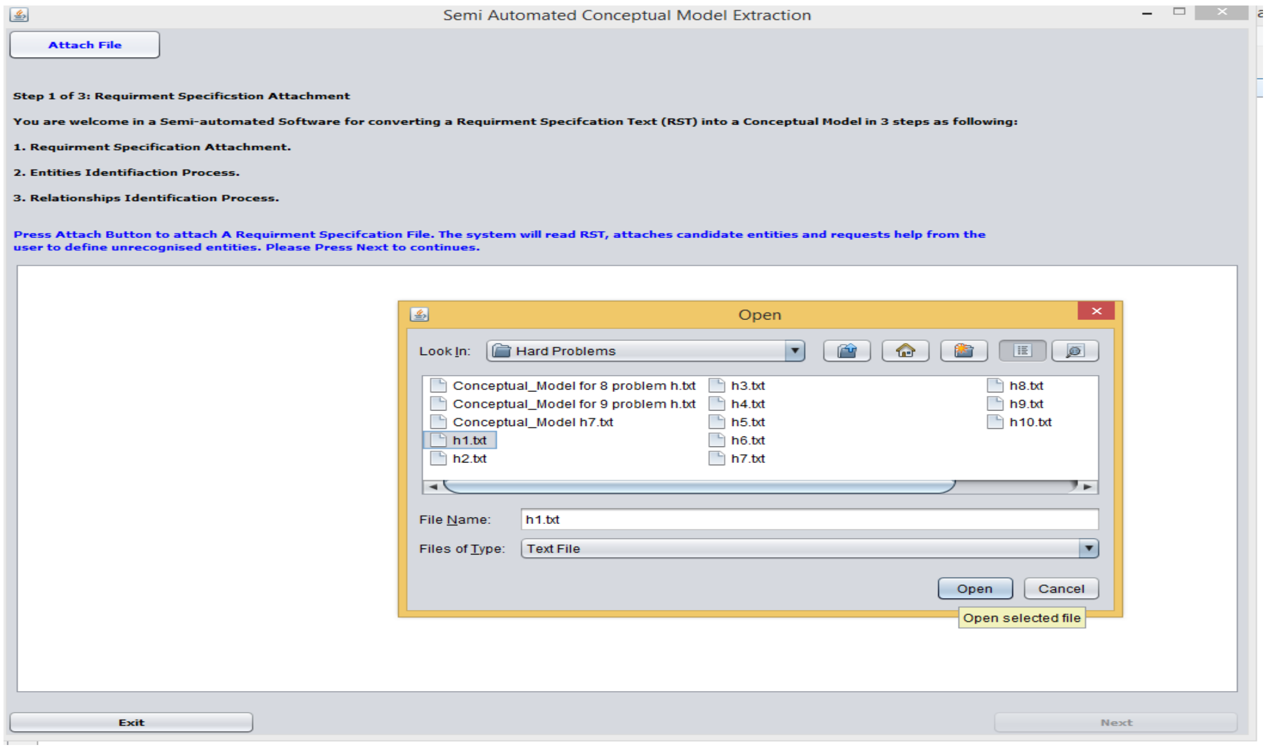
4.2 Step-by-Step Case Study

In this section, SACMES is used to map natural language text into a conceptual model. The case study that is used for this demonstration is illustrated in Figure 4.6.



A company is organized into departments. Each department has a unique number, name and a manager. Each manager has a start date. A department may have several locations. A department controls a number of projects. Each project has a unique name, a unique number and a single location. We store each employee's name, social security number, address, salary, gender and birth date. An employee is assigned to one department. However each employee may work on several projects, which are not necessarily controlled by the same department. Each project needs some parts supplied by some suppliers. Each supplier has a unique name, a contact person first and last name and an address. A part has a unique part number, a description and a cost.

Figure 4.6 A Company Database (Du, 2008, p. 170)



Step 1 of 3: Requirment Specificstion Attachment

You are welcome in a Semi-automated Software for converting a Requirment Specifcation Text (RST) into a Conceptual Model in 3 steps as following:

1. Requirment Specification Attachment.

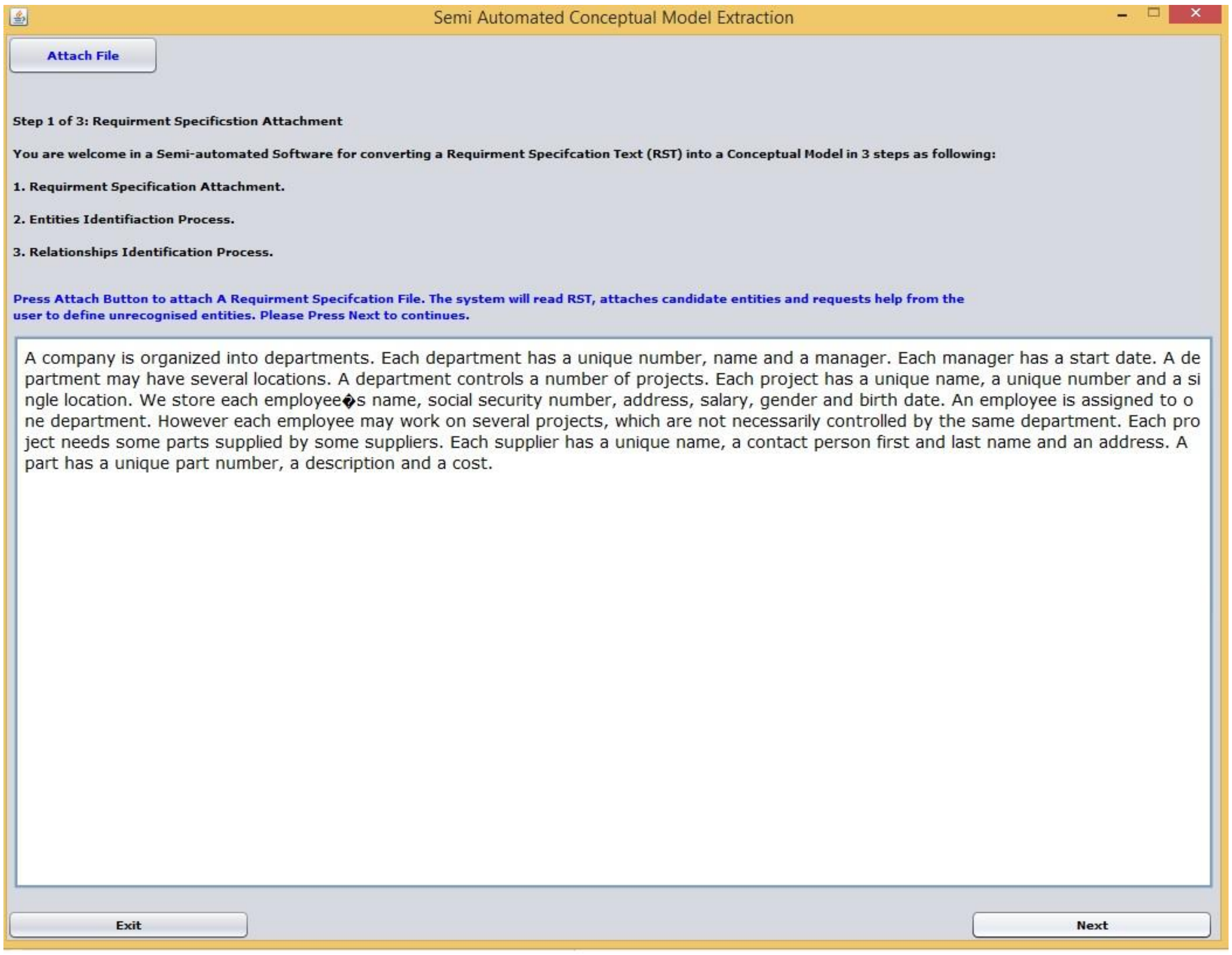
2. Entities Identifiaction Process.

3. Relationships Identification Process.

Press Attach Button to attach A Requirment Specifcation File. The system will read RST, attaches candidate entities and requests help from the user to define unrecognised entities. Please Press Next to continues.

Figure 4.7 Attachment of Requirement Specification Text into SACMES

As shown in Figure 4.7, a user is required to attach a requirement specification text to start the process. The system receives the text file as an input. The system then reads and displays the text as illustrated in Figure 4.8.



Step 1 of 3: Requirment Specificstion Attachment

You are welcome in a Semi-automated Software for converting a Requirment Specifcation Text (RST) into a Conceptual Model in 3 steps as following:

1. Requirment Specification Attachment.

2. Entities Identifiaction Process.

3. Relationships Identification Process.

Press Attach Button to attach A Requirment Specifcation File. The system will read RST, attaches candidate entities and requests help from the user to define unrecognised entities. Please Press Next to continues.

A company is organized into departments. Each department has a unique number, name and a manager. Each manager has a start date. A de partment may have several locations. A department controls a number of projects. Each project has a unique name, a unique number and a si ngle location. We store each employee�s name, social security number, address, salary, gender and birth date. An employee is assigned to o ne department. However each employee may work on several projects, which are not necessarily controlled by the same department. Each pro ject needs some parts supplied by some suppliers. Each supplier has a unique name, a contact person first and last name and an address. A part has a unique part number, a description and a cost.

Figure 4.8 SACMES Displays the RST to the User

The system then performs the pre-processing stage by applying the steps illustrated in Figure 4.3, in order to achieve the entities identification stage as described in Figure 4.4. The first step is that Stanford PoS, which is part of Stanford CoreNLP, defines a list of nouns and NPs as demonstrated in Table 4.1.

S. No	Noun Phrase	Frequency
1	Address date	1
2	Address	1
3	Birth date	1
4	Company	1
5	Contact	1
6	Cost	1
7	Department	4
8	Description	1

S. No	Noun Phrase	Frequency
9	Employee	3
10	Gender date	1
11	Location	2
12	Manager	2
13	Name	3
14	Name date	1
15	Number	3
16	Part	2
17	Part number	1
18	Person	1
19	Project	3
20	Salary date	1
21	Security number date	1
22	Start date	1
23	Supplier	2

Table 4.1 Noun Phrases Defined by Stanford PoS from Company Database Scenario

The list of noun phrases in Table 4.1 does not include any system indicative nouns or any nouns belonging to improbable NER classes. However, ‘name’ and ‘number’ are found in the list and both are indicative of attributes, so they are removed. Furthermore, ‘address date’, ‘birth date’, ‘gender date’, ‘name date’, ‘part number’, ‘salary date’, ‘security number date’ and ‘start date’ are all indicative of attributes, so the system also removes these from the list. After this filtration, the list of nouns is updated as shown in Table 4.2.

S. No	Noun Phrase	Frequency
1	Address	1
2	Company	1
3	Contact	1
4	Cost	1
5	Department	4
6	Description	1
7	Employee	3
8	Location	2

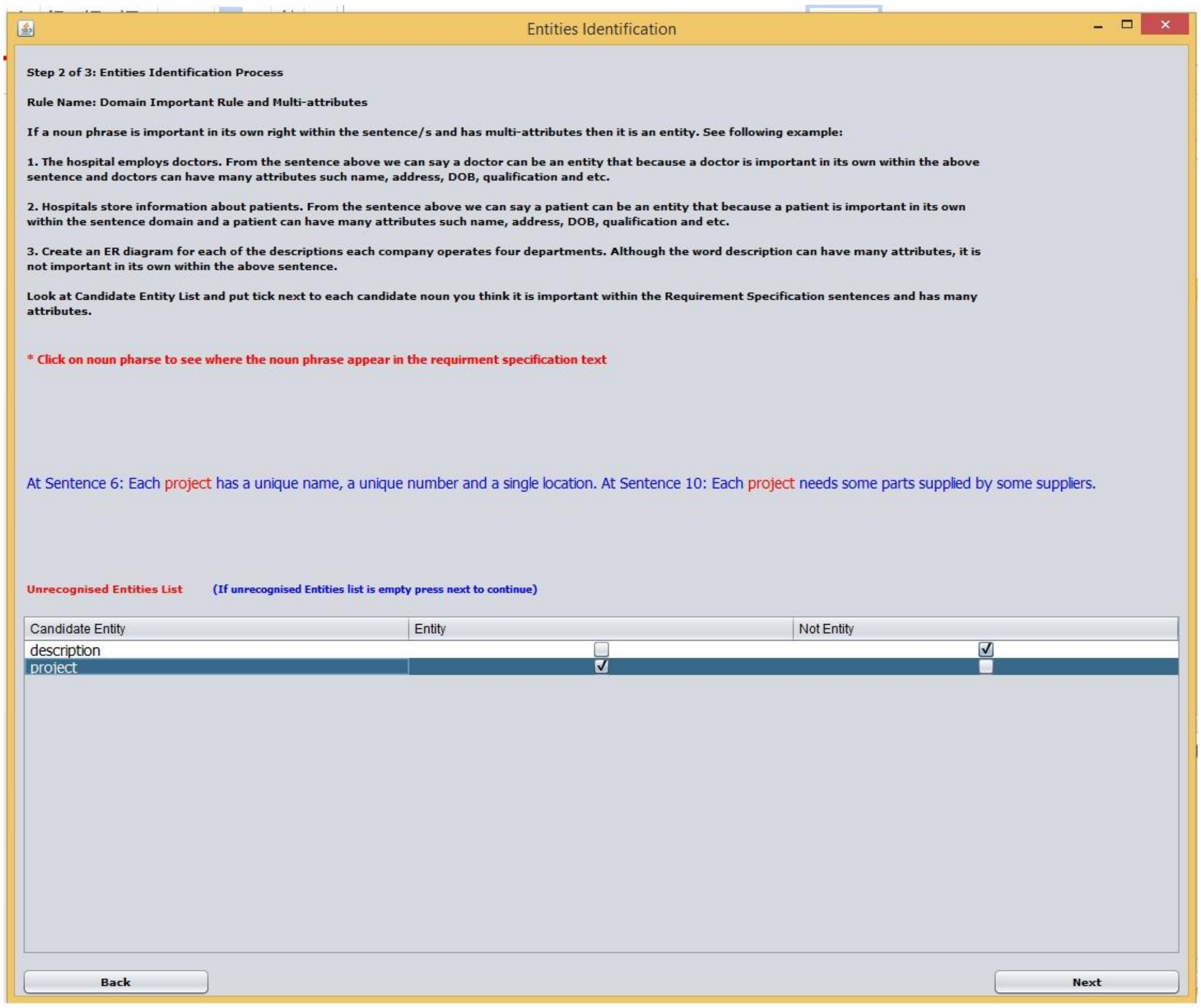
S. No	Noun Phrase	Frequency
9	Manager	2
10	Part	2
11	Person	1
12	Project	3
13	Supplier	2

Table 4.2 Entities List Defined from Company Database Scenario after Filtration

After the above work, the system moves on to applying the process represented by the flowchart in Figure 4.4, the Entities Identification Stage (EIS). The system searches the CMO for the candidate nouns included in Table 4.2. If any of the nouns are found in the CMO, then they are marked as entities. It can be assumed that if the CMO is empty, none of the candidate nouns will be marked as entities. The use of WordNet, however, identifies the nouns ‘address’, ‘company’, ‘contact’, ‘department’, ‘employee’, ‘location’, ‘manager’, ‘part’, ‘person’ and ‘supplier’ as belonging to the strong entities group, and therefore, they are marked as entities. The noun ‘cost’ belongs to the weak entities group, so it is removed from the list. ‘Description’ and ‘project’ belong to the mid-entities group, so the system needs to use human intervention to decide whether they should be marked as entities or removed from the list. After the above filtering, the entities list comprises:

Address
Company
Contact
Department
Employee
Location
Manager
Part
Person
Supplier

As a further part of the entities identification stage, the system also uses linguistic rules and the EHKB. These are both applied by human intervention to define candidate nouns about which SACMES is unable to make a decision. Figure 4.9 demonstrates how the system requests the user to apply the domain importance rule and multi-attributes rules to help decide whether to accept a noun phrase as an entity or to reject it. Based on the EHKB, the system tries to recommend a decision about each candidate noun. In this scenario, the system requested the user to make a decision about the two nouns ‘description’ and ‘project’. When the user clicks on each noun, the system displays sentences that show where the noun appears in the RST, highlighted in red to distinguish it from other text. The system then displays information and examples on the form to explain to the user how to use the domain importance and multi-attributes rules to make appropriate decisions. The system gives warning messages if (1) the user presses ‘Next’ without making a decision about each noun phrase or (2) the user selects a single noun phrase to be both an entity and not an entity at the same time. In response to the system, the author played the role of designer and selected ‘description’ as not being an entity, whereas ‘project’ was selected to be an entity. Therefore, the noun ‘project’ was added to the entities list.



Step 2 of 3: Entities Identification Process

Rule Name: Domain Important Rule and Multi-attributes

If a noun phrase is important in its own right within the sentence/s and has multi-attributes then it is an entity. See following example:

1. The hospital employs doctors. From the sentence above we can say a doctor can be an entity that because a doctor is important in its own within the above sentence and doctors can have many attributes such name, address, DOB, qualification and etc.

2. Hospitals store information about patients. From the sentence above we can say a patient can be an entity that because a patient is important in its own within the sentence domain and a patient can have many attributes such name, address, DOB, qualification and etc.

3. Create an ER diagram for each of the descriptions each company operates four departments. Although the word description can have many attributes, it is not important in its own within the above sentence.

Look at Candidate Entity List and put tick next to each candidate noun you think it is important within the Requirement Specification sentences and has many attributes.

* Click on noun pharse to see where the noun phrase appear in the requirment specification text

At Sentence 6: Each project has a unique name, a unique number and a single location. At Sentence 10: Each project needs some parts supplied by some suppliers.

Unrecognised Entities List (If unrecognised Entities list is empty press next to continue)

Candidate Entity	Entity	Not Entity
description	☐	✓
project	✓	☐

Figure 4.9 Human Intervention for Entities Identification Stage

In Figure 4.9, the system starts to apply the process that appears in Figure 4.5 (Relationships Identification Stage). In the first part of the flowchart process, the system finds all possible binary relationships between entities, as shown in Table 4.3. The system then removes all reverse order relationships, cases where entities have a relationship with themselves, and relationships between entities that are not mentioned in the same sentence. In this scenario, there are 121 binary relationships. Those with reverse order relationships are written in bold font; cases where entities have relationships with themselves are written in italic font; and relationships involving entities that do not appear in the same sentence are identified by bold italic font. After removing the relationships written in bold, italic and bold italic, only sixteen relationships remain.

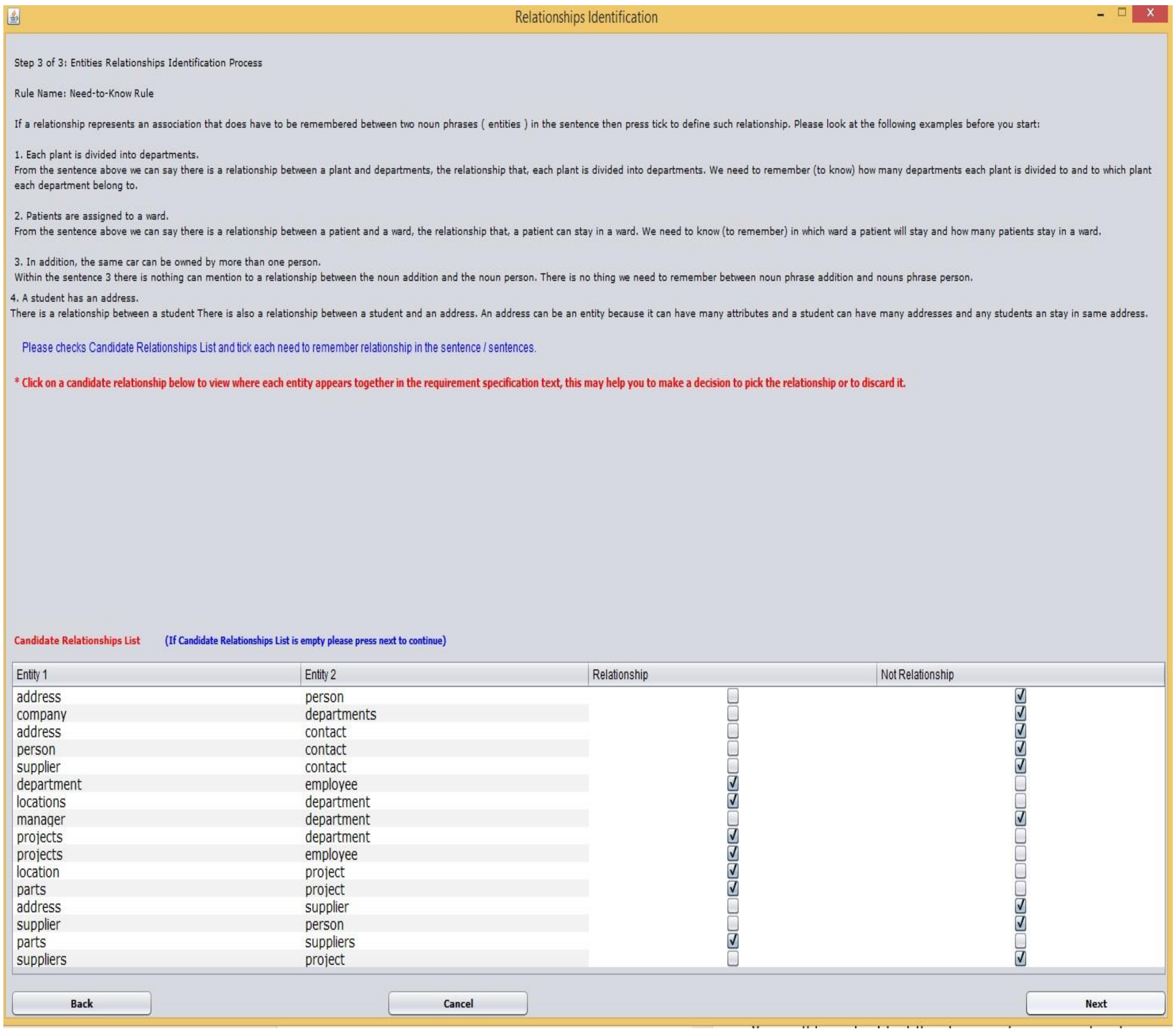
(Address, Address)	**(company, address)**	**(contact, company)**	**(department, address)**	**(employee, address)**	**(location, address)**	**(manager, address)**	**(part, address)**	**(person, address)**	**(supplier, address)**	**(project, address)**
(Address, company)	*(company, company)*	**(contact, address)**	**(department, company)**	**(employee, company)**	**(location, company)**	**(manager, company)**	**(part, company)**	**(person, company)**	**(supplier, company)**	**(project, company)**
(Address, contact)	***(company, contact)***	*(contact, contact)*	**(department, contact)**	**(employee, contact)**	**(location, contact)**	**(manager, contact)**	**(part, contact)**	(person, contact)	(supplier, contact)	**(project, contact)**
(address, department)	(company, department)	***(contact, department)***	*(department, department)*	**(employee, department)**	(location, department)	(manager, department)	**(part, department)**	**(person, department)**	**(supplier, department)**	(project, department)
(address, employee)	***(company, employee)***	***(contact, employee)***	(department, employee)	*(employee, employee)*	**(location, employee)**	**(manager, employee)**	**(part, employee)**	**(person, employee)**	**(supplier, employee)**	(project, employee)
(Address, location)	***(company, location)***	***(contact, location)***	**(department, location)**	***(employee, location)***	*(location, location)*	**(manager, location)**	**(part, location)**	**(person, location)**	**(supplier, location)**	**(project, location)**
(address, manager)	***(company, manager)***	***(contact, manager)***	**(department, manager)**	***(employee, manager)***	***(location, manager)***	*(manager, manager)*	**(part, manager)**	**(person, manager)**	**(supplier, manager)**	**(project, manager)**
(address, part)	***(company, part)***	***(contact, part)***	***(department, part)***	***(employee, part)***	***(location, part)***	***(manager, part)***	*(part, part)*	**(person, part)**	**(supplier, part)**	**(project, part)**
(address, person)	***(company, person)***	**(contact, person)**	***(department, person)***	***(employee, person)***	***(location, person)***	***(manager, person)***	***(part, person)***	*(person, person)*	(supplier, person)	**(project, person)**
(address, supplier)	***(company, supplier)***	**(contact, supplier)**	***(department, supplier)***	***(employee, supplier)***	***(location, supplier)***	***(manager, supplier)***	(part, supplier)	**(person, supplier)**	*(supplier, supplier)*	**(project, supplier)**
(address, project)	***(company, project)***	***(contact, project)***	**(department, project)**	**(employee, project)**	(location, project)	***(manager, project)***	(part, project)	***(person, project)***	(supplier, project)	*(project, project)*

Table 4.3 Binary Relationships between Entities

As part of the process in Figure 4.5, the system also uses Stanford dependencies to extract relationships. The Stanford dependencies technique extracted the following relationships from the company database specifications:

Control (department, number) means a department controls number
Have (department, manager) means a department has a manager
Have (department, name) means a department has a name
Have (department, number) means a department has a number.

As filtration for the above relationships, and as explained in Figure 4.5, the system removes any relationship involving entities that are not included in the entities list. The entities ‘number’ and ‘name’ were not included in the entities list and consequently, the relationships ‘Control (department, number)’, ‘Have (department, name)’ and ‘Have (department, number)’ are removed from the list. Figure 4.5 also illustrates that the system removes any Stanford relationships in which one of the entities belongs to the mid-entities or weak entities groups. Here, the system used WordNet to define hypernym chains for the nouns ‘department’ and ‘manager’, as both are part of the relationship ‘a department has a manager’. The hypernym chain for ‘manager’ matches the mid-entities group and consequently, the relationship ‘Have (department, manager)’ is removed.
As shown in the flow chart in Figure 4.5, the system also uses linguistic rules and the RHKB to help users make decisions about unknown relationships. Figure 4.10 illustrates how the system allowed the user to make such decisions regarding the sixteen relationships that remained after the filtration process had been completed. The figure includes information and examples showing the user how to apply the need-to-know rule to select association relationships. The system also tries to recommend decisions based on the RHKB.


Relationships Identification

Step 3 of 3: Entities Relationships Identification Process

Rule Name: Need-to-Know Rule

If a relationship represents an association that does have to be remembered between two noun phrases (entities) in the sentence then press tick to define such relationship. Please look at the following examples before you start:

1. Each plant is divided into departments.
From the sentence above we can say there is a relationship between a plant and departments, the relationship that, each plant is divided into departments. We need to remember (to know) how many departments each plant is divided to and to which plant each department belong to.

2. Patients are assigned to a ward.
From the sentence above we can say there is a relationship between a patient and a ward, the relationship that, a patient can stay in a ward. We need to know (to remember) in which ward a patient will stay and how many patients stay in a ward.

3. In addition, the same car can be owned by more than one person.
Within the sentence 3 there is nothing can mention to a relationship between the noun addition and the noun person. There is no thing we need to remember between noun phrase addition and nouns phrase person.

4. A student has an address.
There is a relationship between a student There is also a relationship between a student and an address. An address can be an entity because it can have many attributes and a student can have many addresses and any students an stay in same address.

Please checks Candidate Relationships List and tick each need to remember relationship in the sentence / sentences.

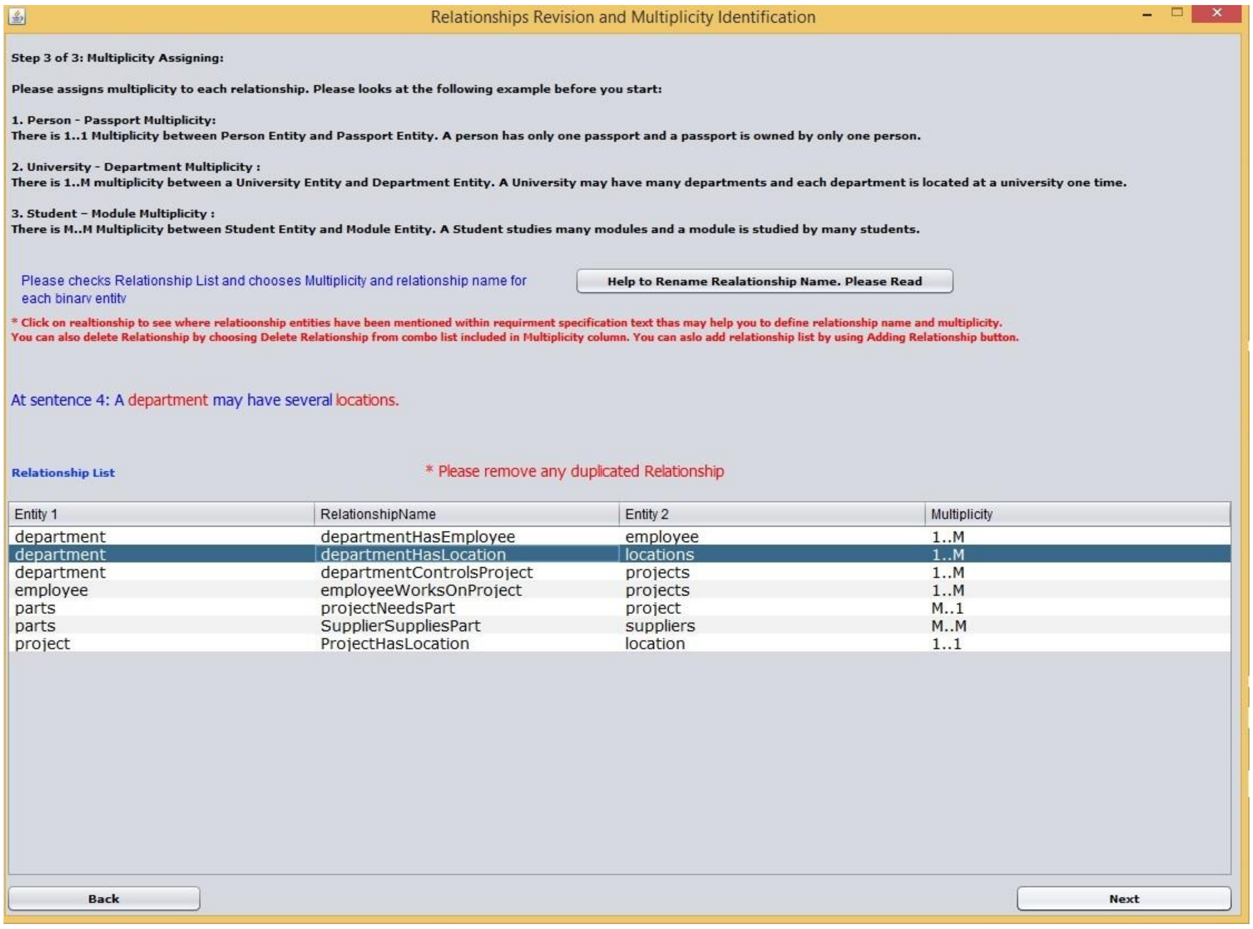
* Click on a candidate relationship below to view where each entity appears together in the requirement specification text, this may help you to make a decision to pick the relationship or to discard it.

Candidate Relationships List (If Candidate Relationships List is empty please press next to continue)

Entity 1	Entity 2	Relationship	Not Relationship
address	person	☐	✓
company	departments	☐	✓
address	contact	☐	✓
person	contact	☐	✓
supplier	contact	☐	✓
department	employee	✓	☐
locations	department	✓	☐
manager	department	☐	✓
projects	department	✓	☐
projects	employee	✓	☐
location	project	✓	☐
parts	project	✓	☐
address	supplier	☐	✓
supplier	person	☐	✓
parts	suppliers	✓	☐
suppliers	project	☐	✓

Figure 4.10 Human Intervention for Defining Relationships

When the user clicks on a specific row in the screen shown in Figure 4.10, the system displays sentences in which both entities appear within the RST. The entities are highlighted in red to distinguish them from the rest of the text. The user can then read the text and apply the need-to-know rule in order to make an appropriate decision for each relationship. The system gives a warning message if the user fails to select a decision about each relationship, or if the user ticks both the ‘relationship’ and ‘not relationship’ options for a row at the same time. Here, the author acted as designer and made the decisions that appear in Figure 4.10.


Relationships Revision and Multiplicity Identification

Step 3 of 3: Multiplicity Assigning:

Please assigns multiplicity to each relationship. Please looks at the following example before you start:

1. Person - Passport Multiplicity:
There is 1..1 Multiplicity between Person Entity and Passport Entity. A person has only one passport and a passport is owned by only one person.

2. University - Department Multiplicity :
There is 1..M multiplicity between a University Entity and Department Entity. A University may have many departments and each department is located at a university one time.

3. Student – Module Multiplicity :
There is M..M Multiplicity between Student Entity and Module Entity. A Student studies many modules and a module is studied by many students.

Please checks Relationship List and chooses Multiplicity and relationship name for each binarv entitv

* Click on realtionship to see where relatioonship entities have been mentioned within requirment specification text thas may help you to define relationship name and multiplicity.
You can also delete Relationship by choosing Delete Relationship from combo list included in Multiplicity column. You can aslo add relationship list by using Adding Relationship button.

At sentence 4: A department may have several locations.

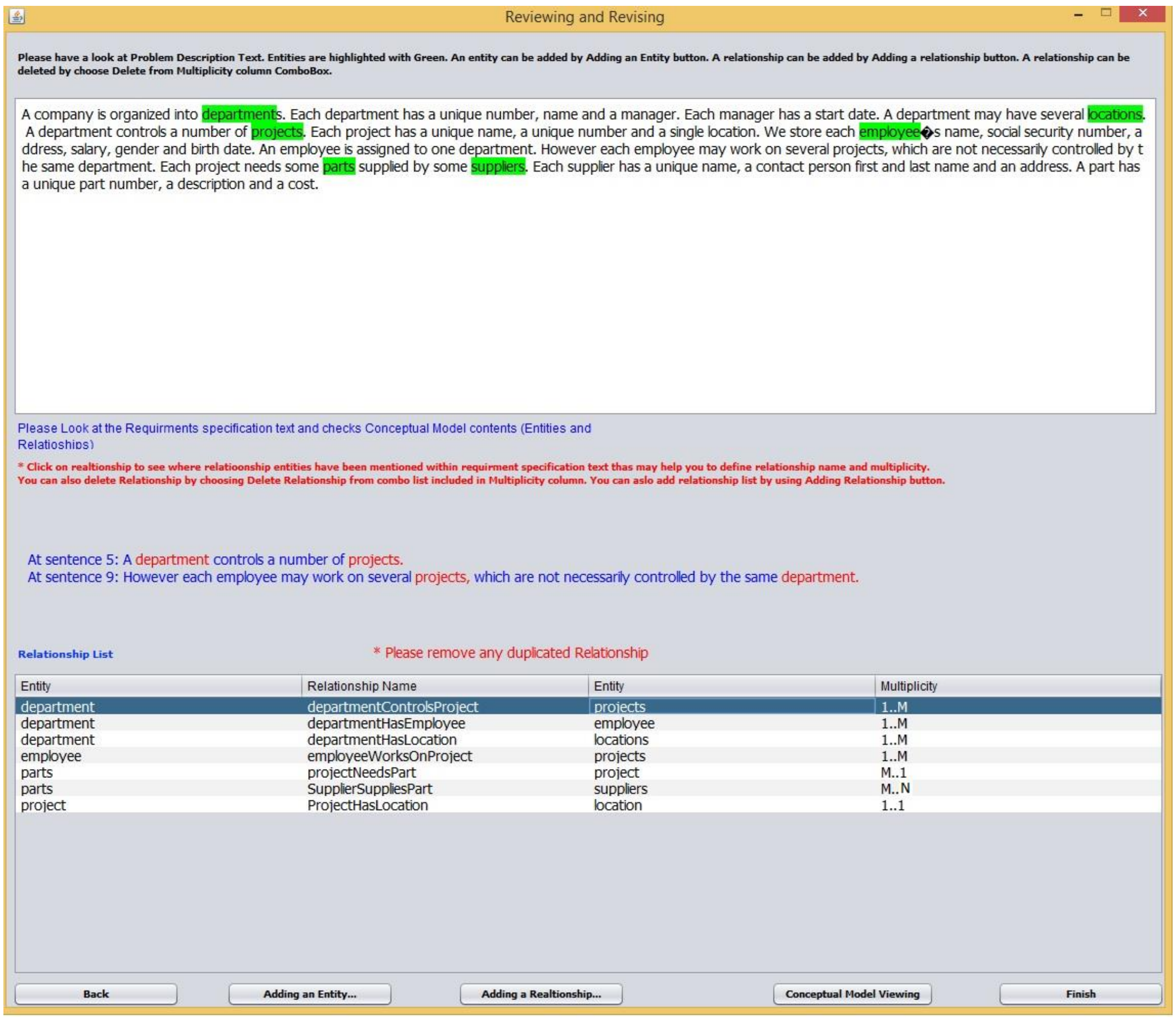
Relationship List

* Please remove any duplicated Relationship

Entity 1	RelationshipName	Entity 2	Multiplicity
department	departmentHasEmployee	employee	1..M
department	departmentHasLocation	locations	1..M
department	departmentControlsProject	projects	1..M
employee	employeeWorksOnProject	projects	1..M
parts	projectNeedsPart	project	M..1
parts	SupplierSuppliesPart	suppliers	M..M
project	ProjectHasLocation	location	1..1

Figure 4.11 Defining Names and Cardinality for Relationships

Figure 4.11 demonstrates how the system shows information and examples to help the user identify a name and cardinality for each relationship. Stanford typed dependencies can help in defining a name for a relationship, and the RHKB can identify cardinality for relationships, but even when Stanford typed dependencies have given a name for a relationship, the user can modify it. Similarly, even though the RHKB may have suggested a cardinality for the relationship, the user can also amend this. Here, the author acted as designer and selected an appropriate name and cardinality for each relationship, as shown in Figure 4.11.


Reviewing and Revising

Please have a look at Problem Description Text. Entities are highlighted with Green. An entity can be added by Adding an Entity button. A relationship can be added by Adding a relationship button. A relationship can be deleted by choose Delete from Multiplicity column ComboBox.

A company is organized into departments. Each department has a unique number, name and a manager. Each manager has a start date. A department may have several locations. A department controls a number of projects. Each project has a unique name, a unique number and a single location. We store each employee�s name, social security number, a ddress, salary, gender and birth date. An employee is assigned to one department. However each employee may work on several projects, which are not necessarily controlled by t he same department. Each project needs some parts supplied by some suppliers. Each supplier has a unique name, a contact person first and last name and an address. A part has a unique part number, a description and a cost.

Please Look at the Requirments specification text and checks Conceptual Model contents (Entities and Relatioships)

*** Click on realtionship to see where relatioonship entities have been mentioned within requirment specification text thas may help you to define relationship name and multiplicity. You can also delete Relationship by choosing Delete Relationship from combo list included in Multiplicity column. You can aslo add relationship list by using Adding Relationship button.**

At sentence 5: A department controls a number of projects.
At sentence 9: However each employee may work on several projects, which are not necessarily controlled by the same department.

Relationship List

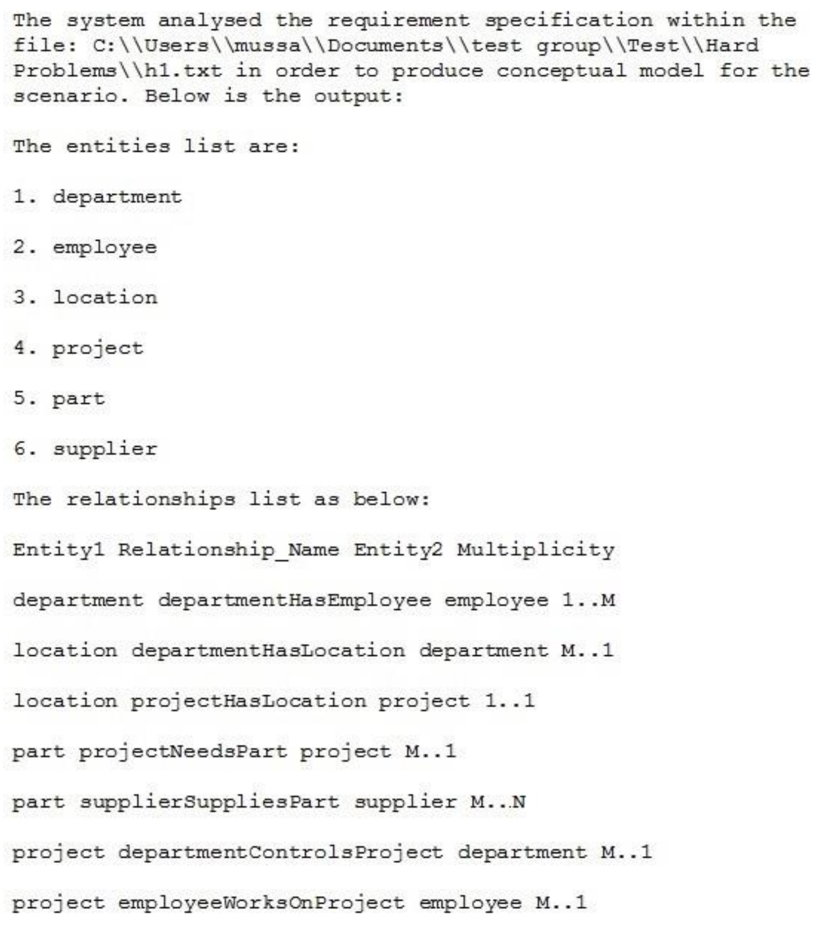
* Please remove any duplicated Relationship

Entity	Relationship Name	Entity	Multiplicity
department	departmentControlsProject	projects	1..M
department	departmentHasEmployee	employee	1..M
department	departmentHasLocation	locations	1..M
employee	employeeWorksOnProject	projects	1..M
parts	projectNeedsPart	project	M..1
parts	SupplierSuppliesPart	suppliers	M..N
project	ProjectHasLocation	location	1..1

Figure 4.12 Review and Revision Form

Figure 4.12 demonstrates how the system gives the user an opportunity to review the conceptual model. The review and revision form shows the requirement specification text. The entities within the text are highlighted in green in order to distinguish them. The form also displays the relationships identified by the designer in the previous steps. At this point, the user can remove or add relationships and update the cardinality. When the user clicks on a specific relationship, the system displays text showing where this relationship appears in the requirement specifications. The user can go back to previous steps by clicking the 'Back' button on the form, or add an entity by clicking on the 'Adding an Entity' button. The user can also add a relationship by clicking on the 'Adding a Relationship' button. When a relationship is added by the user, it is also added into the relationships list. When the user is satisfied with the conceptual model, s/he clicks on the 'Conceptual Model Viewing' button to view a report about the conceptual model for the requirement specification text, as shown in

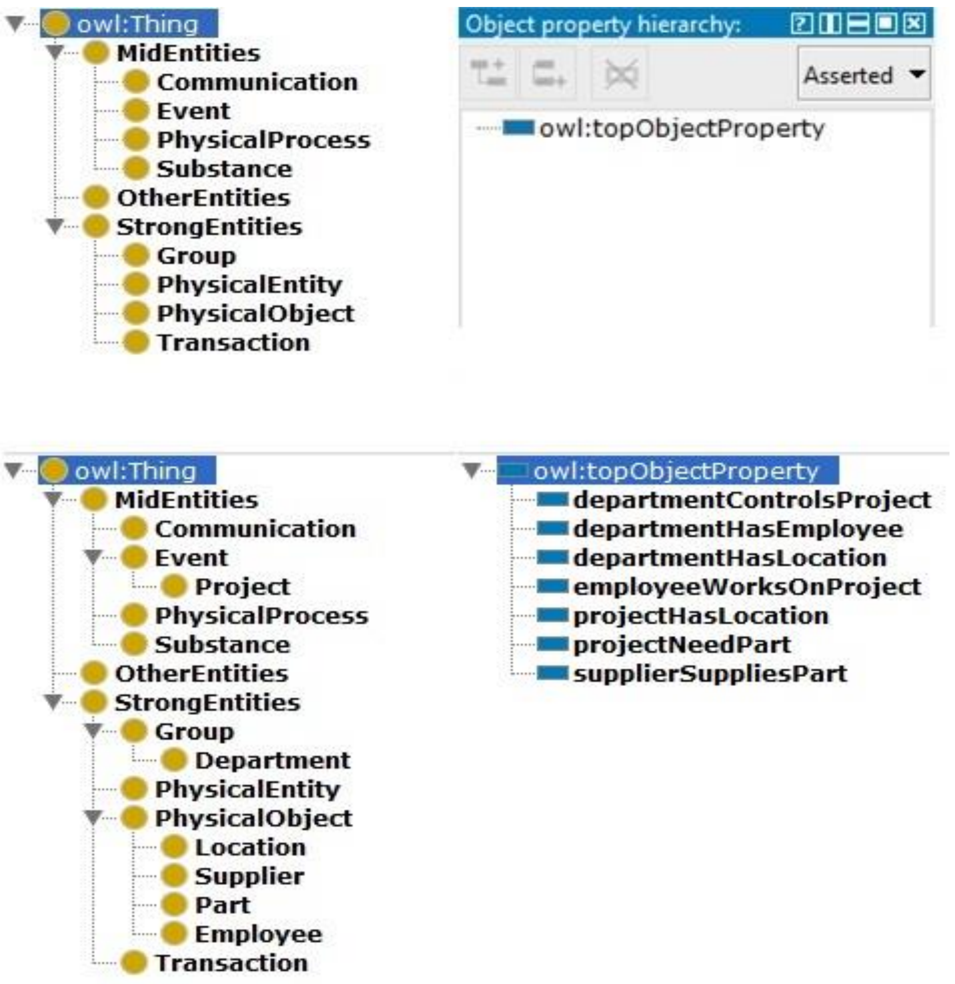
Figure 4.13. Before the report viewing stage, however, the system eliminates each entity that is not included in a relationship unless it has been added by the user.



```
The system analysed the requirement specification within the
file: C:\\Users\\mussa\\Documents\\test group\\Test\\Hard
Problems\\h1.txt in order to produce conceptual model for the
scenario. Below is the output:

The entities list are:

1. department

2. employee

3. location

4. project

5. part

6. supplier

The relationships list as below:

Entity1 Relationship_Name Entity2 Multiplicity

department departmentHasEmployee employee 1..M

location departmentHasLocation department M..1

location projectHasLocation project 1..1

part projectNeedsPart project M..1

part supplierSuppliesPart supplier M..N

project departmentControlsProject department M..1

project employeeWorksOnProject employee M..1
```

Figure 4.13 Report Displaying Information for the Conceptual Model

The conceptual model report is displayed as in Figure 4.13. In addition, the system inserts entities into the CMO to update it, unless the entities already exist within the ontology. The system also inserts relationships into the ontology unless they already appear there. Figure 4.14 shows the hierarchy of the ontology before processing the requirement specification for the company database, and then after the processing has been completed. Before processing, the ontology hierarchy was blank, having no entities and no relationships, whereas after processing the requirements of the company, the entities and relationships extracted for the company's conceptual model have been added. When another requirement is processed by the system, the system will use the information stored in the ontology to advise the user with regard to the creation of a new conceptual model.

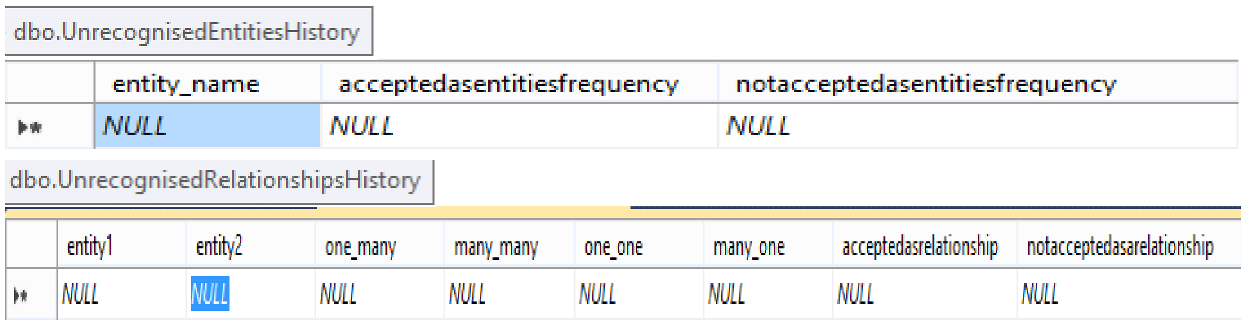

```
owl:Thing
    MidEntities
        Communication
        Event
        PhysicalProcess
        Substance
    OtherEntities
    StrongEntities
        Group
        PhysicalEntity
        PhysicalObject
        Transaction
```

```
owl:topObjectProperty
```

```
owl:Thing
    MidEntities
        Communication
        Event
            Project
        PhysicalProcess
        Substance
    OtherEntities
    StrongEntities
        Group
            Department
        PhysicalEntity
        PhysicalObject
            Location
            Supplier
            Part
            Employee
        Transaction
```

```
owl:topObjectProperty
    departmentControlsProject
    departmentHasEmployee
    departmentHasLocation
    employeeWorksOnProject
    projectHasLocation
    projectNeedPart
    supplierSuppliesPart
```

Figure 4.14 Ontology Hierarchy before and after Processing the Company Requirements

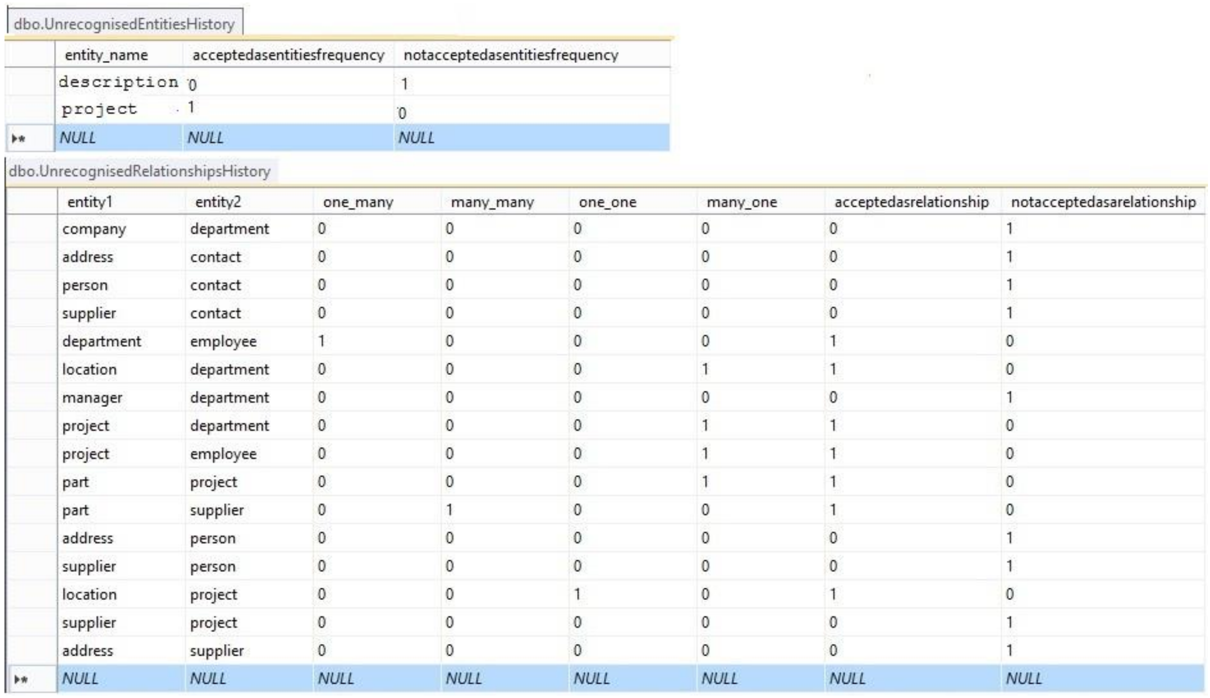

dbo.UnrecognisedEntitiesHistory

	entity_name	acceptedasentitiesfrequency	notacceptedasentitiesfrequency
▸*	NULL	NULL	NULL

dbo.UnrecognisedRelationshipsHistory

	entity1	entity2	one_many	many_many	one_one	many_one	acceptedasrelationship	notacceptedasarelationship
▸*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 4.15 Entities and Relationships History before Processing Company Database



dbo.UnrecognisedEntitiesHistory

entity_name	acceptedasentitiesfrequency	notacceptedasentitiesfrequency
description	0	1
project	1	0
NULL	NULL	NULL

dbo.UnrecognisedRelationshipsHistory

entity1	entity2	one_many	many_many	one_one	many_one	acceptedasrelationship	notacceptedasarelationship
company	department	0	0	0	0	0	1
address	contact	0	0	0	0	0	1
person	contact	0	0	0	0	0	1
supplier	contact	0	0	0	0	0	1
department	employee	1	0	0	0	1	0
location	department	0	0	0	1	1	0
manager	department	0	0	0	0	0	1
project	department	0	0	0	1	1	0
project	employee	0	0	0	1	1	0
part	project	0	0	0	1	1	0
part	supplier	0	1	0	0	1	0
address	person	0	0	0	0	0	1
supplier	person	0	0	0	0	0	1
location	project	0	0	1	0	1	0
supplier	project	0	0	0	0	0	1
address	supplier	0	0	0	0	0	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 4.16 Entities and Relationships History after Processing Company Database

The system also updates the UHKB database. Figure 4.15 shows the UHKB database before processing the company's requirements, while Figure 4.16 shows the entities history database after the processing. The UHKB database was blank before processing the company's requirements, but after processing, new information has been added into the history. In the EHKB (the table that has three columns), the first row means the noun 'description' has not been accepted as an entity once and has been accepted as an entity zero times, whereas the noun 'project' has been accepted as an entity once, and has not been accepted as an entity zero times. For any other requirements processed by the system, when the user is requested to use human intervention to decide whether the noun 'project' should be an entity or not, the system will recommend that the noun 'project' is an entity based on the EHKB available within the system. In the current EHKB, the chance of the noun 'project' being an entity is greater than the chance of it not being an entity. However, if the chance of being an entity is equal to the chance of not being an entity, then the system will not make a recommendation and will rely on the user to decide. The RHKB (the table that has eight columns) was blank before the system processed the company database, whereas after the processing, information has been added into the history. The first row of the RHKB means the relationship between 'company' and 'department' has been considered zero times as a relationship and once as not a relationship. When a future requirement specification is processed, the system will try to retrieve information to help the user make a correct decision based on information found in the RHKB.

4.3 Chapter Summary

In this chapter, the author implements a semi-automated model to help designers create conceptual models from natural language text. The model incorporates a linguistics approach, an ontological approach, natural language processing tools and human intervention, to achieve its goal. The main differences between the present model and earlier models are: (1) the model learns from the designers and from the natural language text that it processes. The model stores entities and relationships that obtained at the end of each mapping in conceptual model ontology and stores designers behaviour in a relational database; (2) the model uses the information that is stored in the ontology and the database to improve its performance and to reduce the need for human intervention. The author expects that, (1) the performance of the designers who use the model will improve when compared to their handcrafted performance, (2) the information that is stored by the model will improve the performance of the model and reduce the need for human intervention. These expectations will be tested in the next chapter.

Chapter 5: Empirical Evaluation of SACMES

This chapter shows how SACMES has been evaluated. The author aims to demonstrate that the performance of designers will be improved when using SACMES, in comparison to their manual performance. The author would also like to show that the information stored by SACMES will help the system to improve its performance and to minimise the need for human intervention.

5.1 Experimental Design One

In this section, an empirical evaluation is conducted to confirm that the performance of designers will be improved when using SACMES, in comparison to their manual performance. A test set of twenty case studies has been established, the case studies having been collected from authentic resources including database textbooks and PhD theses. The test set is divided into easy problems and harder problems, with ten case studies in each sub-set. Clearly, the easy problems are less complex than the harder problems, and the use of both types is intended to demonstrate that the system can deal with both easy and complex cases. Each case study has a set of model answers, which includes entities, relationships and cardinalities of relationships. Some cases were found with their model answers, while other model answers were created by an expert designer[21]. Appendix 2 illustrates the test set with their model answers. The author is confident about the test set count of twenty cases, as some studies similar to this one have used fewer case studies to test the performance of their tools. For example, Elbandack's (2011) study used a test corpus of eight case studies to measure the performance of the Class-Gen tool that maps natural language text into objects/classes. Thonggoom (2011) used a corpus of four case studies to test the performance of the Heuristic-Based Technique (HBT) and Entity Instance Pattern WordNet (EIPW) tools that map natural language text into ERDs. Furthermore, Song et al (2004) used eight case studies to test the performance of Taxonomic Class Modelling (TCM), which identifies classes from natural language. Twenty subjects participated in the experiment, all of whom are novice designers. The author is also confident about the number of the subjects participating in the

[21] Haddeel Jazzaa, Currently (2018) a PhD student in the Informatics Department of the Computing & Engineering School at Huddersfield University in the UK. She worked from 2001 to 2009 in Iraq at the State Company for Information Systems as a database designer and programmer, and from 2009 to 2015 she worked at the Federal Board of Supreme Audit located in Alkarkh-Baghdad, Iraq (http://www.fbsa.gov.iq) as a database designer. She played the role of database designer for this research and designed model answers for the case studies that did not already have them.

study, as Elbandack's (2011) study used just nine subjects to test the performance of the Class-Gen tool.

While expert designers are more capable and skilled at translating natural language specifications into conceptual models, novice designers are less skilled at this task. The author wished to observe how SACMES would support such designers in producing conceptual models, and it was for this reason that the author chose to include novice designers as subjects for the experiment. Each subject was requested to fill in a questionnaire. This questionnaire helped the author to determine the extent to which the subjects were suitable for participation in the experiment, to discover their background with regard to conceptual model creation, and to receive feedback regarding their use of SACMES. The questionnaire was adapted from Thonggoom (2011) and is demonstrated in Appendix 3. All the subjects are students in the Informatics Department of the Computing and Engineering School at the University of Huddersfield in the United Kingdom. Several of them are undergraduate students, while others are postgraduates. None of them have extensive experience in the creation of conceptual models, though the majority have studied conceptual models during their undergraduate or postgraduate courses. The subjects were divided into two groups, namely, Group One and Group Two, with ten subjects in each group. Each subject provided four answers for four different case studies from the test set, two of these case studies being from the easy group and two from the harder group. Two of their answers would be handcrafted answers while the others would be provided by using SACMES. Table 5.1 illustrates the activities undertaken by the subjects during the experiment. For example, subject number one was requested to give answers for four case studies, which comprised: (1) case number one in the easy set, for which the subject would give a handcrafted answer; (2) case number two in the easy set, for which the subject would use SACMES to produce an answer; (3) case number one in the harder set, for which the subject would give a handcrafted answer; and (4) case number two in the harder set, for which the subject would again use SACMES.

Subject	Problem	Problem	Problem	Problem
S1	E1WO	E2W	H1WO	H2W
S2	E1W	E2WO	H1W	H2WO
S3	E3WO	E4W	H3WO	H4W
S4	E3W	E4WO	H3W	H4WO

Subject	Problem	Problem	Problem	Problem
S5	E5WO	E6W	H5WO	H6W
S6	E5W	E6WO	H5W	H6WO
S7	E7WO	E8W	H7WO	H8W
S8	E7W	E8WO	H7W	H8WO
S9	E9WO	E10W	H9WO	H10W
S10	E9W	E10WO	H9W	H10WO
Table Key E: Easy case study H: Harder case study S: Subject W: With using SACMES WO: Without using SACMES				

Table 5.1 Subjects' Activities in the Experiment

The subjects in the first group provided manual answers first and then used the system to provide the other answers, whereas the subjects in the second group started by using the system and then provided their manual answers afterwards. Both the answers that were manually produced, and those provided by the subjects' use of the system, were compared with the model answers in order to determine the extent to which the subjects' performance was affected by using SACMES. Answers provided by subjects with the help of the system are called system answers, while those provided without using the system are called manual answers. The subjects' answers are classified into three classes, which are Correct (COR), Incorrect (INC) and Missed (MISS). An answer is classified as correct when it is found as both a model answer and a system answer, or a model answer and a manual answer. An answer is classified as incorrect when it is found in the system answer or the manual answer but is not included in the model answer. An answer is classified as missed when it is included in the model answer but not found in the system answer or manual answer. Recall and precision are used to evaluate the extent to which system answers and manual answers match model answers. Recall and precession were originally developed for use in evaluating information retrieval systems, but are now most wildly used to evaluate the performance of information extraction systems (Elbendak, 2011). Recall measures to what extent the answers given by the information extraction system are complete, while precision measures to what extent the answers extracted by the information system are correct (Grishman & Sundheim, 1996). They are calculated by using the following equations.

Recall= ($N_{correct}$ / ($N_{correct}$ + N_{missed})) * 100 (Elbendak, 2011)

$N_{correct}$: Total number of correct answers.

N_{missed}: Total number of missed answers.

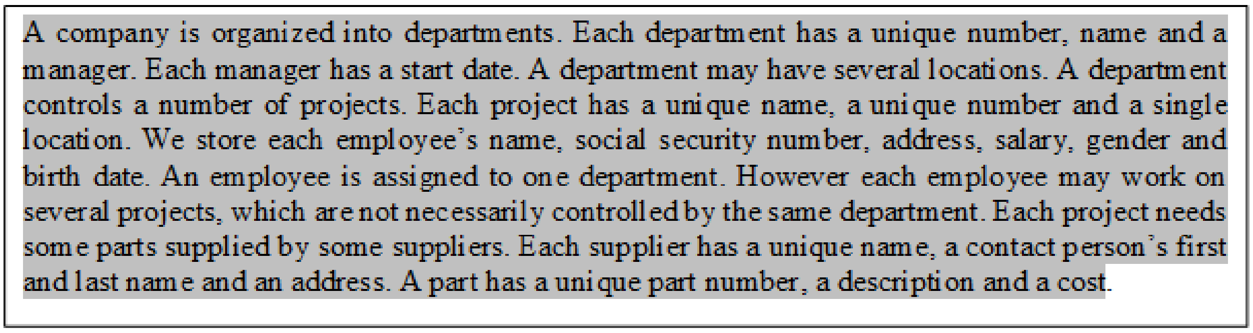
Precision= ($N_{correct}$ / ($N_{correct}$ + $N_{incorrect}$)) * 100 (Elbendak, 2011)

$N_{incorrect}$: Total number of incorrect answers.

5.1.1 First Group Results

5.1.1.1 Entities extraction

Entities in the model answers were compared with the system answers and manual answers. Figure 5.1 shows the requirement specifications for case study number one in the harder set. Figure 5.2 shows the model answer for this case study, Figure 5.3 shows the answer provided manually by a subject without using the system, Figure 5.4 presents the answer provided by a subject with help from the system and Table 5.2 shows a comparison between the answers.



A company is organized into departments. Each department has a unique number, name and a manager. Each manager has a start date. A department may have several locations. A department controls a number of projects. Each project has a unique name, a unique number and a single location. We store each employee's name, social security number, address, salary, gender and birth date. An employee is assigned to one department. However each employee may work on several projects, which are not necessarily controlled by the same department. Each project needs some parts supplied by some suppliers. Each supplier has a unique name, a contact person's first and last name and an address. A part has a unique part number, a description and a cost.

Figure 5.1 Company Database (Du, 2008, p. 170)

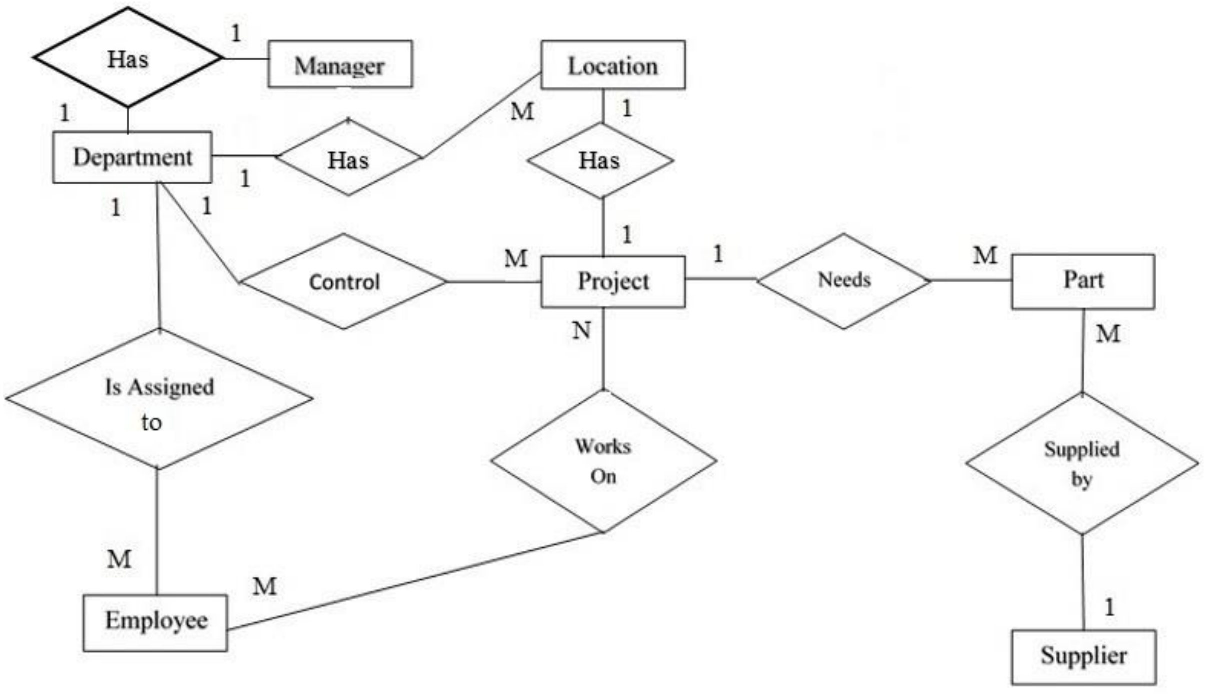


Figure 5.2 Model Answer for Company Database

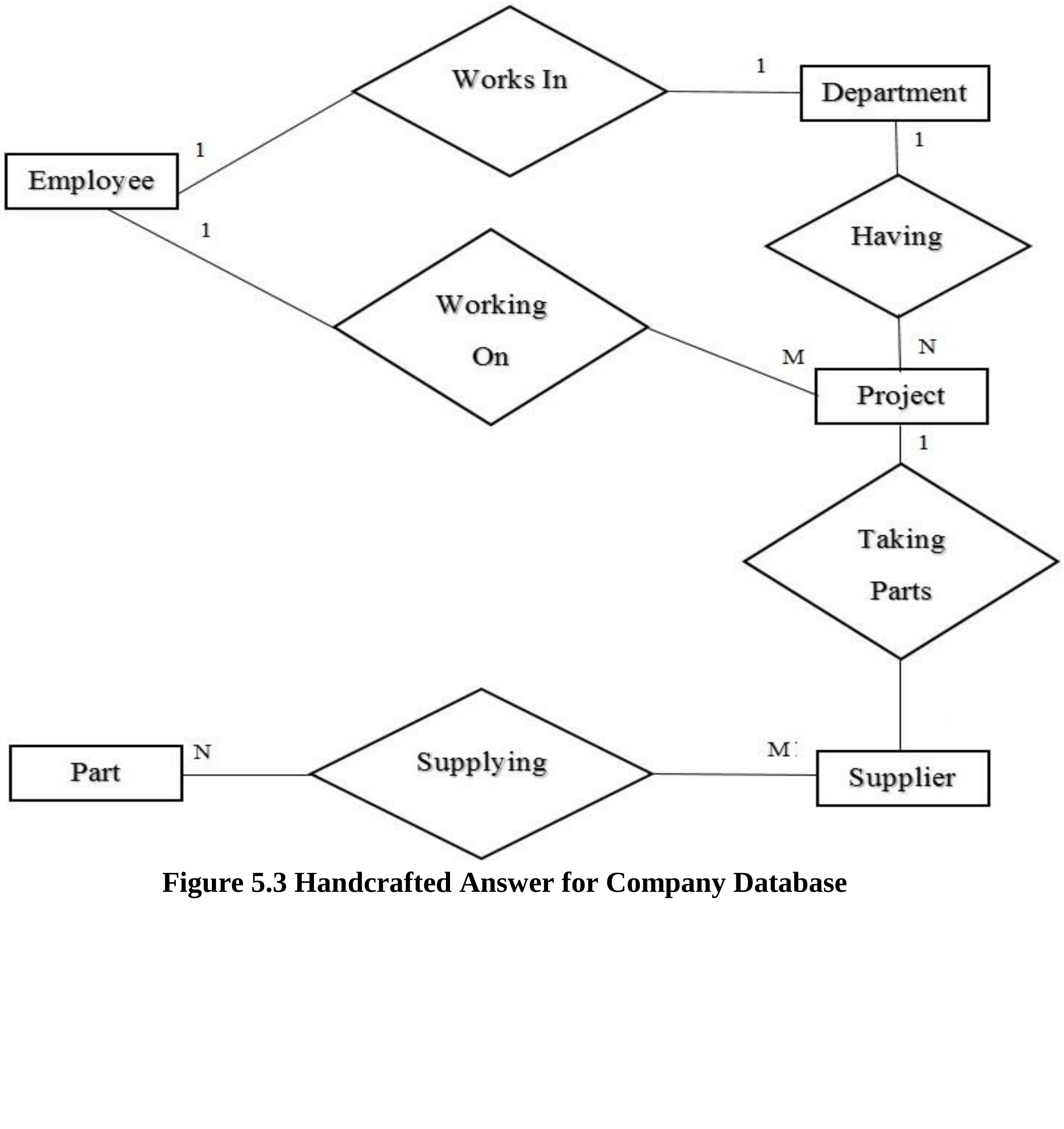
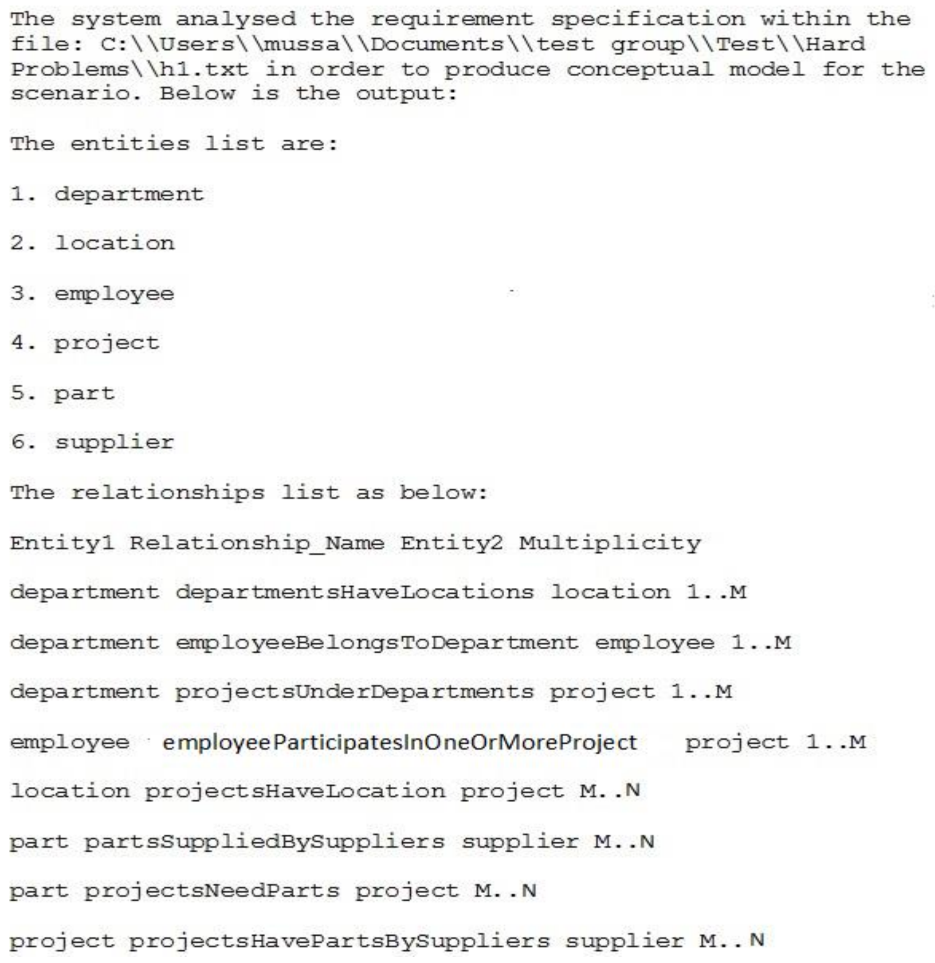


Figure 5.3 Handcrafted Answer for Company Database



```
The system analysed the requirement specification within the
file: C:\\Users\\mussa\\Documents\\test group\\Test\\Hard
Problems\\h1.txt in order to produce conceptual model for the
scenario. Below is the output:

The entities list are:

1. department

2. location

3. employee

4. project

5. part

6. supplier

The relationships list as below:

Entity1 Relationship_Name Entity2 Multiplicity

department departmentsHaveLocations location 1..M

department employeeBelongsToDepartment employee 1..M

department projectsUnderDepartments project 1..M

employee  employeeParticipatesInOneOrMoreProject   project 1..M

location projectsHaveLocation project M..N

part partsSuppliedBySuppliers supplier M..N

part projectsNeedParts project M..N

project projectsHavePartsBySuppliers supplier M..N
```

Figure 5.4 System Answer for Company Database

Model Answer	System Answer	Class	Manual Answer	Class
Department	Department	COR	Department	COR
Manager		MISS		MISS
Location	Location	COR		MISS
Part	Part	COR	Part	COR
Supplier	Supplier	COR	Supplier	COR
Employee	Employee	COR	Employee	COR
Project	Project	COR	Project	COR

Table 5.2 Comparison between System Answer and Manual Answer based on Model Answer for Company Database in Harder Problems Set

In Table 5.2, the first column represents entities that are found in the model answer. The second column represents entities found by a subject as a solution for the company database using the system. The third column represents entities found by a subject as a handcrafted solution without using SACMES. When compared with the model answer, the system answer has six correct answers and one answer missed, whereas the handcrafted answer has five correct answers and two answers missed. Recall and precision were calculated for both the system and manual answers. Recall for the system answer is 85.71% and the precision is 100%, whereas the recall for the manual answer is 71.42% and the precision is 100%. These results show that a better outcome is obtained when the system is used. This process was repeated with the entire test set. The results of these comparisons are presented in Table 5.3.

Subject 1	E1WO + H1WO		E2W + H2W	
	Recall	Precision	Recall	Precision
	50%	100%	60%	100%
	71.42%	100%	100%	100%
Total	121.42	200	160	200
Average	60.71%	100%	80%	100%
Subject 2	E1W + H1W		E2WO + H2WO	
	Recall	Precision	Recall	Precision
	62.5%	100%	60%	100%
	85.71%	100%	50%	75%
Total	148.21	171.42	110	175
Average	74.10%	85.71%	55%	87.5%
Subject 3	E3WO + H3WO		E4W + H4W	
	Recall	Precision	Recall	Precision
	100%	100%	100%	71.42%
	100%	70%	100%	100%
Total	200	170	200	171.42
Average	100%	85%	100%	85.71%
Subject 4	E3W + H3W		E4WO + H4WO	
	Recall	Precision	Recall	Precision
	75%	100%	60%	50%
	85.71%	75%	100%	57.14%
Total	160.71	175	160	107.14
Average	80.35%	87.5%	80%	53.57%
Subject 5	E5WO + H5WO		E6W + H6W	
	Recall	Precision	Recall	Precision

	100%	100%	83.33%	100%
	80%	100%	83.33%	83.33%
Total	180	200	166.66	183.33
Average	90%	100%	83.33%	91.66%
Subject 6	E5W + H5W		E6WO + H6WO	
	Recall	Precision	Recall	Precision
	100%	83.33%	50%	75%
	80%	100%	66.66%	80%
Total	180	183.33	116.66	155
Average	90%	91.66%	58.33%	77.5%
Subject 7	E7WO + H7WO		E8W + H8W	
	Recall	Precision	Recall	Precision
	80%	100%	80%	66.66%
	25%	25%	90.90%	83.83%
Total	105	125	170.9	150.49
Average	52.5%	60.5%	85.45%	75.24%
Subject 8	E7W + H7W		E8WO + H8WO	
	Recall	Precision	Recall	Precision
	100%	62.5%	80%	80%
	50%	50%	90.90%	90.90%
Total	150	112.5	170.9	170.9
Average	75%	56.25%	85.45%	85.45%
Subject 9	E9WO + H9WO		E10W + H10W	
	Recall	Precision	Recall	Precision
	80%	100%	100%	100%
	72.72%	100%	80%	100%
Total	152.72	200	180	200
Average	76.36%	100%	90%	100%
Subject 10	E9W + H9W		E10WO + H10WO	
	Recall	Precision	Recall	Precision
	100%	83.33%	80%	80%
	72.72%	88.88%	33.33%	42.85%
Total	172.72	172.21	113.33	122.85
Average	86.36%	86.10%	56.66%	61.42%
	Manual Answers			

	Recall	Precision
Total	1430.03	1625.89
Average	71.50%	81.29%
	System Answers	
	Recall	Precision
Total	1689.2	1748.28
Average	84.46%	87.41%
Table Key E: Easy case study. H: Harder case study. S: Subject W: With using SACMES. WO: Without SACMES.		

Table 5.3 Comparison between System Answers and Manual Answers for Entities Extraction based on Model Answers

From the results displayed in Table 5.3, it can be concluded that novice designers' performance in entities extraction improved when using SACMES. The recall improved from 71.50% to 84.46% and precision improved from 81.29% to 87.41%. An average was taken to measure the performance of each subject when using the system and when providing handcrafted answers. The results show that the overall performance of the subjects improved when they used the system. For example, subject number one was requested to answer case study number one in the easy set and case study number one in the harder set by giving handcrafted answers. The average for the handcrafted answers is 60.71% for recall and 100% for precision, whereas for the subject requested to answer case study number two in the easy set and case study number two in the harder set by using SACMES, the average for the SACMES answers is 80% for recall and 100% for precision. Only subject numbers five and eight achieved a better performance when providing handcrafted answers than when using the system. The author did not expect that the subjects' performance when using the system would always be better than when not using it. However, it was expected that their overall performance would be better when using the system than when using a manual approach to obtain conceptual models from natural language text. This overall improvement was demonstrated by the results.

5.1.1.2 Relationships extraction

Relationships in the model answers were compared with the system answers and manual answers. Table 5.4 shows a comparison between a system answer and a manual answer based on relationships found in the key answer.

Model Answer	System Answer	Class	Manual Answer	Class
Department Has Location	Department Departments HaveLocations Location	COR		MISS
Department Has Manager		MISS		MISS
Project Has Location	Location ProjectsHaveLocation Project	COR		MISS
Department Controls Project	Department ProjectsUnderDepartments Project	COR		MISS
Employee Is Assigned To Department	Department EmployeeBelongsToDepartment Employee	COR	Employee Works In Department	COR
Employee Works On Project	Employee EmployeeParticipatesInOneOr-MoreProjects Project	COR	Employee Works On Project	COR
Project Needs Part	Part ProjectsNeedParts Project	COR		MISS
Supplier Supplies Part	Part PartsSuppliedBySuppliers Supplier	COR	Supplier Supplying Part	COR
	Project ProjectsHavePartsBySuppliers Supplier	INC		
			Department Having Project	INC
			Project Taking Parts Supplier	INC

Table 5.4 Comparing Relationships Found in System Answer and Handcrafted Answer based on Model Answer for Company Database Case Study

When compared with the model answer, the system answer has seven correct answers, one missed answer and one incorrect answer. When comparing the manual answer with the model answer, there are three correct answers, five missed answers and two incorrect answers. Recall and precision are both 87.5% for the system answer compared to the model answer, whereas recall is 37.5% and precision is 60% for the manual answer in comparison with the model answer. These results indicate that there is improvement in the performance when the system used. This process was repeated with all case studies in the test set and the results of the comparisons are represented in Table 5.5.

Subject 1	E1WO + H1WO		E2W + H2W	
	Recall	Precision	Recall	Precision
	20%	40%	50%	100%
	37.5%	60%	100%	100%
Total	57.5	100	150	200
Average	28.75%	50%	75%	100%
Subject 2	E1W+ H1W		E2WO+ H2WO	
	Recall	Precision	Recall	Precision
	10%	14.28%	25%	50%
	87.5%	87.5%	20%	33.33%
Total	97.5	101.78	45%	83.33
Average	48.75%	50.89%	22.5%	41.66%
Subject 3	E3WO + H3WO		E4W + H4W	
	Recall	Precision	Recall	Precision
	100%	100%	75%	42.85%
	54.54%	50%	50%	66.66%
Total	154.54	150	125	109.51
Average	77.27%	75%	62.5%	54.75%
Subject 4	E3W + H3W		E4WO + H4WO	
	Recall	Precision	Recall	Precision
	66.66	100	25%	25%
	63.63	63.63	50%	40%
Total	130.29	163.63	75	65
Average	65.14%	81.81%	35%	32.5%
Subject 5	E5WO + H5WO		E6W + H6W	
	Recall	Precision	Recall	Precision
	100%	100%	80%	80%
	80%	100%	50%	37.5%
Total	180	200	130	117.5
Average	90%	100%	65%	58.75%
Subject 6	E5W + H5W		E6WO + H6WO	
	Recall	Precision	Recall	Precision
	100%	75%	0%	0%
	80%	66.66%	16.66 %	20%
Total	180	141.66	16.66	20

Average	90%	70.83%	8.33%	10%
Subject 7	E7WO + H7WO		E8W + H8W	
	Recall	Precision	Recall	Precision
	33.33%	33.33%	50%	40%
	8.33%	8.33%	84.61%	61.11%
Total	41.66	41.66	134.61	101.11
Average	20.83%	20.83%	67.30%	50.5%
Subject 8	E7W + H7W		E8WO + H8WO	
	Recall	Precision	Recall	Precision
	100%	37.5	50%	50%
	16.66%	22.22	69.23%	69.23%
Total	116.66	59.72	119.23	119.23
Average	58.33%	29.86%	59.61%	59.61%
Subject 9	E9WO + H9WO		E10W + H10W	
	Recall	Precision	Recall	Precision
	50%	75%	100%	100%
	40%	57.14%	33.33%	80%
Total	90	132.14	133.33	180
Average	45%	66.07%	66.66%	90%
Subject 10	E9W + H9W		E10WO + H10WO	
	Recall	Precision	Recall	Precision
	33.33%	40%	50%	50%
	50%	35.71%	0%	0%
Total	83.33	75.71	50	50
Average	41.66%	37.85%	25%	25%
	Manual Answers			
	Recall		Precision	
Total	829.59		961.36	
Average	41.47		48.06	
	System Answers			
	Recall		Precision	
Total	1280.72		1250.35	
Average	64.03%		62.51%	

Table 5.5 Comparison between System Answers and Manual Answers for Relationship Extraction based on Model Answers

The results presented in Table 5.5 show that most subjects' performance in extracting relationships improved when they used the system. Recall improved from 41.47% to 64.03% and precision improved from 48.06% to 62.51%. The performance of subject numbers one, two, four, six, seven, nine and ten improved when they used the system. Only the performance of subject numbers three, five and eight was better when they did not use the system than when they did use it. The author is not concerned about the performance of these subjects, since it was not expected that every subject's performance would improve when using the system compared to when not using it. However, it was anticipated that the subjects' overall performance would be imrpoved when using SACMES and this is what has been demonstrated. Furthermore, as the system learns from the natural language text that it processes, the author is confident that the performance of the system will improve as it processes many more case studies. Therefore it is very encouraging that, even though the system had so far only processed a few case studies, the average performance of the subjects still improved when using it.

5.1.1.3 Cardinalities extraction

Cardinalities in the model answers were compared with those in the system answers and manual answers. Table 5.6 shows a comparison between a system answer and manual answer based on the relationship cardinalities found in the model answer for case study number one in the harder set.

Model Answer	System Answer	Class	Manual Answer	Class
Department Has Location(1-M)	Department DepartmentsHaveLocations Location 1..M	COR		MISS
Department Has Manager (1-1)		MISS		MISS
Project has Location (1-1)	Location ProjectsHaveLocation Project M..N	INC		MISS
Department Controls Project (1-M)	Department ProjectsUnderDepartments Project 1..M	COR		MISS
Employee Is Assigned To Department (M-1)	Department EmployeeBelongsToDepartment Employee 1..M	COR	Employee Works in department (1-1)	INC
Employee Works On Project (M-N)	Employee EmployeeParticipatesInOneOrM-oreProjects Project 1..M	INC	Employee Works On Project (1-N)	INC
Project Needs Part (1-M)	Part ProjectsNeedParts project M..N	INC		MISS

Model Answer	System Answer	Class	Manual Answer	Class
Supplier Supplies Part (1-M)	Part PartsSuppliedBySuppliers supplier M..N	INC	Supplier Supplying Part (M-N)	INC
	Project ProjectsHavePartsBySuppliers Supplier M..N	INC		
			Department Having Project (1-N)	INC
			Project Taking Parts Supplier (1-N)	INC

Table 5.6 Comparing Relationship Cardinalities Found in System Answer and Manual Answer based on Relationship Cardinalities Found in Model Answer for Company Database Scenario

The result is 75% for recall and 37.5% precision when the system answer is compared to the model answer, whereas the result is Zero% for both recall and precision when the manual answer is compared to the model answer. This demonstrates that performance for extracting the cardinalities of relationships improved when the system was used. This procedure was repeated with all case studies in the test set. The results of these comparisons are represented in Table 5.7.

Subject 1	E1WO + H1WO		E2W + H2W	
	Recall	Precision	Recall	Precision
	20%	40%	33.33%	50%
	0%	0%	100%	100%
Total	20	40	133.33	150
Average	10%	20%	66.66%	75%
Subject 2	E1W + H1W		E2WO + H2WO	
	Recall	Precision	Recall	Precision
	0%	0%	0%	0%
	75%	37.5%	0%	0%
Total	75	37.5	0	0
Average	35.5%	18.75 %	0%	0%
Subject 3	E3WO + H3WO		E4W + H4W	
	Recall	Precision	Recall	Precision
	100%	100%	66.66%	28.57%
	33.33%	30%	33.33%	33.33%

Total	133.33	130	99.99	61.9
Average	66.66%	65%	49.99%	30.95%
Subject 4	E3W + H3W		E4WO + H4WO	
	Recall	Precision	Recall	Precision
	0%	0%	0%	0%
	33.33%	18.18%	33.33%	20%
Total	33.33	18.18	33.33%	20
Average	16.66%	9.09%	16.66%	10%
Subject 5	E5WO + H5WO		E6W + H6W	
	Recall	Precision	Recall	Precision
	100%	50%	0%	0%
	100%	75%	25%	12.5%
Total	200	125	25	12.5
Average	100%	62.5%	12.5%	6.25%
Subject 6	E5W + H5W		E6WO + H6WO	
	Recall	Precision	Recall	Precision
	0%	0%	0%	0%
	75%	50%	16.66%	20%
Total	75	50	16.66	20%
Average	37.5%	25%	8.33%	10%
Subject 7	E7WO + H7WO		E8W + H8W	
	Recall	Precision	Recall	Precision
	33.33%	33.33%	33.33%	20%
	0%	0%	80%	47.05%
Total	33.33	33.33	113.33	67.05
Average	16.66%	16.66%	56.66%	33.52%
Subject 8	E7W + H7W		E8WO + H8WO	
	Recall	Precision	Recall	Precision
	100%	22.22%	33.33%	25%
	9.09%	11.11%	63.63%	53.84%
Total	109.09	33.33	96.96	78.84
Average	54.54%	16.66%	48.48%	39.42%
Subject 9	E9WO + H9WO		E10W + H10W	
	Recall	Precision	Recall	Precision
	40%	50%	100%	75%

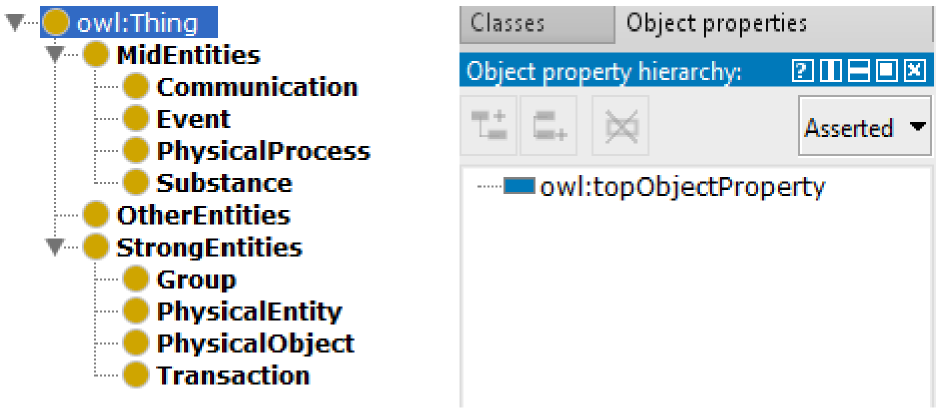
	25%	28.57%	27.27%	60%
Total	65	78.57	127.27	135
Average	32.5%	39.28 %	63.63%	67.5%
Subject 10	E9W + H9W		E10WO + H10W	
	Recall	Precision	Recall	Precision
	20%	20%	50%	50%
	44.44%	28.57%	0%	0%
Total	64.44	48.57	50	50
Average	32.22%	24.28%	25%	25%
	Manual Answers			
	Recall		Precision	
Total	648.61		575.14	
Average	32.43%		28.75%	
	System Answers			
	Recall		Precision	
Total	855.45		614.03	
Average	42.77		30.70	

Table 5.7 Comparison between System Answers and Manual Answers for Cardinalities of Relationships Extraction based on Model Answers

After considering the results found in Table 5.7, it can be concluded that the overall performance of subjects for extracting cardinalities of relationships improved when they used the system. Recall improved from 32.43% when not using the system to 42.77% when the subjects used the system. However, there was not a big improvement in the precision, which only increased from 28.75% when not using the system to 30.70% when the subjects used the system. The performance of five of the ten subjects improved when they used the system in comparison with when they did not use it. For subject numbers one, two, six, seven and nine, their performance when they used the system was better than when they did not. However, the performance of subject numbers three, four, five, eight and ten was better when they did not use the system than when they did use it. The author expectation is that, the overall performance of subjects when they use the system will improve comparing to their performance when they use handcrafted answers and this what has been obtained. Furthermore, as the system learns from natural language text that it processed, the author is confident the performance of the system will improve as the system process many and many case studies. Although, the system has processed several case studies, the average performance of the subjects improved when they use the system. Therefore, obtained result is very encouraging.

Overall, the novice designers' performance in extracting entities improved when they used the system. Their performance in extracting relationships and cardinalities of relationships also improved when they used the system. This result supports the hypothesis that the performance of novice designers will improve when they use SACMES comparing to their manual performance.

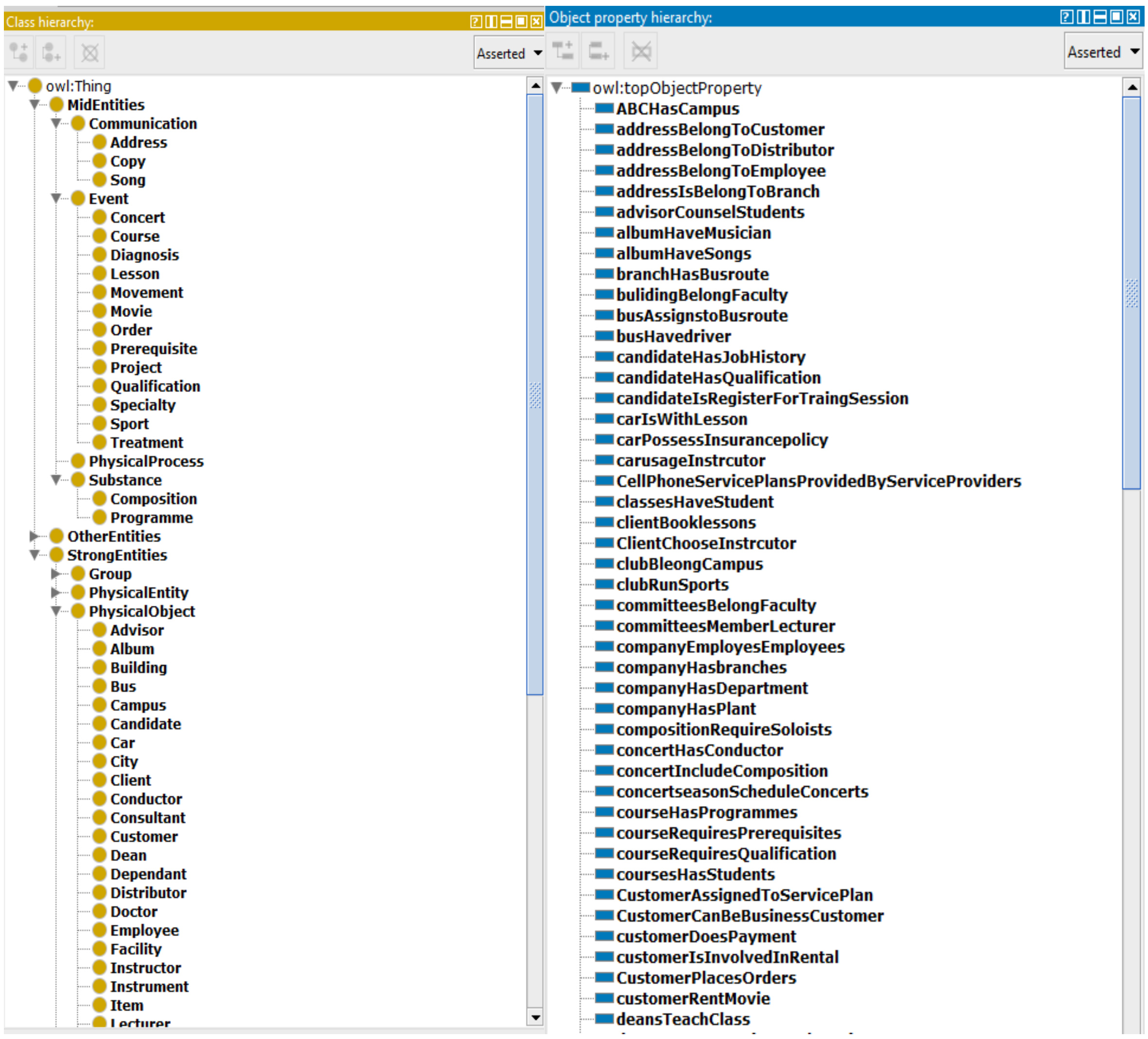
By comparing the CMO before and after the experiment, it can be noted that many entities have been added to the ontology, as well as many relationships. Figure 5.5 shows a screenshot of the ontology before the experiment, and Figure 5.6 shows a screenshot of the ontology after the experiment.



```
owl:Thing
    MidEntities
        Communication
        Event
        PhysicalProcess
        Substance
    OtherEntities
    StrongEntities
        Group
        PhysicalEntity
        PhysicalObject
        Transaction
```

```
owl:topObjectProperty
```

Figure 5.5 Screenshot of Entities Hierarchy and Relationships Hierarchy before the Experiment


Class hierarchy:

Asserted

- owl:Thing
  - MidEntities
    - Communication
      - Address
      - Copy
      - Song
    - Event
      - Concert
      - Course
      - Diagnosis
      - Lesson
      - Movement
      - Movie
      - Order
      - Prerequisite
      - Project
      - Qualification
      - Specialty
      - Sport
      - Treatment
    - PhysicalProcess
    - Substance
      - Composition
      - Programme
  - OtherEntities
  - StrongEntities
    - Group
    - PhysicalEntity
    - PhysicalObject
      - Advisor
      - Album
      - Building
      - Bus
      - Campus
      - Candidate
      - Car
      - City
      - Client
      - Conductor
      - Consultant
      - Customer
      - Dean
      - Dependant
      - Distributor
      - Doctor
      - Employee
      - Facility
      - Instructor
      - Instrument
      - Item
      - Lecturer

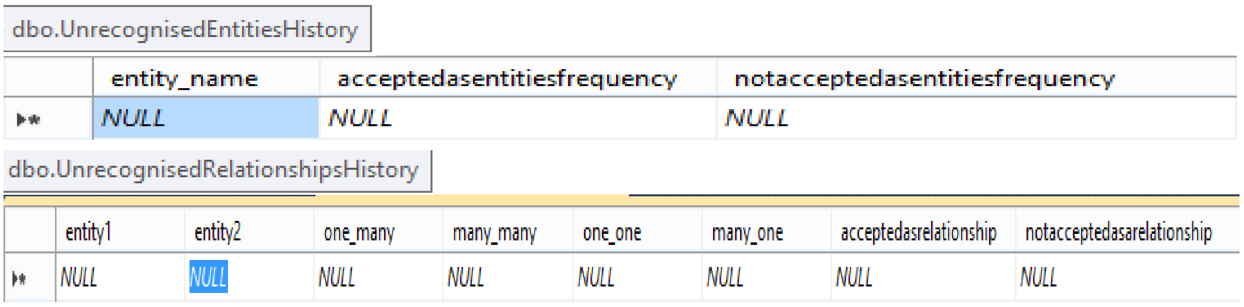
Object property hierarchy:

Asserted

- owl:topObjectProperty
  - ABCHasCampus
  - addressBelongToCustomer
  - addressBelongToDistributor
  - addressBelongToEmployee
  - addressIsBelongToBranch
  - advisorCounselStudents
  - albumHaveMusician
  - albumHaveSongs
  - branchHasBusroute
  - bulidingBelongFaculty
  - busAssignstoBusroute
  - busHavedriver
  - candidateHasJobHistory
  - candidateHasQualification
  - candidateIsRegisterForTraingSession
  - carIsWithLesson
  - carPossessInsurancepolicy
  - carusageInstrcutor
  - CellPhoneServicePlansProvidedByServiceProviders
  - classesHaveStudent
  - clientBooklessons
  - ClientChooseInstrcutor
  - clubBleongCampus
  - clubRunSports
  - committeesBelongFaculty
  - committeesMemberLecturer
  - companyEmployesEmployees
  - companyHasbranches
  - companyHasDepartment
  - companyHasPlant
  - compositionRequireSoloists
  - concertHasConductor
  - concertIncludeComposition
  - concertseasonScheduleConcerts
  - courseHasProgrammes
  - courseRequiresPrerequisites
  - courseRequiresQualification
  - coursesHasStudents
  - CustomerAssignedToServicePlan
  - CustomerCanBeBusinessCustomer
  - customerDoesPayment
  - customerIsInvolvedInRental
  - CustomerPlacesOrders
  - customerRentMovie
  - deansTeachClass

Figure 5.6 Screenshot of Part of Entities Hierarchy and Relationships Hierarchy after the Experiment

In Figure 5.5, the ontology is blank and there are no entities or relationships, whereas Figure 5.6 shows that many entities have been added to the ontology, such as ‘song’, ‘movie’ and ‘adviser’. Many relationships have also been added to the ontology, such as ‘ClubRunsSport’, ‘CustomerRentMovie’. Furthermore, by comparing the EHKB and RHKB in the UHKB database, it can be seen that the database tables before the experiment do not include information, whereas the tables after the experiment clearly show that information has been added. Figure 5.7 presents a screenshot of the tables before the experiment and Figure 5.8 shows a screenshot of a section of the tables after the experiment.

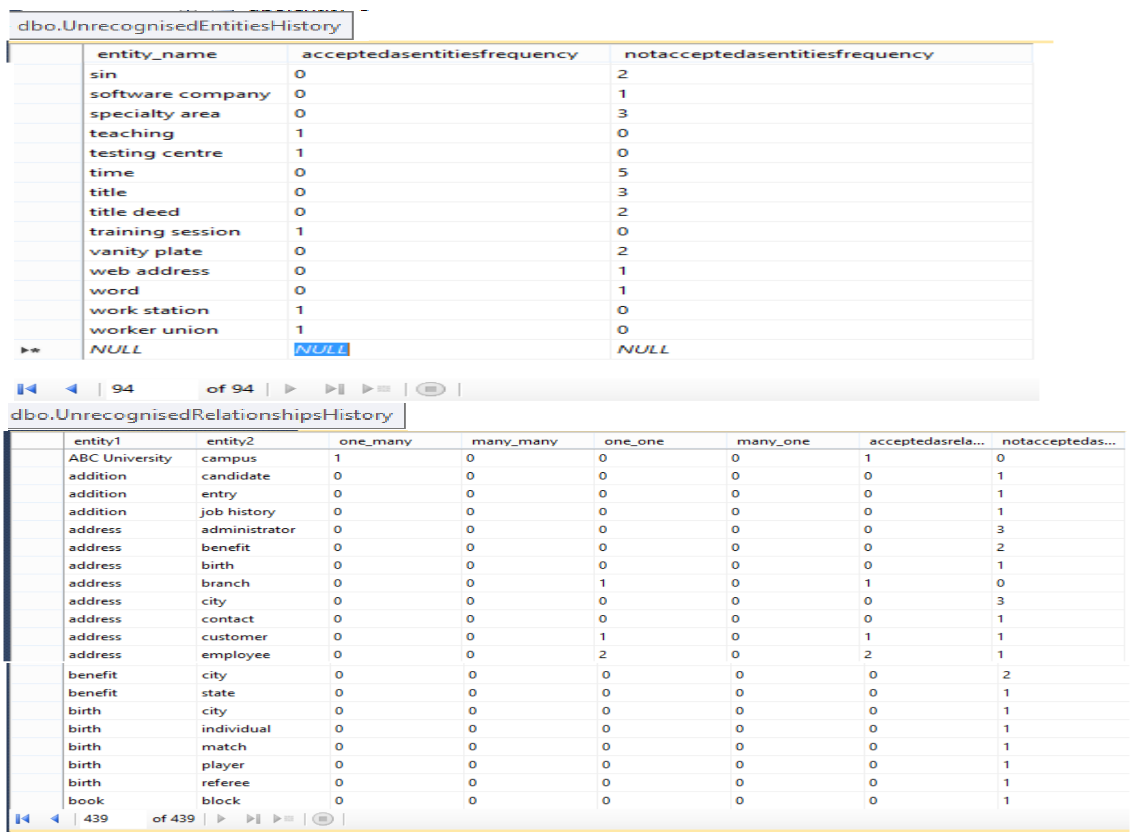

dbo.UnrecognisedEntitiesHistory

	entity_name	acceptedasentitiesfrequency	notacceptedasentitiesfrequency
▸*	NULL	NULL	NULL

dbo.UnrecognisedRelationshipsHistory

	entity1	entity2	one_many	many_many	one_one	many_one	acceptedasrelationship	notacceptedasarelationship
▸*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 5.7 Screenshot of the UHKB Database before the Experiment


dbo.UnrecognisedEntitiesHistory

	entity_name	acceptedasentitiesfrequency	notacceptedasentitiesfrequency
	sin	0	2
	software company	0	1
	specialty area	0	3
	teaching	1	0
	testing centre	1	0
	time	0	5
	title	0	3
	title deed	0	2
	training session	1	0
	vanity plate	0	2
	web address	0	1
	word	0	1
	work station	1	0
	worker union	1	0
▸*	NULL	NULL	NULL

94 of 94

dbo.UnrecognisedRelationshipsHistory

	entity1	entity2	one_many	many_many	one_one	many_one	acceptedasrela...	notacceptedas...
	ABC University	campus	1	0	0	0	1	0
	addition	candidate	0	0	0	0	0	1
	addition	entry	0	0	0	0	0	1
	addition	job history	0	0	0	0	0	1
	address	administrator	0	0	0	0	0	3
	address	benefit	0	0	0	0	0	2
	address	birth	0	0	0	0	0	1
	address	branch	0	0	1	0	1	0
	address	city	0	0	0	0	0	3
	address	contact	0	0	0	0	0	1
	address	customer	0	0	1	0	1	1
	address	employee	0	0	2	0	2	1
	benefit	city	0	0	0	0	0	2
	benefit	state	0	0	0	0	0	1
	birth	city	0	0	0	0	0	1
	birth	individual	0	0	0	0	0	1
	birth	match	0	0	0	0	0	1
	birth	player	0	0	0	0	0	1
	birth	referee	0	0	0	0	0	1
	book	block	0	0	0	0	0	1

439 of 439

Figure 5.8 Screenshot of UHKB Database Relationships Table after the Experiment

It can therefore be said that the CMO within SACMES and the UHKB components in SACMES have stored information from the natural language scenarios processed by the subjects. The extent to which the information stored by the system will be useful in improving the performance of the system in creation of conceptual models from natural language text will be discussed in Section 5.2.

5.1.2 Second Group Results

Before the subjects in the second group started, the information learnt by the system and stored in the CMO and UHKB database was deleted so that it would not affect these subjects' performance. The steps performed by the second group of subjects were the same as those followed by the first group. The only difference was that the subjects in the first group gave their handcrafted answers first, before using the system, whereas the subjects in the second group started by using the system and then gave their handcrafted answers afterwards. The reason behind requesting the second group of subjects to start with the system was that the author wished to ensure that the improvement in performance shown by the first group was not because they were doing the job of creating conceptual models for the second time, having learnt from the first time. If the performance of subjects in the second group was also improved by using the system, despite starting by using it, then it can be presumed that the subjects' improvement was purely due to their use of the system.

5.1.2.1 Entities extraction

Comparisons were made between the system answers and manual answers for entity extraction, based on entities found in the model answers for the test set. The results of these comparisons are presented in Table 5.8.

Subject 1	E1WO + H1WO		E2W + H2W	
	Recall	Precision	Recall	Precision
	50%	80%	100%	71.42%
	85.71%	100%	100%	85.71%
Total	135.71	180	200	157.13
Average	67.85%	90%	100%	78.56%
Subject 2	E1W + H1W		E2WO + H2Wo	
	Recall	Precision	Recall	Precision
	62.5%	83.33%	60%	100%
	85.71%	100%	100%	100%
Total	148.21	183.33	160	200
Average	74.10%	91.66%	80%	100%
Subject 3	E3WO + H3WO		E4W + H4W	
	Recall	Precision	Recall	Precision
	100%	66.66%	100%	62.5%
	85.71%	66%	100%	66.66%
Total	185.71	132.66	200	129.16
Average	92.85%	66.33%	100%	64.58%
Subject 4	E3W + H3W		E4WO + H4WO	

	Recall	Precision	Recall	Precision
	100%	100%	100%	83.33%
	85.71%	66.66%	100%	40%
Total	185.71	166.66	200	123.33
Average	92.85%	83.33%	100%	61.66%
Subject 5	E5WO + H5WO		E6W + H6W	
	Recall	Precision	Recall	Precision
	60%	75%	100%	100%
	80%	100%	100%	100%
Total	140	175	200	200
Average	70%	87.5%	100%	100%
Subject 6	E5W + H5W		E6WO + H6WO	
	Recall	Precision	Recall	Precision
	100%	100%	66.66%	100%
	80%	80%	100%	100%
Total	180	180	166.66	200
Average	90%	90%	83.33%	100%
Subject 7	E7WO + H7WO		E8W + H8W	
	Recall	Precision	Recall	Precision
	100%	100%	80%	100%
	62.5%	100%	100%	91.66
Total	162.5	200	180	191.66
Average	81.25%	100%	90%	95.83%
Subject 8	E7W + H7W		E8WO + H8WO	
	Recall	Precision	Recall	Precision
	60%	60%	80%	80%
	62.5%	62.5%	81.81%	81.81%
Total	122.5	122.5	161.81	161.81
Average	61.25%	61.25%	80.90 %	80.90%
Subject 9	E9WO + H9WO		E10W + H10W	
	Recall	Precision	Recall	Precision
	60%	60%	100%	100%
	72.72%	88.88%	90%	100%
Total	132.72	148.88	190	200
Average	66.36%	74.44%	95%	100%

Subject 10	E9W + H9W		E10WO + H10WO	
	Recall	Precision	Recall	Precision
	100 %	83.33%	100%	100%
	72.72%	80%	70%	87.5%
Total	172.72	163.33	170	187.5
Average	86.36%	81.66%	85%	93.75%
	Manual Answers			
	Recall		Precision	
Total	1579.11		1709.18	
Average	78.95%		85.45%	
	System Answers			
	Recall		Precision	
Total	1779.14		1693.77	
Average	88.95%		84.68%	

Table 5.8 Comparison between System Answers and Handcrafted Answers for Entities Extraction based on Key Answers

From the results obtained, it can be concluded that the novice designers' performance in entities extraction improved when they used SACMES. The recall improved from 78.95%, with 85.45% precision to 88.95%, with 84.68% precision. An average was taken to measure each subject's performance when using the system, and this was compared with their average when providing handcrafted answers. The performance of six of the ten subjects improved when they used the system, whereas for four subjects, their handcrafted performance was better than when they used the system.

5.1.2.2 Relationships extraction

Comparisons were made between system answers and manual answers for relationships extraction, based on relationships found in the model answers for the test set. The results of these comparisons are presented in Table 5.9.

Subject 1	E1WO+ H1WO		E2W+ H2W	
	Recall	Precision	Recall	Precision
	0%	0%	75%	60%
	75%	100%	100%	83.33%
Total	75	100	175	143.33
Average	37.5%	50%	87.5%	71.66%

Subject 2	E1W+ H1W		E2WO+ H2WO	
	Recall	Precision	Recall	Precision
	10%	25%	50%	100%
	87.5%	100%	100%	100%
Total	97.5	125	150	200
Average	48.75%	62.5%	75%	100%
Subject 3	E3WO+ H3WO		E4W+ H4W	
	Recall	Precision	Recall	Precision
	66.66%	66.66%	50%	33.33%
	54.54%	50%	100%	57.14%
Total	121.2	116.66	150	90.47
Average	60.6%	58.33%	75%	45.23%
Subject 4	E3W+ H3W		E4WO+ H4WO	
	Recall	Precision	Recall	Precision
	100%	100%	75%	75%
	54.54%	66.66%	100%	28.57%
Total	154.54	166.66	175	103.57
Average	77.27%	83.33%	87.5%	51.78%
Subject 5	E5WO+ H5WO		E6W+ H6W	
	Recall	Precision	Recall	Precision
	14.28%	20%	60%	60%
	60%	75%	60%	80%
Total	74.28	95	120	140
Average	37.14%	47.5%	60%	70%
Subject 6	E5W+ H5W		E6WO+ H6WO	
	Recall	Precision	Recall	Precision
	100%	85.71%	60%	75%
	60%	60%	66.66%	57.14%
Total	160	145.71	126.66	132.14
Average	80%	72.85%	63.33%	66.22%
Subject 7	E7WO+ H7WO		E8W+H8W	
	Recall	Precision	Recall	Precision
	100%	100%	75%	100%
	50%	60%	100	86.66%
Total	150	160	175	186.66

Average	75%	80%	87.5%	93.33%
Subject 8	E7W+ H7W		E8WO+ H8WO	
	Recall	Precision	Recall	Precision
	66.66%	40%	75%	75%
	33.33%	44.44%	69.23%	75%
Total	99.99	84.44	144.23	150
Average	49.99%	42.22%	72.11%	75%
Subject 9	E9WO+ H9WO		E10W+H10W	
	Recall	Precision	Recall	Precision
	16.66%	12.5%	100%	100%
	50%	35.71%	58.33%	77.77%
Total	66.66	48.21	158.33	177.77
Average	33.33%	24.10%	79.16%	88.88%
Subject 10	E9W+ H9W		E10WO+ H10WO	
	Recall	Precision	Recall	Precision
	66.66%	80%	100%	100%
	60%	33.33%	33.33%	44.44%
Total	126.66	113.33	133.33	144.44
Average	63.33%	56.66%	66.66%	72.22%
	Manual Answers			
	Recall		Precision	
Total	1216.36		1250.02	
Average	60.81%		62.50%	
	System Answers			
	Recall		Recall	
Total	1417.02		1373.37	
Average	70.85%		68.66%	

Table 5.9 Comparison between System Answers and Handcrafted Answers for Relationships Extraction based on Model Answers

From the results obtained, it can be concluded that the novice designers' performance in relationships extraction improved when they used SACMES. The recall improved from 60.81% to 70.85%, and precision improved from 62.50% to 68.66%. The average for each subject was taken to measure their performance when using the system and when providing handcrafted answers. The performance of six of the ten subjects improved when they used the system, whereas for three subjects, there was no improvement compared to their handcrafted performance. Subject number three's performance for recall when s/he used the system was better than when s/he did not use it, but in terms of precision, there was no improvement when using the system.

5.1.2.3 Cardinalities extraction

Comparisons were made between system answers and manual answers for cardinalities extraction, based on the cardinalities found in the model answers for the test set. The results of these comparisons are provided in Table 5.10.

Subject 1	E1WO+ H1WO		E2W+ H2W	
	Recall	Precision	Recall	Precision
	0%	0%	75%	60%
	71.42%	83.33%	100%	83.33%
Total	71.42	83.33	175	143.33
Average	35.71%	41.66%	87.5%	71.66%
Subject 2	E1W+ H1W		E2WO+ H2WO	
	Recall	Precision	Recall	Precision
	10%	25%	50%	100%
	83.33%	71.42%	100%	80%
Total	93.33	96.42	150	180
Average	46.66%	48.21%	75%	90%
Subject 3	E3WO+ H3WO		E4W+ H4W	
	Recall	Precision	Recall	Precision
	66%	66%	50%	33.33%
	44.44%	33.33%	100%	42.85%
Total	110.44	99.33	150	76.18
Average	55.22%	49.66%	75%	38.09%
Subject 4	E3W+ H3W		E4WO+ H4WO	
	Recall	Precision	Recall	Precision
	100%	100%	75%	75%
	54.54%	66.66%	100%	7.14%
Total	154.54	166.66	175	82.14
Average	77.27%	83.33%	87.5%	41.07%
Subject 5	E5WO+ H5WO		E6W+ H6W	

	Recall	Precision	Recall	Precision
	14.28%	20%	50%	40%
	60%	75%	66.66%	80%
Total	74.28	95	116.66	120
Average	37.14%	47.5%	58.33%	60%
Subject 6	E5W+ H5W		E6WO+H6WO	
	Recall	Precision	Recall	Precision
	100%	42.58%	0%	0%
	50%	40%	66.66%	57.14%
Total	150	82.58%	66.66	57.14
Average	75 %	41.29%	33.33%	28.57%
Subject 7	E7WO+ H7WO		E8W+ H8W	
	Recall	Precision	Recall	Precision
	100%	100%	75%	100%
	50%	60%	100%	80%
Total	150	160	175	180
Average	75%	80%	87.5%	90%
Subject 8	E7W+ H7W		E8WO+ H8WO	
	Recall	Precision	Recall	Precision
	66.66%	40%	66.66%	50%
	33.33%	44.44%	66.66%	66.66%
Total	99.99	84.44	133.32	116.66
Average	49.99%	42.22%	66.66%	58.33%
Subject 9	E9WO+ H9WO		E10W+H10W	
	Recall	Precision	Recall	Precision
	16.66%	12.5%	100%	100%
	50%	35.71%	37.5%	33.33%
Total	66.66	48.21	137.5	133.33
Average	33.33%	24.10%	68.75%	66.66%
Subject 10	E9W+ H9W		E10WO+H10WO	
	Recall	Precision	Recall	Precision
	33.33%	20%	100%	100%
	55.55%	27.77%	20%	22.22%
Total	88.88	47.77	120	122.22
Average	44.44%	23.88%	60%	61.11%

	Manual Answers	
	Recall	Precision
Total	1117.78	1044.03
Average	55.88%	52.20%
	System Answers	
	Recall	Precision
Total	1340.9	1130.71
Average	67.04%	56.53%

Table 5.10 Comparison between System Answers and Handcrafted Answers for Cardinalities Extraction based on Model Answers

From the results obtained, it can be concluded that the novice designers' performance in cardinalities extraction improved when they used SACMES. The recall improved from 55.88% to 67.04%, and precision improved from 52.20% to 56.53%. The average for each subject was taken to measure their performance when using the system and when providing handcrafted answers. The performance of seven of the ten subjects improved when they used the system, while only three subjects' handcrafted performance was better than when they used the system.

The results obtained from the second group of subjects show that novice designers' performance in extracting entities improved when they used the system. Their performance in extracting relationships and cardinalities of relationships also improved when they used the system. This result supports the hypothesis that the performance of novice designers will improve when they use SACMES comparing to their manual performance.

5.2 Experimental Design Two

In this section, the author attempts to provide evidence that the knowledge and information stored by SACMES helps to improve the performance of the system and minimise human intervention. In order to provide evidence of this, it was necessary to train the system to learn and then measure the performance of the system after it had learnt. To train a system to learn, a training set must be developed. For this purpose, a collection of fifty case studies, taken from authentic resources such as database textbooks and PhD theses, which could be used as a training set. Appendix 4 shows the case studies in the collection. The training set was divided into ten groups, each with five case studies. Another collection of five case studies was used as test set. Two of these case studies were found with their answers, while model answers for the other three were provided by a human expert[21]. Appendix 5 demonstrates the test set with the model answers. The case studies used in the test set were different from those used in the training set. Before starting this experiment, the author considered the subjects who would participate in the experiment. The initial intention was to find fifty students as subjects to train the system, but students are busy with their studies and the majority of them were not interested in participating in the experiment. As an alternative it was decided that the author, who has some experience in the creation of conceptual models, having studied this during his undergraduate course, would be eligible to participate in the study. The author therefore played the role of designer and performed the tasks required to train the system.

The system does need human intervention to complete the process of extracting conceptual models from natural language text. The system identifies entities, but then becomes unable to decide whether some nouns are entities or not. Therefore, the system requests user intervention as shown in Figure 4.1. The system also identifies relationships but again, is sometimes unable to define relationships between entities. Thus, the system again requires human intervention, as demonstrated in Figure 4.1. Previous behaviours make the outputs of the system differ from one user to another, and as a result, each system output depends on the user. In this experiment, however, it was important for the output of the system to rely on the knowledge stored in SACMES. Therefore, the author needed to use two different versions of SACMES. The first version, depicted in Figure 4.1, was used by the author to train the system. The second version differs from the first in that it does not require human intervention and does not store the outputs of the system in the CMO and UHKB database, as shown in Figure 5.9.

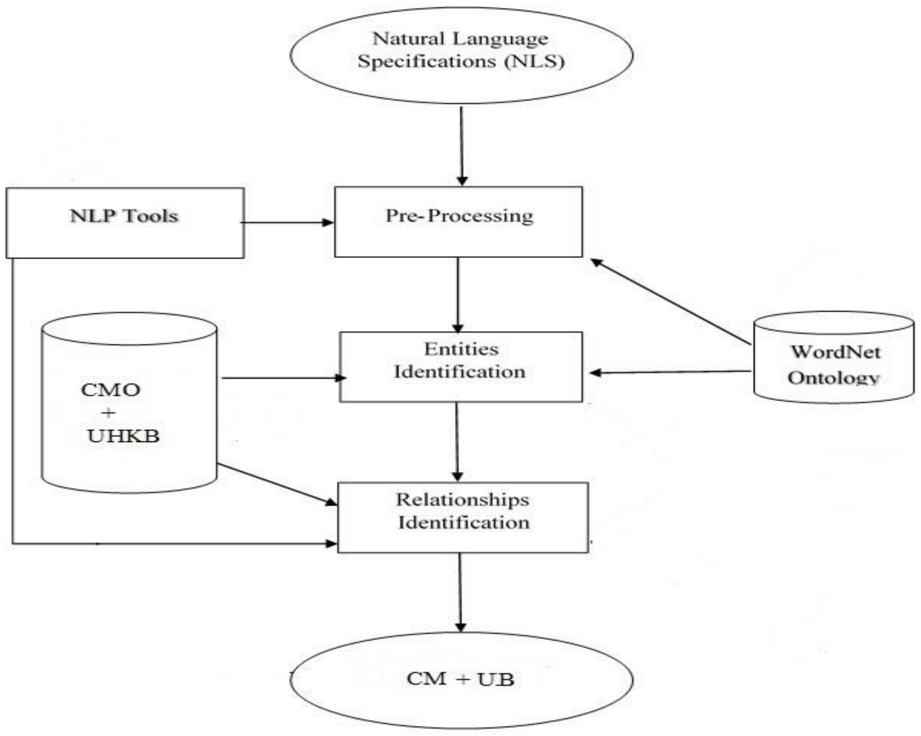


Figure 5.9 System Architecture for KBCMES

In the architecture illustrated in Figure 4.1, in addition to knowledge found in the CMO and UHKB, the system requests human intervention to define entities and relationships that it is unable to define. In contrast, the system shown in Figure 5.9 defines entities and relationships based on knowledge in the CMO and UHKB database, and does not use human intervention. Consequently, the results of this system will be dependent on its knowledge, rather than on the user of the system. Furthermore, in the version shown in Figure 4.1, the outputs of the system are stored in the CMO and UHKB, whereas in that shown in Figure 5.9, the outputs are not stored in the CMO and UHKB. This is so the author can ensure no information is added into the system apart from that which is added during each training stage. This version of the system is called Knowledge-Based Conceptual Model Extraction System (KBCMES). The author used KBCMES to extract conceptual models for the test set after finishing each of the training stages.

Before SACMES was trained on the training set, KBCMES was used to obtain conceptual models for the test set. The recall and precision for each case study within the test set were recorded. Next, SACMES was used to train the CMO and UHKB by using group number one of the training set, which includes five case studies. Fresh copies of the CMO and RHKB were then used and trained on ten case studies from the training set. Further copies of the CMO and RHKB were used and trained on fifteen case studies from the training set. This process was repeated for ten copies of the CMO and RHKB. At the end of the training, the author had obtained ten copies of the CMO and RHKB, the first copy trained on five case studies from the training set, the second copy trained on ten case studies, the third copy trained on fifteen case studies and the tenth copy trained on fifty case studies. It was noted that as the number of case studies on which the system was trained increased, the information in the CMO and UHKB database also increased.

KBCMES was integrated with copy number one of the CMO and RHKB to obtain conceptual models for the test set, and the recall and precision for each case study were recorded. KBCMES was then integrated with copy number two to obtain conceptual models for the test set and again, the recall and precision for each case study were recorded. This process was repeated with each copy of the CMO and RHKB, from copy number one to number ten.

5.2.1 Results

Table 5.11 represents the results obtained by using KBCMES to extract conceptual models for case studies within the test set before the training, when there was no information in the CMO or UHKB database.

Unrecognised Entities						
Case study name	Unrecognised entities count	Correct answers	Incorrect answers	Missed	Recall	Precision
VedMed Hospital	13	0	0	13	0%	0%
DreamHome	14	0	0	14	0%	0%
Airline	18	0	0	18	0%	0%
Florida Mall	9	0	0	9	0%	0%
Coca Cola	14	0	0	14	0%	0%
Total	68	0	0	68	0	0
Average					0%	0%
Unrecognised Relationships						
Case study name	Unrecognised relationships count	Correct answers	Incorrect answers	Missed	Recall	Precision
VEDMED	61	0	0	61	0%	0%
DreamHome	121	0	0	121	0%	0%
Airline	58	0	0	58	0%	0%

Florida Mall	70	0	0	70	0%	0%
Coca Cola	111	0	0	111	0%	0%
Total	421	0	0	421	0	0
Average					0%	0%
Entities						
Case study name	Entities count	Correct answers	Incorrect answers	Missed	Recall	Precision
VEDMED	4	0	0	4	0%	0%
DreamHome	8	0	0	8	0%	0%
Airline	7	2	0	5	28.57%	100%
Florida Mall	7	0	0	7	0%	0%
Coca Cola	9	0	0	9	0%	0%
Total	35	2	0	33	0.2857	100
Average					5.71%	20%
Relationships						
Case study name	Relationships count	Correct answers	Incorrect answers	Missed	Recall	Precision
VEDMED	3	0	0	3	0%	0%
DreamHome	10	0	0	10	0%	0%
Airline	7	0	0	7	0%	0%
Florida Mall	9	0	0	9	0%	0%
Coca Cola	12	0	0	12	0%	0%
Total	41	0	0	41	0	0
Average					0%	0%
Cardinalities						
Case study name	Entities count	Correct answers	Incorrect answers	Missed	Recall	Precision
VEDMED	3	0	1	3	0 %	0 %
DreamHome	10	0	0	10	0%	0%
Airline	7	0	0	7	0%	0%
Florida Mall	6	0	0	6	0%	0%
Coca Cola	12	0	0	12	0%	0%
Total	38	0	1	38	0	0
Average					0%	0%

Table 5.11 Summary of Results Obtained for Test Set from KBCMES before Training

Table 5.11 is divided into five subsections, namely, unrecognised entities, unrecognised relationships, entities, relationships and cardinalities. In the unrecognised entities section, the average for recall and precision is zero. The average for recall and precision is also zero for unrecognised relationships. The results obtained for entities extraction from the test set by KBCMES is 5.7% for recall and 20% for precision, whereas the results obtained for relationships extraction and cardinalities of relationships is zero percent for both recall and precision. Table 5.11 also shows that in the unrecognised entities section, the unrecognised entities count should be thirteen for the VedMed case study. However, the correct answers count for unrecognised entities is zero and the number of missed answers is thirteen, which means that the system failed to retrieve any correct or incorrect answers related to unrecognised entities for the VedMed case study. This indicates that the system needs human intervention to obtain a correct answer for each entity in the unrecognised entities list. In the unrecognised relationships section, the unrecognised relationships count should be sixty-one for the VedMed case study. Correct answers for unrecognised relationships for this case study are equal to zero and missed answers are equal to sixty-one, which means that the system failed to retrieve any correct or incorrect answers related to unrecognised relationships for this case study. In the entities section, the entities count should be four for the VedMed case study. The number of correctly extracted entities for the case study is zero and there are four missed answers, which means that the system failed to retrieve any correct answers or incorrect answers related to entities for the VedMed case study. In the relationships section, the relationships count should be three for the VedMed case study. Correctly extracted relationships are equal to zero and missed answers are equal to three, which means that the system failed to retrieve any correct or incorrect answers related to relationships for the case study. In the cardinalities section, the cardinalities of relationships count should be three for the VedMed case study. The number of correctly extracted answers for the case study is zero and missed answers are equal to three, which means the system failed to retrieve any correct or incorrect answers related to cardinalities of relationships for the VedMed case study.

Table 5.12 represents a results summary obtained from using KBCMES integrated with a CMO and UHKB database trained on fifty case studies.

Result Summary after Training 50 Case studies on the System						
Unrecognised Entities						
Case study name	Unrecognised entities count	Correct Answers	Incorrect Answers	MISSED	Recall	Precision
VedMed Hospital	13	8	0	5	61.53	100
DreamHome	13	4	0	9	30.76	100
Airline	18	4	0	14	22.22	100
Florida Mall	9	2	0	7	22.22	100
Coca Cola	14	5	0	9	35.71	100
Total	67	23	0	44	172.44	500
Average					35.47	100
Unrecognised Relationships						
Case study name	Unrecognised relationships count	Correct Answers	Incorrect Answers	MISSED	Recall	Precision
VEDMED	61	15	1	45	25	93.75
DreamHome	127	22	2	103	17.6	91.66
Airline	58	14	1	43	24.56	93.33
Florida Mall	70	9	3	58	13.43	75
Coca Cola	111	33	3	75	30.55	91.66
Total	427	93	10	324	111.14	445.4
Average					22.22	89.08
Entities						
Case study name	Entities Count	Correct Answers	Incorrect Answers	MISSED	Recall	Precision
VEDMED	4	1	1	3	25	50
DreamHome	8	1	2	7	12.5	33.33
Airline	7	3	1	4	42.85	75
Florida Mall	7	3	2	4	42.85	60
Coca Cola	9	7	0	2	77.77	100
Total	35	15	6	20	200.97	318.33
Average					40.19	63.66

Relationships						
Case study name	Relationships count	Correct Answers	Incorrect Answers	MISSED	Recall	Precision
VEDMED	3	0	1	3	0	0
DreamHome	10	0	2	10	0	0
Airline	7	0	2	7	0	0
Florida Mall	9	1	3	8	11.11	25
Coca Cola	12	5	1	7	41.66	83.33
Total	41	6	9	35	52.77	108.33
Average					10.55	21.66
Cardinality						
Case study name	Entities Count	Correct Answers	Incorrect Answers	MISSED	Recall	Precision
VEDMED	3	0	1	3	0	0
DreamHome	10	0	2	10	0	0
Airline	7	0	1	7	0	0
Florida Mall	6	1	3	5	16.66	25
Coca Cola	12	4	2	7	36.36	66.66
Total	38	5	9	32	53.02	91.66
Average					10.60	18.33

Table 5.12 Summary of Results Obtained for Test Set from KBCMES after Training

As Table 5.12 illustrates, in the unrecognised entities section, the average result for recall is 35.47% and for precision is 100%. For unrecognised relationships, the average for recall is 22.22% and for precision is 89.08%. The results obtained for extraction of entities from the test set by KBCMES are 40.19% for recall and 63.66% for precision, while the results obtained for relationships extraction from the test set by KBCMES are 10.55% for recall and 21.66% for precision. The results obtained for extraction of cardinalities of relationships from the test set by KBCMES are 10.60% for recall and 18.33% for precision.

Table 5.12 also shows that in the unrecognised entities section, where the unrecognised entities count should be thirteen for the VedMed case study, the correct answers count for unrecognised entities is eight and the missed answer count is five, which means that the system has successfully extracted eight out of thirteen answers for the VedMed case study. This indicates that the system’s performance has improved and its need for human intervention has reduced in comparison with the result for the same section in Table 5.11. In the unrecognised relationships section, the unrecognised relationships count is sixty-one for the VedMed case study. The number of correct answers for extraction of unrecognised relationships from the case study is fifteen, the number of missed answers is forty-five and the incorrect answers count is one. The system has therefore been successful in retrieving fifteen out of sixty-one answers, in addition to one incorrect answer. This demonstrates that the system’s performance has improved and the need for human intervention has reduced in comparison to the result for the same section in Table 5.11, before the training was completed. In the entities section, the entities count is four for the VedMed case study. The number of entities correctly extracted from the case study is one and there are three missed answers. Therefore, the system successfully retrieved one correct answer, though there was also one incorrect answer related to entities for the VedMed case study. This result again shows some improvement in the system’s performance compared to the result found in Table 5.11. In the relationships and cardinalities sections, however, there is no improvement in the results compared to those found in Table 5.11. The only difference is that the system retrieved one incorrect relationship and one incorrect cardinality.

Table 5.13 contains the averages for recall and precision obtained before and after the training for each case study within the test set.

Case study name	Criterion	Before training		After training	
		Recall	Precision	Recall	Precision
VEDMED Hospital	Unrecognised entities extraction	0%	0%	61.53%	100%
	Unrecognised relationships	0%	0%	25%	93.75%
	Entities extraction	0%	0%	25%	50%
	Relationships extraction	0%	0%	0%	0%
	Cardinalities of relationships extraction	0%	0%	0%	0%
DreamHome	Unrecognised entities extraction	0%	0%	30.76%	100%
	Unrecognised relationships	0%	0%	17.6%	91.66%

Case study name	Criterion	Before training		After training	
		Recall	Precision	Recall	Precision
	Entities extraction	0%	0%	12.5%	33.33%
	Relationships extraction	0%	0%	0%	0%
	Cardinalities of relationships extraction	0%	0%	0%	0%
Airline	Unrecognised entities extraction	0%	0%	22.22%	100%
	Unrecognised relationships	0%	0%	24.56%	93.33%
	Entities extraction	28.57%	100%	42.85%	75%
	Relationships extraction	0%	0%	0%	0%
	Cardinalities of relationships extraction	0%	0%	0%	0%
Florida Mall	Unrecognised entities extraction	0%	0%	22.22%	100%
	Unrecognised relationships	0%	0%	13.43%	75%
	Entities extraction	0%	0%	42.85%	60%
	Relationships extraction	0%	0%	11.11%	25%
	Cardinalities of relationships extraction	0%	0%	16.66%	25%
Coca Cola	Unrecognised entities extraction	0%	0%	35.71%	100%
	Unrecognised relationships	0%	0%	30.55%	91.66%
	Entities extraction	0%	0%	77.77%	100%
	Relationships extraction	0%	0%	41.66%	83.33%
	Cardinalities of relationships extraction	0%	0%	36.36%	66.66%

Table 5.13 Comparison of Results for Extraction of Unrecognised Entities, Unrecognised Relationships, Entities, Relationships and Cardinalities from Test Set by KBCMES before and after Training

From the results displayed in Table 5.13, it is clear that extraction improved after the training. This result supports the hypothesis that the information stored by SACMES helps to improve the performance of the system and assists in minimising the level of human intervention.

Table 5.14 shows the relationship between unrecognised entities extraction and the count of case studies on which the system is trained.

Result	Count of case studies used in training										
	0	5	10	15	20	25	30	35	40	45	50
Recall	0%	7.66%	8.83%	23.61%	25.15%	25.15%	26.57%	26.57%	28.11%	28.11%	34.48%
Precision	0%	100%	100%	100%	100 %	100%	100%	100%	100%	100%	100%

Table 5.14 Relationship between Unrecognised Entities Extraction and Count of Case Studies on which System is Trained

Table 5.14 demonstrates a proportional relationship between the count of cases on which the system is trained and the extraction result. As the count of training cases increases, the accuracy is increased. When the count of the cases on which the system was trained was equal to zero, the average recall and precision were equal to zero as well. When the system was trained using the training set, the average started to increase. Recall represents the average extent to which the system answers match model answers produced by the system analyser. Precision represents the average recall accuracy. When the system had been trained on five case studies, the recall improved from zero percent to 7.66% and the accuracy for that percentage was 100%. When the system had been trained on twenty-five case studies, the recall reached 25.15% and the accuracy for that percentage was 100 %. When the system had been trained on fifty case studies, the recall reached 34.48% and the accuracy for that percentage was 100%. The precision was 100% at all times, meaning that all the answers extracted by the system were correct. However, this does not mean that all the required answers were extracted by the system, as it missed some of them. For example, if a model answer has thirty answers, but the system succeeds in extracting only five correct answers out of the thirty, then the precision for these five will be 100% even though some of the answers were not extracted by the system. However, when all the answers are extracted correctly by the system, the recall will be 100%. Figure 5.10 represents the relationship between the count of case studies on which the system is trained and the average recall and precision in defining unrecognised entities. From Figure 5.10 it can be seen how the recall increases as the system is trained on more case studies.

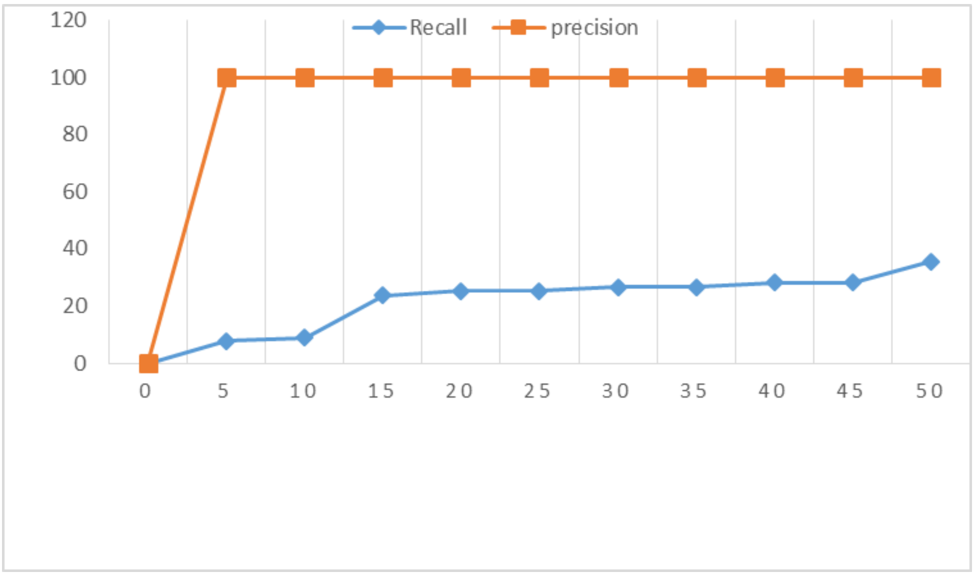


Figure 5.10 Relationship between Count of Case Studies on which the System is Trained and Average Recall and Precision in Defining Unrecognised Entities

Table 5.15 shows the relationship between unrecognised relationships extraction and the count of case studies on which the system is trained.

Result	Count of case studies used in training										
	0	5	10	15	20	25	30	35	40	45	50
Recall	0%	1.31%	5.91%	10.03%	11.17%	13.22%	17.00%	17.47%	18.79%	21.21%	22.22%
Precision	0%	50%	72.47%	78.81%	83.38%	85.01%	84.14%	85.63%	86.72%	88.54%	89.08%

Table 5.15 Relationship between Unrecognised Relationships Extraction and Count of Case Studies on which System is Trained

From Table 5.15 it can be seen that there is also a proportional relationship between the count of cases on which the system has been trained and the results for extraction of unrecognised relationships. As the count of cases used in training increases, the accuracy is also increased. When the count of cases on which the system was trained was zero, the average for recall and precision was zero as well. When the system had been trained using the training set, the average started to increase. When the system had been trained on five case studies, the recall improved from zero percent to 1.31% and the accuracy for that recall was 50%. When the system had been trained on twenty-five case studies, the recall reached 13.22% and the accuracy for that percentage was 85.01%. When the system had been trained on fifty case studies, the recall reached 22.22% and the accuracy for that percentage was 89.08%. Figure 5.11 reflects the information found in Table 5.15, representing the relationship between the count of case studies on which the system is trained and the average recall and precision in defining unrecognised relationships. From Figure 5.11 it can be seen that the rate of recall rises as the system is trained on more case studies.

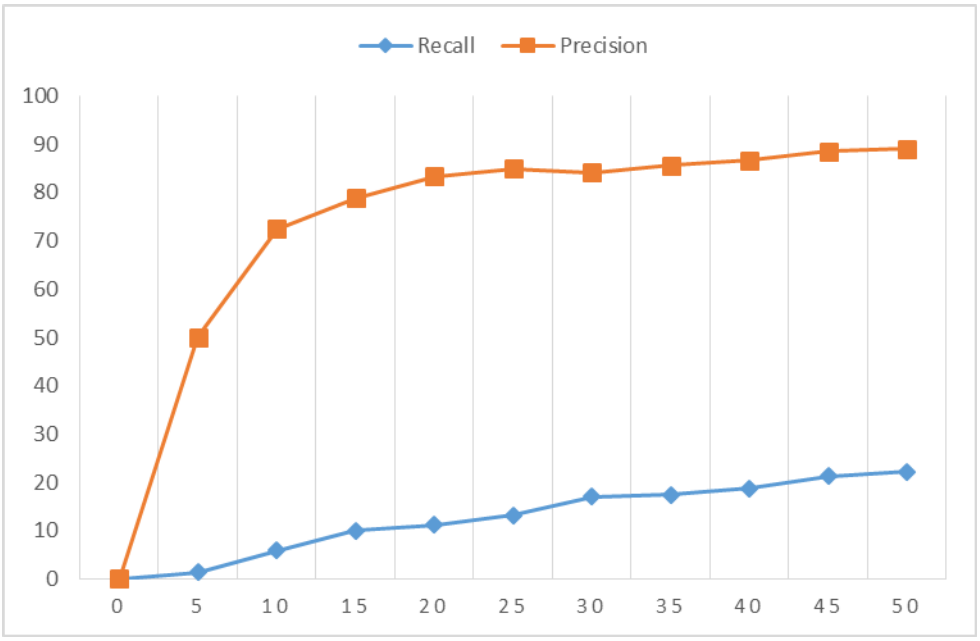


Figure 5.11 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Unrecognised Relationships

Table 5.16 shows the relationship between entities extraction and the count of case studies on which the system is trained.

Result	Count of case studies used in training										
	0	5	10	15	20	25	30	35	40	45	50
Recall	5.71%	15.87%	28.45%	33.52%	33.52%	31.30%	42.97%	37.97%	37.97%	40.19%	40.19%
Precision	100%	50%	61.66%	60.33%	63.66%	63.66%	63.66%	63.66%	63.66%	63.66%	63.66%

Table 5.16 Relationship between Entities Extraction and Count of Case Studies on which System is Trained

From Table 5.16 it can be seen that there is a proportional relationship between the count of cases on which the system is trained and the results extraction of entities. When the count of training cases increases, the accuracy is also increased. Although the recall sometimes decreases as the count of case studies on which the system is trained increases, it begins to increase again when the count of cases increases further. When the count of cases on which the system had been trained was zero, the average recall was 5.71% with 100% precision for that recall, but when the system had been trained using the training set, the average started to increase. When the system had been trained on five case studies, the recall improved from 5.71% to 15.87% and the accuracy for this recall was 50%. When the system had been trained on twenty-five case studies, the recall reached 31.30% and the accuracy for that percentage was 63.66%. When the system had been trained on fifty case studies, the recall reached 40.19% and the accuracy for that percentage was 63.66%.

It is interesting to note that the average for entities extraction decreased from 33.52% when the system had been trained on twenty case studies, to 31.30% when it had been trained on twenty-five case studies, as shown in Table 5.16. However, the author is not concerned about this decrease because the averages for entities extraction started to increase again when the system was trained on further case studies. In addition, there was no decrease in other areas, such as the averages for unrecognised entities extraction, relationships extraction and cardinality extraction. The average for unrecognised entities extraction was 25.15% both when the system had been trained on twenty case studies and when it had been trained on twenty-five case studies, as shown in Table 5.14. The average for relationships was 5.55% when the system had been trained on twenty case studies and on twenty-five case studies, as demonstrated in Table 5.17. Similarly, the average for cardinality was 6.66% when the system had been trained on twenty case studies and on twenty-five case studies, as demonstrated in Table 5.18. Moreover, there was improvement elsewhere, such as in the average for unrecognised relationships extraction, which increased from 11.17% when the system had been trained on twenty case studies to 13.22% when it had been trained on twenty-five case studies, as shown in Table 5.15.

A similar situation occurred later, when the average for entities extraction decreased from 42.97% after the system had been trained on thirty case studies to 37.97% when it had been trained on thirty-five case studies. Again, the author is not worried about this decrease because there was no decrease in the averages for other areas, such as for unrecognised entities extraction as shown in Table 5.14, for relationships as shown in Table 5.17 or for cardinality of relationships as shown in Table 5.18. Moreover, there was improvement in the area of unrecognised relationships extraction, as shown in Table 5.15. Therefore, the author expected the average for entities extraction to increase again, as had happened when the case study count used in training increased from twenty-five to thirty, shown in Table 5.16.

Figure 5.12 reflects the information found in Table 5.16. The figure represents the relationship between the count of case studies on which the system is trained and the average recall and precision in defining entities. It can be seen from the figure that the rate of recall increases as the system is trained on more case studies.

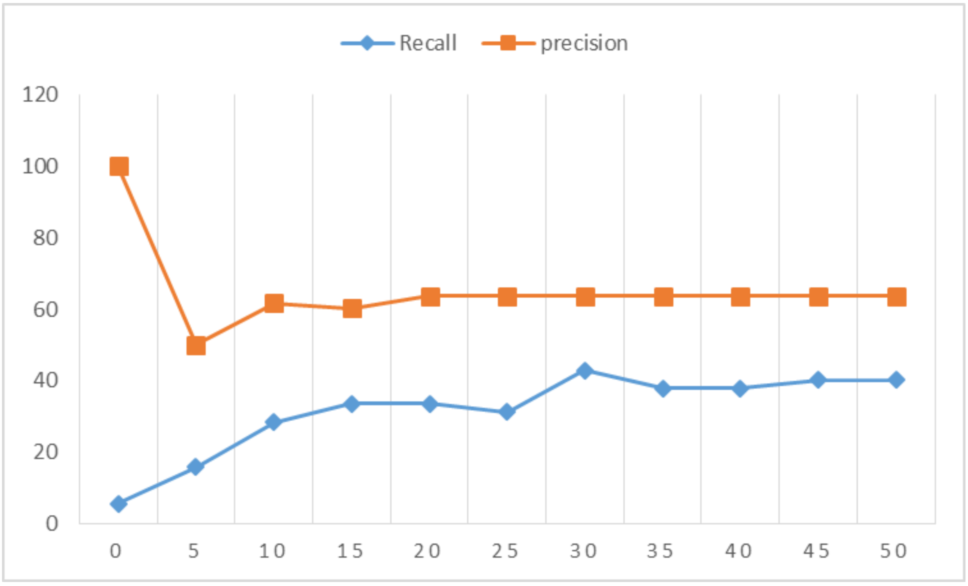


Figure 5.12 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Entities

Table 5.17 shows the relationship between relationship extraction and the count of case studies on which the system is trained.

	Count of case studies used in training										
	0	5	10	15	20	25	30	35	40	45	50
Recall	0%	3.88%	5.55%	7.22%	5.55%	5.55%	8.88%	8.88%	8.88%	10.55%	10.55%
Precision	0%	26.66%	26.66%	20%	18.33%	25%	21%	21%	21%	21.66%	21.66%

Table 5.17 Relationship between Relationships Extraction and Count of Case Studies on which System is Trained

From Table 5.17 it can be seen that there is a proportional relationship between the count of cases on which the system is trained and the extraction of relationships. When the count of cases increases, the accuracy is also increased. Although the recall is sometimes decreased by increasing the count of case studies on which the system is trained, it begins to increase again as the count of training cases is increased further. When the count of cases on which the system had been trained was zero, the average recall and precision was zero as well, but when the system had been trained using the training set, the average started to increase. When the system had been trained on five case studies, the recall improved from zero percent to 3.88%

and the accuracy for that recall was 26.66%. When the system had been trained on twenty-five case studies, the recall reached 5.55% and the accuracy for that percentage was 25%. When the system had been trained on fifty case studies, the recall reached 10.55% and the accuracy for that percentage was 21.66%.

While training the system on between twenty and thirty case studies, there was fluctuation in the average for relationships extraction. The average decreased to 5.22% when the system had been trained on twenty and twenty-five case studies, but then increased to 8.88% when the system had been trained on thirty case studies. The author is not concerned about this fluctuation because although the average decreased, it then increased as the system was trained on further case studies. Furthermore, although there was fluctuation in the average for relationships extraction, at the same time there were increases in other areas, such as the average for unrecognised entities extraction, as shown in Table 5.14, the average for unrecognised relationships extraction, as shown in Table 5.15, the average for entities extraction, as shown in Table 5.16, and the average for cardinalities extraction, as shown in Table 5.18.

Figure 5.13 reflects the information found in Table 5.17. It represents the relationship between the count of case studies on which the system has been trained and the average recall and precision in defining relationships. From the figure, it can be seen that the recall increases as the system is trained on more case studies.

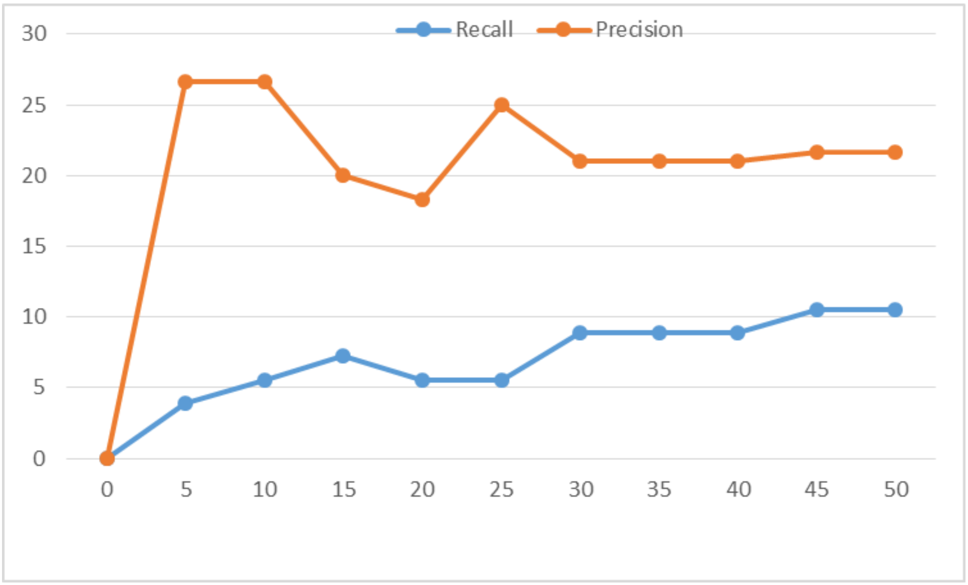


Figure 5.13 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Relationships

Table 5.18 shows the relationship between cardinalities of relationships extraction and the count of case studies on which the system has been trained.

	Count of case studies used in training										
	0	5	10	15	20	25	30	35	40	45	50
Recall	0%	4.99%	6.66%	8.33%	6.66%	6.66%	8.78%	8.78%	8.78%	10.60%	10.60%
Precision	0%	26.66%	26.66%	20%	18.33%	25%	17%	17%	17%	18.33%	18.33%

Table 5.18 Relationship between Cardinalities of Relationships Extraction and Count of Case Studies on which System is Trained

From Table 5.18 it can be seen that there is a proportional relationship between the count of cases on which the system has been trained and the results for extraction of cardinalities of relationships. In general, as the count of training cases increases, the accuracy is also increased. Sometimes the recall is decreased by increasing the count of case studies used in training the system, while at other times it remains constant. However, the recall begins increasing again as the count of cases used in training the system increases further. When the count of cases on which the system was trained was zero, the average recall and precision was zero as well, but when the system had been trained using the training set, the average started to increase. When the system had been trained on five case studies, the recall improved from zero percent to 4.99% and the accuracy for that recall was 26.66%. When the system had been trained on twenty-five case studies, the recall reached 6.66% and the accuracy for that percentage was 25%. When the system had been trained on fifty case studies, the recall reached 10.60% and the accuracy for that percentage was 18.33%.

After training the system on fifteen case studies, the average for cardinality extraction increased to 8.33%, but after training it on twenty and twenty-five case studies, this average decreased to 6.66%. However, the average increased again to 8.78% when the system had been trained on thirty case studies. The decrease is therefore not a cause for concern because it was followed by an increase when the system was trained on further case studies. Furthermore, at the same time, there were increases in other areas, such as in the averages for unrecognised entities extraction and unrecognised relationships extraction. In general, even when there is a decrease in the average for a particular area, there are improved or constant averages in other areas. Therefore, there is improvement in the system's overall performance as it is trained on further case studies.

Figure 5.14 reflects the information found in Table 5.18. It represents the relationship between the count of case studies on which the system has been trained and the average recall and precision in defining cardinalities of relationships. From Figure 5.15 it can be seen that the rate of recall rises as the system is trained on more case studies. In other words, the system's performance improves as the count of cases processed by the system increases.

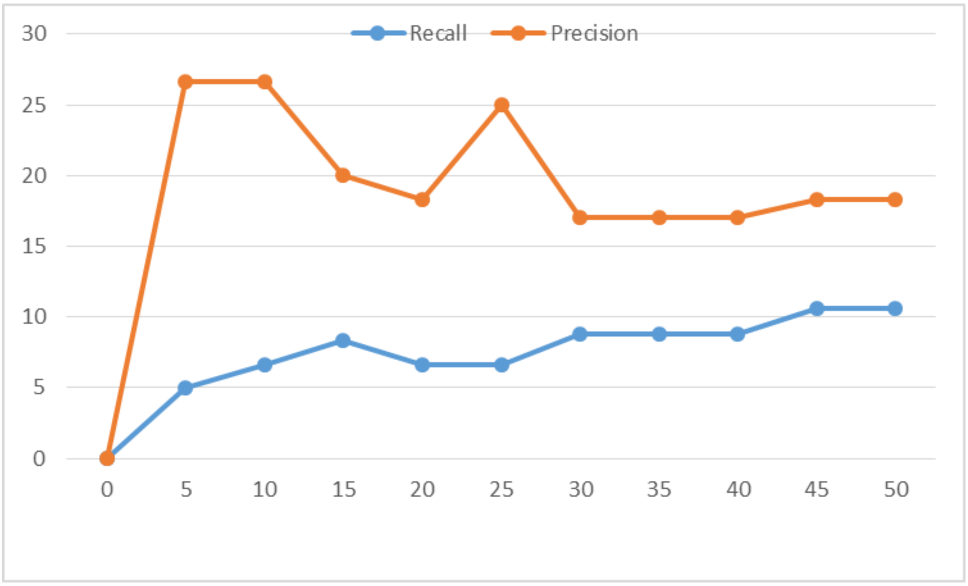


Figure 5.14 Relationship between Count of Case Studies on which System is Trained and Average Recall and Precision in Defining Cardinalities of Relationships

To summarise, results were obtained for extraction of unrecognised entities, unrecognised relationships, entities, relationships and cardinalities of relationships in the following situations: (1) before the training; (2) during training by increasing the count of case studies on which the system was trained by five case studies each time; and (3) when the training was complete (the system had been trained on all case studies in the training set). These results show that the system learns from the natural language specifications processed by users, and uses the knowledge stored from these specifications to improve the extraction of ERDs from specifications that will be processed in the future. As a result, the system's performance is enhanced. Sometimes the system's performance decreased as the number of specifications processed by the system increased, and at other times the rate of improvement stalled, but then the system's performance began to improve again as the count of specifications processed by the system increased further. This is positive proof that the information stored by SACMES in the CMO and UHKB, can help the system to improve its performance, minimise the need for human intervention and enable it to produce more relevant information to advise users in the creation of conceptual models. Although there was improvement in the performance of the system, this improvement is not very high. This is because the training set is not very large. Furthermore, the training set contains many domains that are completely different from the domains included in the test set. Despite there being no systematic relationship between the domains included in the training set and test set, system performance still improved. This is a positive point, however the specification that will be processed by the system might not already be processed by the system in a systematic approach. In such a situation, in order for this improvement to reach high accuracy, the system needs to be processed on very large set of natural language specifications. However, such improvements can reach good accuracy for a smaller set of specifications if the system is trained on a domain and test set, which will be in the same domain.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

Conceptual model creation is an important stage in the development of a system. The conceptual model shows the main actors in the system and the relationships between them. In other words, by looking at the conceptual model, the system can be understood. In order to be a good conceptual model, it must have the ability to reflect the real world environment. Furthermore, errors in conceptual models must be corrected at an early stage, as it is costly to make such corrections in the advanced stages of a system's development (Boehm, 1981).

Conceptual models can be created by analysing the requirement specifications for a problem, which are generally written using natural language. There are about eighty notations that can describe the requirement specifications of a system, among which the ERD and UML are the most commonly used in practice (Neill & Laplante, 2003). The ERD, suggested by Chen in 1976, is extensively used to define conceptual models for database design because it is easy to understand and a powerful means of modelling natural language specifications (Chen, 1976). Despite its significance, however, designing a conceptual model can be very problematic (Thonggoom, 2011), as the process can face many difficulties, as identified below.

1. The complexity of relationships between the concepts of a conceptual model can be very difficult for both novice and expert designers to identify.

2. Natural language rules for conceptual model extraction are incomplete and overlapped, which means there is no reliable set of linguistic rules that can be used to transfer natural language specifications into a conceptual model.

3. Semantic relationships in natural language text can be complex. This means that not every relationship mentioned in the requirement specification needs to be mapped into a relationship in the conceptual model while, conversely, some relationships that are not mentioned in the requirements do need to be included.

4. Lack of domain knowledge and experience can cause difficulties in the creation of conceptual models, particularly for novice designers.

5. There can be different solutions to the same problem, because conceptual models reflect the designer's viewpoint and this may differ from one designer to another. The fact that two points of view might be correct makes it very difficult to define one optimal solution for a problem.

6. Natural language, which is widely used for writing requirement specifications in industry, contains inherent problems, such as noise, silence, overspecification, contradiction, forward reference, wishful thinking and ambiguity.

Due to the problems faced by designers, particularly novice designers, in the development of conceptual models, technologies have become involved in conceptual model creation, as well as in mapping from conceptual models to logical or physical models. There are many commercial graphical CASE tools that can be used to automatically map a conceptual model into a logical or a physical model (Thonggoom, 2011). However, there is no tool, commercial or otherwise, which can automatically map natural language text into a conceptual model. As a substitute, various semi-automated approaches are used for this purpose.

The purpose of this thesis is to improve the creation of conceptual models by producing a tool to assist designers in this process. In order to achieve this goal, the author set five objectives as follows:

1. To explore and analyse the approaches that are currently used for extracting conceptual models from natural language text, to examine their strengths and weaknesses, and to identify features that could be integrated in a new tool.

To achieve the above objective, a review of approaches used in mapping natural language into conceptual models was conducted and the findings of the review are summarised here. Firstly, there is significant need for a tool to help designers create conceptual models that contain fewer errors than those created manually. Although researchers have made good progress in mapping natural language text into conceptual models, no fully automated tool can yet achieve this. Therefore, a minimum level of human intervention needs to be included in the process. The review also demonstrates that the majority of systems used to map natural language into conceptual models include linguistic rules and natural language techniques to facilitate the mapping. Furthermore, the majority of tools used for this process rely on reuse of previous designs that have been stored in a knowledge base to be used by tools. The literature further reveals that the range of approaches used to map natural language into conceptual models includes linguistics-based, knowledge-based and multiple approaches (the latter type integrates more than one approach). The linguistic approach is domain independent but does not include a knowledge base. The knowledge-based approach can use different methods, such as pattern-based, case-based and ontology-based techniques, to obtain knowledge that can support designers in mapping natural language text into conceptual models. While the use of ontologies is a common technique for capturing knowledge, several researchers have integrated more than one approach to improve performance. However, there is no domain-independent knowledge base that can be used to support designers in the creation of conceptual models, because creating such knowledge is difficult and time consuming. The author has therefore worked to fill this gap.

Accordingly, in this research, the author has built a model that can learn from the natural language texts that it processes, and which uses the learnt information to update its knowledge base and improve its performance. Achieving this aim would require the integration of natural language processing, ontology, linguistic rules and human intervention within a model. The author reviewed NLP tools and selected Stanford CoreNLP[6] for incorporation in the proposed model. Existing ontology types were also reviewed, and a lightweight ontology was selected to enable the storage of domain-independent knowledge about entities and relationships within the model. Next, methods used for the creation of an ontology were reviewed, and a semi-automated method that would use linguistic techniques to train the ontology was selected. From a review of existing ontologies, WordNet was selected, and from existing ontology languages, OWL was adopted for use in formalising the ontology that would be incorporated in the proposed model.

2. The second objective was to examine the linguistic rules that are used in mapping natural language requirements into conceptual models, to identify their strengths and weaknesses, and to determine which rules would be suitable for use.

To achieve this, the author conducted a review of the relevant linguistic rules, which produced two main findings. The first was that the rules for mapping natural language into conceptual models are not complete. For example, entities can be represented by nouns, but not every noun in a problem description will be mapped into an entity. The second finding was that the rules are overlapped. For example, nouns represent entities but can also represent attributes. At this stage, the rules to be used in the proposed model were selected. For extraction of entities, a WordNet ontology was chosen, in addition to human intervention by applying domain-independent rules, such as the domain-importance and Multi-attributes rules to define entities. For extraction of relationships, Stanford typed dependencies were selected in combination with human application of the need-to-know rule.

3. To design a semi-automated, domain-independent methodology that attempts to tackle the limitations of current methodologies.

4. To implement a prototype for the methodology.

In order to achieve objectives three and four, SACMES was implemented as a prototype of the proposed model using Java programming language. Chapter 4 includes more detail and information about the architecture of the proposed model and its implementation. The purpose of the system is to provide a model that can learn from the natural language specifications that it processes, and which can use the learnt information to improve its performance in mapping conceptual models from natural language requirements and minimise the need for human intervention. To achieve this task, the model integrates natural language processing tools, an ontology, WordNet and human intervention. The input of the model is the natural language text of a specific problem. The system is divided into three stages, namely, the pre-processing stage, the entities identification stage and the relationships identification stage. The pre-processing stage takes the natural language text input and performs textual analysis in order to define a candidate list of nouns that could be mapped into entities. This is a fully automated stage performed by natural language tools. The entities identification stage takes the candidate entities defined in the pre-processing stage and converts them into entities. This stage is semi-automated and supported by WordNet, a CMO, a UHKB and linguistic rules applied by human intervention. At the relationship identification stage, the system then produces a list of relationships. This stage is also semi-automated and, in order for the stage to extract an appropriate list of relationships, it is supported by natural language processing tools, the CMO, the UHKB database and linguistic rules applied by human intervention. The entities and relationships that are extracted from the system are stored in the CMO, and the behaviour of the user is stored in the UHKB. When the system processes further requirement specifications, the system will use the stored information to extract entities and relationships. Thus, the more case studies the system processes, the more it will learn from users, and consequently its performance will improve in terms of being able to predict entities and relationships with less human intervention.

5. To conduct an empirical evaluation of the methodology using the prototype to ascertain the effectiveness of its implementation.

5.1 One of the goals of this evaluation was to determine whether the proposed model could improve the performance of designers in creating conceptual models. To achieve this objective, a test set of case studies and their model answers was used. Twenty novice designers were involved as subjects in the experiment. The subjects were divided into two groups, each group having ten subjects. Each designer was requested to provide answers for two case studies in the test set by using the system, and answers for another two cases without using the system. The answers provided by subjects when using the system, and those given by subjects without using it, were compared to model answers for the case studies provided by human experts in system analysis. It was found that the average performance of the designers was improved when they used the system. More details about the experiment and its results are available in Section 5.1.

5.2. The second goal of the evaluation was to determine whether the knowledge stored by SACMES from natural language texts could improve the performance of the system and reduce the need for human intervention. To achieve this objective, a training set of fifty case studies was prepared. A test set of five case studies, including their model answers, was also prepared. The training set was divided into ten groups, each group consisting of five case studies. The system was used to find answers for the test set of case studies, and the recall and precision were recorded. The system was then trained on five case studies from the training set, after which the system was again used on the test set and the recall and precision recorded. Each time the training set count was increased by five case studies, the system was further tested using the test case studies. The purpose of testing the system each time the number of training case studies increased was to determine whether the information stored by the system would help to improve its performance. The results demonstrated that the performance of the system did indeed improve as the count of case studies used to train the system increased.

6.2 Limitations and Future Work

1. At the end of the process, the system produces a text report containing a list of entities and a list of relationships. It would be better if the system were able to draw the entities relationship diagram in Chen's notation, rather than just providing a report listing entities and relationships.

2. In terms of relationships, the system currently supports basic types including one-to-one, one-to-many and many-to-many relationships. It would be interesting if other relationships were added, such as generalisation, specialisation and aggregation.

3. Experiments were conducted on the use of SACMES by novice designers, and the results showed improvement in the designers' performance when they used the system compared to when not using it. It would be valuable if SACMES could also be tested by expert designers, in order to see how their performance is affected by using SACMES in comparison with their creation of handcrafted models.

4. In the evaluation of SACMES, the subjects were requested to provide answers for a test set of case studies both by using the system and without using it. The answers obtained from the subjects when using and when not using the system were compared to model answers provided by human experts. The author has noted that many of the answers provided by the subjects when using the system could have been correct answers, but were classified as incorrect because they were not identified in the model answers. This reduces the precision of the results. It would have been better if the author had been able to measure the answers given by experts, those given by subjects using SACMES and those provided by subjects without using it, based on a set of criteria to determine the best performance.

5. During the relationships identification stage, the user is requested to give a name to unnamed relationships. These relationships are then stored in the CMO. Sometimes, however, users may give names for relationships that contain spelling mistakes, which will be stored as such in the CMO. This information will be retrieved when requested and because of the spelling mistakes, the user may not understand it. It would be helpful if techniques for checking spelling and grammar could be used to ensure the user has given valid names and non-redundant names for relationships before they are stored in the CMO.

6. As the system is being used, the information stored by the system increases. As the information stored by the system increases, retrieval techniques need to be developed so that the precise information required can be retrieved. The retrieval techniques currently used by the system rely on spelling matching. For example, if the entities ‘student’ and ‘module’ are mentioned within the requirement specifications of a university system, the system will retrieve every relationship that includes ‘student’ and ‘module’. Future work could be undertaken to develop the retrieval techniques so that just the information needed is retrieved and any unnecessary information is eliminated.

7. The evaluation conducted to prove that information stored by the system will be useful in improving the system’s performance was achieved with a training set consisting of fifty case studies. The results of this evaluation would be more valuable if a larger training set had been used. Furthermore, instead of the author playing the role of designer to perform the evaluation, it would have improved validity if system designers had participated in the evaluation.

8. SACMES learns from the natural language requirements that it processes, stores the learnt information in the CMO and UHKB, and then uses this information to improve its performance in extracting conceptual models from natural language text. The results obtained after training the system on the training set showed that the performance of the system improved by increasing the number of case studies used to train it. However, the author cannot claim that this is machine learning, because no machine learning algorithms were included in the training. Future research could be conducted to determine whether it is possible to use machine learning techniques and algorithms to allow the system to learn from natural language specifications. Using such algorithms may improve the performance of the system more effectively than storing user behaviour in the CMO and UHKB and using this information when requested, which is how the current version of SACMES learns.

List of Appendices

Appendix 1:

List of sixty-eight documents which cited the paper entitled 'English Sentence Structure and Entity Relationship Diagrams' and were identified to be read in more depth and detail.

NO	Article title	Type of Document
1.	Conceptual modelling through linguistic analysis using LIDA (Overmyer et al., 2001).	Conference paper
2.	On the Systematic Analysis of Natural Language Requirements with CIRCE (ASE) (Ambriola & Gervasi, 2006).	Journal paper
3.	Generating Natural Language specifications from UML class diagrams (Meziane, Athanasakis, & Ananiadou, 2008).	Journal paper
4.	Transformation of requirement specifications expressed in natural language into an EER model (Tjoa & Berger, 1994).	Conference paper
5.	Semantic parameterization: A process for modelling domain descriptions (Breaux, Antón, & Doyle, 2008).	Journal paper
6.	Conceptual predesign bridging the gap between requirements and conceptual design (Kop & Mayr, 1998).	Conference paper
7.	Heuristic-based entity-relationship modelling through natural language processing (Omar et al., 2004).	Conference paper
8.	A system for the semiautomatic generation of ER models from natural language specifications (Gomez et al., 1999).	Journal article
9.	Applying a natural language dialogue tool for designing databases (Buchholz et al., 1995).	International Workshop
10.	English, Chinese and ER diagrams (Chen, 1997)	Journal article
11.	Analyzing informal requirements specifications: a first step towards conceptual modelling (Burg & Van de Riet, 1996).	Journal article
12.	English sentence structures and EER modelling (Hartmann & Link, 2007).	Conference paper
13.	A taxonomic class modelling methodology for object-oriented analysis (Song et al., 2004).	Conference paper
14.	On mapping natural language constructs into relational algebra through ER representation (Tseng et al., 1992).	Journal article

NO	Article title	Type of Document
15.	Extracting conceptual graphs from Japanese documents for software requirements modelling (Hasegawa, Kitamura, Kaiya, & Saeki, 2009).	Conference paper
16.	Finding comparatively important concepts between texts (Lecoeuche, 2000).	Conference paper
17.	Parsed use case descriptions as a basis for object-oriented class model generation (Elbendak, Vickers, & Rossiter, 2011).	Journal article
18.	From user requirements to UML class diagram (Herchi & Abdessalem, 2012).	Conference paper
19.	A method for the definition and treatment of conceptual schema quality issues (Aguilera, Gómez, & Olivé, 2012).	Conference paper
20.	MOODD, a method for object-oriented database design (Silva & Carlson, 1995).	Journal Article
21.	Schema Methodology for Large Entity-Relationship Diagrams (Gilberg, 1985).	Conference paper
22.	Semi-automatic conceptual data modelling using entity and relationship instance repositories (Thonggoom et al., 2011b).	Conference paper
23.	An automated multi-component approach to extracting entity relationships from database requirement specification documents (Du & Metzler, 2006).	Conference paper
24.	A complete set of guidelines for naming UML conceptual schema elements (Aguilera, Gómez, & Olivé, 2013).	Journal article
25.	Semantic analysis in the automation of ER modelling through natural language processing (Omar, Hanna, & Mc Kevitt, 2006).	Conference paper
26.	Automatic acquisition of linguistic patterns for conceptual modelling (Zhou & Zhou, 2004) .	Journal article
27.	Guidelines for NL-Based requirements specifications in NIBA (Fliedl, Kop, Mayerthaler, Mayr, & Winkler, 2000).	Conference paper
28.	Enriching the class diagram concepts to capture natural language semantics for database access (Tseng & Chen, 2008).	Journal article
29.	A linguistic approach to conceptual modelling with semantic types and ontoUML (Castro, Baião, & Guizzardi, 2010).	Conference paper
30.	On the automatization of database conceptual modelling through linguistic engineering (Martínez & García-Serrano, 2000).	Conference paper
31.	Extracting Entity Relationship Diagram (ERD) from relational database schema (Al-Masree, 2015).	Journal article

NO	Article title	Type of Document
32.	Building Natural Language Interface to an ER Database (Luk, 1989).	Conference paper
33.	Extending the UML concepts to transform natural language queries with fuzzy semantics into SQL (Tseng & Chen, 2006).	Journal article
34.	Automatic generation of extended er diagram using natural language processing (Shahbaz, Ahsan, Shaheen, Nawab, & Masood, 2011).	Journal article
35.	Towards the automated business model-driven conceptual database design (Brdjanin & Maric, 2013).	Conference paper
36.	Automatic builder of class diagram (ABCD): an application of UML generation from functional requirements (Ben Abdessalem Karaa et al., 2016).	Journal article
37.	Application of conceptual structures in requirements modelling (Bogatyrev & Nuriahmetov, 2011).	Conference paper
38.	The Circe approach to the systematic analysis of NL requirements (Ambriola & Gervasi, 2003).	Technical-report
39.	NTS-based derivation of KCPM cardinalities: From natural language to conceptual predesign (Fliedi, Kop, Mayerthaler, Mayer, & Winkler, 1996).	Journal article
40.	From Natural Language Requirements to a Conceptual Model (Kop, Fliedl, & Mayr, 2010).	Conference paper
41.	Formalization and classification of product requirements using axiomatic theory of design modelling (Chen, 2006).	Master thesis
42.	The representation of rules in the ER model (Monarchi & Smith, 1992).	Journal article
43.	Extracting Domain Models from Natural-Language Requirements: Approach and Industrial Evaluation (Arora, Sabetzadeh, Briand, & Zimmer, 2016).	Conference paper
44.	Concept extraction from business documents for software engineering projects (Ménard & Ratté, 2016).	Journal article
45.	Implementing database access control policy from unconstrained natural language text (Slankas, 2013).	Conference paper
46.	Conceptual modelling & natural language analysis (Rolland, 2013).	Book section
47.	Natural language discourse generation in a support tool for conceptual modelling (Dalianis, 1992).	Conference paper
48.	Conceptual modelling tool for novice designers (Kop, 2008).	Journal article
49.	Modelling, extraction, and transformation of semantics in computer aided engineering systems (Zeng, Kim, Raskin, Fung, & Kitamura, 2013).	Journal article

NO	Article title	Type of Document
50.	Automated Enterprise Data Model by Formulating Requirements (Lee, 2009).	Journal article
51.	Bridging the gap between natural and information modelling languages: an informal approach to information modelling learning (Kern & Ramos, 2002).	Journal article
52.	Extracting Entity Relationship Diagram (ERD) from English Sentences (Al-Btoush, 2015).	Journal article
53.	The use of semantic heuristics in the automation of ER modelling (Omar, Muhammad, & Yahya, 2007).	Conference paper
54.	Validating Documentation with Domain Ontologies (Kof & Pizka, 2005).	Conference paper
55.	Requirements Modelling: From Natural Language to Conceptual Models Using Recursive Object Model (ROM) Analysis (Wang, 2013).	PhD thesis
56.	A Survey on Conceptual Modelling (Castro et al., 2009).	Journal article
57.	Methodologies for Semi-automated Conceptual Data Modelling from Requirements (Song et al., 2015).	Conference paper
58.	Automatic Construction of Conceptual Models to Support Early Stages of Software Development (Chioasca, 2015).	PhD thesis
59.	An algorithm for Finding a Relationship Between Entities: Semi-Automated Schema Integration Approach (Chan, 2017).	PhD thesis
60.	Implementing a Database from a Requirement Specification (Omer & Wilson, 2015).	Journal article
61.	Design a Data Model Diagram from Textual Requirements (Abdullah & Saleem, 2013).	Journal article
62.	Requirement-Oriented Entity Relationship Modelling (Lee & Shin, 2010).	Journal article
63.	Survey of works that transform requirements into UML diagrams (Abdouli, Karaa, & Ghezala, 2016).	Conference paper
64.	From Natural Language to Object Oriented Requirements: an Annotated Bibliography (Mich & Giuliani , 1995).	Journal article
65.	ER—A Historical Perspective and Future Directions (Davis, Jajodia, Ng, & Yeh, 1983).	Conference paper
66.	Conceptual schema extraction using POS annotations and weighted edit distance algorithm (Shinde, Kulkarni, Patwardhan, Sarda, & Mantri, 2015).	Conference paper
67.	Heuristic rules for transforming preconceptual schemas into uml 2.0 diagrams: a C# implementation (Zapata & Cardona, 2008).	Journal article

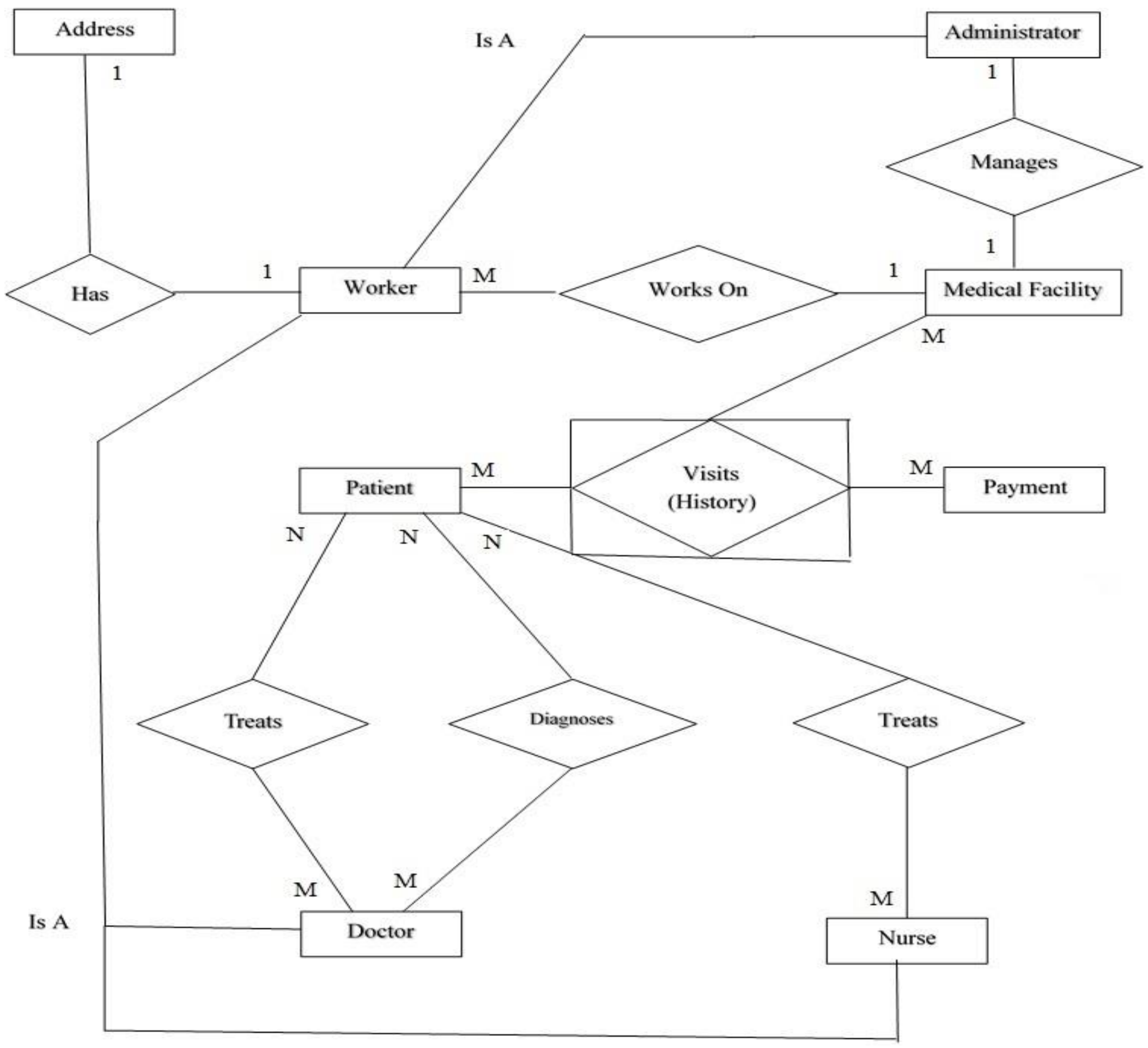
NO	Article title	Type of Document
68.	An environment for automated UML diagrams obtaining from a controlled language (Zapata & Arango, 2007).	Journal article

Appendix 2:

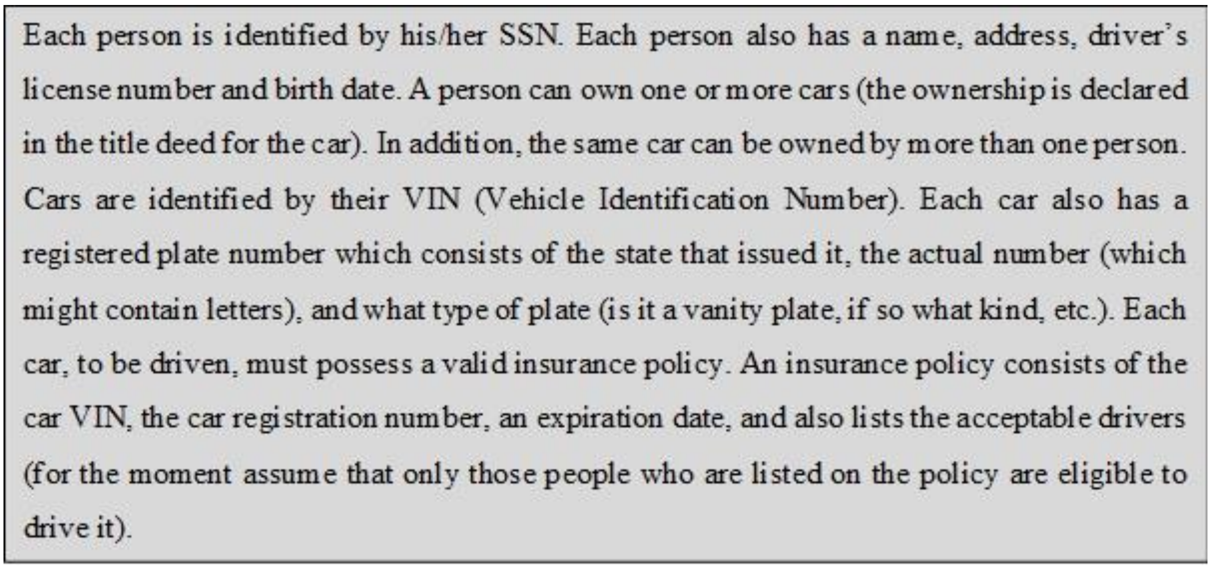
Test set for Experimental One

> Medical workers work in a medical facility. A facility has a name, address, possibly a specialty area, and the name of an administrator. A patient visits a medical facility for a diagnosis of a health problem. A patient has name, id number (ssn), address (street, city, state, and zip), phone (day and evening), employer (company) name, employer address, type of benefits eligible for, and method for payment. Doctors are allowed to perform any kind of diagnosis and treatment based on their specialty. Nurses participate in treatment.

Appendix Figure 1 Problem One in Easy Set (Du, 2008, p.169)

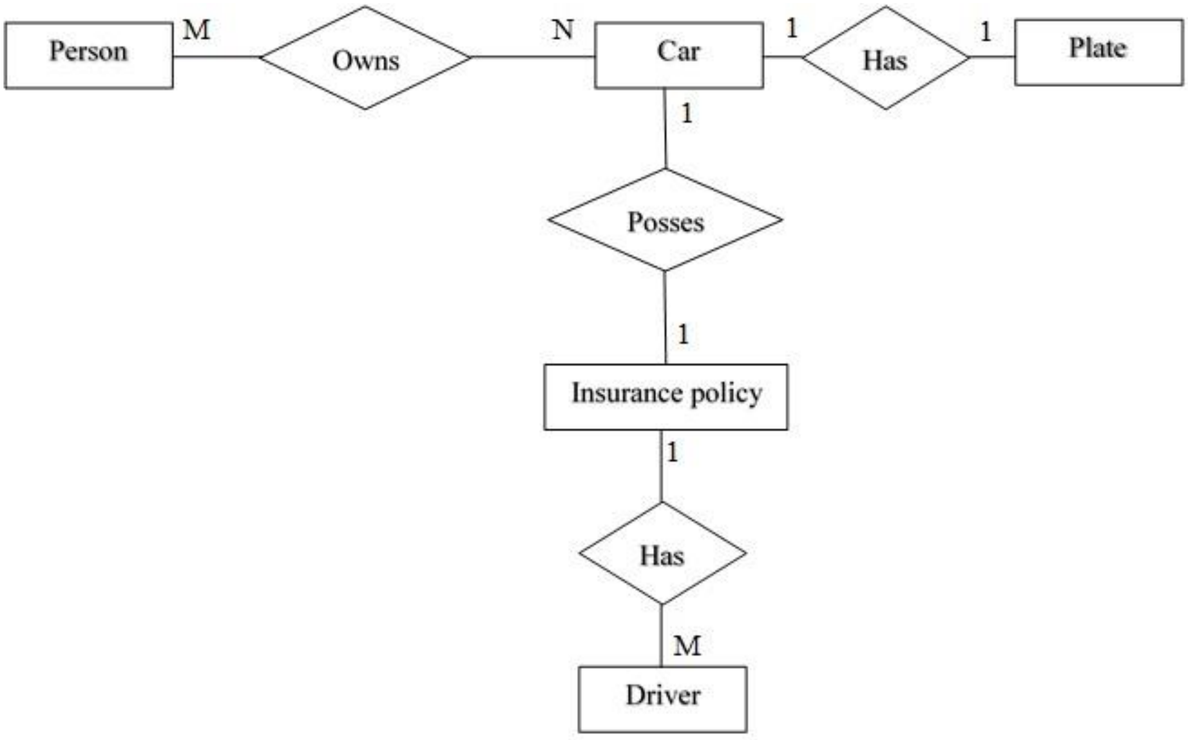


Appendix Figure 2 Model Answer for Problem One in Easy Set provided by Database Designer[21]

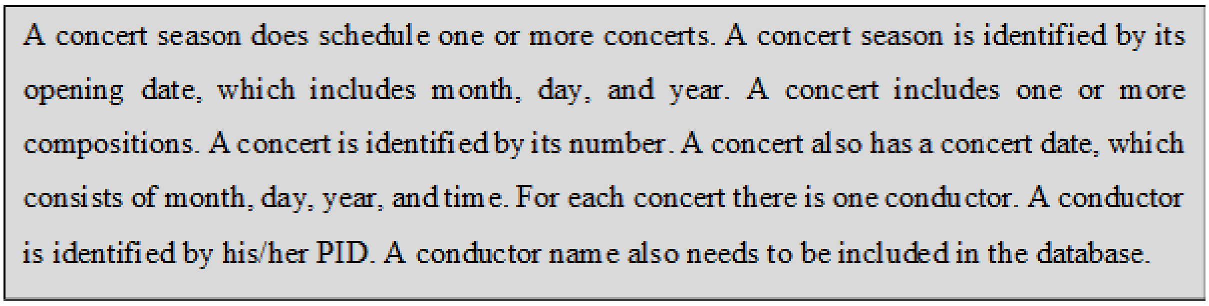


Each person is identified by his/her SSN. Each person also has a name, address, driver's license number and birth date. A person can own one or more cars (the ownership is declared in the title deed for the car). In addition, the same car can be owned by more than one person. Cars are identified by their VIN (Vehicle Identification Number). Each car also has a registered plate number which consists of the state that issued it, the actual number (which might contain letters), and what type of plate (is it a vanity plate, if so what kind, etc.). Each car, to be driven, must possess a valid insurance policy. An insurance policy consists of the car VIN, the car registration number, an expiration date, and also lists the acceptable drivers (for the moment assume that only those people who are listed on the policy are eligible to drive it).

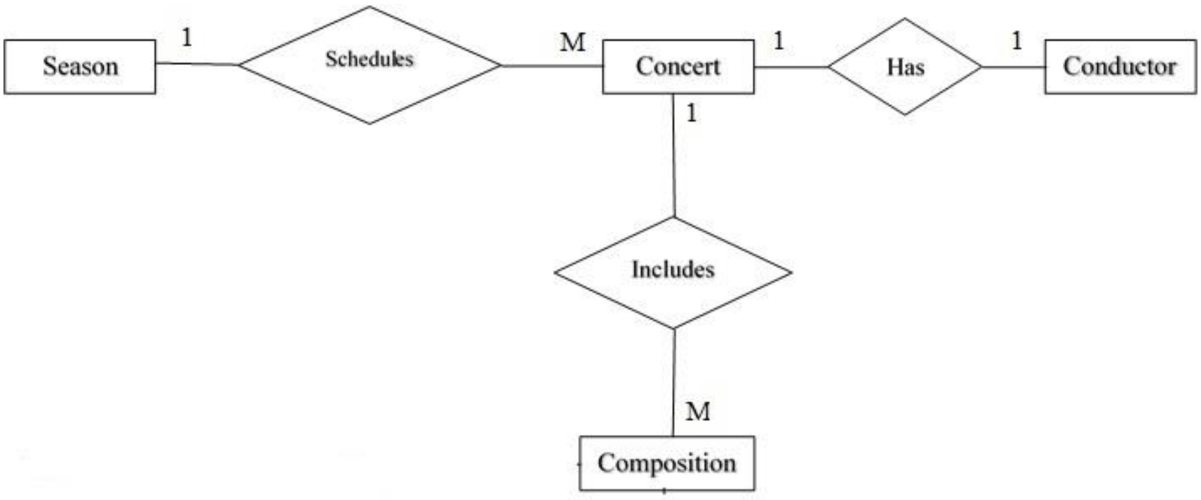
Appendix Figure 3 Problem Two in Easy Set (Du, 2008, p.172)



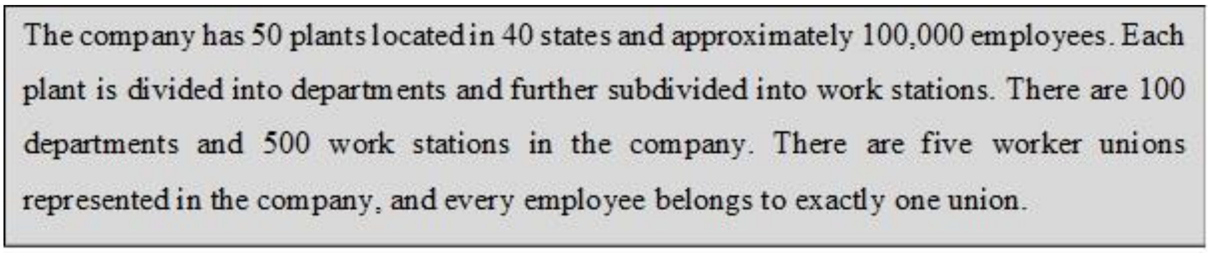
Appendix Figure 4 Solution for Problem Two in Easy Set provided by Database Designer[21]


A concert season does schedule one or more concerts. A concert season is identified by its opening date, which includes month, day, and year. A concert includes one or more compositions. A concert is identified by its number. A concert also has a concert date, which consists of month, day, year, and time. For each concert there is one conductor. A conductor is identified by his/her PID. A conductor name also needs to be included in the database.

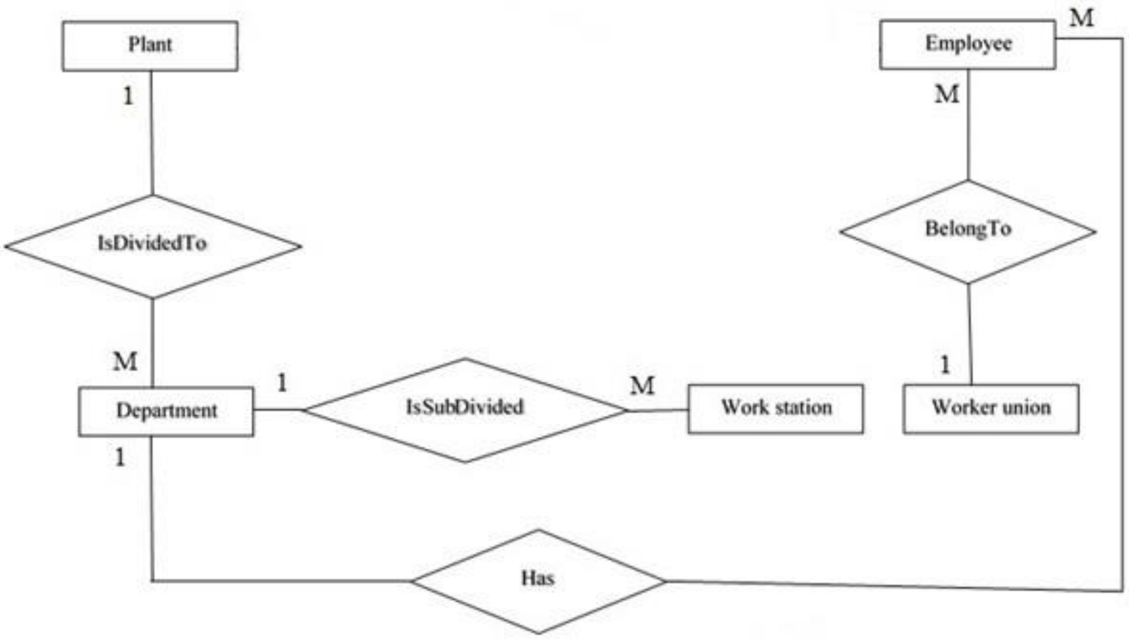
Appendix Figure 5 Problem Three in Easy Set (Du, 2008, p. 167)



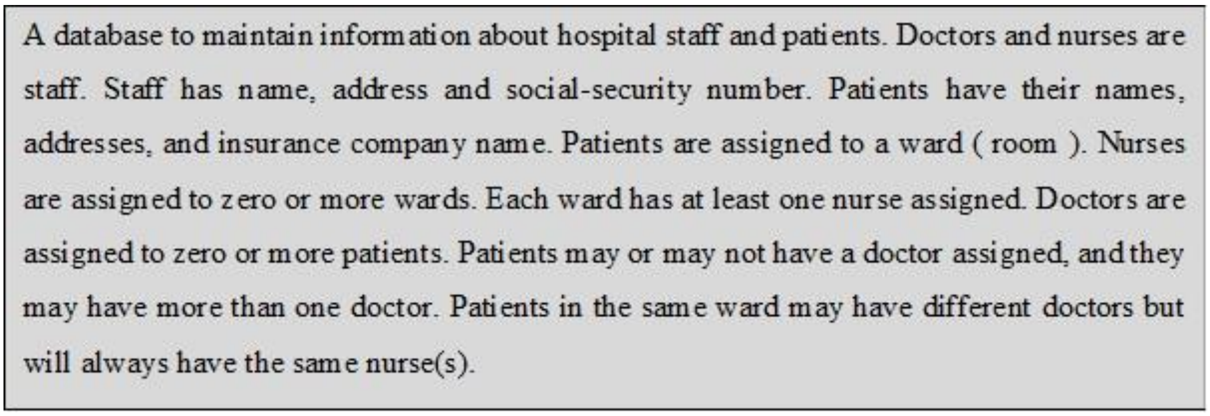
Appendix Figure 6 Solution for Problem Three in Easy Set provided by Database Designer[21]


The company has 50 plants located in 40 states and approximately 100,000 employees. Each plant is divided into departments and further subdivided into work stations. There are 100 departments and 500 work stations in the company. There are five worker unions represented in the company, and every employee belongs to exactly one union.

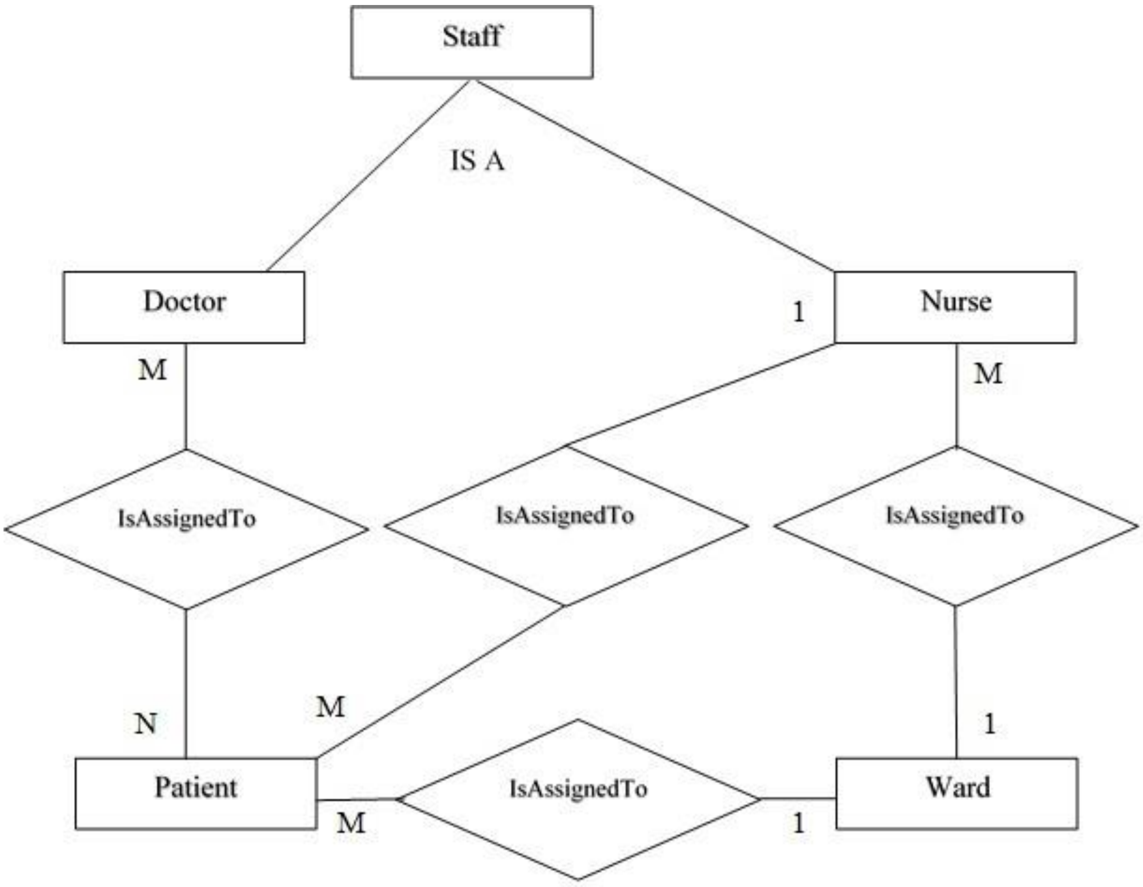
Appendix Figure 7 Problem Four in Easy Set (Du, 2008, p. 167)



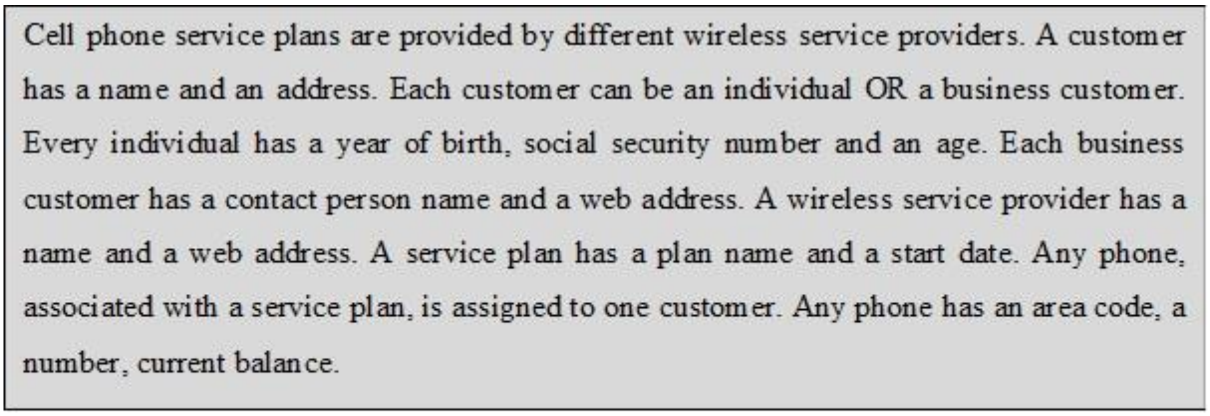
Appendix Figure 8 Solution for Problem Four in Easy Set provided by Database Designer[21]


A database to maintain information about hospital staff and patients. Doctors and nurses are staff. Staff has name, address and social-security number. Patients have their names, addresses, and insurance company name. Patients are assigned to a ward (room). Nurses are assigned to zero or more wards. Each ward has at least one nurse assigned. Doctors are assigned to zero or more patients. Patients may or may not have a doctor assigned, and they may have more than one doctor. Patients in the same ward may have different doctors but will always have the same nurse(s).

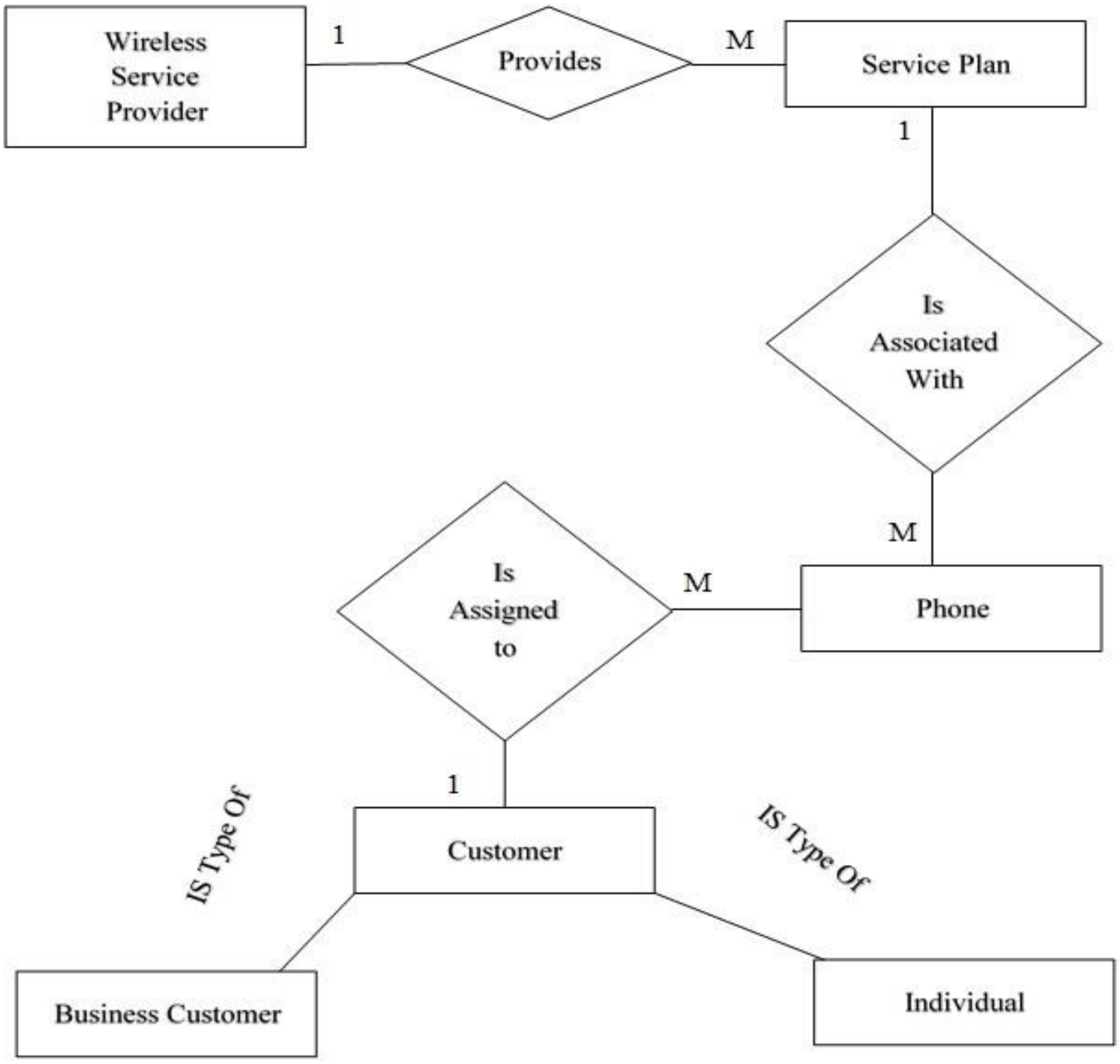
Appendix Figure 9 Problem Five in Easy Set (Du, 2008, p. 167)



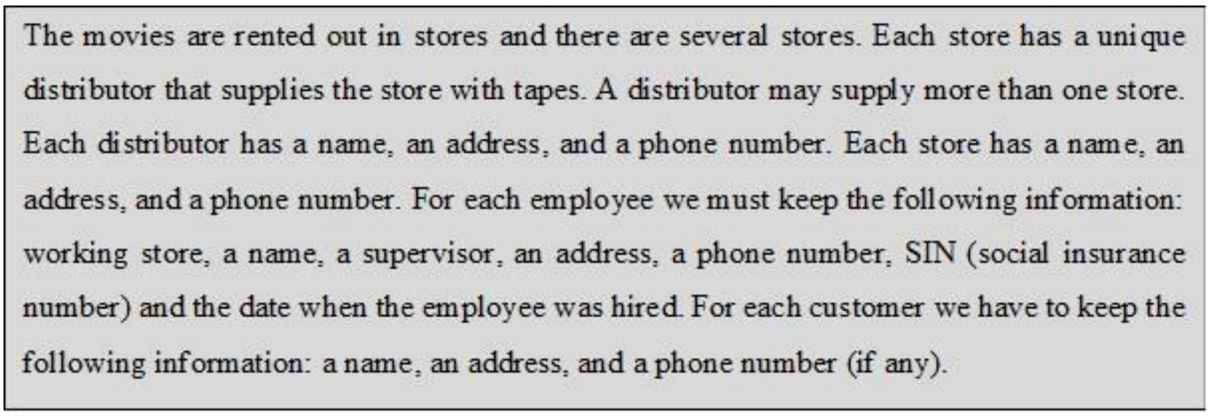
Appendix Figure 10 Solution for Problem Five in Easy Set provided by Database Designer[21]


Cell phone service plans are provided by different wireless service providers. A customer has a name and an address. Each customer can be an individual OR a business customer. Every individual has a year of birth, social security number and an age. Each business customer has a contact person name and a web address. A wireless service provider has a name and a web address. A service plan has a plan name and a start date. Any phone, associated with a service plan, is assigned to one customer. Any phone has an area code, a number, current balance.

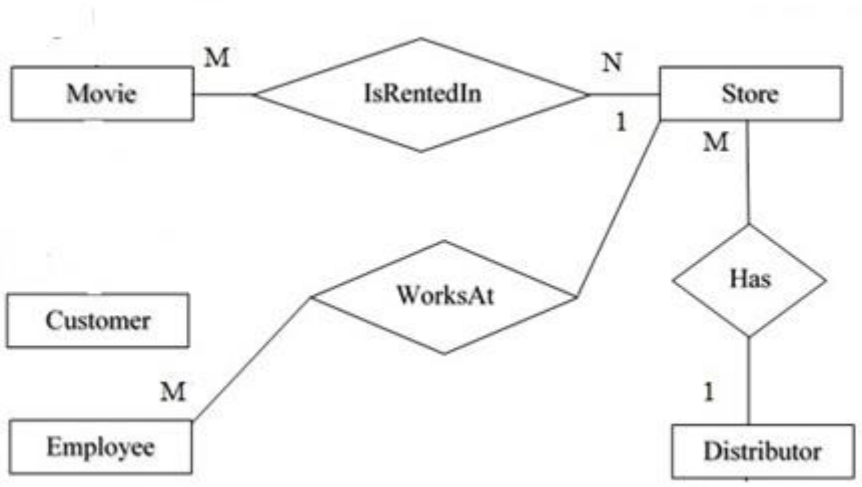
Appendix Figure 11 Problem Six in Easy Set (Du, 2008, p. 168)



Appendix Figure 12 Solution for Problem Six in Easy Set provided by Database Designer[21]

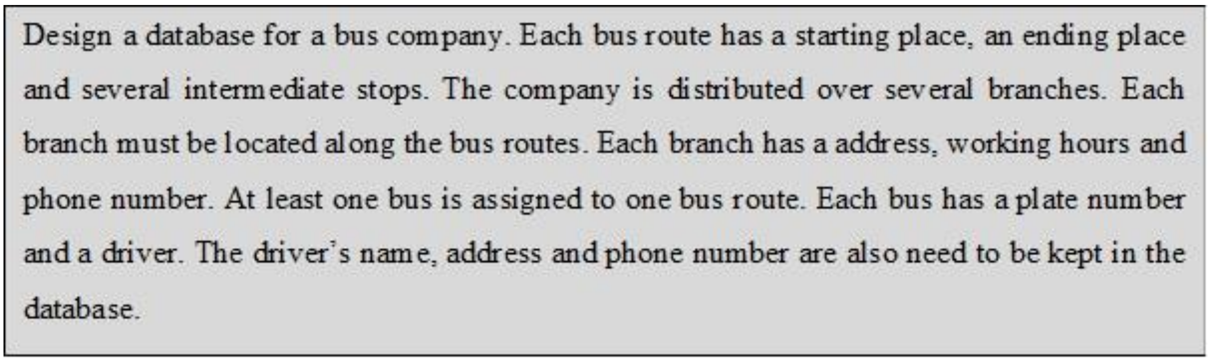

The movies are rented out in stores and there are several stores. Each store has a unique distributor that supplies the store with tapes. A distributor may supply more than one store. Each distributor has a name, an address, and a phone number. Each store has a name, an address, and a phone number. For each employee we must keep the following information: working store, a name, a supervisor, an address, a phone number, SIN (social insurance number) and the date when the employee was hired. For each customer we have to keep the following information: a name, an address, and a phone number (if any).

Appendix Figure 13 Problem Seven in Easy Set (Zhang, 2012, p. 34)

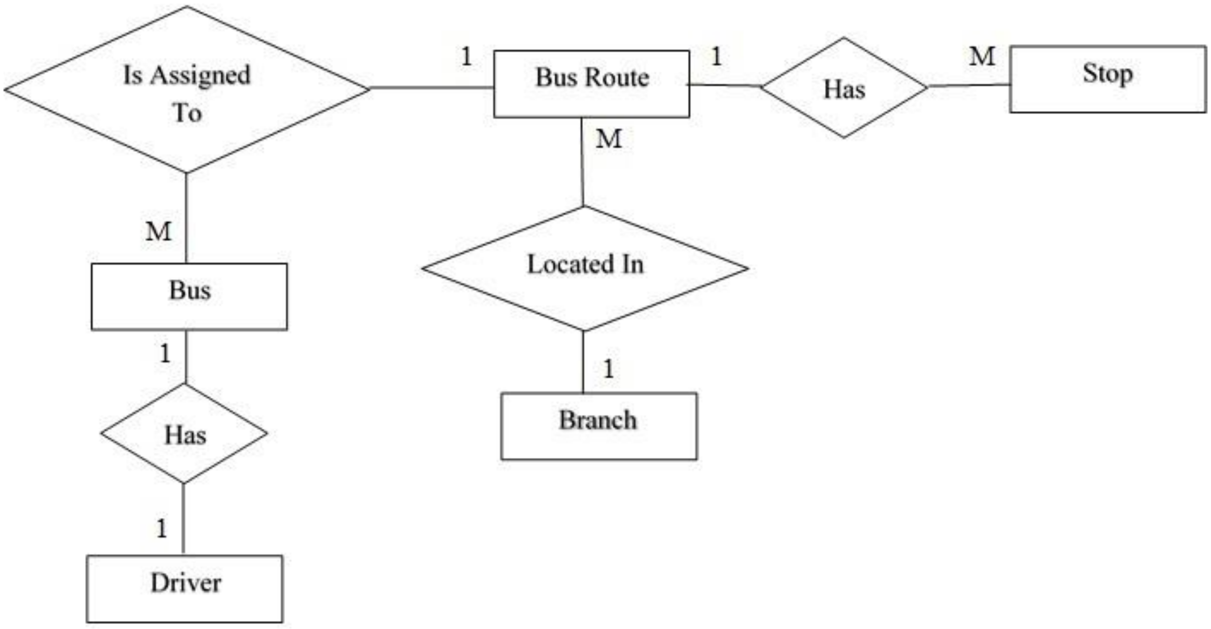


Appendix Figure 14 Solution for Problem Seven in Easy Set (Zhang, 2012)[22]

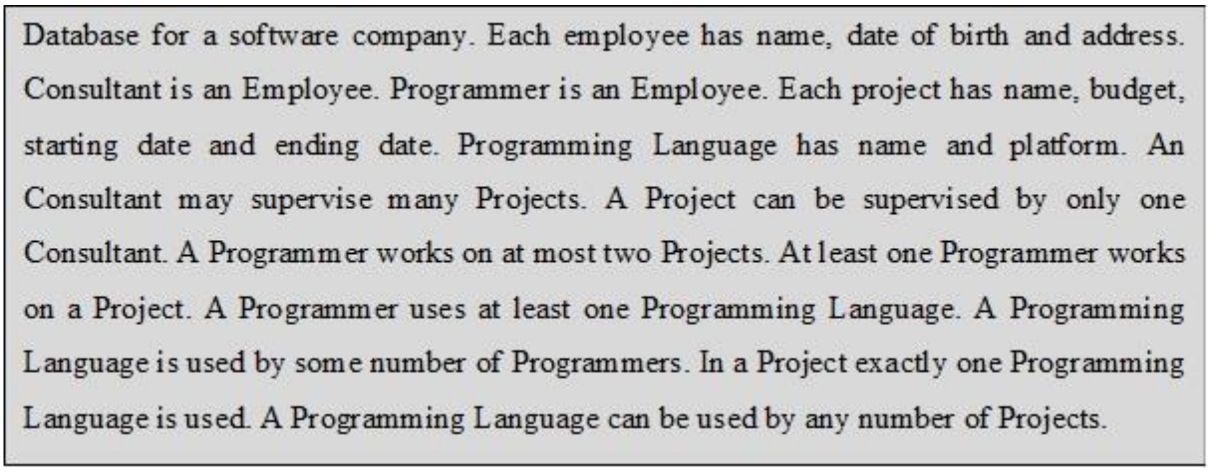
[22] https://macsphere.mcmaster.ca/bitstream/11375/11862/1/fulltext.pdf


Design a database for a bus company. Each bus route has a starting place, an ending place and several intermediate stops. The company is distributed over several branches. Each branch must be located along the bus routes. Each branch has a address, working hours and phone number. At least one bus is assigned to one bus route. Each bus has a plate number and a driver. The driver's name, address and phone number are also need to be kept in the database.

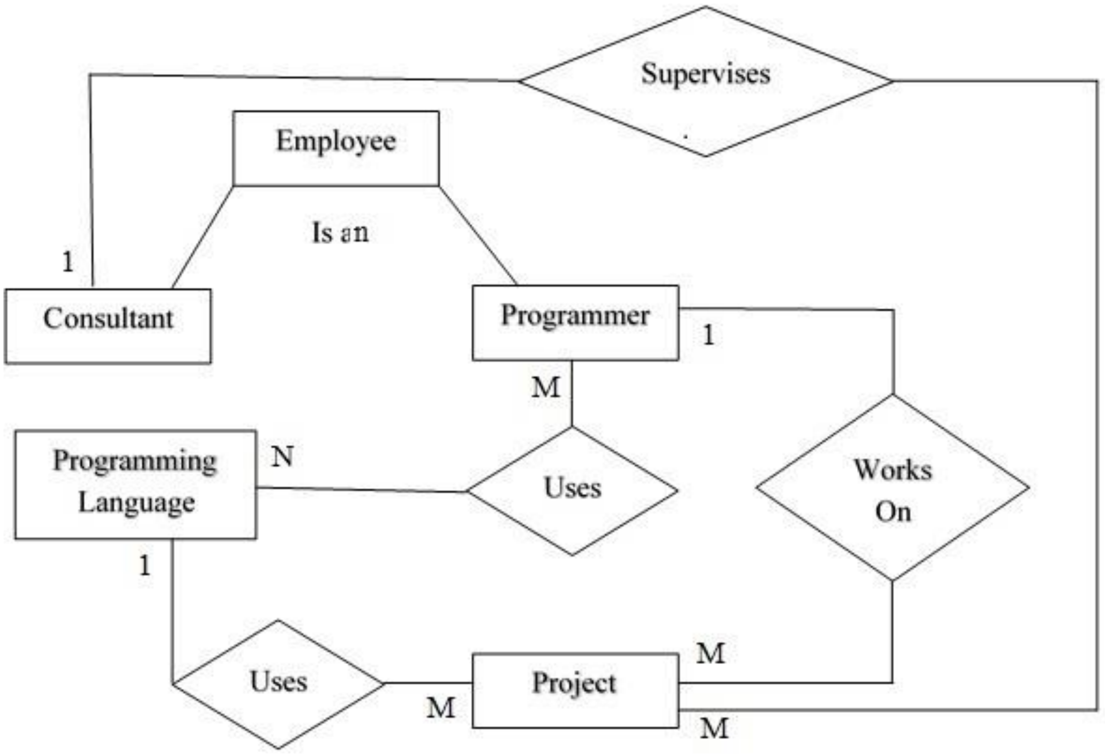
Appendix Figure 15 Problem Eight in Easy Set (Du, 2008, p. 168)



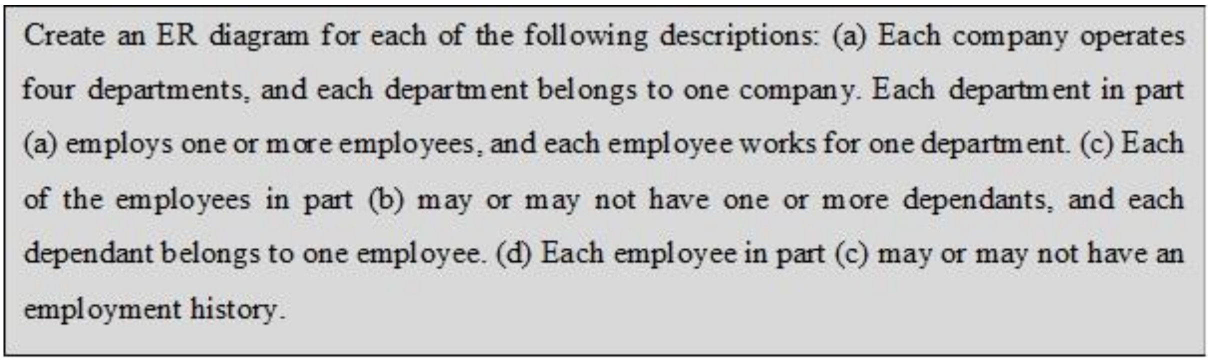
Appendix Figure 16 Solution for Problem Eight in Easy Set provided by Database Designer[21]


Database for a software company. Each employee has name, date of birth and address. Consultant is an Employee. Programmer is an Employee. Each project has name, budget, starting date and ending date. Programming Language has name and platform. An Consultant may supervise many Projects. A Project can be supervised by only one Consultant. A Programmer works on at most two Projects. At least one Programmer works on a Project. A Programmer uses at least one Programming Language. A Programming Language is used by some number of Programmers. In a Project exactly one Programming Language is used. A Programming Language can be used by any number of Projects.

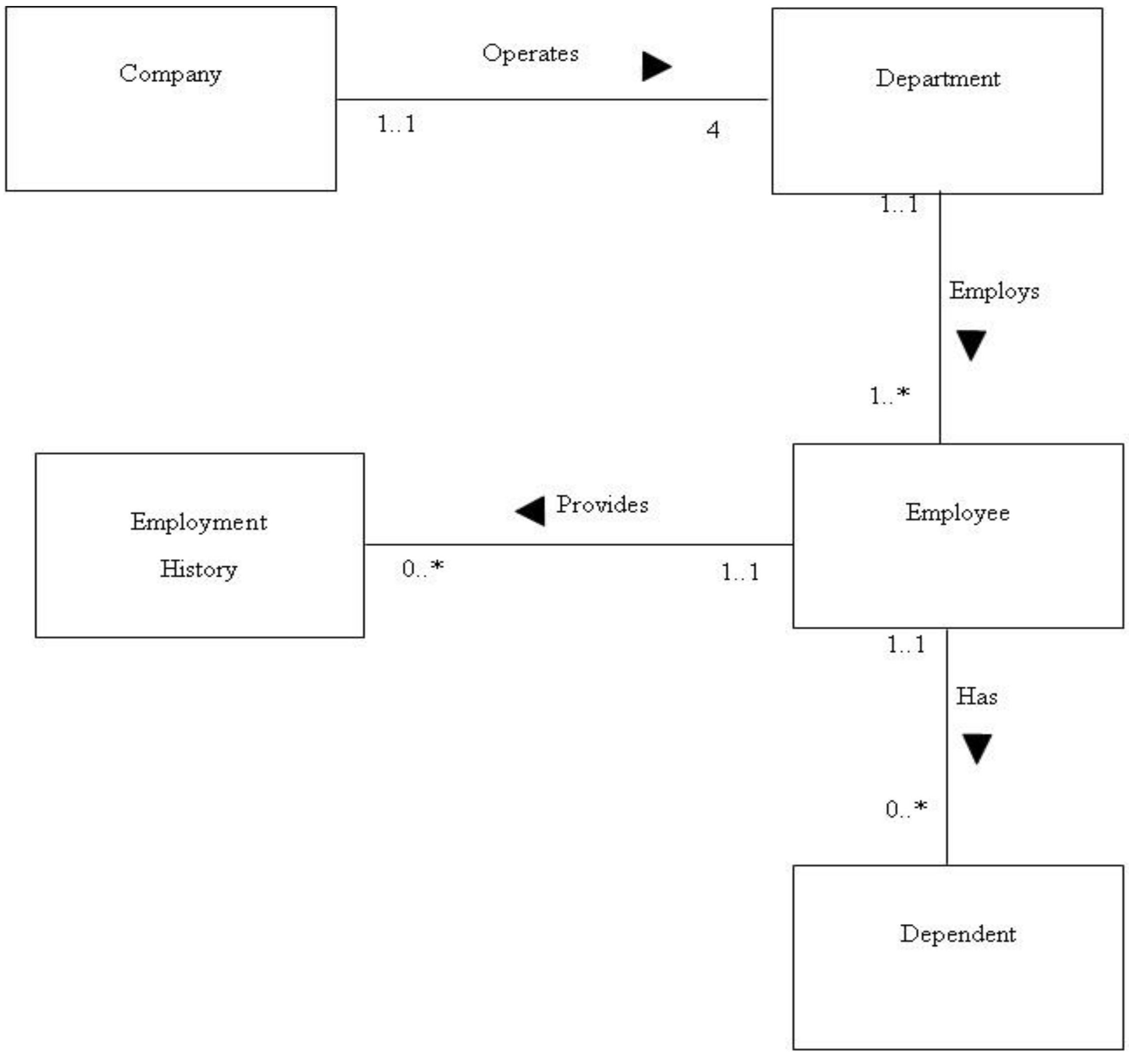
Appendix Figure 17 Problem Nine in Easy Set (Du, 2008, p. 169)



Appendix Figure 18 Solution for Problem Nine in Easy Set provided by Database Designer[21]

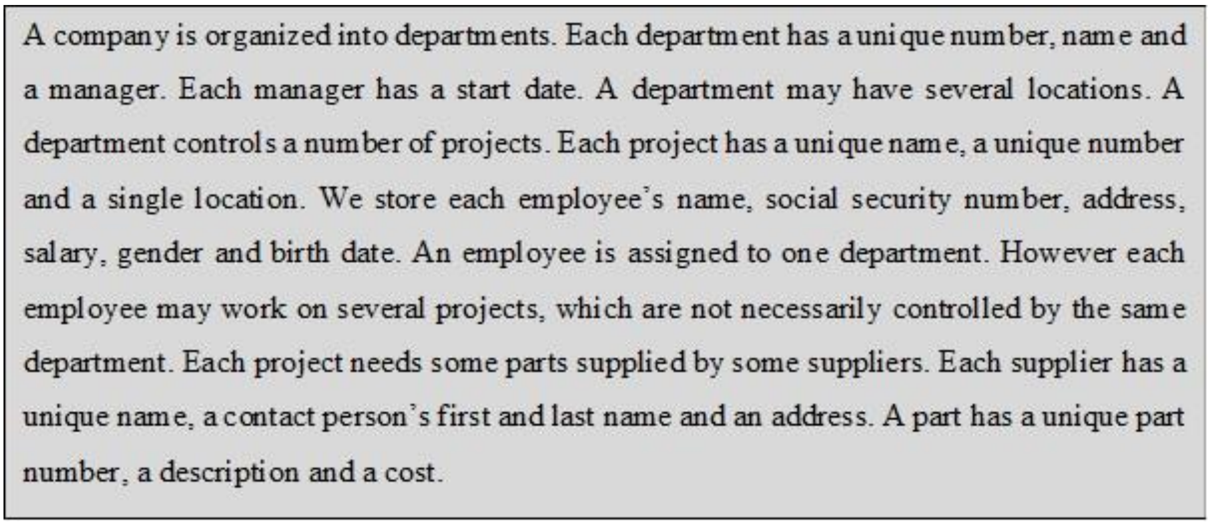

Create an ER diagram for each of the following descriptions: (a) Each company operates four departments, and each department belongs to one company. Each department in part (a) employs one or more employees, and each employee works for one department. (c) Each of the employees in part (b) may or may not have one or more dependants, and each dependant belongs to one employee. (d) Each employee in part (c) may or may not have an employment history.

Appendix Figure 19 Problem Ten in Easy Set (Connolly & Begg, 2015, p. 431)

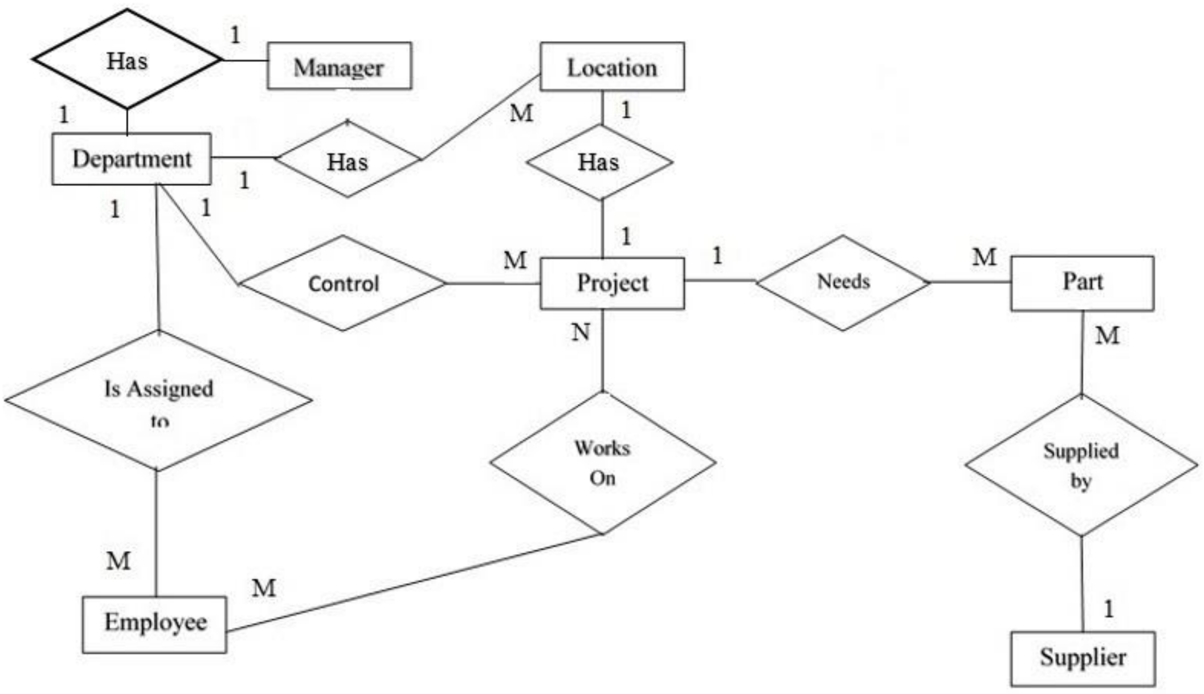


Appendix Figure 20 Solution for Problem Ten in Easy Set[23]

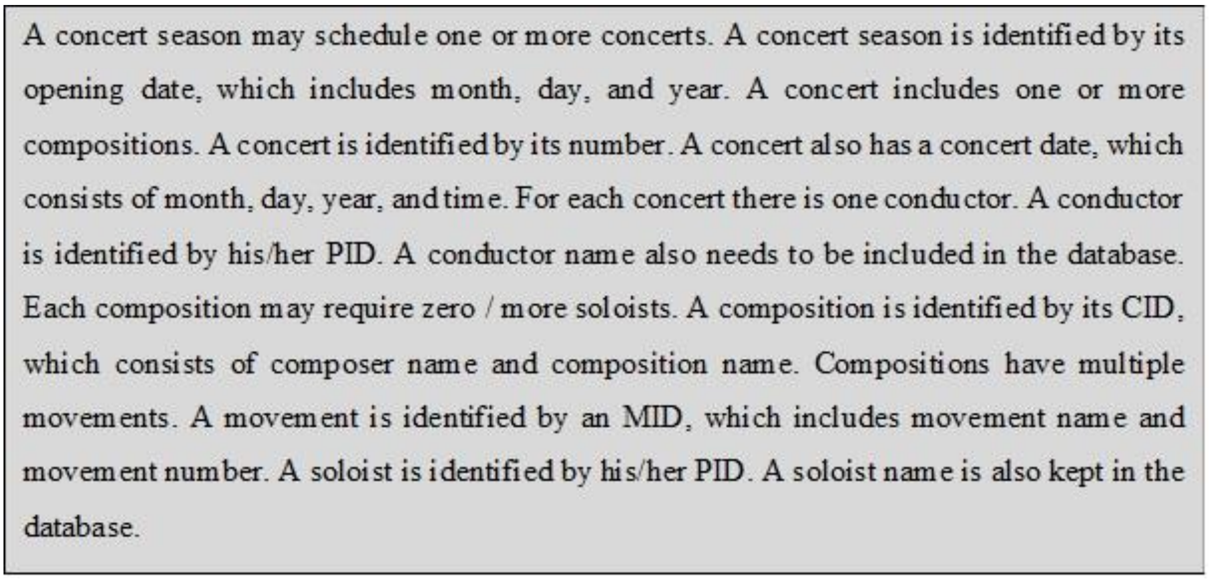
[23] https://www.scribd.com/document/170295338/Solution-Er


A company is organized into departments. Each department has a unique number, name and a manager. Each manager has a start date. A department may have several locations. A department controls a number of projects. Each project has a unique name, a unique number and a single location. We store each employee's name, social security number, address, salary, gender and birth date. An employee is assigned to one department. However each employee may work on several projects, which are not necessarily controlled by the same department. Each project needs some parts supplied by some suppliers. Each supplier has a unique name, a contact person's first and last name and an address. A part has a unique part number, a description and a cost.

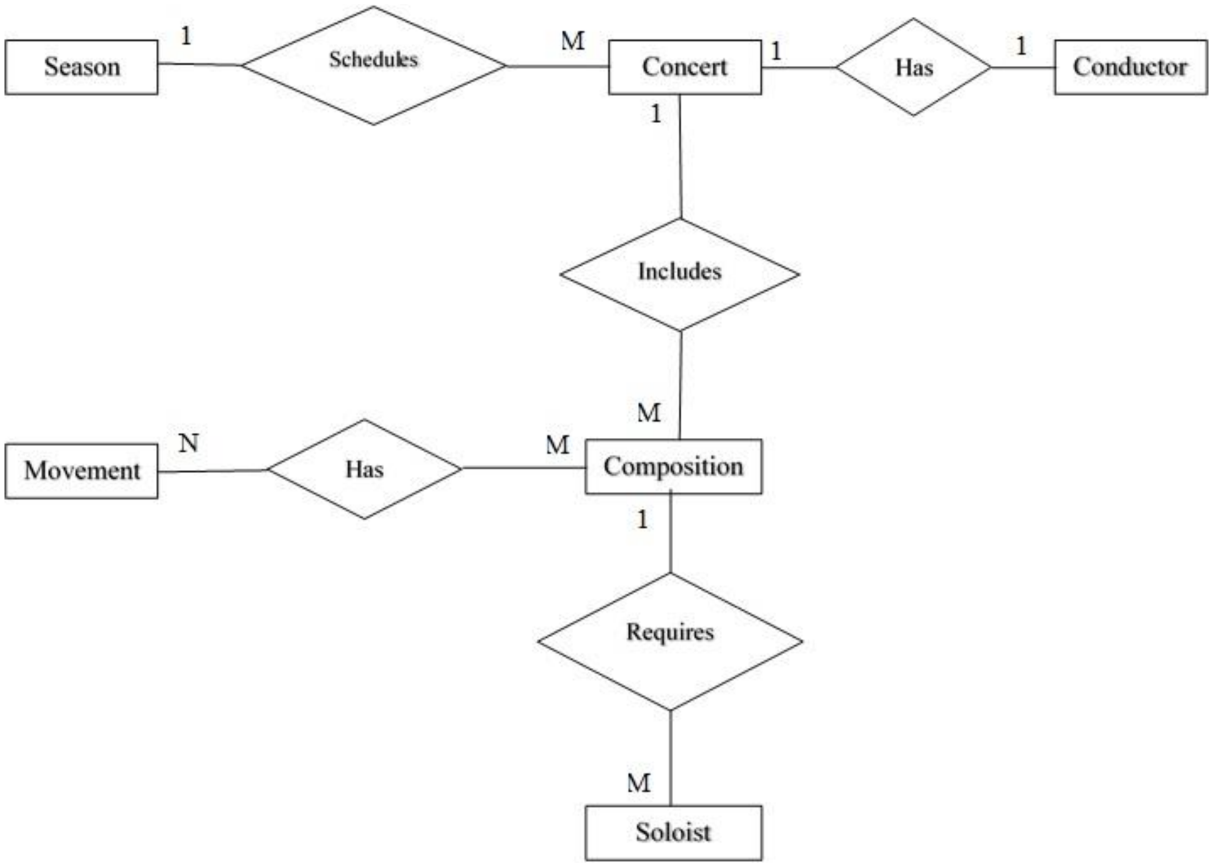
Appendix Figure 21 Problem One in Harder Set (Du, 2008, p. 170)



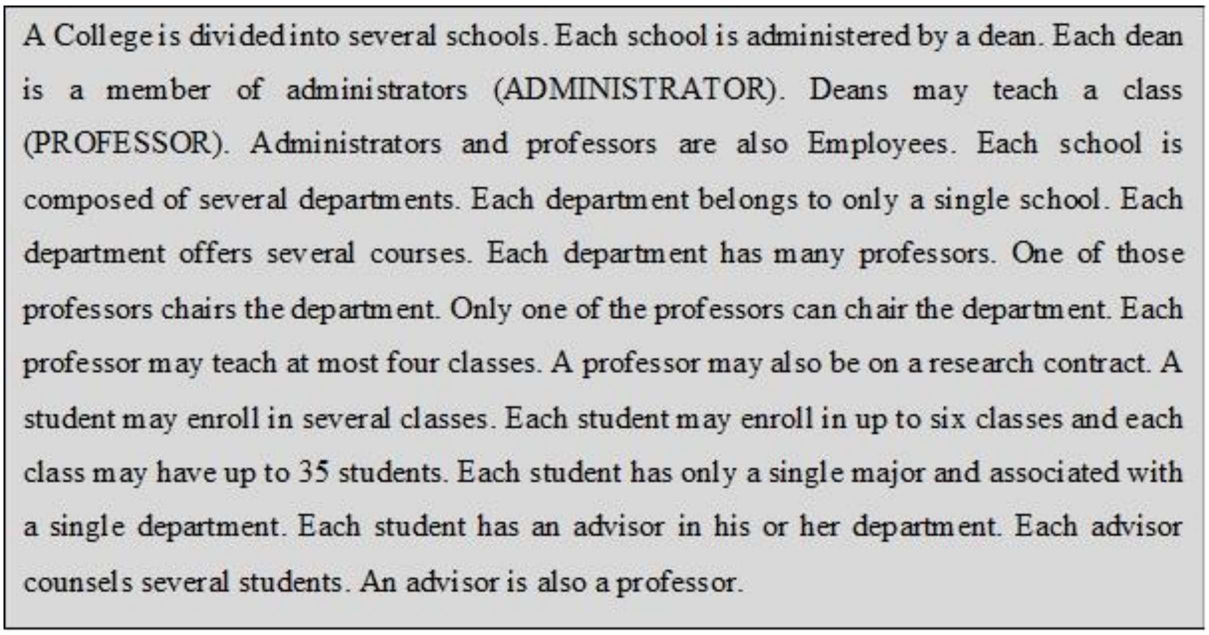
Appendix Figure 22 Solution for Problem One in Harder Set provided by Database Designer[21]


A concert season may schedule one or more concerts. A concert season is identified by its opening date, which includes month, day, and year. A concert includes one or more compositions. A concert is identified by its number. A concert also has a concert date, which consists of month, day, year, and time. For each concert there is one conductor. A conductor is identified by his/her PID. A conductor name also needs to be included in the database. Each composition may require zero / more soloists. A composition is identified by its CID, which consists of composer name and composition name. Compositions have multiple movements. A movement is identified by an MID, which includes movement name and movement number. A soloist is identified by his/her PID. A soloist name is also kept in the database.

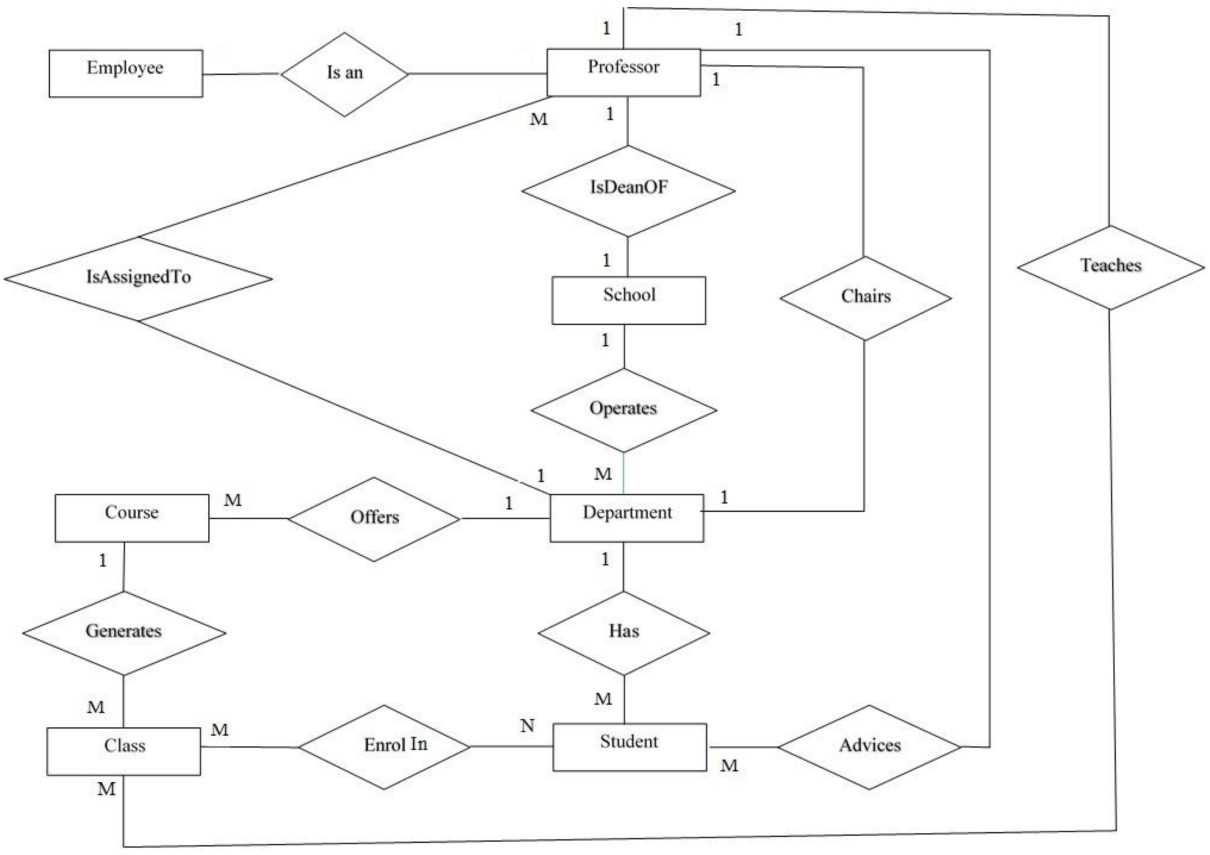
Appendix Figure 23 Problem Two in Harder Set (Du, 2008, p. 98)



Appendix Figure 24 Solution for Problem Two in Harder Set provided by Database Designer[21]

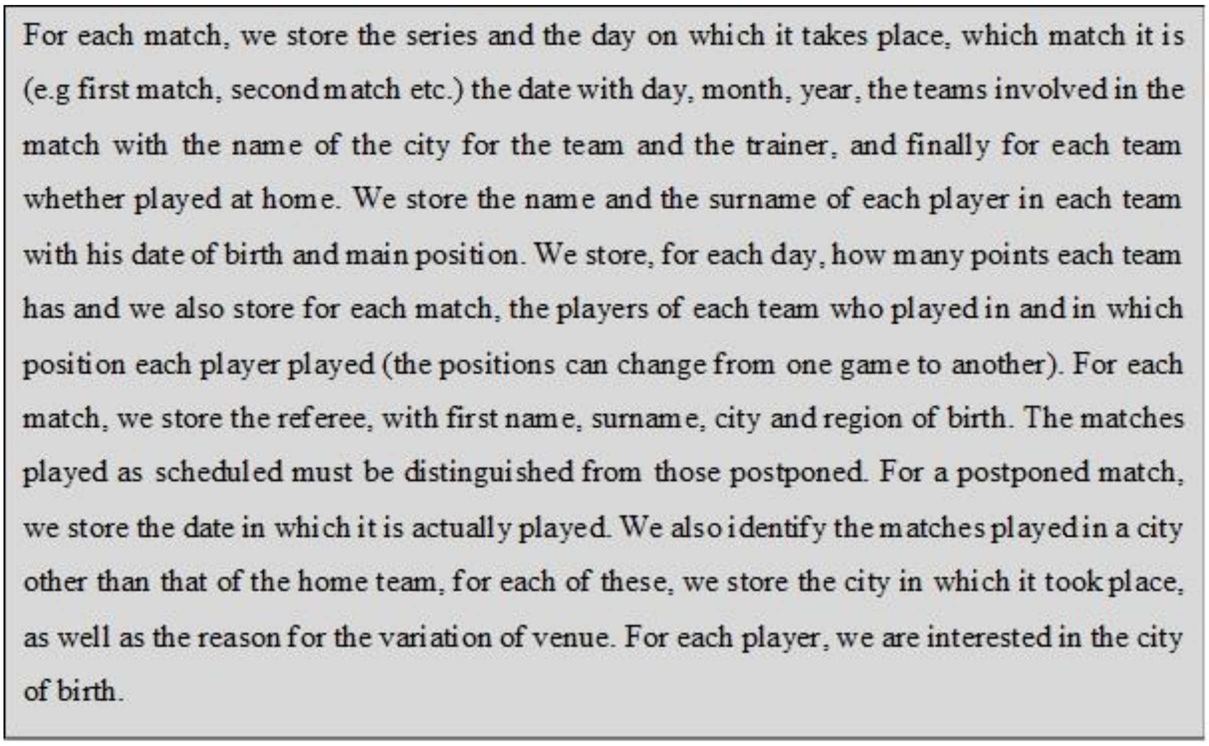

A College is divided into several schools. Each school is administered by a dean. Each dean is a member of administrators (ADMINISTRATOR). Deans may teach a class (PROFESSOR). Administrators and professors are also Employees. Each school is composed of several departments. Each department belongs to only a single school. Each department offers several courses. Each department has many professors. One of those professors chairs the department. Only one of the professors can chair the department. Each professor may teach at most four classes. A professor may also be on a research contract. A student may enroll in several classes. Each student may enroll in up to six classes and each class may have up to 35 students. Each student has only a single major and associated with a single department. Each student has an advisor in his or her department. Each advisor counsels several students. An advisor is also a professor.

Appendix Figure 25 Problem Three in Harder Set (Du, 2008, p.172)

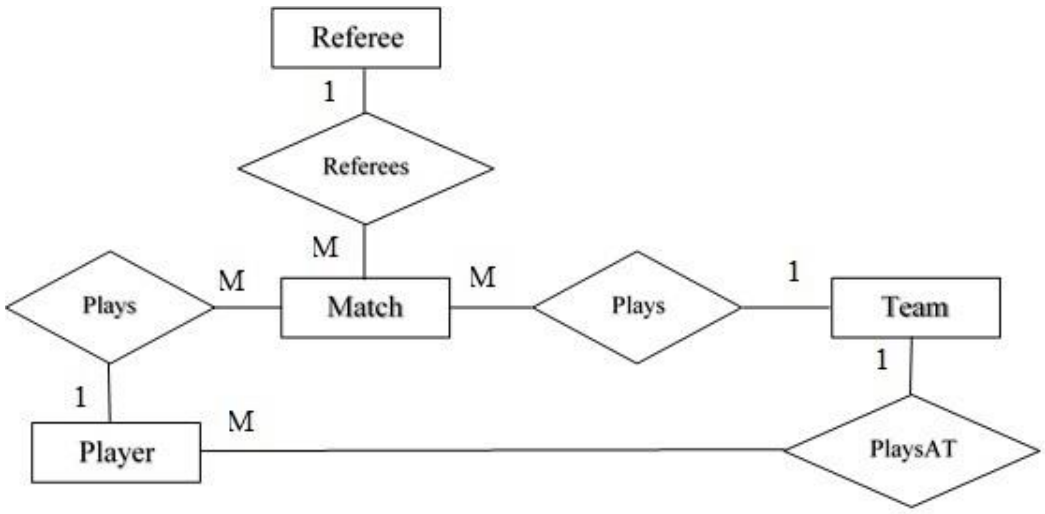


Appendix Figure 26 Solution for Problem Three in Easy Set[24]

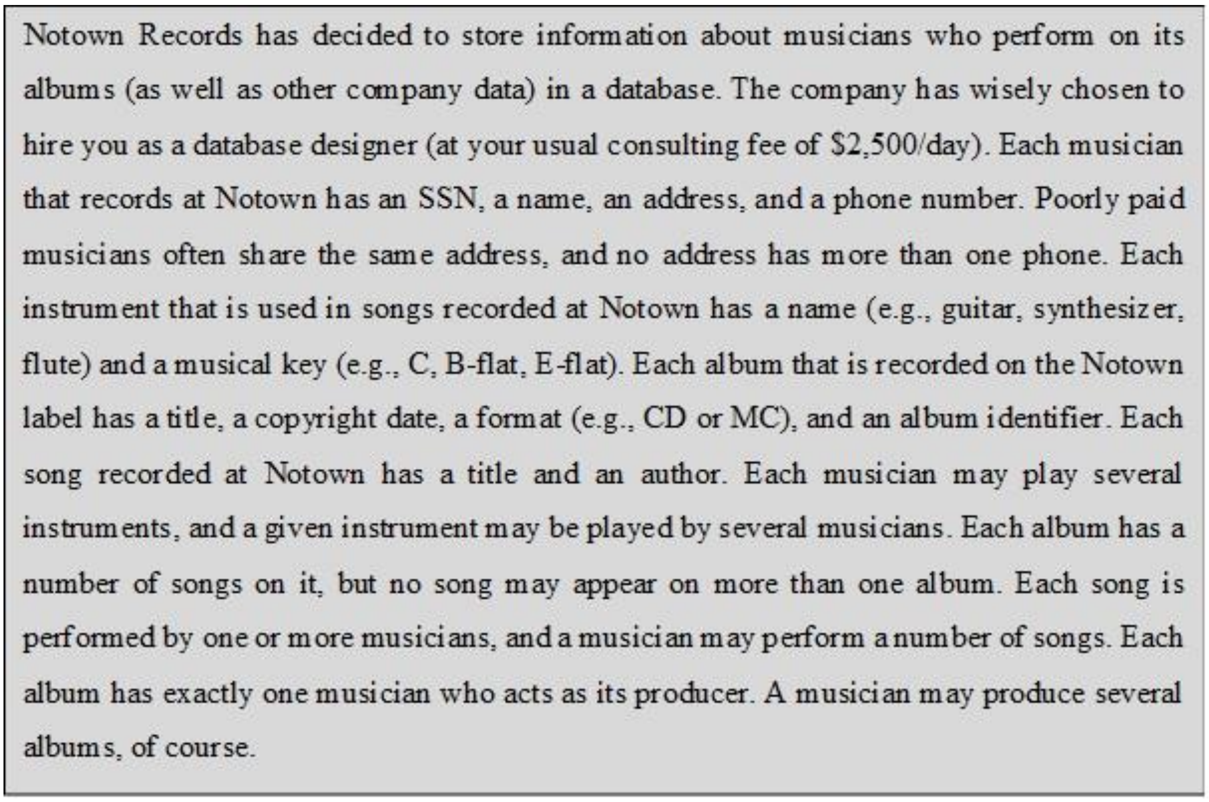
[24] https://www.shsu.edu/~csc_tjm/summer2000/cs334/Chapter04/part2/Chapter4b.html


For each match, we store the series and the day on which it takes place, which match it is (e.g first match, second match etc.) the date with day, month, year, the teams involved in the match with the name of the city for the team and the trainer, and finally for each team whether played at home. We store the name and the surname of each player in each team with his date of birth and main position. We store, for each day, how many points each team has and we also store for each match, the players of each team who played in and in which position each player played (the positions can change from one game to another). For each match, we store the referee, with first name, surname, city and region of birth. The matches played as scheduled must be distinguished from those postponed. For a postponed match, we store the date in which it is actually played. We also identify the matches played in a city other than that of the home team, for each of these, we store the city in which it took place, as well as the reason for the variation of venue. For each player, we are interested in the city of birth.

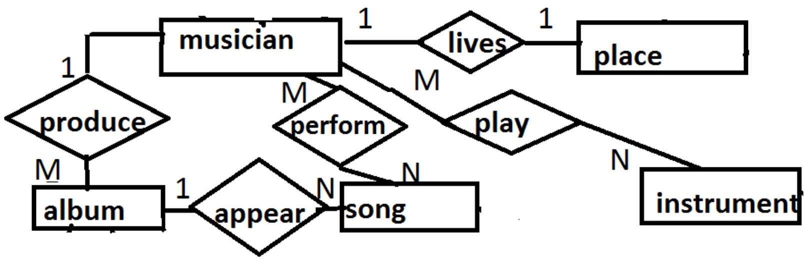
Appendix Figure 27 Problem Four in Harder Set (Atzeni, 1999, p. 213)



Appendix Figure 28 Solution for Problem Four in Harder Set provided by Database Designer[21]

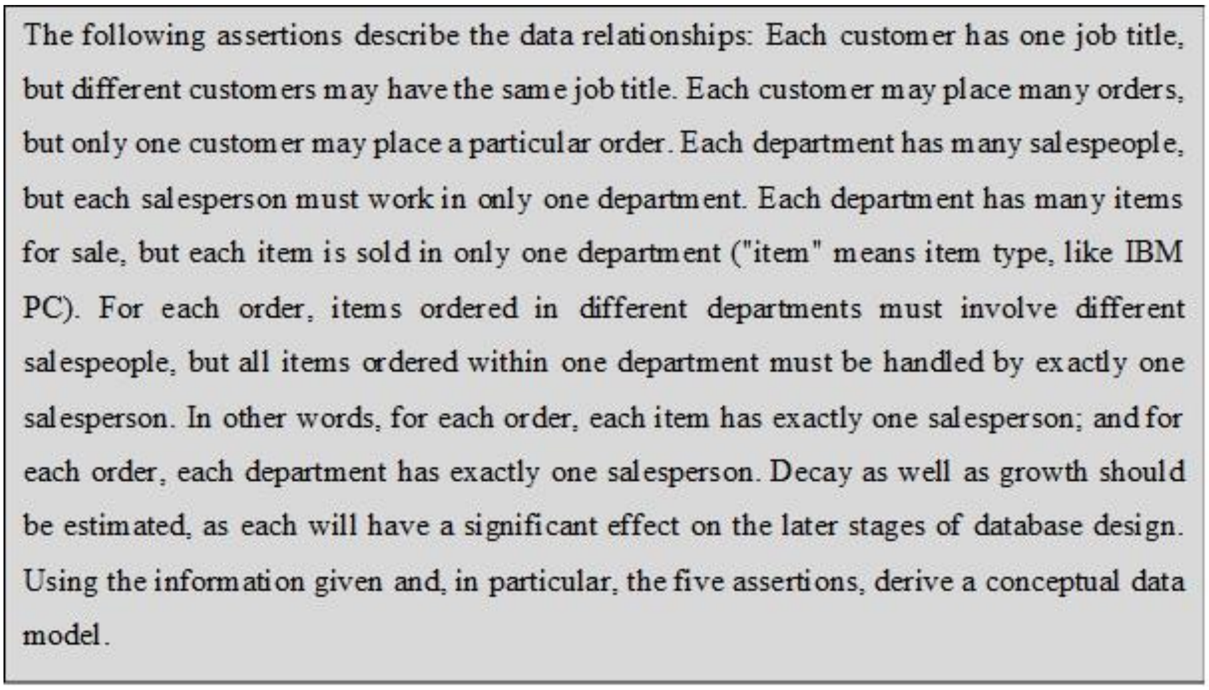

Notown Records has decided to store information about musicians who perform on its albums (as well as other company data) in a database. The company has wisely chosen to hire you as a database designer (at your usual consulting fee of $2,500/day). Each musician that records at Notown has an SSN, a name, an address, and a phone number. Poorly paid musicians often share the same address, and no address has more than one phone. Each instrument that is used in songs recorded at Notown has a name (e.g., guitar, synthesizer, flute) and a musical key (e.g., C, B-flat, E-flat). Each album that is recorded on the Notown label has a title, a copyright date, a format (e.g., CD or MC), and an album identifier. Each song recorded at Notown has a title and an author. Each musician may play several instruments, and a given instrument may be played by several musicians. Each album has a number of songs on it, but no song may appear on more than one album. Each song is performed by one or more musicians, and a musician may perform a number of songs. Each album has exactly one musician who acts as its producer. A musician may produce several albums, of course.

Appendix Figure 29 Problem Five in Harder Set (Gehrke, 2002, p. 8)



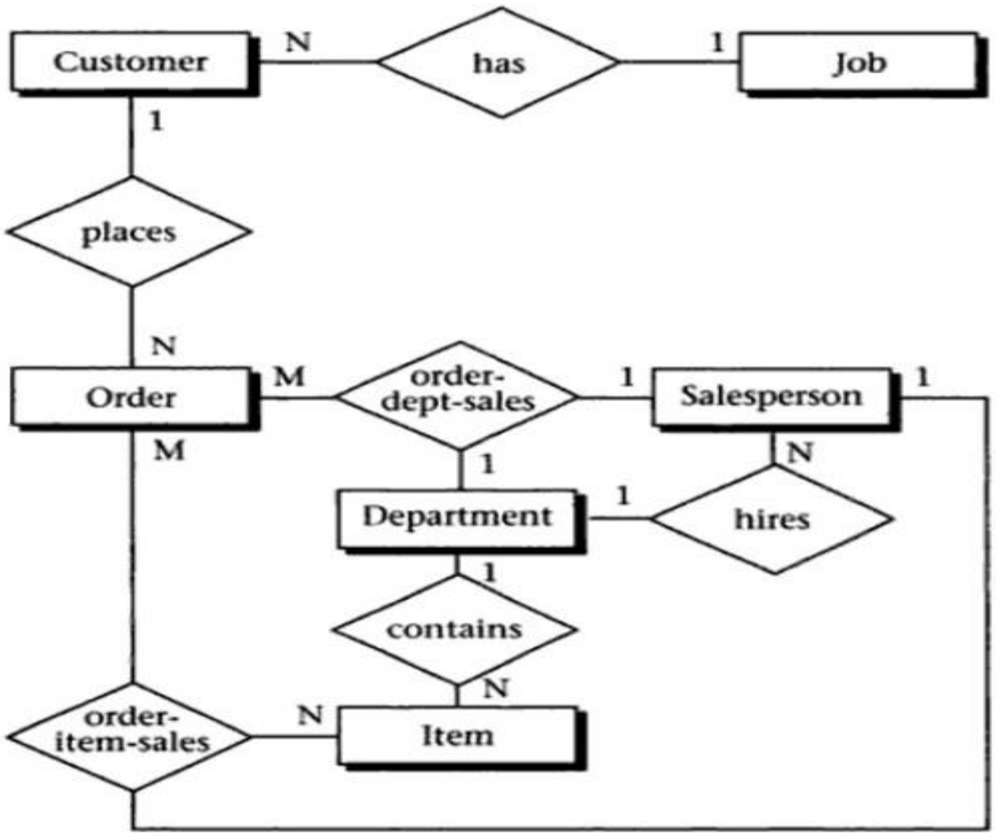
Appendix Figure 30 Solution for Problem Five in Harder Set[25]

[25] https://lbsitbytes2010.files.wordpress.com/2013/09/m11.png

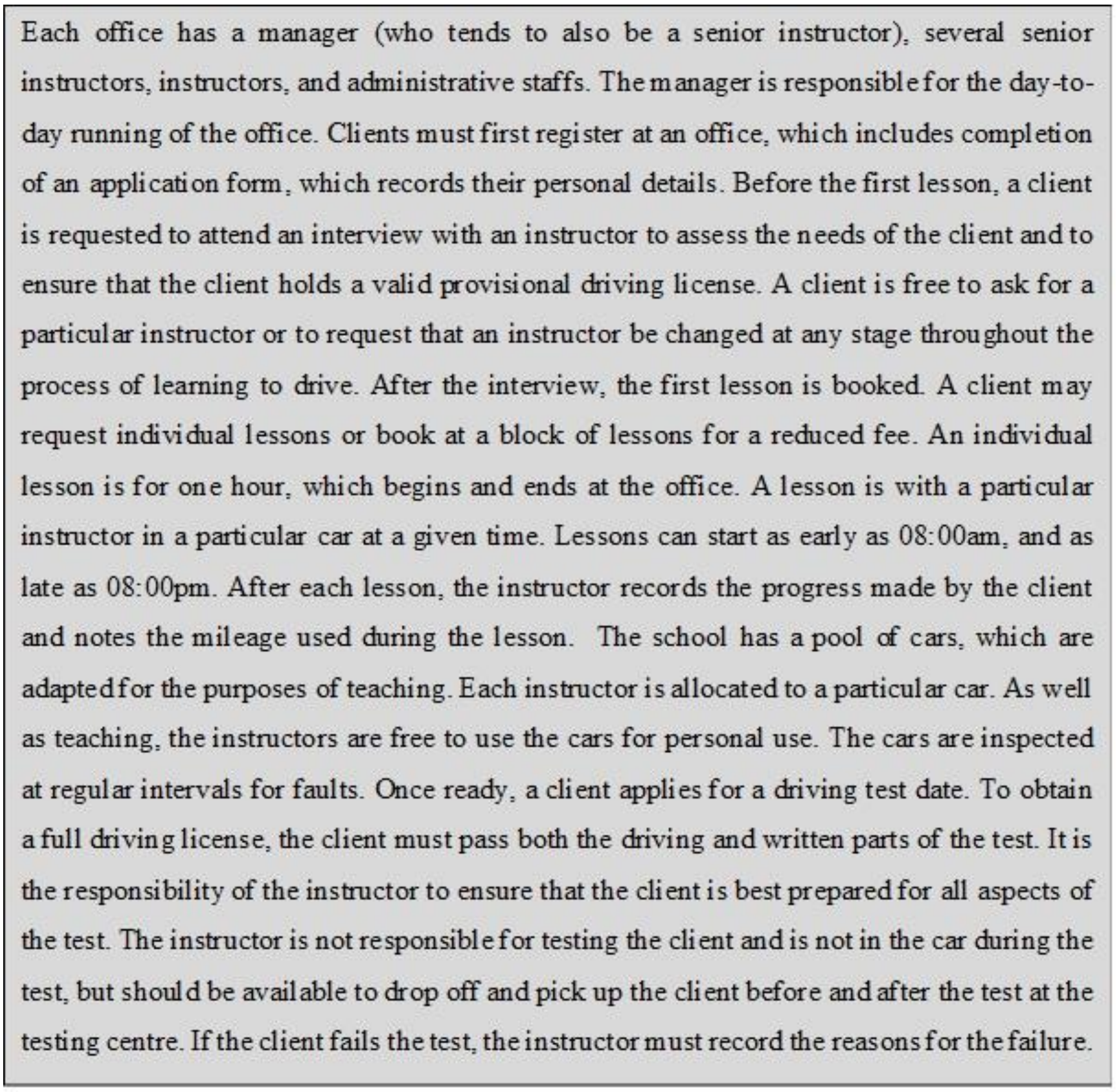


The following assertions describe the data relationships: Each customer has one job title, but different customers may have the same job title. Each customer may place many orders, but only one customer may place a particular order. Each department has many salespeople, but each salesperson must work in only one department. Each department has many items for sale, but each item is sold in only one department ("item" means item type, like IBM PC). For each order, items ordered in different departments must involve different salespeople, but all items ordered within one department must be handled by exactly one salesperson. In other words, for each order, each item has exactly one salesperson; and for each order, each department has exactly one salesperson. Decay as well as growth should be estimated, as each will have a significant effect on the later stages of database design. Using the information given and, in particular, the five assertions, derive a conceptual data model.

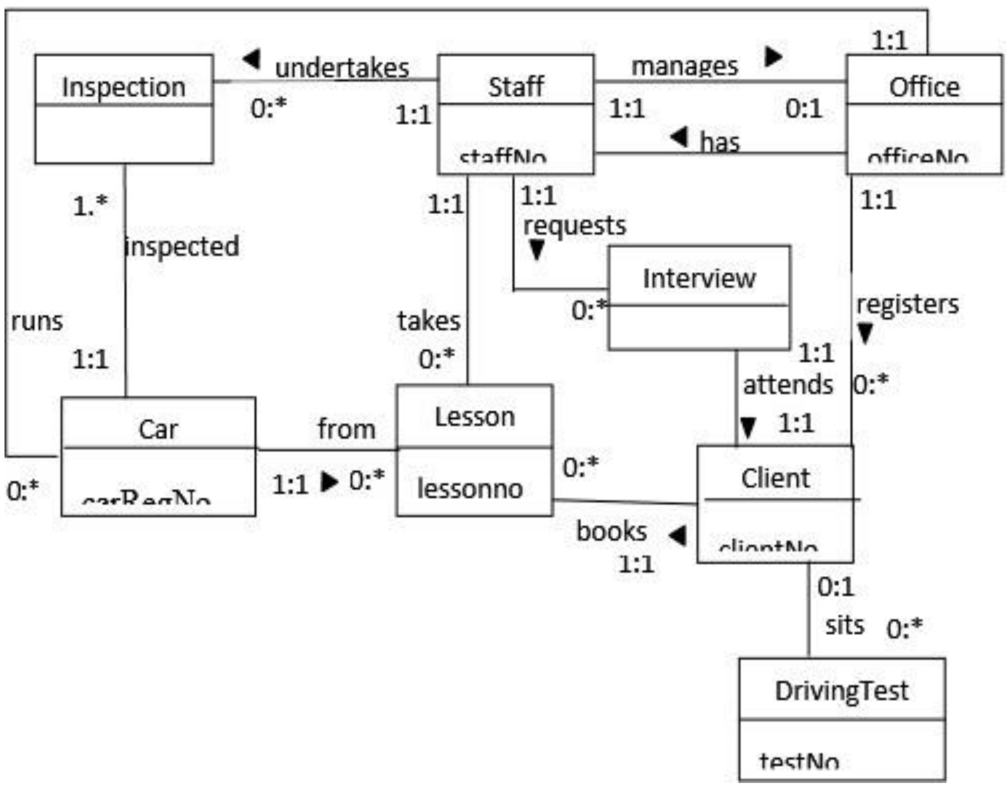
Appendix Figure 31 Problem Six in Harder Set (Teorey, Lightstone, Nadeau, & Jagadish, 2005, p. 131)



Appendix Figure 32 Solution for Problem Six in Harder Set (Teorey et al., 2005, p. 133)

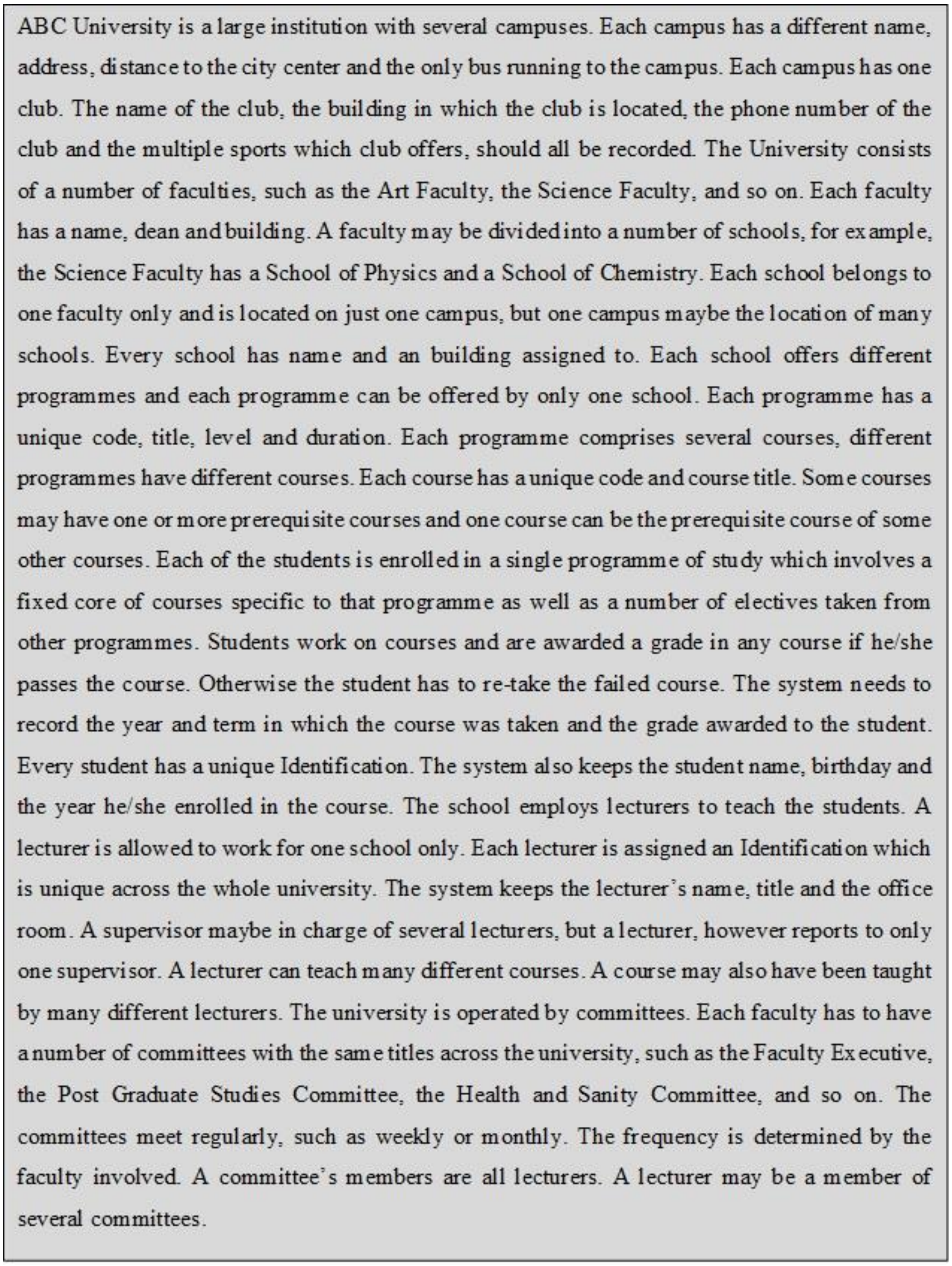

Each office has a manager (who tends to also be a senior instructor), several senior instructors, instructors, and administrative staffs. The manager is responsible for the day-to-day running of the office. Clients must first register at an office, which includes completion of an application form, which records their personal details. Before the first lesson, a client is requested to attend an interview with an instructor to assess the needs of the client and to ensure that the client holds a valid provisional driving license. A client is free to ask for a particular instructor or to request that an instructor be changed at any stage throughout the process of learning to drive. After the interview, the first lesson is booked. A client may request individual lessons or book at a block of lessons for a reduced fee. An individual lesson is for one hour, which begins and ends at the office. A lesson is with a particular instructor in a particular car at a given time. Lessons can start as early as 08:00am, and as late as 08:00pm. After each lesson, the instructor records the progress made by the client and notes the mileage used during the lesson. The school has a pool of cars, which are adapted for the purposes of teaching. Each instructor is allocated to a particular car. As well as teaching, the instructors are free to use the cars for personal use. The cars are inspected at regular intervals for faults. Once ready, a client applies for a driving test date. To obtain a full driving license, the client must pass both the driving and written parts of the test. It is the responsibility of the instructor to ensure that the client is best prepared for all aspects of the test. The instructor is not responsible for testing the client and is not in the car during the test, but should be available to drop off and pick up the client before and after the test at the testing centre. If the client fails the test, the instructor must record the reasons for the failure.

Appendix Figure 33 Problem Seven in Harder Set (Connolly & Begg, 2015, p. B-6)



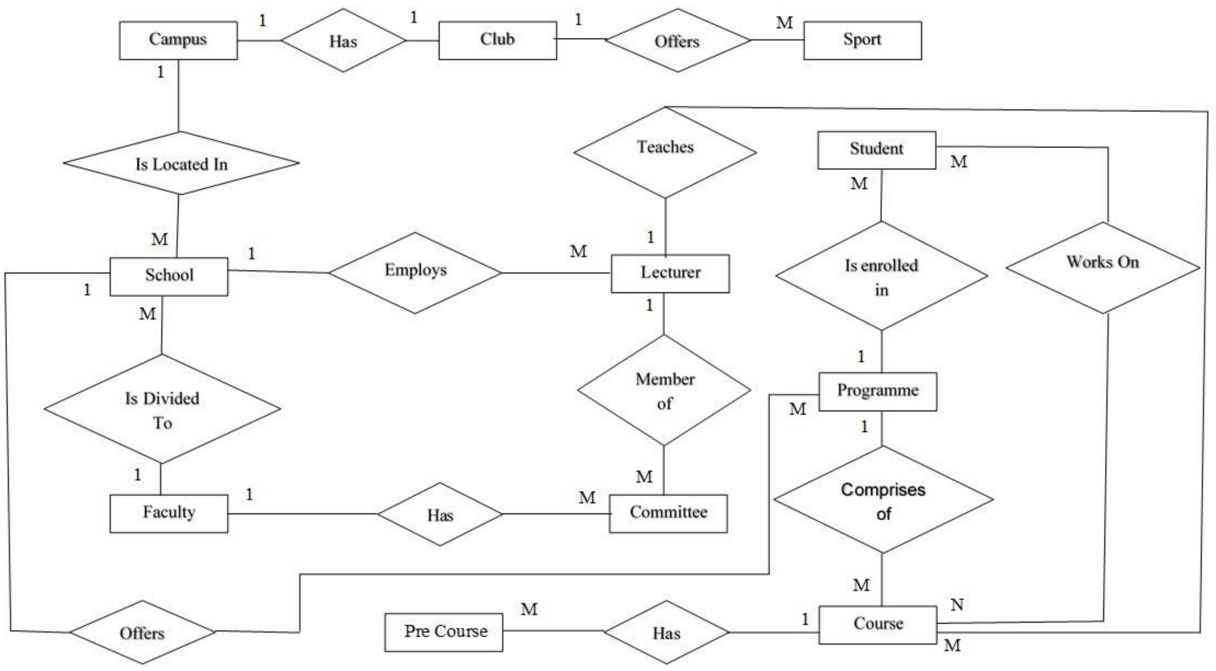
Appendix Figure 34 Solution for Problem Seven in Harder Set[26]

[26]https://www.google.co.uk/search?q=EasyDrive+School+of+Motoring+case+study&dcr=0&tbm=isch&tbo=u&source=univ&sa=X&ved=0ahUKEwiJ3WJ1v_ZAhWLIMAKHW_7BF0QsAQITw&biw=1239&bih=606#imgrc=7daotfWWnOEtFM:&spf=1521713153272

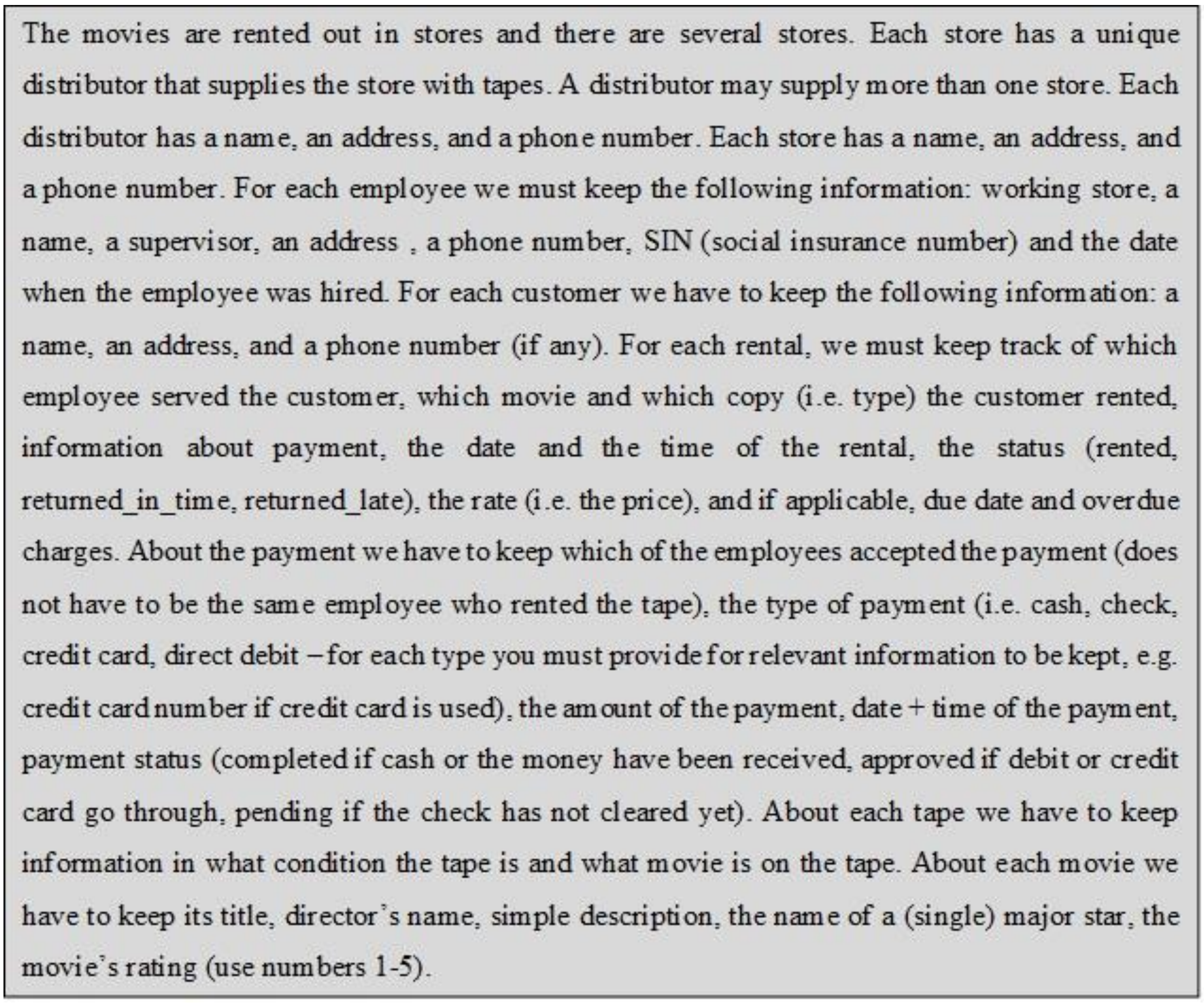


ABC University is a large institution with several campuses. Each campus has a different name, address, distance to the city center and the only bus running to the campus. Each campus has one club. The name of the club, the building in which the club is located, the phone number of the club and the multiple sports which club offers, should all be recorded. The University consists of a number of faculties, such as the Art Faculty, the Science Faculty, and so on. Each faculty has a name, dean and building. A faculty may be divided into a number of schools, for example, the Science Faculty has a School of Physics and a School of Chemistry. Each school belongs to one faculty only and is located on just one campus, but one campus maybe the location of many schools. Every school has name and an building assigned to. Each school offers different programmes and each programme can be offered by only one school. Each programme has a unique code, title, level and duration. Each programme comprises several courses, different programmes have different courses. Each course has a unique code and course title. Some courses may have one or more prerequisite courses and one course can be the prerequisite course of some other courses. Each of the students is enrolled in a single programme of study which involves a fixed core of courses specific to that programme as well as a number of electives taken from other programmes. Students work on courses and are awarded a grade in any course if he/she passes the course. Otherwise the student has to re-take the failed course. The system needs to record the year and term in which the course was taken and the grade awarded to the student. Every student has a unique Identification. The system also keeps the student name, birthday and the year he/she enrolled in the course. The school employs lecturers to teach the students. A lecturer is allowed to work for one school only. Each lecturer is assigned an Identification which is unique across the whole university. The system keeps the lecturer's name, title and the office room. A supervisor maybe in charge of several lecturers, but a lecturer, however reports to only one supervisor. A lecturer can teach many different courses. A course may also have been taught by many different lecturers. The university is operated by committees. Each faculty has to have a number of committees with the same titles across the university, such as the Faculty Executive, the Post Graduate Studies Committee, the Health and Sanity Committee, and so on. The committees meet regularly, such as weekly or monthly. The frequency is determined by the faculty involved. A committee's members are all lecturers. A lecturer may be a member of several committees.

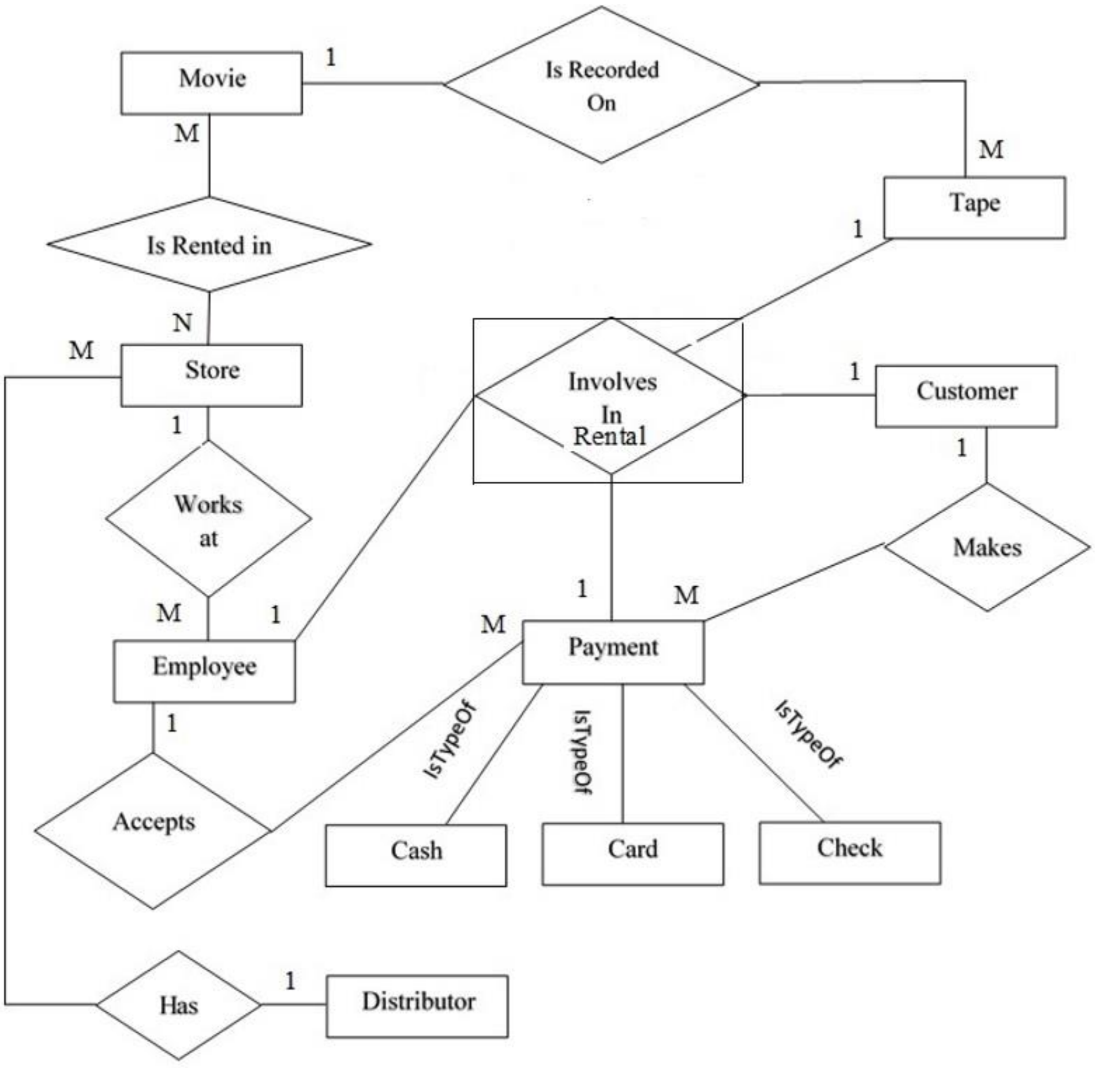
Appendix Figure 35 Problem Eight in Harder Set (Zhang, 2012, p. 8)



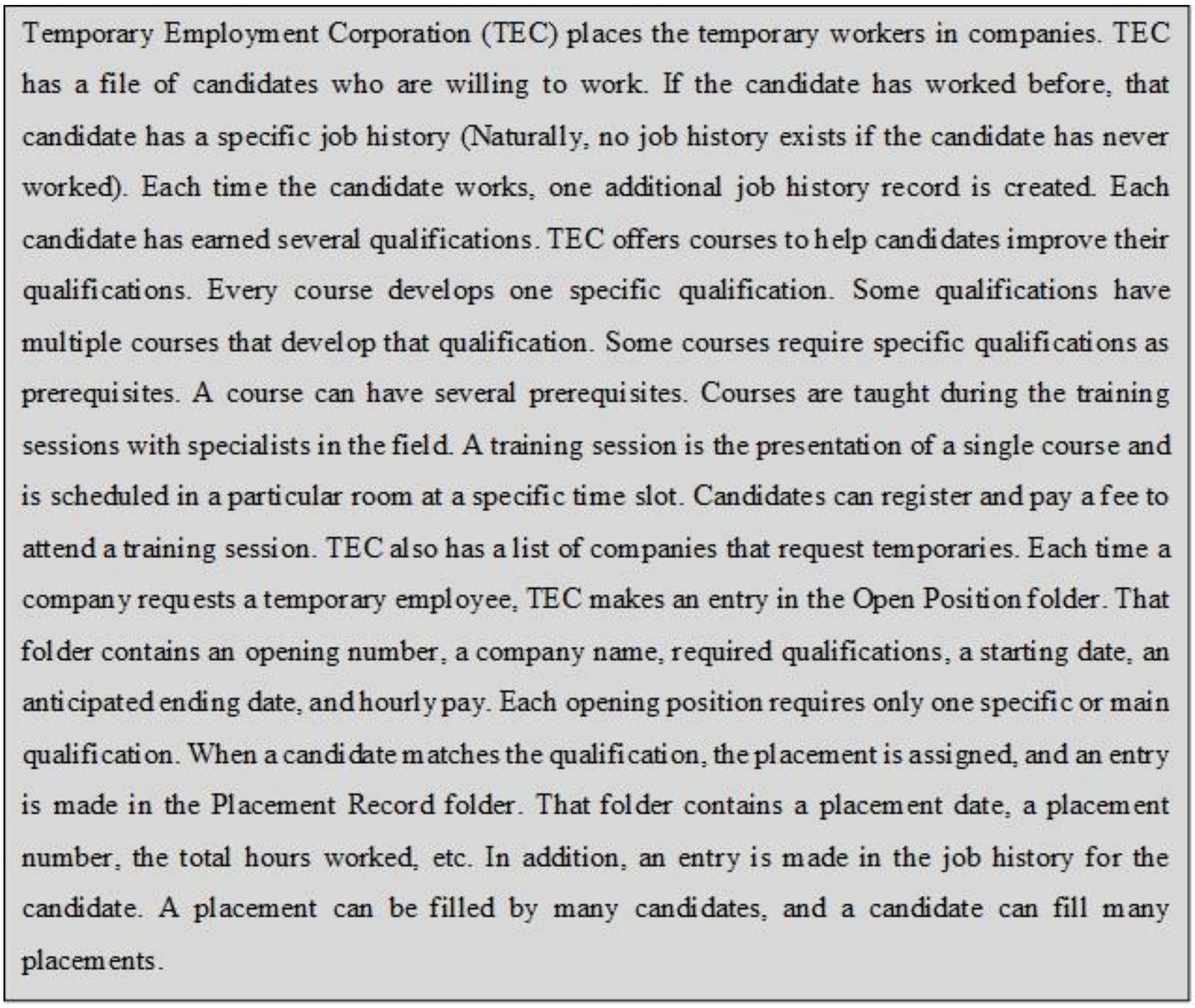
Appendix Figure 36 Solution for Problem Eight in Harder Set (Zhang, 2012, p. 10)


The movies are rented out in stores and there are several stores. Each store has a unique distributor that supplies the store with tapes. A distributor may supply more than one store. Each distributor has a name, an address, and a phone number. Each store has a name, an address, and a phone number. For each employee we must keep the following information: working store, a name, a supervisor, an address , a phone number, SIN (social insurance number) and the date when the employee was hired. For each customer we have to keep the following information: a name, an address, and a phone number (if any). For each rental, we must keep track of which employee served the customer, which movie and which copy (i.e. type) the customer rented, information about payment, the date and the time of the rental, the status (rented, returned_in_time, returned_late), the rate (i.e. the price), and if applicable, due date and overdue charges. About the payment we have to keep which of the employees accepted the payment (does not have to be the same employee who rented the tape), the type of payment (i.e. cash, check, credit card, direct debit – for each type you must provide for relevant information to be kept, e.g. credit card number if credit card is used), the amount of the payment, date + time of the payment, payment status (completed if cash or the money have been received, approved if debit or credit card go through, pending if the check has not cleared yet). About each tape we have to keep information in what condition the tape is and what movie is on the tape. About each movie we have to keep its title, director's name, simple description, the name of a (single) major star, the movie's rating (use numbers 1-5).

Appendix Figure 37 Problem Nine in Harder Set (Zhang, 2012, p. 34)

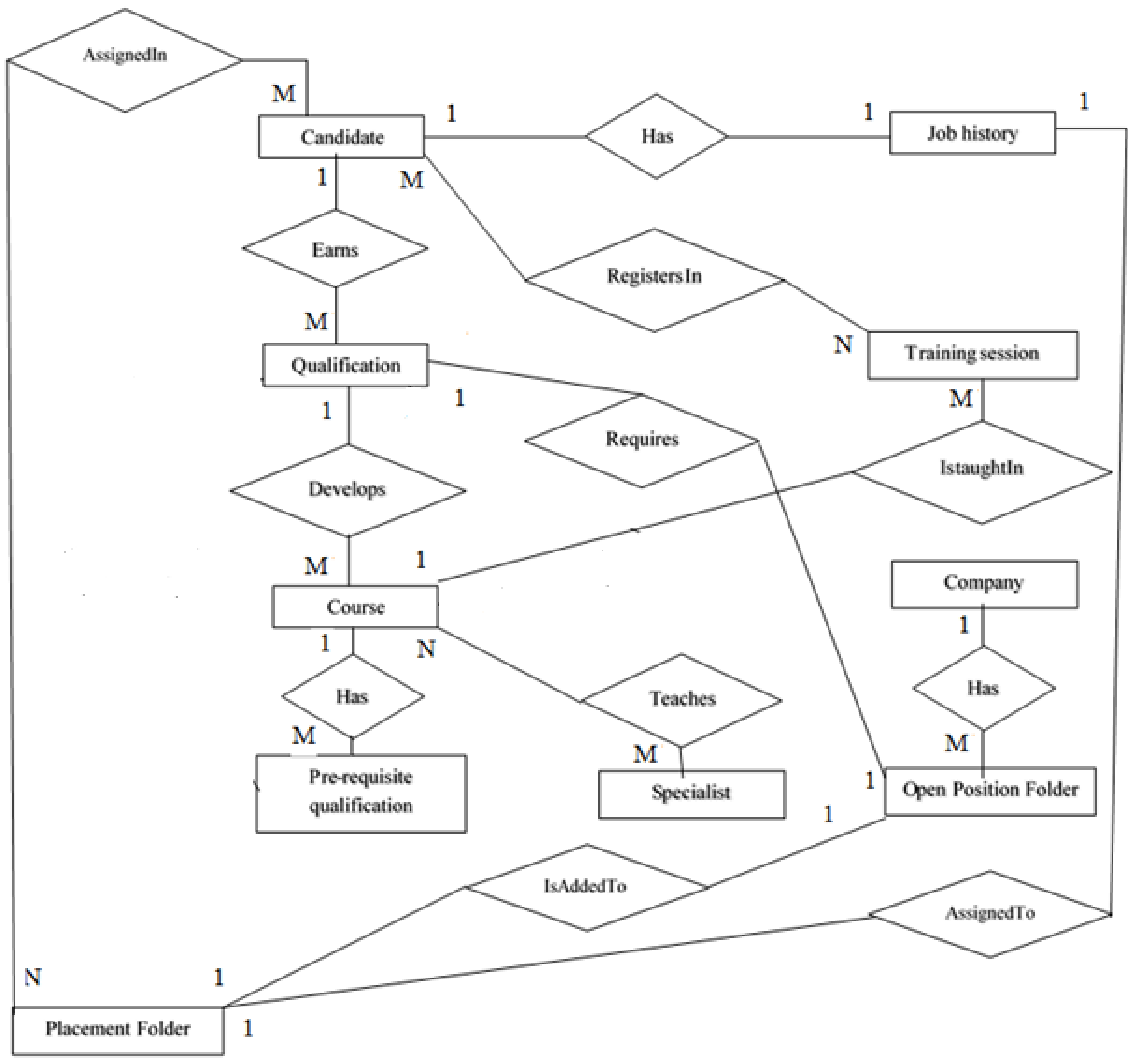


Appendix Figure 38 Solution for Problem Nine in Harder Set (Zhang, 2012, p. 35)



Temporary Employment Corporation (TEC) places the temporary workers in companies. TEC has a file of candidates who are willing to work. If the candidate has worked before, that candidate has a specific job history (Naturally, no job history exists if the candidate has never worked). Each time the candidate works, one additional job history record is created. Each candidate has earned several qualifications. TEC offers courses to help candidates improve their qualifications. Every course develops one specific qualification. Some qualifications have multiple courses that develop that qualification. Some courses require specific qualifications as prerequisites. A course can have several prerequisites. Courses are taught during the training sessions with specialists in the field. A training session is the presentation of a single course and is scheduled in a particular room at a specific time slot. Candidates can register and pay a fee to attend a training session. TEC also has a list of companies that request temporaries. Each time a company requests a temporary employee, TEC makes an entry in the Open Position folder. That folder contains an opening number, a company name, required qualifications, a starting date, an anticipated ending date, and hourly pay. Each opening position requires only one specific or main qualification. When a candidate matches the qualification, the placement is assigned, and an entry is made in the Placement Record folder. That folder contains a placement date, a placement number, the total hours worked, etc. In addition, an entry is made in the job history for the candidate. A placement can be filled by many candidates, and a candidate can fill many placements.

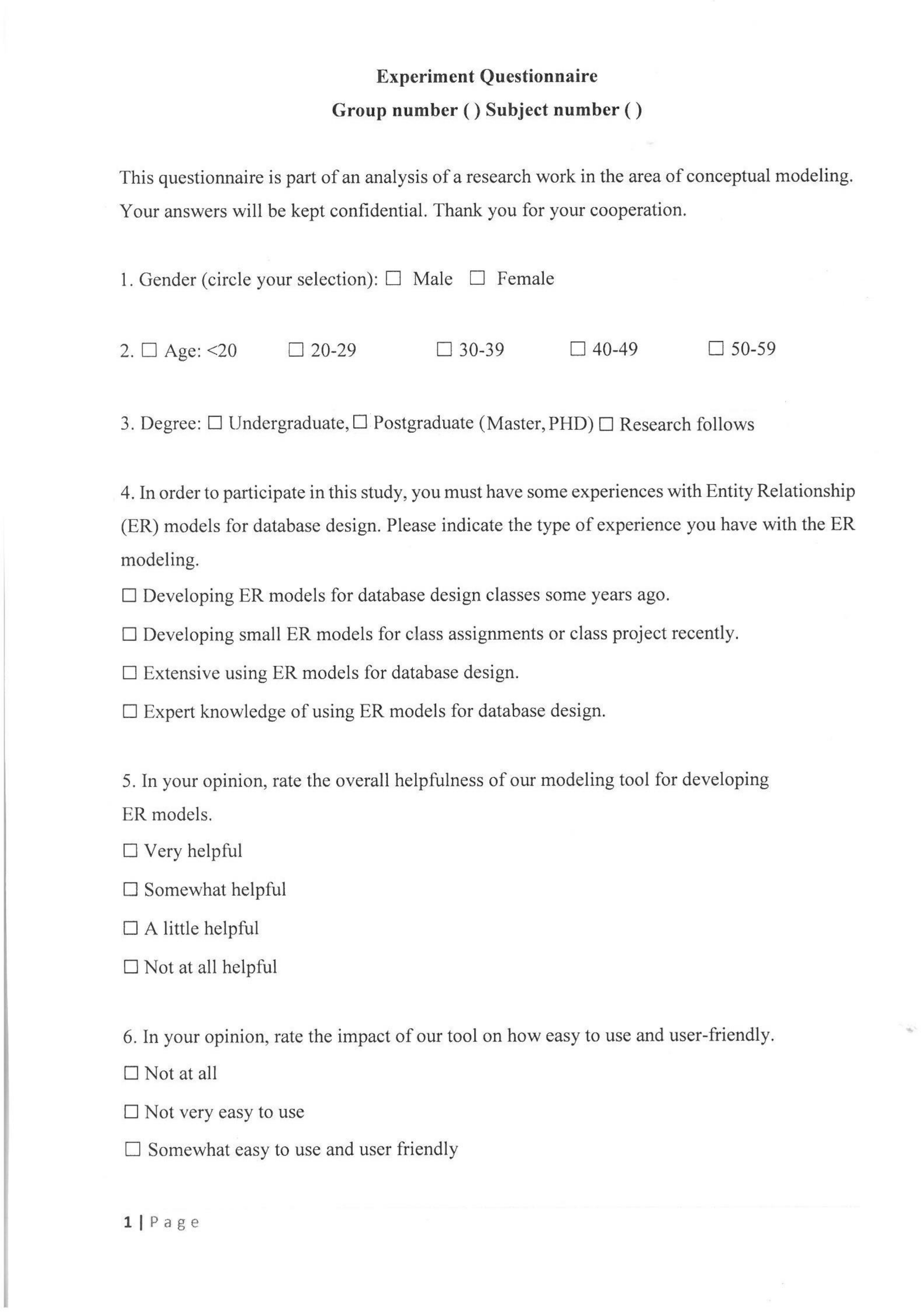
Appendix Figure 39 Problem Ten in Harder Set (Thonggoom, 2011, p. 132)



Appendix Figure 40 Solution for Problem Ten in Harder Set provided by Database Designer[21]

Appendix 3:

Questionnaire Form used in Experimental One



Experiment Questionnaire

Group number ( ) Subject number ( )

This questionnaire is part of an analysis of a research work in the area of conceptual modeling. Your answers will be kept confidential. Thank you for your cooperation.

1. Gender (circle your selection): ☐ Male ☐ Female

2. ☐ Age: <20 ☐ 20-29 ☐ 30-39 ☐ 40-49 ☐ 50-59

3. Degree: ☐ Undergraduate, ☐ Postgraduate (Master, PHD) ☐ Research follows

4. In order to participate in this study, you must have some experiences with Entity Relationship (ER) models for database design. Please indicate the type of experience you have with the ER modeling.

☐ Developing ER models for database design classes some years ago.

☐ Developing small ER models for class assignments or class project recently.

☐ Extensive using ER models for database design.

☐ Expert knowledge of using ER models for database design.

5. In your opinion, rate the overall helpfulness of our modeling tool for developing ER models.

☐ Very helpful

☐ Somewhat helpful

☐ A little helpful

☐ Not at all helpful

6. In your opinion, rate the impact of our tool on how easy to use and user-friendly.

☐ Not at all

☐ Not very easy to use

☐ Somewhat easy to use and user friendly

1 | Page

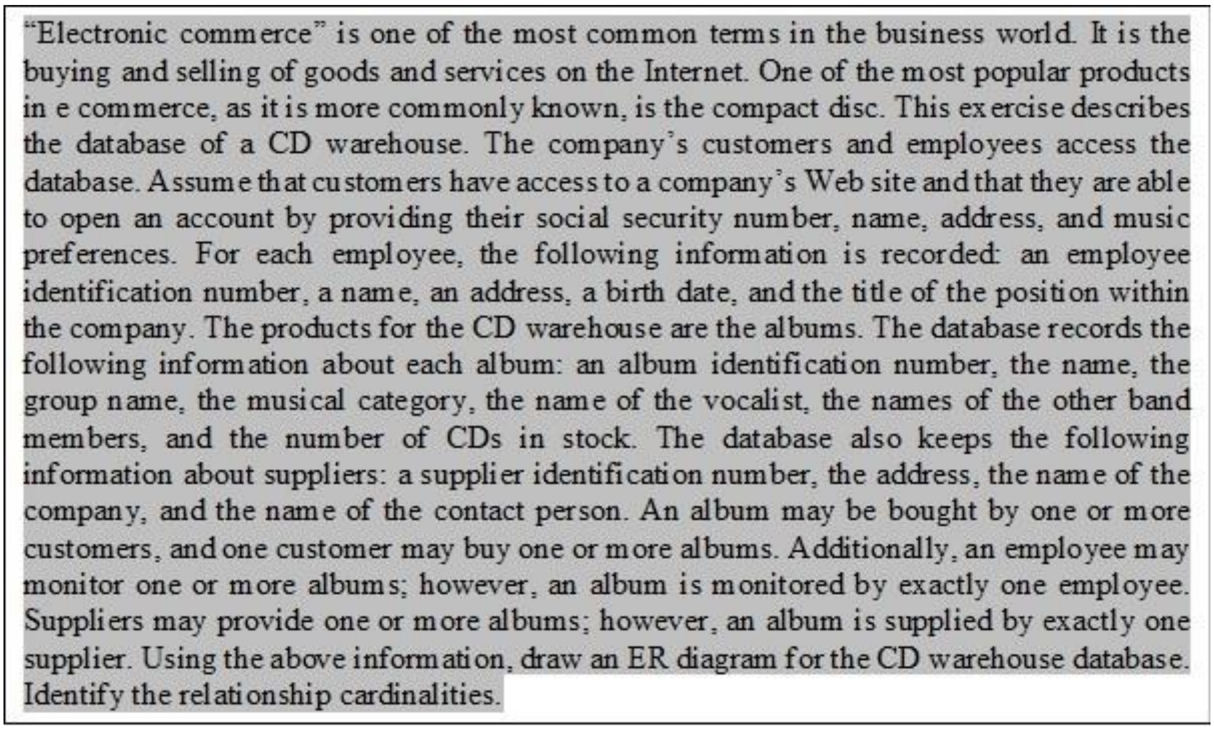
☐ Very easy to use and user friendly

7. What are the main advantages of using our tool?

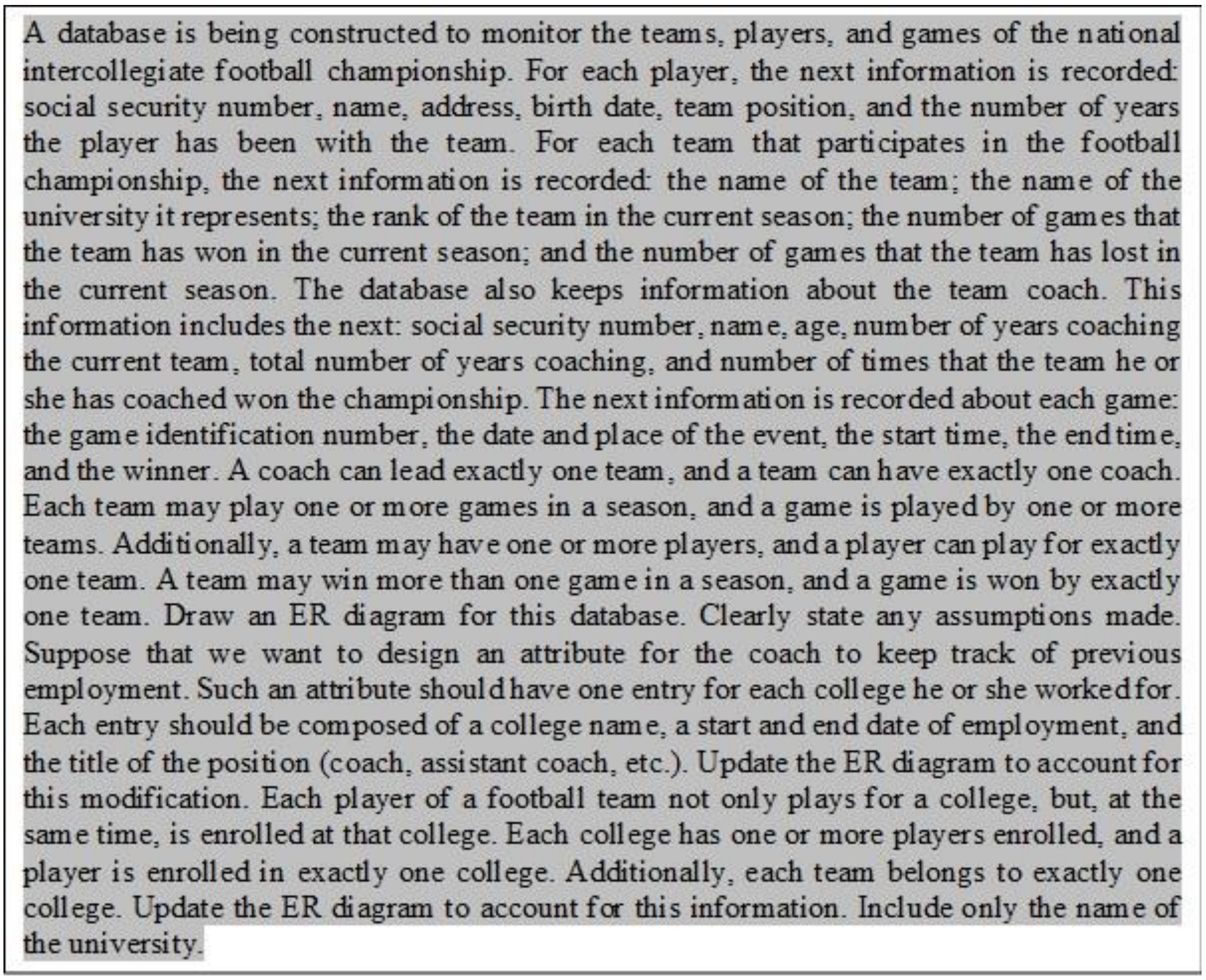
8. Identify areas where you believe that the tools could be improved.

Appendix 4:

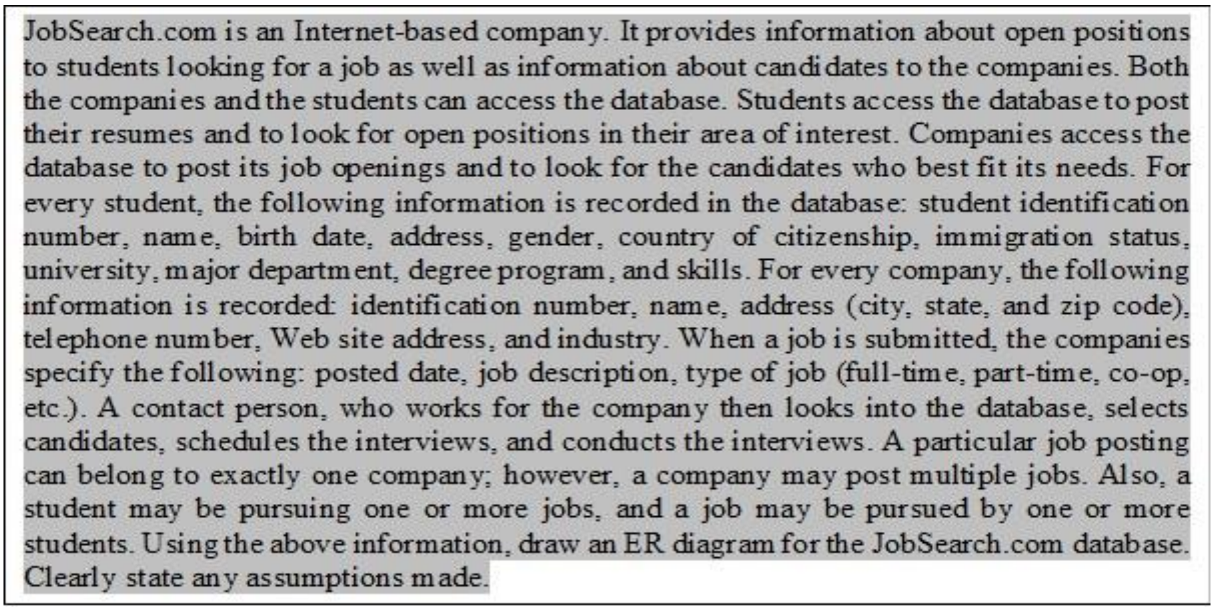
This appendix demonstrates the training set used for Experimental Two. In addition to seventeen out of the twenty case studies used in Experimental One, thirty-three case studies were added to the collection. This brought the total number of case studies used in the second experimental to fifty. Of the seventeen case studies reused from Experimental One, all ten of the harder set of case studies were included, but case numbers six, seven and ten from the easy set were omitted. The other thirty-three cases added for Experimental Two are listed as follows.


"Electronic commerce" is one of the most common terms in the business world. It is the buying and selling of goods and services on the Internet. One of the most popular products in e commerce, as it is more commonly known, is the compact disc. This exercise describes the database of a CD warehouse. The company's customers and employees access the database. Assume that customers have access to a company's Web site and that they are able to open an account by providing their social security number, name, address, and music preferences. For each employee, the following information is recorded: an employee identification number, a name, an address, a birth date, and the title of the position within the company. The products for the CD warehouse are the albums. The database records the following information about each album: an album identification number, the name, the group name, the musical category, the name of the vocalist, the names of the other band members, and the number of CDs in stock. The database also keeps the following information about suppliers: a supplier identification number, the address, the name of the company, and the name of the contact person. An album may be bought by one or more customers, and one customer may buy one or more albums. Additionally, an employee may monitor one or more albums; however, an album is monitored by exactly one employee. Suppliers may provide one or more albums; however, an album is supplied by exactly one supplier. Using the above information, draw an ER diagram for the CD warehouse database. Identify the relationship cardinalities.

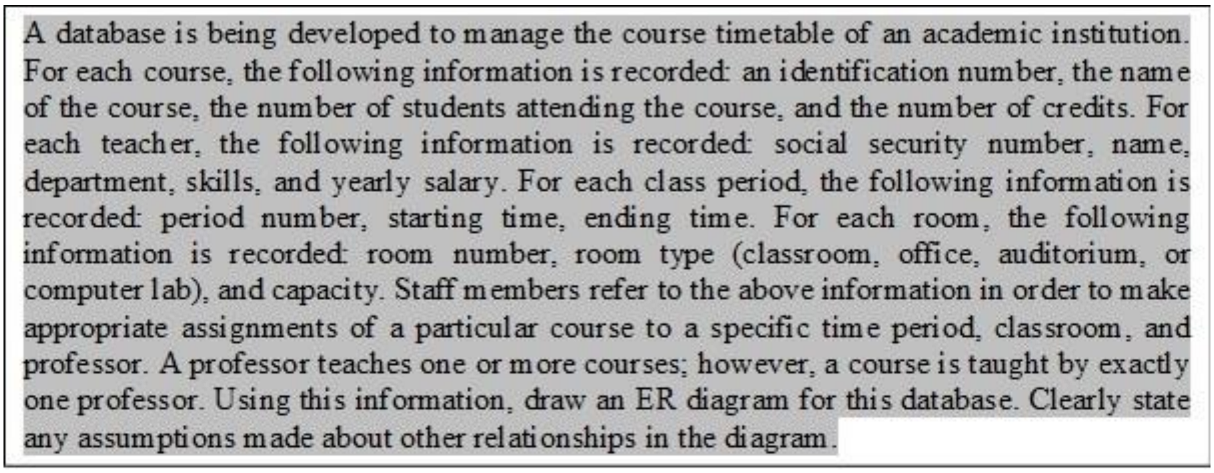
Appendix Figure 41 Electronic Commerce Case Study (Pol & Ahuja, 2007, p. 73)


A database is being constructed to monitor the teams, players, and games of the national intercollegiate football championship. For each player, the next information is recorded: social security number, name, address, birth date, team position, and the number of years the player has been with the team. For each team that participates in the football championship, the next information is recorded: the name of the team; the name of the university it represents; the rank of the team in the current season; the number of games that the team has won in the current season; and the number of games that the team has lost in the current season. The database also keeps information about the team coach. This information includes the next: social security number, name, age, number of years coaching the current team, total number of years coaching, and number of times that the team he or she has coached won the championship. The next information is recorded about each game: the game identification number, the date and place of the event, the start time, the end time, and the winner. A coach can lead exactly one team, and a team can have exactly one coach. Each team may play one or more games in a season, and a game is played by one or more teams. Additionally, a team may have one or more players, and a player can play for exactly one team. A team may win more than one game in a season, and a game is won by exactly one team. Draw an ER diagram for this database. Clearly state any assumptions made. Suppose that we want to design an attribute for the coach to keep track of previous employment. Such an attribute should have one entry for each college he or she worked for. Each entry should be composed of a college name, a start and end date of employment, and the title of the position (coach, assistant coach, etc.). Update the ER diagram to account for this modification. Each player of a football team not only plays for a college, but, at the same time, is enrolled at that college. Each college has one or more players enrolled, and a player is enrolled in exactly one college. Additionally, each team belongs to exactly one college. Update the ER diagram to account for this information. Include only the name of the university.

Appendix Figure 42 Intercollegiate Football Championship Case Study (Pol & Ahuja, 2007, p. 74)

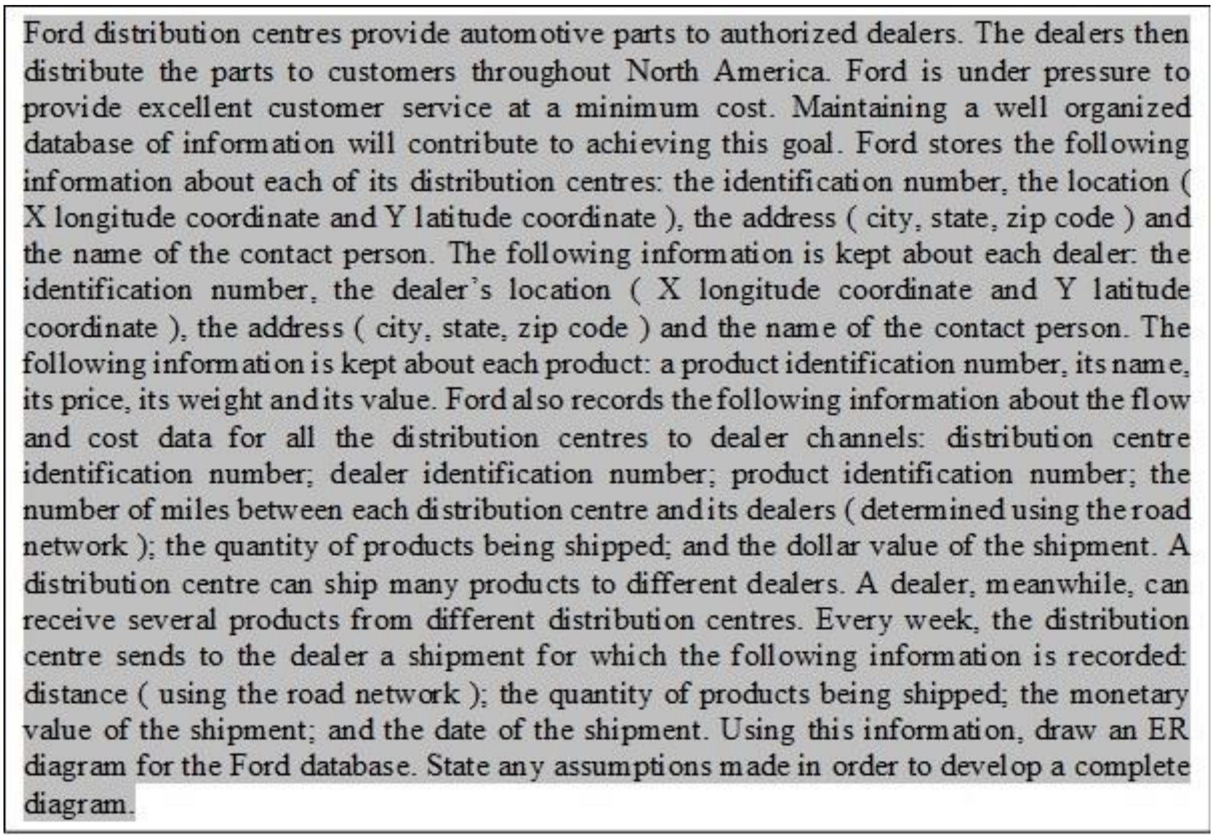

JobSearch.com is an Internet-based company. It provides information about open positions to students looking for a job as well as information about candidates to the companies. Both the companies and the students can access the database. Students access the database to post their resumes and to look for open positions in their area of interest. Companies access the database to post its job openings and to look for the candidates who best fit its needs. For every student, the following information is recorded in the database: student identification number, name, birth date, address, gender, country of citizenship, immigration status, university, major department, degree program, and skills. For every company, the following information is recorded: identification number, name, address (city, state, and zip code), telephone number, Web site address, and industry. When a job is submitted, the companies specify the following: posted date, job description, type of job (full-time, part-time, co-op, etc.). A contact person, who works for the company then looks into the database, selects candidates, schedules the interviews, and conducts the interviews. A particular job posting can belong to exactly one company; however, a company may post multiple jobs. Also, a student may be pursuing one or more jobs, and a job may be pursued by one or more students. Using the above information, draw an ER diagram for the JobSearch.com database. Clearly state any assumptions made.

Appendix Figure 43 JobSearch Case Study (Pol & Ahuja, 2007, p. 75)



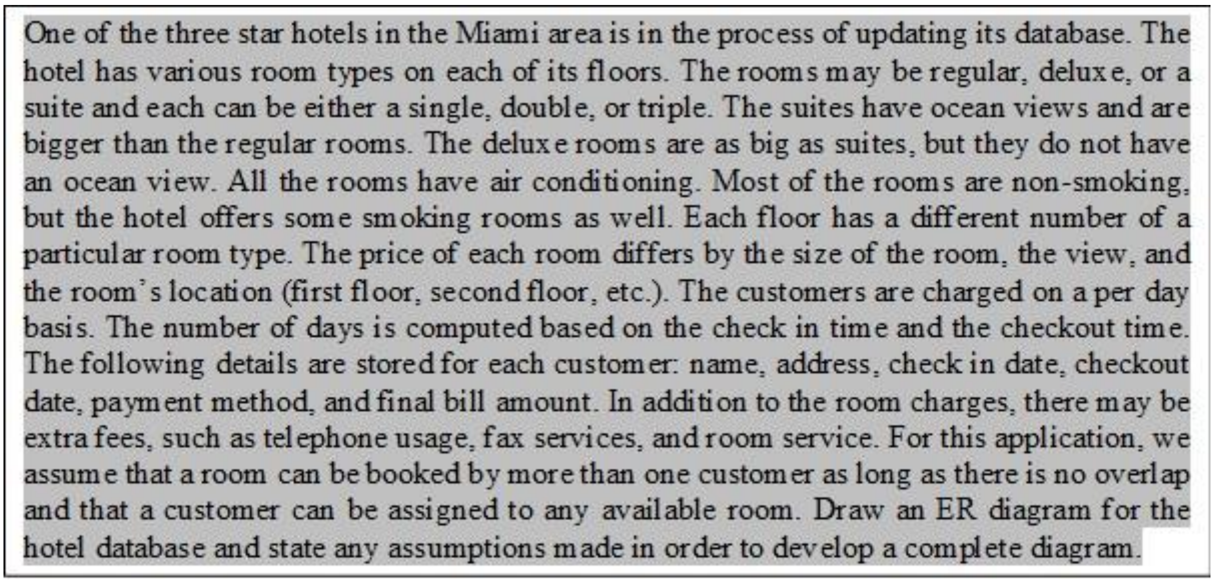
A database is being developed to manage the course timetable of an academic institution. For each course, the following information is recorded: an identification number, the name of the course, the number of students attending the course, and the number of credits. For each teacher, the following information is recorded: social security number, name, department, skills, and yearly salary. For each class period, the following information is recorded: period number, starting time, ending time. For each room, the following information is recorded: room number, room type (classroom, office, auditorium, or computer lab), and capacity. Staff members refer to the above information in order to make appropriate assignments of a particular course to a specific time period, classroom, and professor. A professor teaches one or more courses; however, a course is taught by exactly one professor. Using this information, draw an ER diagram for this database. Clearly state any assumptions made about other relationships in the diagram.

Appendix Figure 44 Course Timetable Case study (Pol & Ahuja, 2007, p. 74)

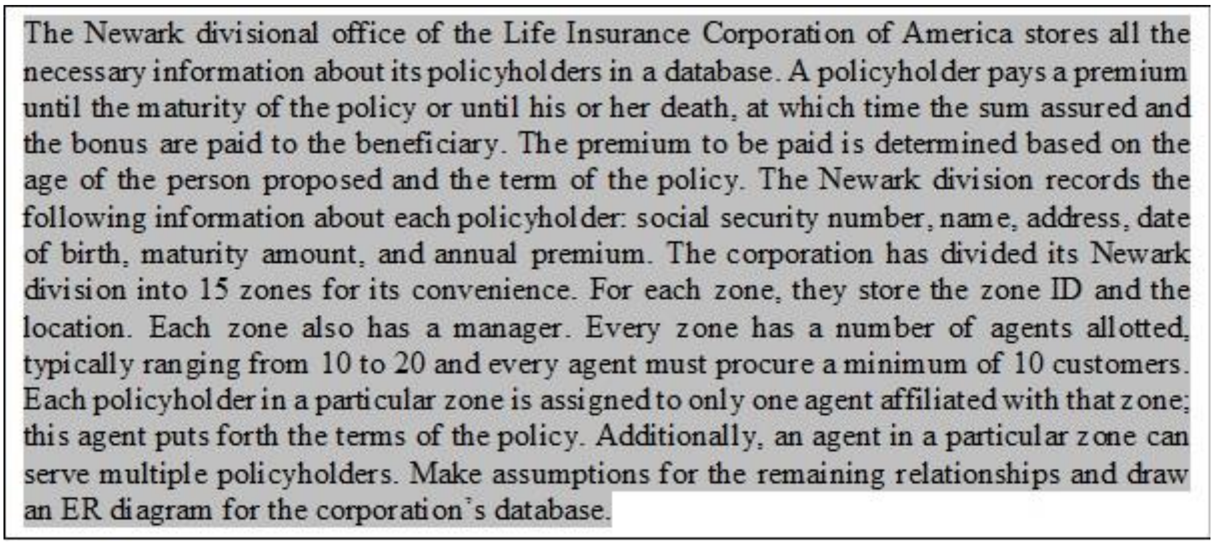


Ford distribution centres provide automotive parts to authorized dealers. The dealers then distribute the parts to customers throughout North America. Ford is under pressure to provide excellent customer service at a minimum cost. Maintaining a well organized database of information will contribute to achieving this goal. Ford stores the following information about each of its distribution centres: the identification number, the location (X longitude coordinate and Y latitude coordinate), the address (city, state, zip code) and the name of the contact person. The following information is kept about each dealer: the identification number, the dealer's location (X longitude coordinate and Y latitude coordinate), the address (city, state, zip code) and the name of the contact person. The following information is kept about each product: a product identification number, its name, its price, its weight and its value. Ford also records the following information about the flow and cost data for all the distribution centres to dealer channels: distribution centre identification number; dealer identification number; product identification number; the number of miles between each distribution centre and its dealers (determined using the road network); the quantity of products being shipped; and the dollar value of the shipment. A distribution centre can ship many products to different dealers. A dealer, meanwhile, can receive several products from different distribution centres. Every week, the distribution centre sends to the dealer a shipment for which the following information is recorded: distance (using the road network); the quantity of products being shipped; the monetary value of the shipment; and the date of the shipment. Using this information, draw an ER diagram for the Ford database. State any assumptions made in order to develop a complete diagram.

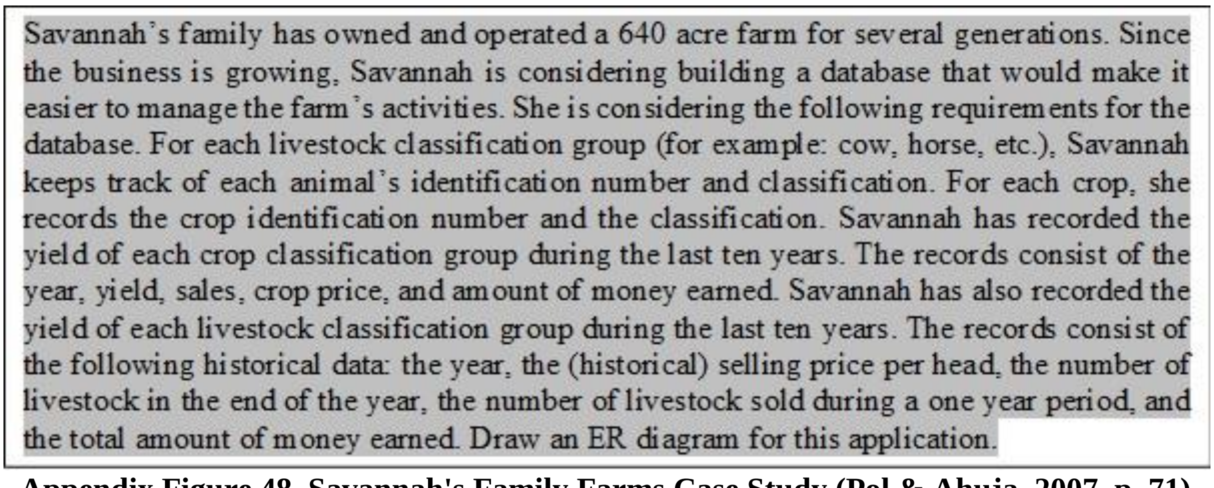
Appendix Figure 45 Ford Distribution Centres Case study (Pol & Ahuja, 2007, p. 73)


One of the three star hotels in the Miami area is in the process of updating its database. The hotel has various room types on each of its floors. The rooms may be regular, deluxe, or a suite and each can be either a single, double, or triple. The suites have ocean views and are bigger than the regular rooms. The deluxe rooms are as big as suites, but they do not have an ocean view. All the rooms have air conditioning. Most of the rooms are non-smoking, but the hotel offers some smoking rooms as well. Each floor has a different number of a particular room type. The price of each room differs by the size of the room, the view, and the room's location (first floor, second floor, etc.). The customers are charged on a per day basis. The number of days is computed based on the check in time and the checkout time. The following details are stored for each customer: name, address, check in date, checkout date, payment method, and final bill amount. In addition to the room charges, there may be extra fees, such as telephone usage, fax services, and room service. For this application, we assume that a room can be booked by more than one customer as long as there is no overlap and that a customer can be assigned to any available room. Draw an ER diagram for the hotel database and state any assumptions made in order to develop a complete diagram.

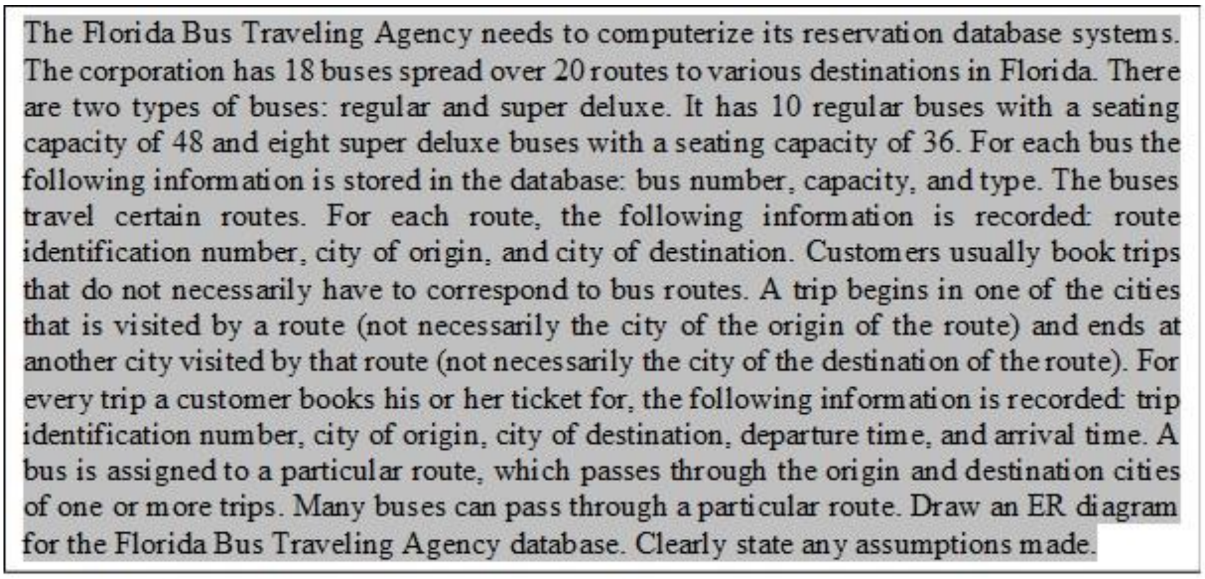
Appendix Figure 46 Miami Hotel Case Study (Pol & Ahuja, 2007, p. 73)


The Newark divisional office of the Life Insurance Corporation of America stores all the necessary information about its policyholders in a database. A policyholder pays a premium until the maturity of the policy or until his or her death, at which time the sum assured and the bonus are paid to the beneficiary. The premium to be paid is determined based on the age of the person proposed and the term of the policy. The Newark division records the following information about each policyholder: social security number, name, address, date of birth, maturity amount, and annual premium. The corporation has divided its Newark division into 15 zones for its convenience. For each zone, they store the zone ID and the location. Each zone also has a manager. Every zone has a number of agents allotted, typically ranging from 10 to 20 and every agent must procure a minimum of 10 customers. Each policyholder in a particular zone is assigned to only one agent affiliated with that zone; this agent puts forth the terms of the policy. Additionally, an agent in a particular zone can serve multiple policyholders. Make assumptions for the remaining relationships and draw an ER diagram for the corporation's database.

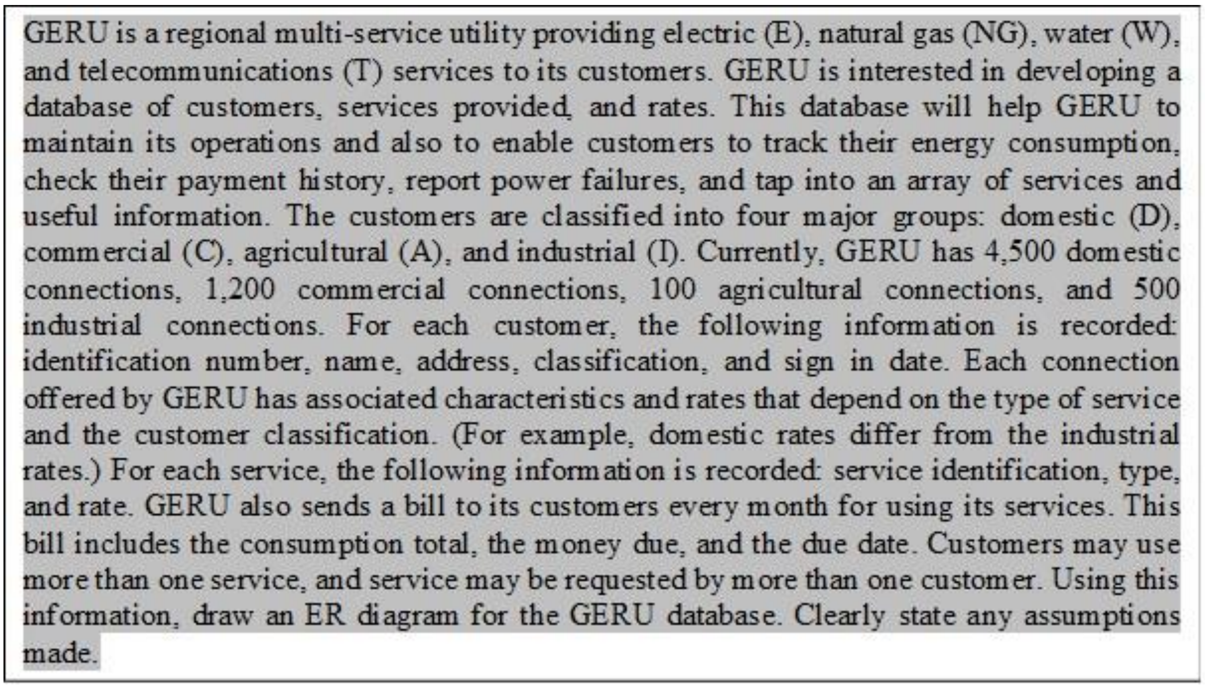
Appendix Figure 47 Newark Divisional Office Case Study (Pol & Ahuja, 2007, p. 73)


Savannah's family has owned and operated a 640 acre farm for several generations. Since the business is growing, Savannah is considering building a database that would make it easier to manage the farm's activities. She is considering the following requirements for the database. For each livestock classification group (for example: cow, horse, etc.), Savannah keeps track of each animal's identification number and classification. For each crop, she records the crop identification number and the classification. Savannah has recorded the yield of each crop classification group during the last ten years. The records consist of the year, yield, sales, crop price, and amount of money earned. Savannah has also recorded the yield of each livestock classification group during the last ten years. The records consist of the following historical data: the year, the (historical) selling price per head, the number of livestock in the end of the year, the number of livestock sold during a one year period, and the total amount of money earned. Draw an ER diagram for this application.

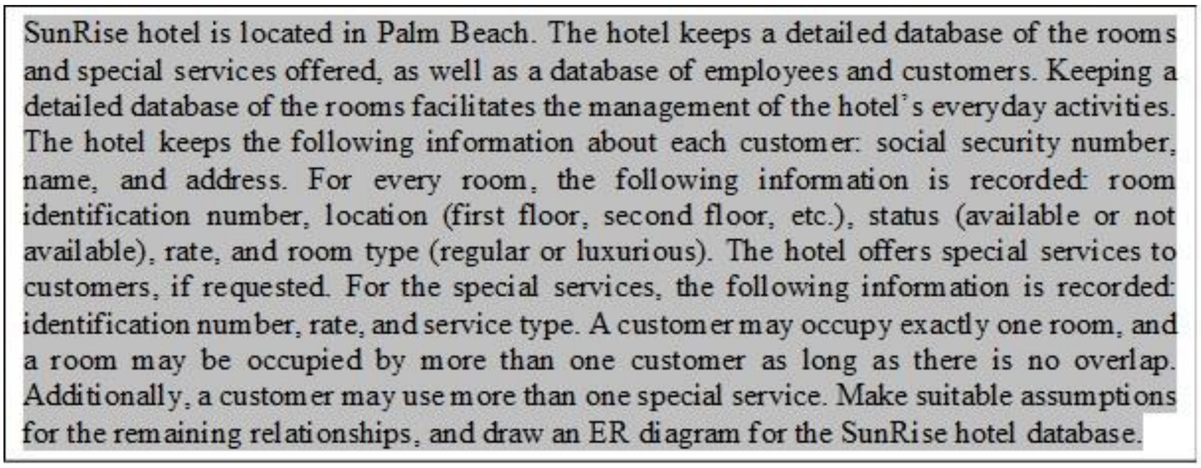
Appendix Figure 48 Savannah's Family Farms Case Study (Pol & Ahuja, 2007, p. 71)


The Florida Bus Traveling Agency needs to computerize its reservation database systems. The corporation has 18 buses spread over 20 routes to various destinations in Florida. There are two types of buses: regular and super deluxe. It has 10 regular buses with a seating capacity of 48 and eight super deluxe buses with a seating capacity of 36. For each bus the following information is stored in the database: bus number, capacity, and type. The buses travel certain routes. For each route, the following information is recorded: route identification number, city of origin, and city of destination. Customers usually book trips that do not necessarily have to correspond to bus routes. A trip begins in one of the cities that is visited by a route (not necessarily the city of the origin of the route) and ends at another city visited by that route (not necessarily the city of the destination of the route). For every trip a customer books his or her ticket for, the following information is recorded: trip identification number, city of origin, city of destination, departure time, and arrival time. A bus is assigned to a particular route, which passes through the origin and destination cities of one or more trips. Many buses can pass through a particular route. Draw an ER diagram for the Florida Bus Traveling Agency database. Clearly state any assumptions made.

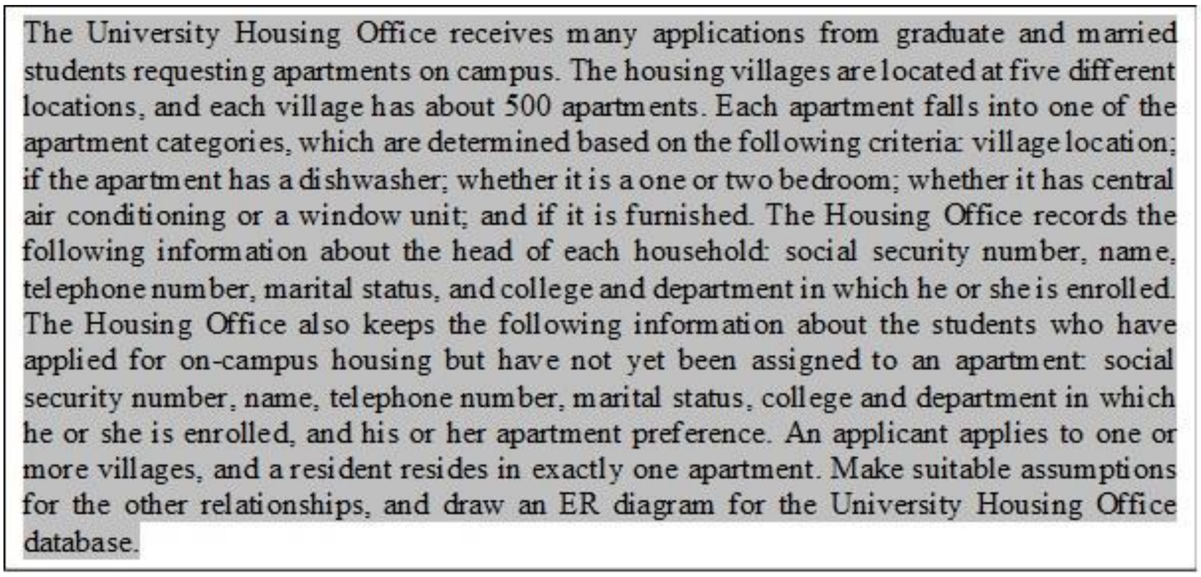
Appendix Figure 49 Florida Bus Traveling Agency Case Study (Pol & Ahuja, 2007, p.75)


GERU is a regional multi-service utility providing electric (E), natural gas (NG), water (W), and telecommunications (T) services to its customers. GERU is interested in developing a database of customers, services provided, and rates. This database will help GERU to maintain its operations and also to enable customers to track their energy consumption, check their payment history, report power failures, and tap into an array of services and useful information. The customers are classified into four major groups: domestic (D), commercial (C), agricultural (A), and industrial (I). Currently, GERU has 4,500 domestic connections, 1,200 commercial connections, 100 agricultural connections, and 500 industrial connections. For each customer, the following information is recorded: identification number, name, address, classification, and sign in date. Each connection offered by GERU has associated characteristics and rates that depend on the type of service and the customer classification. (For example, domestic rates differ from the industrial rates.) For each service, the following information is recorded: service identification, type, and rate. GERU also sends a bill to its customers every month for using its services. This bill includes the consumption total, the money due, and the due date. Customers may use more than one service, and service may be requested by more than one customer. Using this information, draw an ER diagram for the GERU database. Clearly state any assumptions made.

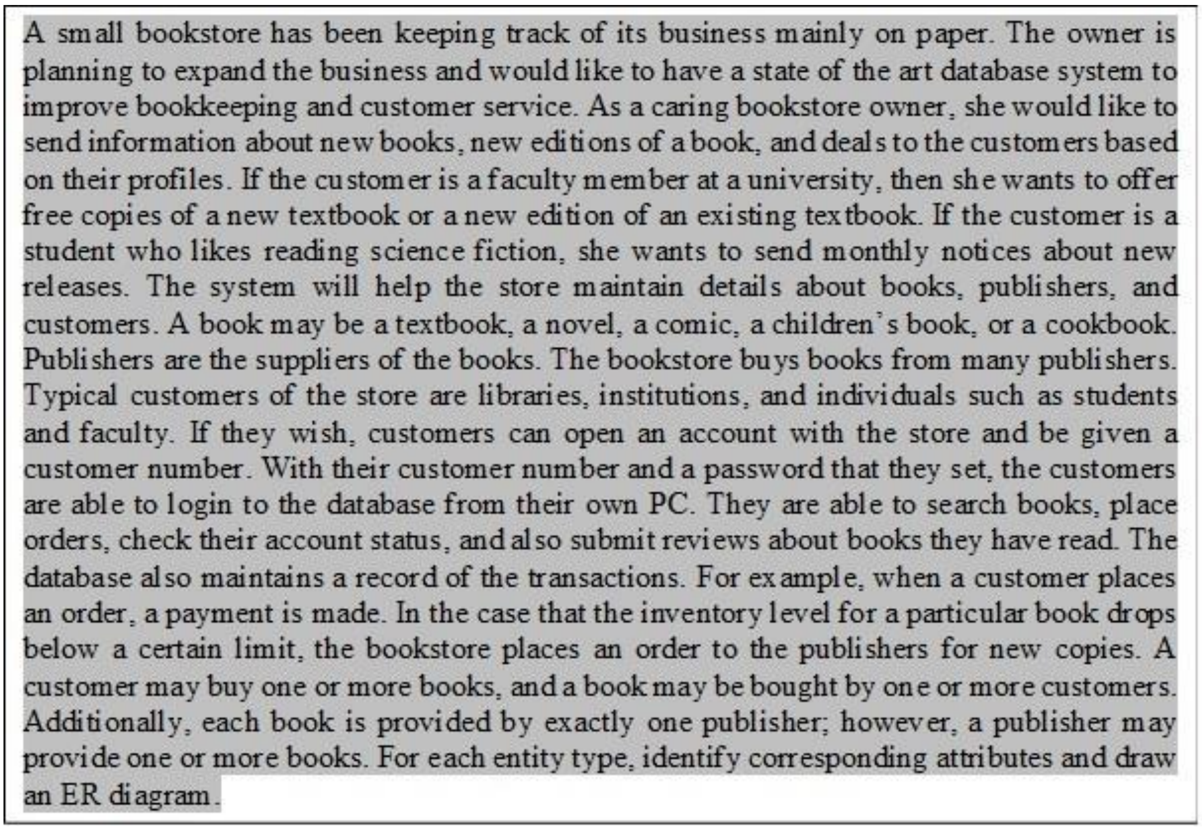
Appendix Figure 50 GERU Company Case Study (Pol & Ahuja, 2007, p. 76)


SunRise hotel is located in Palm Beach. The hotel keeps a detailed database of the rooms and special services offered, as well as a database of employees and customers. Keeping a detailed database of the rooms facilitates the management of the hotel's everyday activities. The hotel keeps the following information about each customer: social security number, name, and address. For every room, the following information is recorded: room identification number, location (first floor, second floor, etc.), status (available or not available), rate, and room type (regular or luxurious). The hotel offers special services to customers, if requested. For the special services, the following information is recorded: identification number, rate, and service type. A customer may occupy exactly one room, and a room may be occupied by more than one customer as long as there is no overlap. Additionally, a customer may use more than one special service. Make suitable assumptions for the remaining relationships, and draw an ER diagram for the SunRise hotel database.

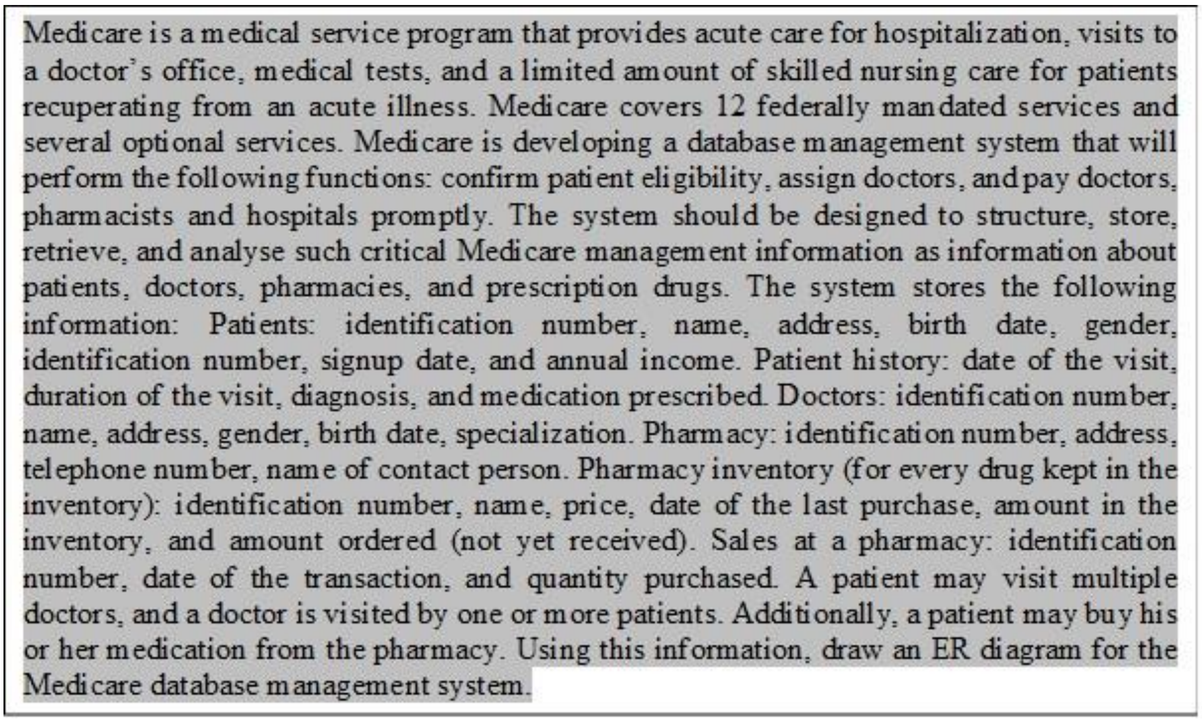
Appendix Figure 51 SunRise Hotel Case study (Pol & Ahuja, 2007, p. 76)


The University Housing Office receives many applications from graduate and married students requesting apartments on campus. The housing villages are located at five different locations, and each village has about 500 apartments. Each apartment falls into one of the apartment categories, which are determined based on the following criteria: village location; if the apartment has a dishwasher; whether it is a one or two bedroom; whether it has central air conditioning or a window unit; and if it is furnished. The Housing Office records the following information about the head of each household: social security number, name, telephone number, marital status, and college and department in which he or she is enrolled. The Housing Office also keeps the following information about the students who have applied for on-campus housing but have not yet been assigned to an apartment: social security number, name, telephone number, marital status, college and department in which he or she is enrolled, and his or her apartment preference. An applicant applies to one or more villages, and a resident resides in exactly one apartment. Make suitable assumptions for the other relationships, and draw an ER diagram for the University Housing Office database.

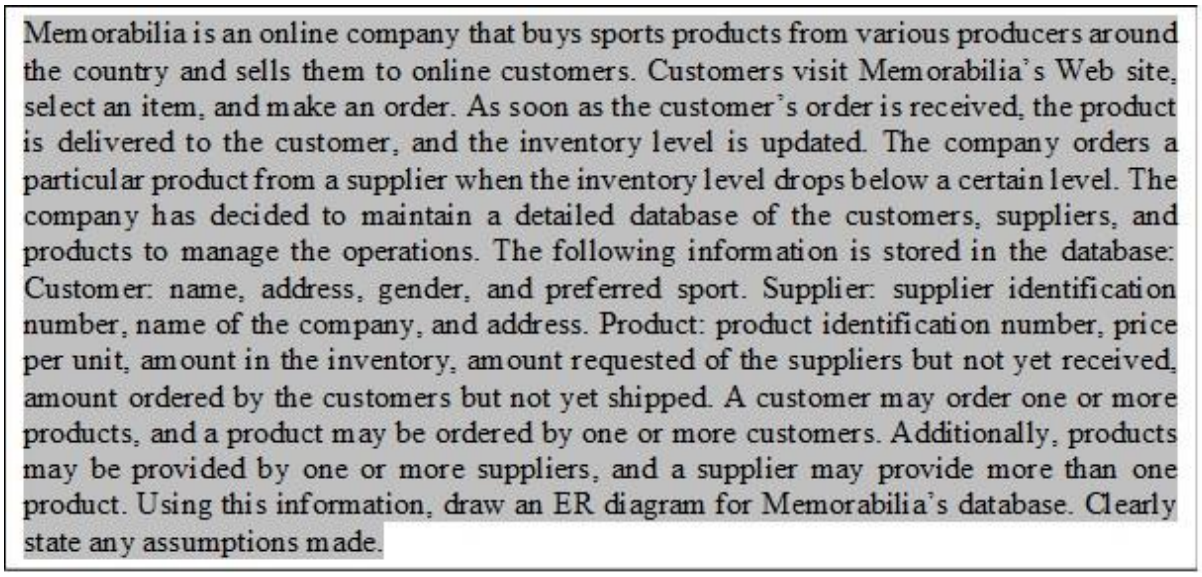
Appendix Figure 52 University Housing Office Case Study (Pol & Ahuja, 2007, p. 74)


A small bookstore has been keeping track of its business mainly on paper. The owner is planning to expand the business and would like to have a state of the art database system to improve bookkeeping and customer service. As a caring bookstore owner, she would like to send information about new books, new editions of a book, and deals to the customers based on their profiles. If the customer is a faculty member at a university, then she wants to offer free copies of a new textbook or a new edition of an existing textbook. If the customer is a student who likes reading science fiction, she wants to send monthly notices about new releases. The system will help the store maintain details about books, publishers, and customers. A book may be a textbook, a novel, a comic, a children's book, or a cookbook. Publishers are the suppliers of the books. The bookstore buys books from many publishers. Typical customers of the store are libraries, institutions, and individuals such as students and faculty. If they wish, customers can open an account with the store and be given a customer number. With their customer number and a password that they set, the customers are able to login to the database from their own PC. They are able to search books, place orders, check their account status, and also submit reviews about books they have read. The database also maintains a record of the transactions. For example, when a customer places an order, a payment is made. In the case that the inventory level for a particular book drops below a certain limit, the bookstore places an order to the publishers for new copies. A customer may buy one or more books, and a book may be bought by one or more customers. Additionally, each book is provided by exactly one publisher; however, a publisher may provide one or more books. For each entity type, identify corresponding attributes and draw an ER diagram.

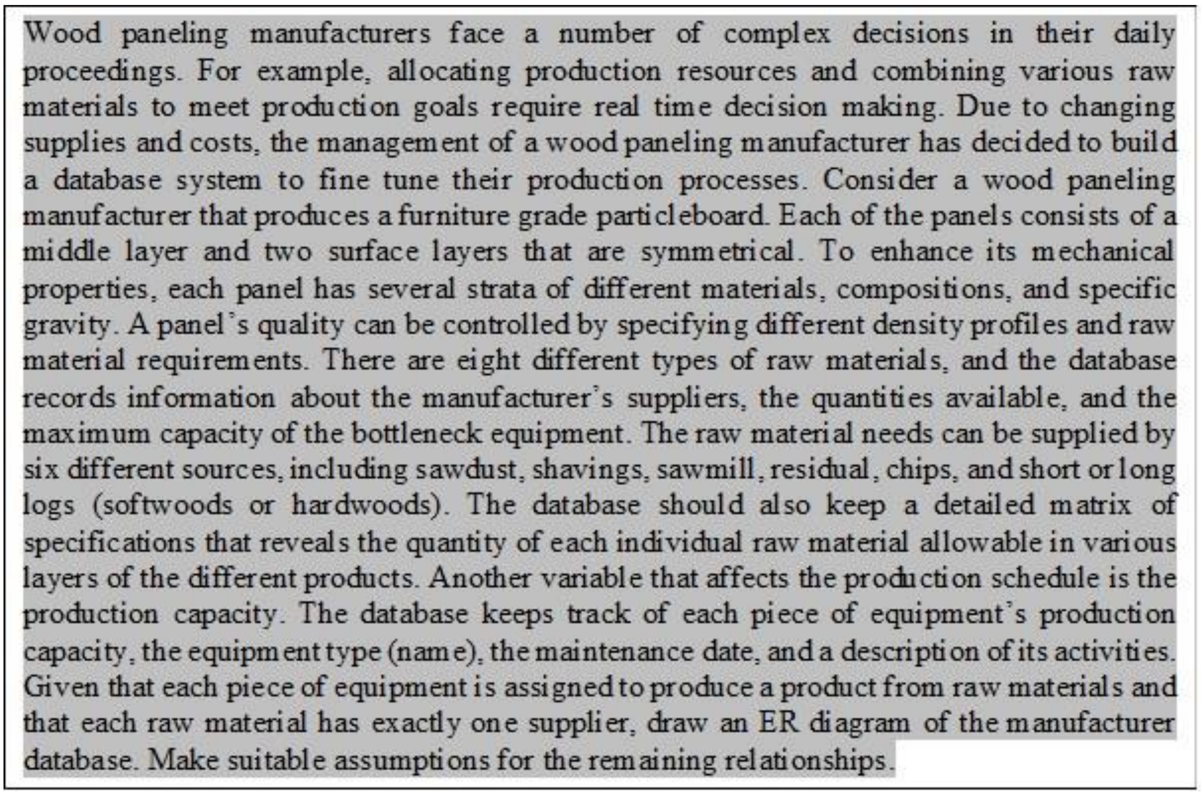
Appendix Figure 53 Bookstore Case Study (Pol & Ahuja, 2007, p. 77)


Medicare is a medical service program that provides acute care for hospitalization, visits to a doctor's office, medical tests, and a limited amount of skilled nursing care for patients recuperating from an acute illness. Medicare covers 12 federally mandated services and several optional services. Medicare is developing a database management system that will perform the following functions: confirm patient eligibility, assign doctors, and pay doctors, pharmacists and hospitals promptly. The system should be designed to structure, store, retrieve, and analyse such critical Medicare management information as information about patients, doctors, pharmacies, and prescription drugs. The system stores the following information: Patients: identification number, name, address, birth date, gender, identification number, signup date, and annual income. Patient history: date of the visit, duration of the visit, diagnosis, and medication prescribed. Doctors: identification number, name, address, gender, birth date, specialization. Pharmacy: identification number, address, telephone number, name of contact person. Pharmacy inventory (for every drug kept in the inventory): identification number, name, price, date of the last purchase, amount in the inventory, and amount ordered (not yet received). Sales at a pharmacy: identification number, date of the transaction, and quantity purchased. A patient may visit multiple doctors, and a doctor is visited by one or more patients. Additionally, a patient may buy his or her medication from the pharmacy. Using this information, draw an ER diagram for the Medicare database management system.

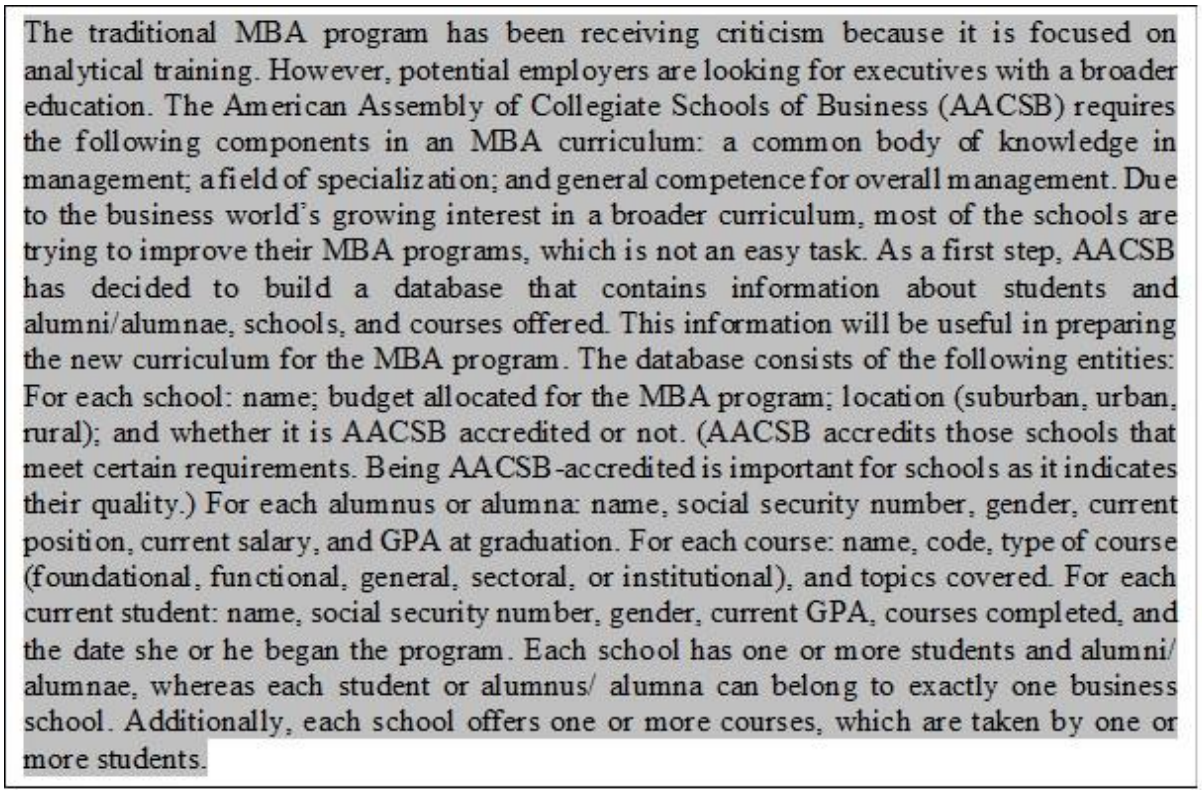
Appendix Figure 54 Medicare Case study (Pol & Ahuja, 2007, p. 77)


Memorabilia is an online company that buys sports products from various producers around the country and sells them to online customers. Customers visit Memorabilia's Web site, select an item, and make an order. As soon as the customer's order is received, the product is delivered to the customer, and the inventory level is updated. The company orders a particular product from a supplier when the inventory level drops below a certain level. The company has decided to maintain a detailed database of the customers, suppliers, and products to manage the operations. The following information is stored in the database: Customer: name, address, gender, and preferred sport. Supplier: supplier identification number, name of the company, and address. Product: product identification number, price per unit, amount in the inventory, amount requested of the suppliers but not yet received, amount ordered by the customers but not yet shipped. A customer may order one or more products, and a product may be ordered by one or more customers. Additionally, products may be provided by one or more suppliers, and a supplier may provide more than one product. Using this information, draw an ER diagram for Memorabilia's database. Clearly state any assumptions made.

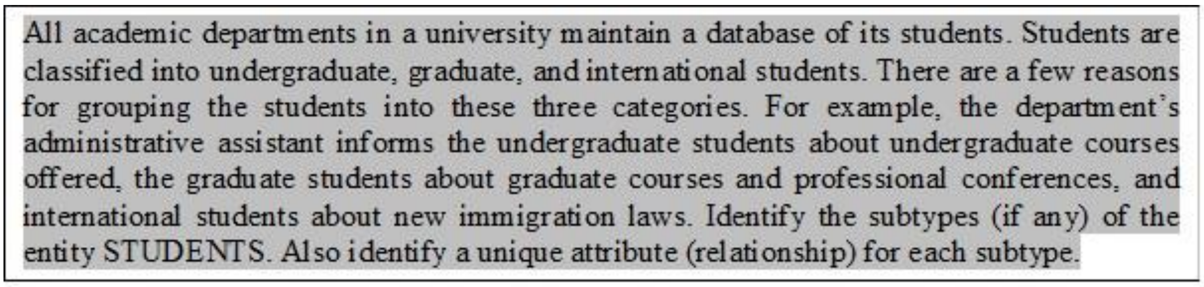
Appendix Figure 55 Memorabilia Company Case Study (Pol & Ahuja, 2007, p. 76)


Wood paneling manufacturers face a number of complex decisions in their daily proceedings. For example, allocating production resources and combining various raw materials to meet production goals require real time decision making. Due to changing supplies and costs, the management of a wood paneling manufacturer has decided to build a database system to fine tune their production processes. Consider a wood paneling manufacturer that produces a furniture grade particleboard. Each of the panels consists of a middle layer and two surface layers that are symmetrical. To enhance its mechanical properties, each panel has several strata of different materials, compositions, and specific gravity. A panel's quality can be controlled by specifying different density profiles and raw material requirements. There are eight different types of raw materials, and the database records information about the manufacturer's suppliers, the quantities available, and the maximum capacity of the bottleneck equipment. The raw material needs can be supplied by six different sources, including sawdust, shavings, sawmill, residual, chips, and short or long logs (softwoods or hardwoods). The database should also keep a detailed matrix of specifications that reveals the quantity of each individual raw material allowable in various layers of the different products. Another variable that affects the production schedule is the production capacity. The database keeps track of each piece of equipment's production capacity, the equipment type (name), the maintenance date, and a description of its activities. Given that each piece of equipment is assigned to produce a product from raw materials and that each raw material has exactly one supplier, draw an ER diagram of the manufacturer database. Make suitable assumptions for the remaining relationships.

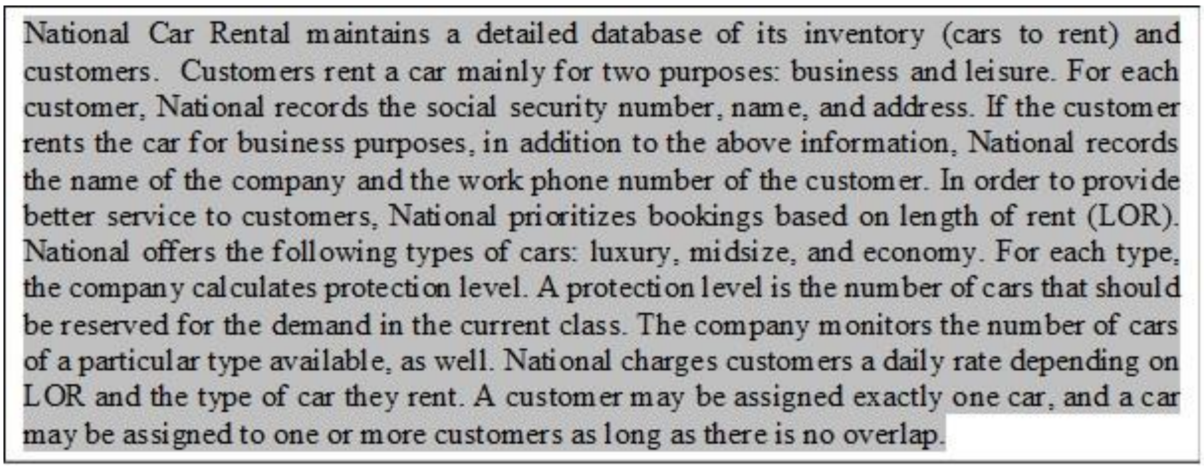
Appendix Figure 56 Wood Paneling Manufacturers Case study (Pol & Ahuja, 2007, p.78)


The traditional MBA program has been receiving criticism because it is focused on analytical training. However, potential employers are looking for executives with a broader education. The American Assembly of Collegiate Schools of Business (AACSB) requires the following components in an MBA curriculum: a common body of knowledge in management; a field of specialization; and general competence for overall management. Due to the business world's growing interest in a broader curriculum, most of the schools are trying to improve their MBA programs, which is not an easy task. As a first step, AACSB has decided to build a database that contains information about students and alumni/alumnae, schools, and courses offered. This information will be useful in preparing the new curriculum for the MBA program. The database consists of the following entities: For each school: name; budget allocated for the MBA program; location (suburban, urban, rural); and whether it is AACSB accredited or not. (AACSB accredits those schools that meet certain requirements. Being AACSB-accredited is important for schools as it indicates their quality.) For each alumnus or alumna: name, social security number, gender, current position, current salary, and GPA at graduation. For each course: name, code, type of course (foundational, functional, general, sectoral, or institutional), and topics covered. For each current student: name, social security number, gender, current GPA, courses completed, and the date she or he began the program. Each school has one or more students and alumni/alumnae, whereas each student or alumnus/ alumna can belong to exactly one business school. Additionally, each school offers one or more courses, which are taken by one or more students.

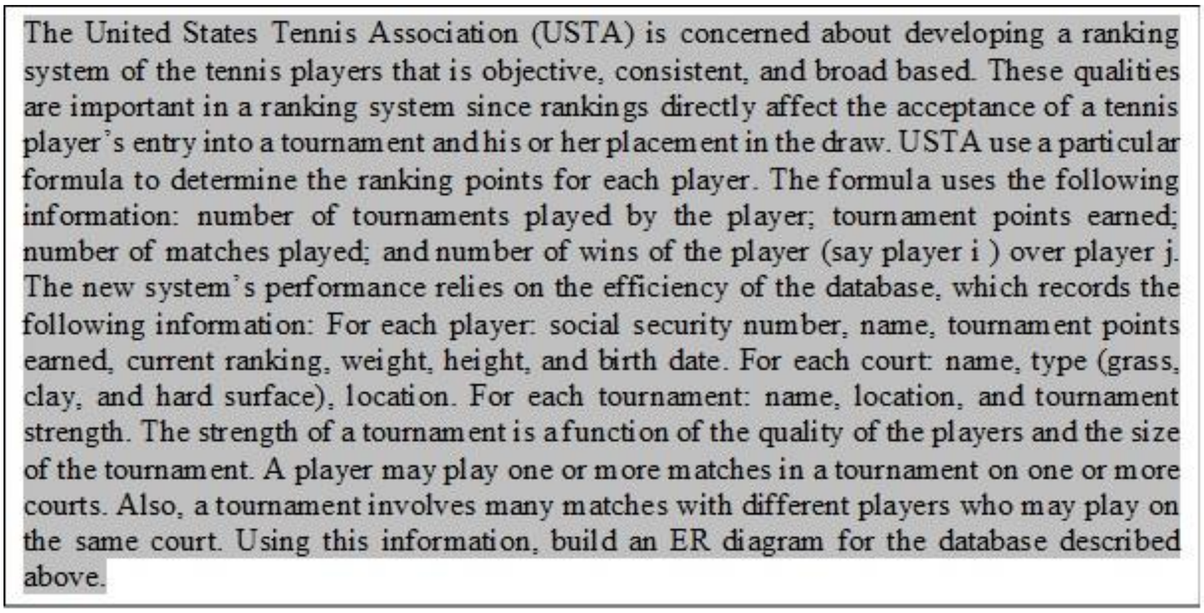
Appendix Figure 57 AACSB Case Study (Pol & Ahuja, 2007, p. 79)


All academic departments in a university maintain a database of its students. Students are classified into undergraduate, graduate, and international students. There are a few reasons for grouping the students into these three categories. For example, the department's administrative assistant informs the undergraduate students about undergraduate courses offered, the graduate students about graduate courses and professional conferences, and international students about new immigration laws. Identify the subtypes (if any) of the entity STUDENTS. Also identify a unique attribute (relationship) for each subtype.

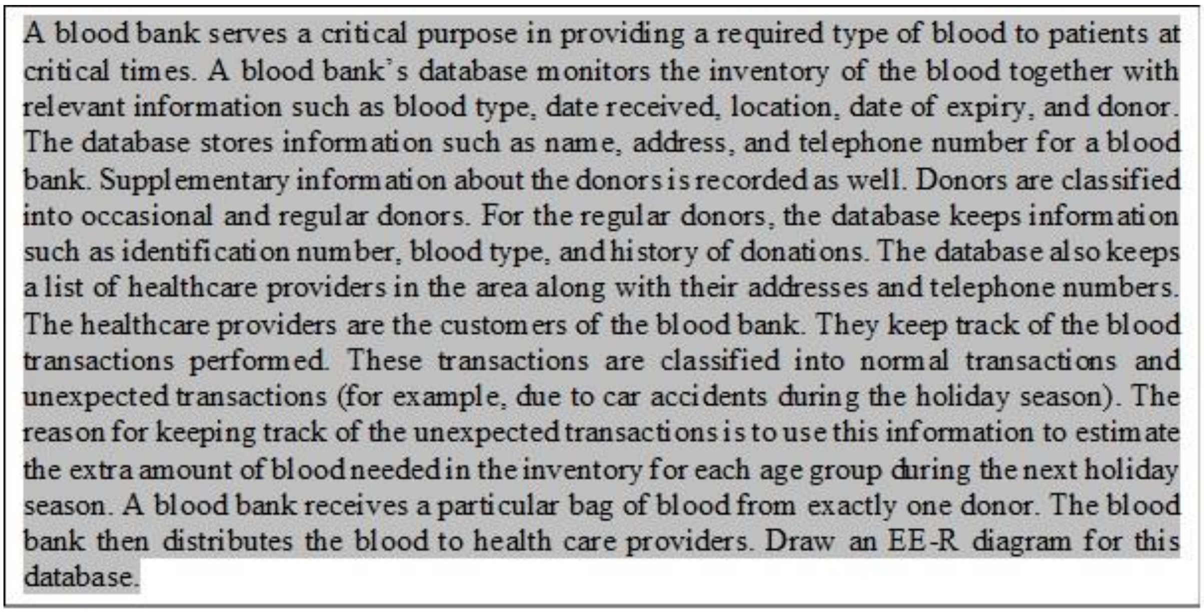
Appendix Figure 58 University Database Case Study (Pol & Ahuja, 2007, p. 81)


National Car Rental maintains a detailed database of its inventory (cars to rent) and customers. Customers rent a car mainly for two purposes: business and leisure. For each customer, National records the social security number, name, and address. If the customer rents the car for business purposes, in addition to the above information, National records the name of the company and the work phone number of the customer. In order to provide better service to customers, National prioritizes bookings based on length of rent (LOR). National offers the following types of cars: luxury, midsize, and economy. For each type, the company calculates protection level. A protection level is the number of cars that should be reserved for the demand in the current class. The company monitors the number of cars of a particular type available, as well. National charges customers a daily rate depending on LOR and the type of car they rent. A customer may be assigned exactly one car, and a car may be assigned to one or more customers as long as there is no overlap.

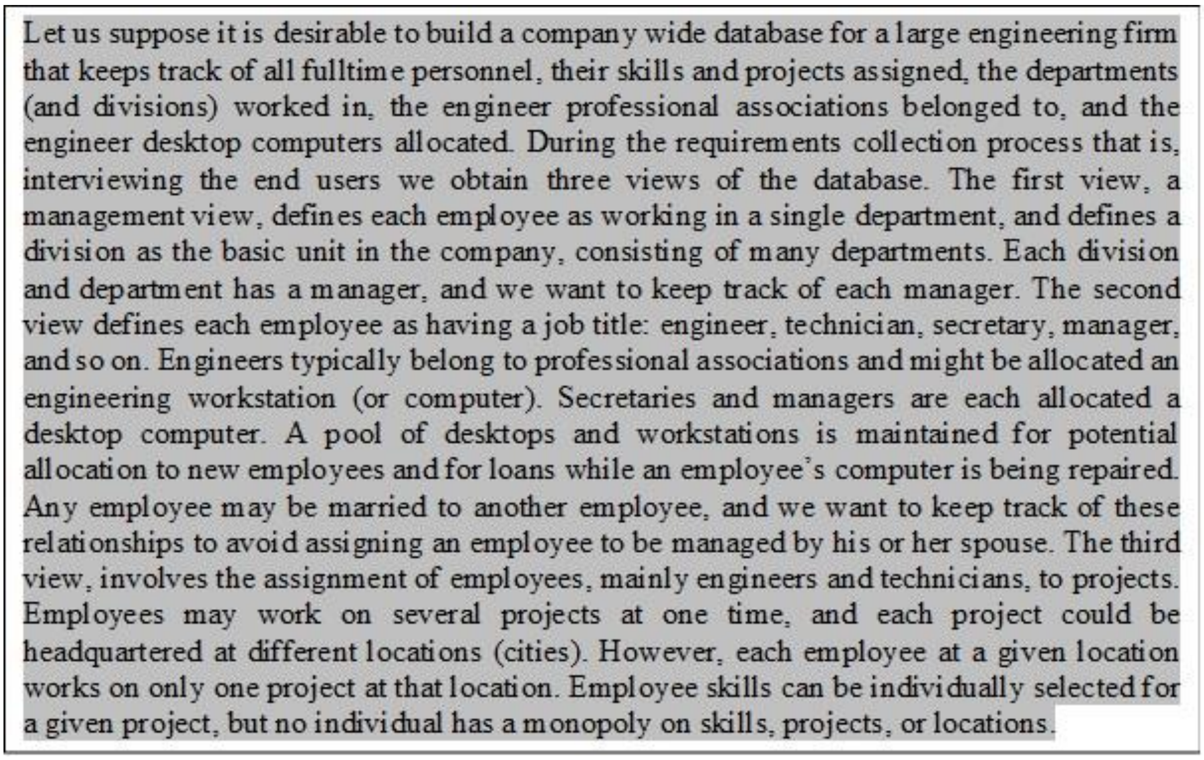
Appendix Figure 59 National Car Rental Case Study (Pol & Ahuja, 2007, p. 81)


The United States Tennis Association (USTA) is concerned about developing a ranking system of the tennis players that is objective, consistent, and broad based. These qualities are important in a ranking system since rankings directly affect the acceptance of a tennis player's entry into a tournament and his or her placement in the draw. USTA use a particular formula to determine the ranking points for each player. The formula uses the following information: number of tournaments played by the player; tournament points earned; number of matches played; and number of wins of the player (say player i) over player j. The new system's performance relies on the efficiency of the database, which records the following information: For each player: social security number, name, tournament points earned, current ranking, weight, height, and birth date. For each court: name, type (grass, clay, and hard surface), location. For each tournament: name, location, and tournament strength. The strength of a tournament is a function of the quality of the players and the size of the tournament. A player may play one or more matches in a tournament on one or more courts. Also, a tournament involves many matches with different players who may play on the same court. Using this information, build an ER diagram for the database described above.

Appendix Figure 60 USTA Case Study (Pol & Ahuja, 2007, p. 79)

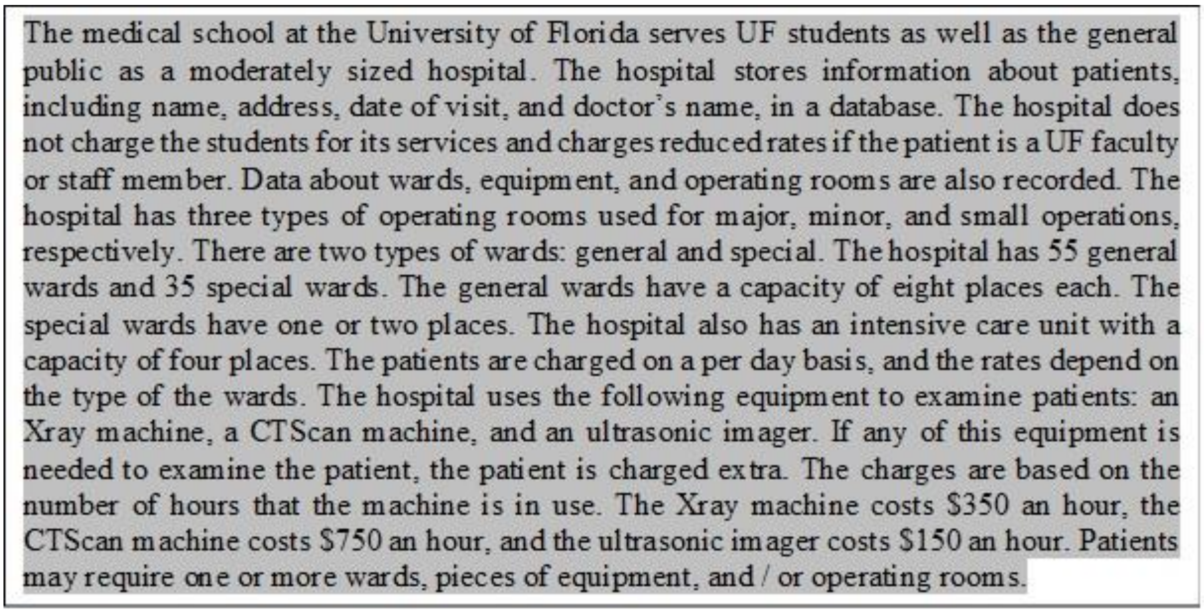

A blood bank serves a critical purpose in providing a required type of blood to patients at critical times. A blood bank's database monitors the inventory of the blood together with relevant information such as blood type, date received, location, date of expiry, and donor. The database stores information such as name, address, and telephone number for a blood bank. Supplementary information about the donors is recorded as well. Donors are classified into occasional and regular donors. For the regular donors, the database keeps information such as identification number, blood type, and history of donations. The database also keeps a list of healthcare providers in the area along with their addresses and telephone numbers. The healthcare providers are the customers of the blood bank. They keep track of the blood transactions performed. These transactions are classified into normal transactions and unexpected transactions (for example, due to car accidents during the holiday season). The reason for keeping track of the unexpected transactions is to use this information to estimate the extra amount of blood needed in the inventory for each age group during the next holiday season. A blood bank receives a particular bag of blood from exactly one donor. The blood bank then distributes the blood to health care providers. Draw an EE-R diagram for this database.

Appendix Figure 61 Blood Bank Case Study (Pol & Ahuja, 2007, p. 82)



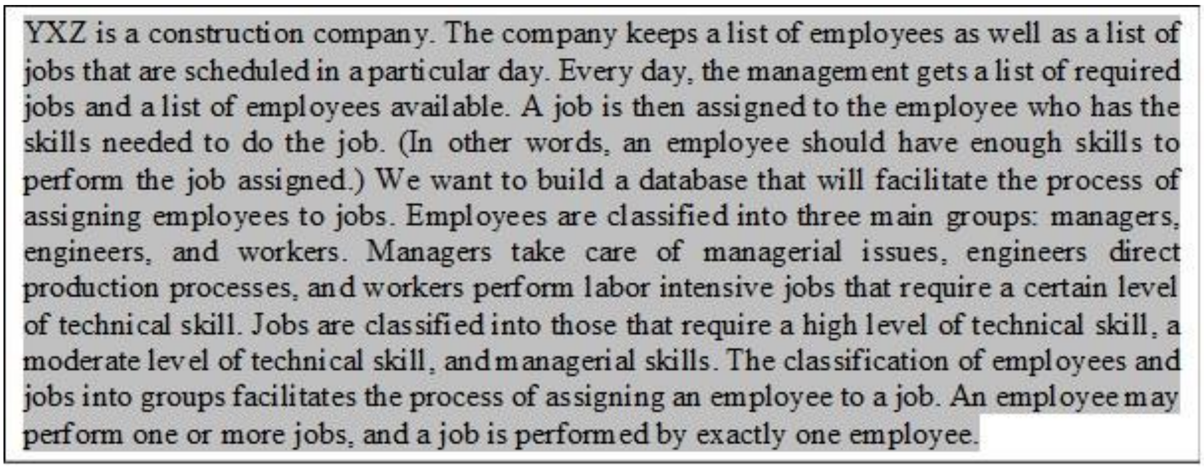
Let us suppose it is desirable to build a company wide database for a large engineering firm that keeps track of all fulltime personnel, their skills and projects assigned, the departments (and divisions) worked in, the engineer professional associations belonged to, and the engineer desktop computers allocated. During the requirements collection process that is, interviewing the end users we obtain three views of the database. The first view, a management view, defines each employee as working in a single department, and defines a division as the basic unit in the company, consisting of many departments. Each division and department has a manager, and we want to keep track of each manager. The second view defines each employee as having a job title: engineer, technician, secretary, manager, and so on. Engineers typically belong to professional associations and might be allocated an engineering workstation (or computer). Secretaries and managers are each allocated a desktop computer. A pool of desktops and workstations is maintained for potential allocation to new employees and for loans while an employee's computer is being repaired. Any employee may be married to another employee, and we want to keep track of these relationships to avoid assigning an employee to be managed by his or her spouse. The third view, involves the assignment of employees, mainly engineers and technicians, to projects. Employees may work on several projects at one time, and each project could be headquartered at different locations (cities). However, each employee at a given location works on only one project at that location. Employee skills can be individually selected for a given project, but no individual has a monopoly on skills, projects, or locations.

Appendix Figure 62 Company Wide Database Case Study (Teorey et al., 2005, p. 64)



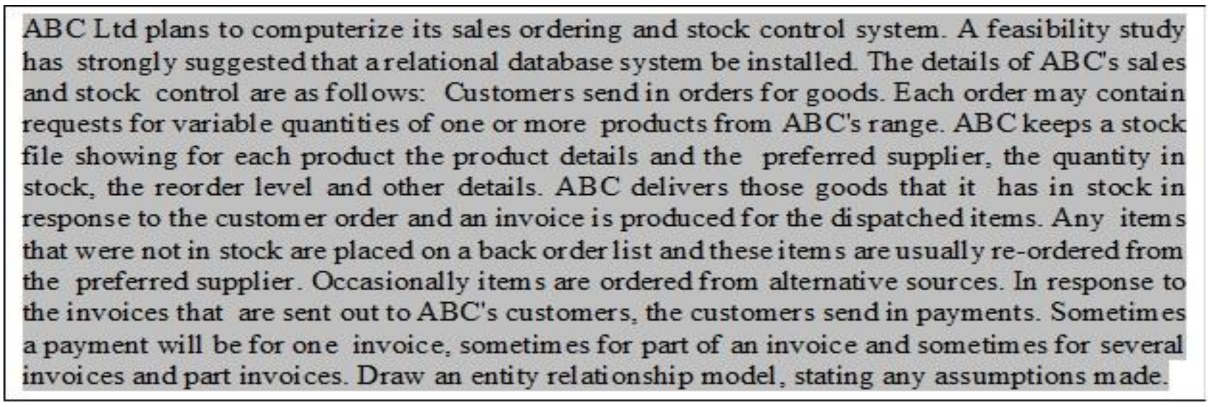
The medical school at the University of Florida serves UF students as well as the general public as a moderately sized hospital. The hospital stores information about patients, including name, address, date of visit, and doctor's name, in a database. The hospital does not charge the students for its services and charges reduced rates if the patient is a UF faculty or staff member. Data about wards, equipment, and operating rooms are also recorded. The hospital has three types of operating rooms used for major, minor, and small operations, respectively. There are two types of wards: general and special. The hospital has 55 general wards and 35 special wards. The general wards have a capacity of eight places each. The special wards have one or two places. The hospital also has an intensive care unit with a capacity of four places. The patients are charged on a per day basis, and the rates depend on the type of the wards. The hospital uses the following equipment to examine patients: an Xray machine, a CTScan machine, and an ultrasonic imager. If any of this equipment is needed to examine the patient, the patient is charged extra. The charges are based on the number of hours that the machine is in use. The Xray machine costs $350 an hour, the CTScan machine costs $750 an hour, and the ultrasonic imager costs $150 an hour. Patients may require one or more wards, pieces of equipment, and / or operating rooms.

Appendix Figure 63 Medical School Case Study (Pol & Ahuja, 2007, p. 81)



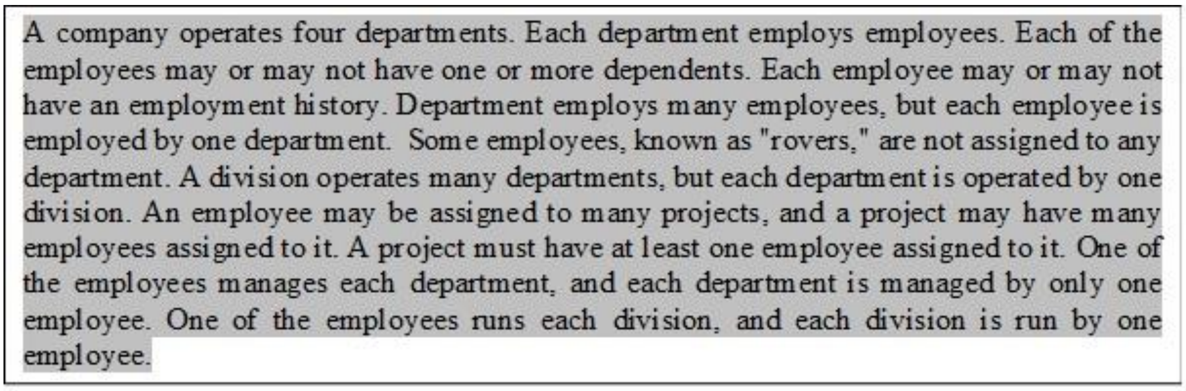
YXZ is a construction company. The company keeps a list of employees as well as a list of jobs that are scheduled in a particular day. Every day, the management gets a list of required jobs and a list of employees available. A job is then assigned to the employee who has the skills needed to do the job. (In other words, an employee should have enough skills to perform the job assigned.) We want to build a database that will facilitate the process of assigning employees to jobs. Employees are classified into three main groups: managers, engineers, and workers. Managers take care of managerial issues, engineers direct production processes, and workers perform labor intensive jobs that require a certain level of technical skill. Jobs are classified into those that require a high level of technical skill, a moderate level of technical skill, and managerial skills. The classification of employees and jobs into groups facilitates the process of assigning an employee to a job. An employee may perform one or more jobs, and a job is performed by exactly one employee.

Appendix Figure 64 YXZ Company Case Study (Pol & Ahuja, 2007, p. 82)



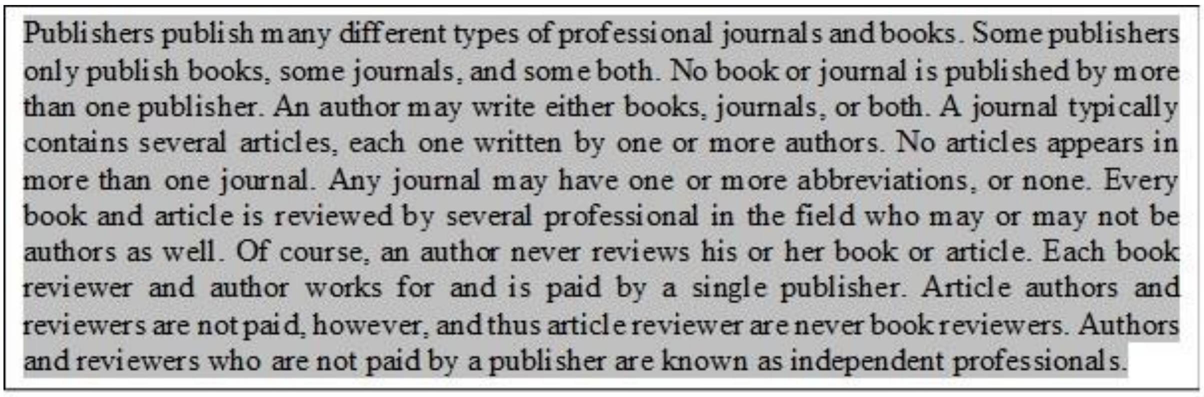
ABC Ltd plans to computerize its sales ordering and stock control system. A feasibility study has strongly suggested that a relational database system be installed. The details of ABC's sales and stock control are as follows: Customers send in orders for goods. Each order may contain requests for variable quantities of one or more products from ABC's range. ABC keeps a stock file showing for each product the product details and the preferred supplier, the quantity in stock, the reorder level and other details. ABC delivers those goods that it has in stock in response to the customer order and an invoice is produced for the dispatched items. Any items that were not in stock are placed on a back order list and these items are usually re-ordered from the preferred supplier. Occasionally items are ordered from alternative sources. In response to the invoices that are sent out to ABC's customers, the customers send in payments. Sometimes a payment will be for one invoice, sometimes for part of an invoice and sometimes for several invoices and part invoices. Draw an entity relationship model, stating any assumptions made.

Appendix Figure 65 ABC Ltd Case Study Needs Page (Carter, 2003, p. 39)

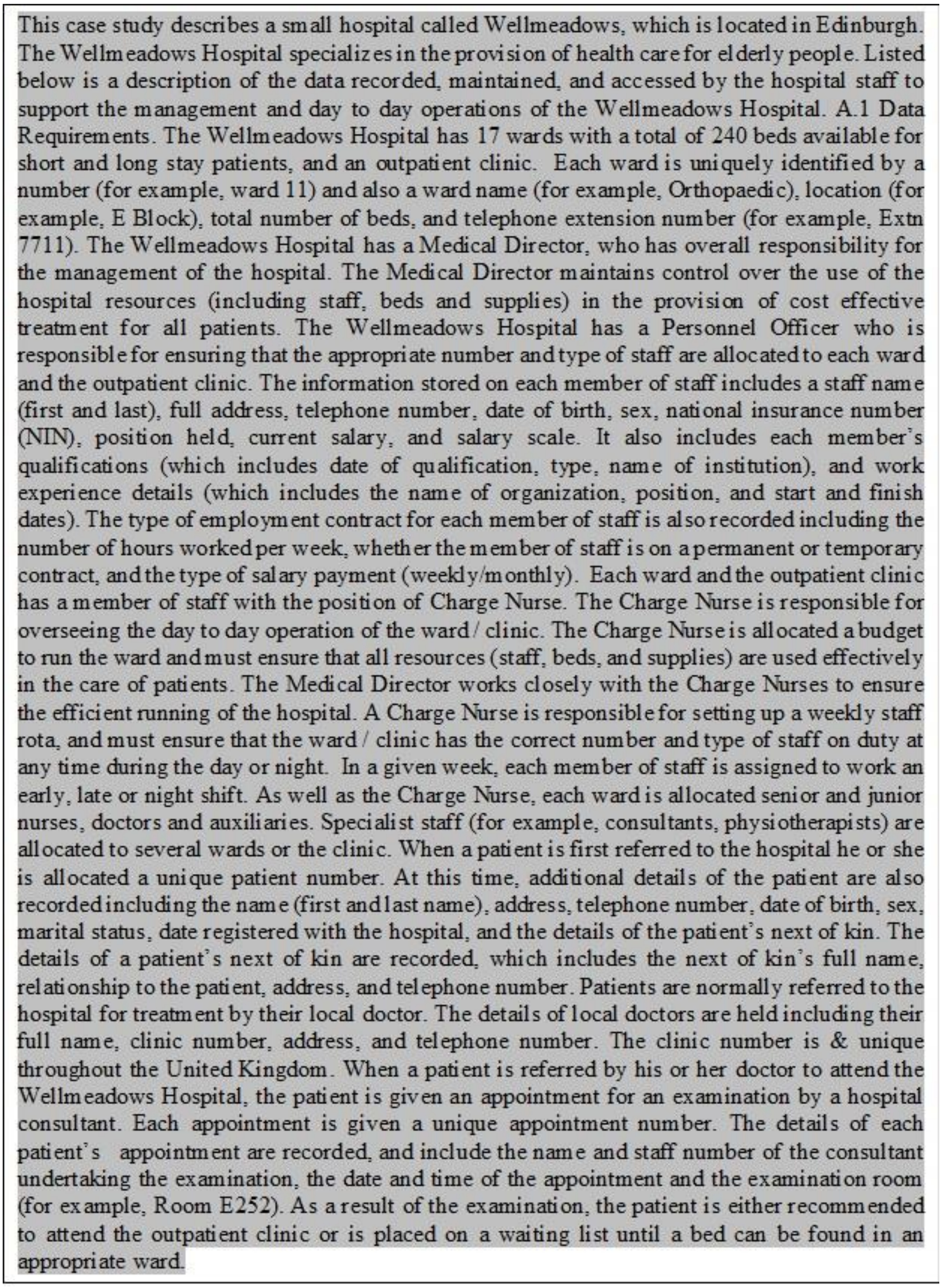


A company operates four departments. Each department employs employees. Each of the employees may or may not have one or more dependents. Each employee may or may not have an employment history. Department employs many employees, but each employee is employed by one department. Some employees, known as "rovers," are not assigned to any department. A division operates many departments, but each department is operated by one division. An employee may be assigned to many projects, and a project may have many employees assigned to it. A project must have at least one employee assigned to it. One of the employees manages each department, and each department is managed by only one employee. One of the employees runs each division, and each division is run by one employee.

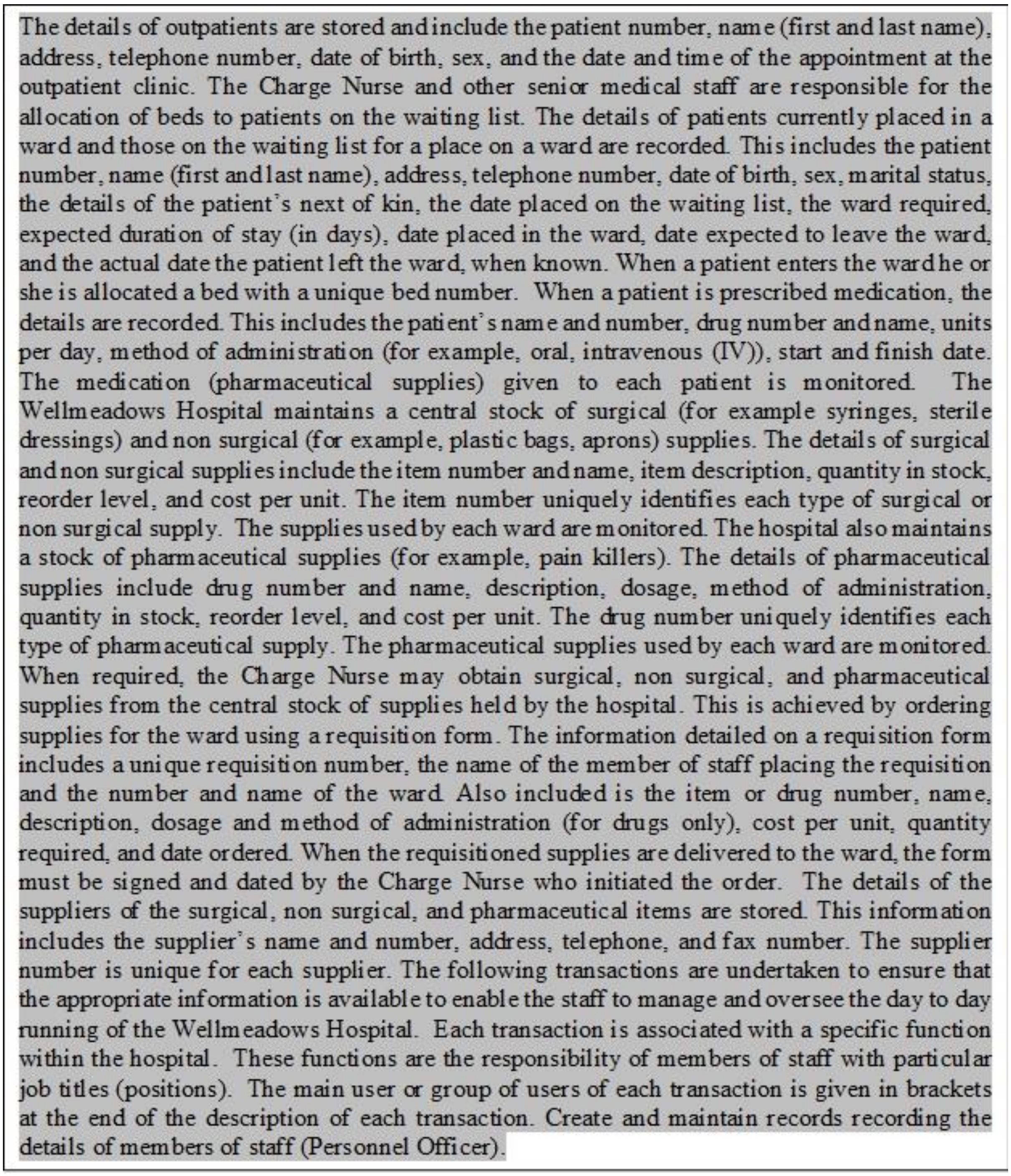
Appendix Figure 66 Company Database Case Study (Rob & Coronel, 2009, p. 142)


Publishers publish many different types of professional journals and books. Some publishers only publish books, some journals, and some both. No book or journal is published by more than one publisher. An author may write either books, journals, or both. A journal typically contains several articles, each one written by one or more authors. No articles appears in more than one journal. Any journal may have one or more abbreviations, or none. Every book and article is reviewed by several professional in the field who may or may not be authors as well. Of course, an author never reviews his or her book or article. Each book reviewer and author works for and is paid by a single publisher. Article authors and reviewers are not paid, however, and thus article reviewer are never book reviewers. Authors and reviewers who are not paid by a publisher are known as independent professionals.

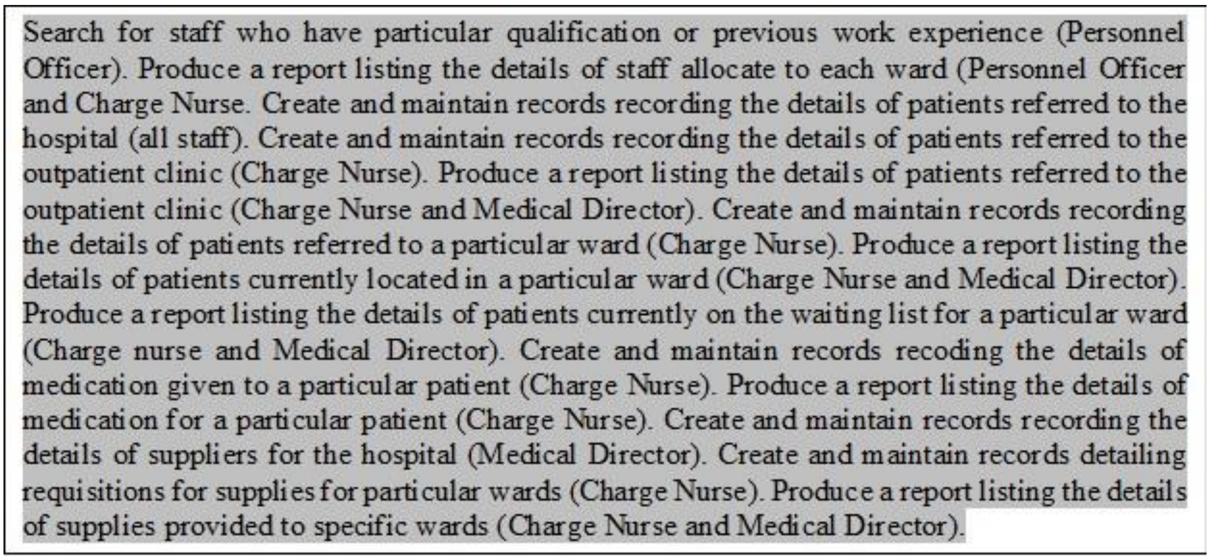
Appendix Figure 67 Publishers Database Case Study (Teorey, 1999, p. 76)


This case study describes a small hospital called Wellmeadows, which is located in Edinburgh. The Wellmeadows Hospital specializes in the provision of health care for elderly people. Listed below is a description of the data recorded, maintained, and accessed by the hospital staff to support the management and day to day operations of the Wellmeadows Hospital. A.1 Data Requirements. The Wellmeadows Hospital has 17 wards with a total of 240 beds available for short and long stay patients, and an outpatient clinic. Each ward is uniquely identified by a number (for example, ward 11) and also a ward name (for example, Orthopaedic), location (for example, E Block), total number of beds, and telephone extension number (for example, Extn 7711). The Wellmeadows Hospital has a Medical Director, who has overall responsibility for the management of the hospital. The Medical Director maintains control over the use of the hospital resources (including staff, beds and supplies) in the provision of cost effective treatment for all patients. The Wellmeadows Hospital has a Personnel Officer who is responsible for ensuring that the appropriate number and type of staff are allocated to each ward and the outpatient clinic. The information stored on each member of staff includes a staff name (first and last), full address, telephone number, date of birth, sex, national insurance number (NIN), position held, current salary, and salary scale. It also includes each member's qualifications (which includes date of qualification, type, name of institution), and work experience details (which includes the name of organization, position, and start and finish dates). The type of employment contract for each member of staff is also recorded including the number of hours worked per week, whether the member of staff is on a permanent or temporary contract, and the type of salary payment (weekly/monthly). Each ward and the outpatient clinic has a member of staff with the position of Charge Nurse. The Charge Nurse is responsible for overseeing the day to day operation of the ward / clinic. The Charge Nurse is allocated a budget to run the ward and must ensure that all resources (staff, beds, and supplies) are used effectively in the care of patients. The Medical Director works closely with the Charge Nurses to ensure the efficient running of the hospital. A Charge Nurse is responsible for setting up a weekly staff rota, and must ensure that the ward / clinic has the correct number and type of staff on duty at any time during the day or night. In a given week, each member of staff is assigned to work an early, late or night shift. As well as the Charge Nurse, each ward is allocated senior and junior nurses, doctors and auxiliaries. Specialist staff (for example, consultants, physiotherapists) are allocated to several wards or the clinic. When a patient is first referred to the hospital he or she is allocated a unique patient number. At this time, additional details of the patient are also recorded including the name (first and last name), address, telephone number, date of birth, sex, marital status, date registered with the hospital, and the details of the patient's next of kin. The details of a patient's next of kin are recorded, which includes the next of kin's full name, relationship to the patient, address, and telephone number. Patients are normally referred to the hospital for treatment by their local doctor. The details of local doctors are held including their full name, clinic number, address, and telephone number. The clinic number is & unique throughout the United Kingdom. When a patient is referred by his or her doctor to attend the Wellmeadows Hospital, the patient is given an appointment for an examination by a hospital consultant. Each appointment is given a unique appointment number. The details of each patient's appointment are recorded, and include the name and staff number of the consultant undertaking the examination, the date and time of the appointment and the examination room (for example, Room E252). As a result of the examination, the patient is either recommended to attend the outpatient clinic or is placed on a waiting list until a bed can be found in an appropriate ward.

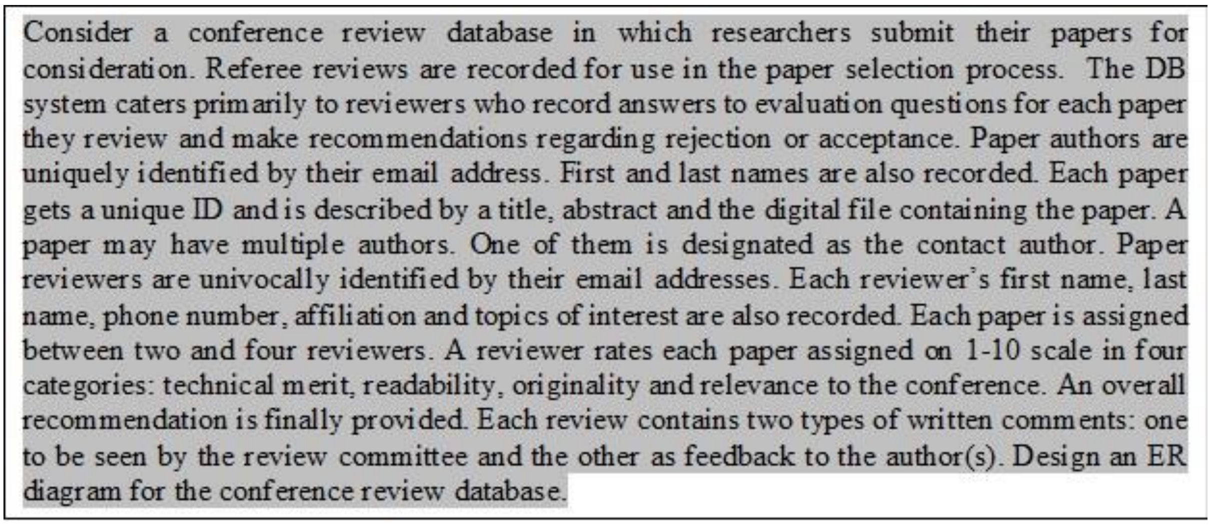
Appendix Figure 68 Wellmeadows Hospital Case Study Part One (Connolly & Begg, 2015, p. B-5)


The details of outpatients are stored and include the patient number, name (first and last name), address, telephone number, date of birth, sex, and the date and time of the appointment at the outpatient clinic. The Charge Nurse and other senior medical staff are responsible for the allocation of beds to patients on the waiting list. The details of patients currently placed in a ward and those on the waiting list for a place on a ward are recorded. This includes the patient number, name (first and last name), address, telephone number, date of birth, sex, marital status, the details of the patient's next of kin, the date placed on the waiting list, the ward required, expected duration of stay (in days), date placed in the ward, date expected to leave the ward, and the actual date the patient left the ward, when known. When a patient enters the ward he or she is allocated a bed with a unique bed number. When a patient is prescribed medication, the details are recorded. This includes the patient's name and number, drug number and name, units per day, method of administration (for example, oral, intravenous (IV)), start and finish date. The medication (pharmaceutical supplies) given to each patient is monitored. The Wellmeadows Hospital maintains a central stock of surgical (for example syringes, sterile dressings) and non surgical (for example, plastic bags, aprons) supplies. The details of surgical and non surgical supplies include the item number and name, item description, quantity in stock, reorder level, and cost per unit. The item number uniquely identifies each type of surgical or non surgical supply. The supplies used by each ward are monitored. The hospital also maintains a stock of pharmaceutical supplies (for example, pain killers). The details of pharmaceutical supplies include drug number and name, description, dosage, method of administration, quantity in stock, reorder level, and cost per unit. The drug number uniquely identifies each type of pharmaceutical supply. The pharmaceutical supplies used by each ward are monitored. When required, the Charge Nurse may obtain surgical, non surgical, and pharmaceutical supplies from the central stock of supplies held by the hospital. This is achieved by ordering supplies for the ward using a requisition form. The information detailed on a requisition form includes a unique requisition number, the name of the member of staff placing the requisition and the number and name of the ward. Also included is the item or drug number, name, description, dosage and method of administration (for drugs only), cost per unit, quantity required, and date ordered. When the requisitioned supplies are delivered to the ward, the form must be signed and dated by the Charge Nurse who initiated the order. The details of the suppliers of the surgical, non surgical, and pharmaceutical items are stored. This information includes the supplier's name and number, address, telephone, and fax number. The supplier number is unique for each supplier. The following transactions are undertaken to ensure that the appropriate information is available to enable the staff to manage and oversee the day to day running of the Wellmeadows Hospital. Each transaction is associated with a specific function within the hospital. These functions are the responsibility of members of staff with particular job titles (positions). The main user or group of users of each transaction is given in brackets at the end of the description of each transaction. Create and maintain records recording the details of members of staff (Personnel Officer).

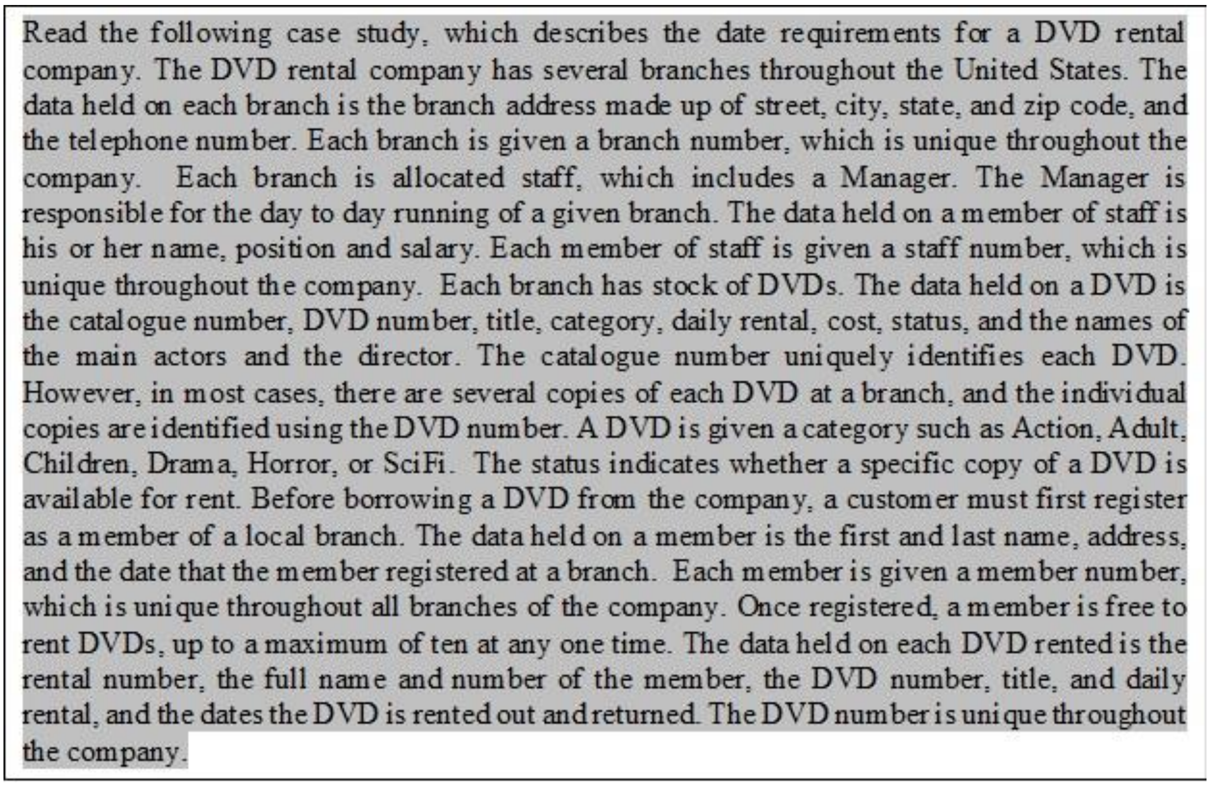
Appendix Figure 69 Wellmeadows Hospital Case Study Part 2 (Connolly & Begg, 2015, p. B-5)


Search for staff who have particular qualification or previous work experience (Personnel Officer). Produce a report listing the details of staff allocate to each ward (Personnel Officer and Charge Nurse. Create and maintain records recording the details of patients referred to the hospital (all staff). Create and maintain records recording the details of patients referred to the outpatient clinic (Charge Nurse). Produce a report listing the details of patients referred to the outpatient clinic (Charge Nurse and Medical Director). Create and maintain records recording the details of patients referred to a particular ward (Charge Nurse). Produce a report listing the details of patients currently located in a particular ward (Charge Nurse and Medical Director). Produce a report listing the details of patients currently on the waiting list for a particular ward (Charge nurse and Medical Director). Create and maintain records recoding the details of medication given to a particular patient (Charge Nurse). Produce a report listing the details of medication for a particular patient (Charge Nurse). Create and maintain records recording the details of suppliers for the hospital (Medical Director). Create and maintain records detailing requisitions for supplies for particular wards (Charge Nurse). Produce a report listing the details of supplies provided to specific wards (Charge Nurse and Medical Director).

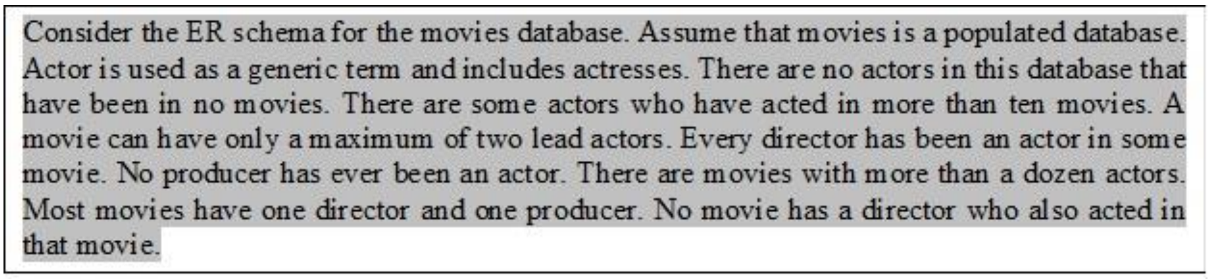
Appendix Figure 70 Wellmeadows Hospital Case Study Part 3 (Connolly & Begg, 2015, p. B-5)


Consider a conference review database in which researchers submit their papers for consideration. Referee reviews are recorded for use in the paper selection process. The DB system caters primarily to reviewers who record answers to evaluation questions for each paper they review and make recommendations regarding rejection or acceptance. Paper authors are uniquely identified by their email address. First and last names are also recorded. Each paper gets a unique ID and is described by a title, abstract and the digital file containing the paper. A paper may have multiple authors. One of them is designated as the contact author. Paper reviewers are univocally identified by their email addresses. Each reviewer's first name, last name, phone number, affiliation and topics of interest are also recorded. Each paper is assigned between two and four reviewers. A reviewer rates each paper assigned on 1-10 scale in four categories: technical merit, readability, originality and relevance to the conference. An overall recommendation is finally provided. Each review contains two types of written comments: one to be seen by the review committee and the other as feedback to the author(s). Design an ER diagram for the conference review database.

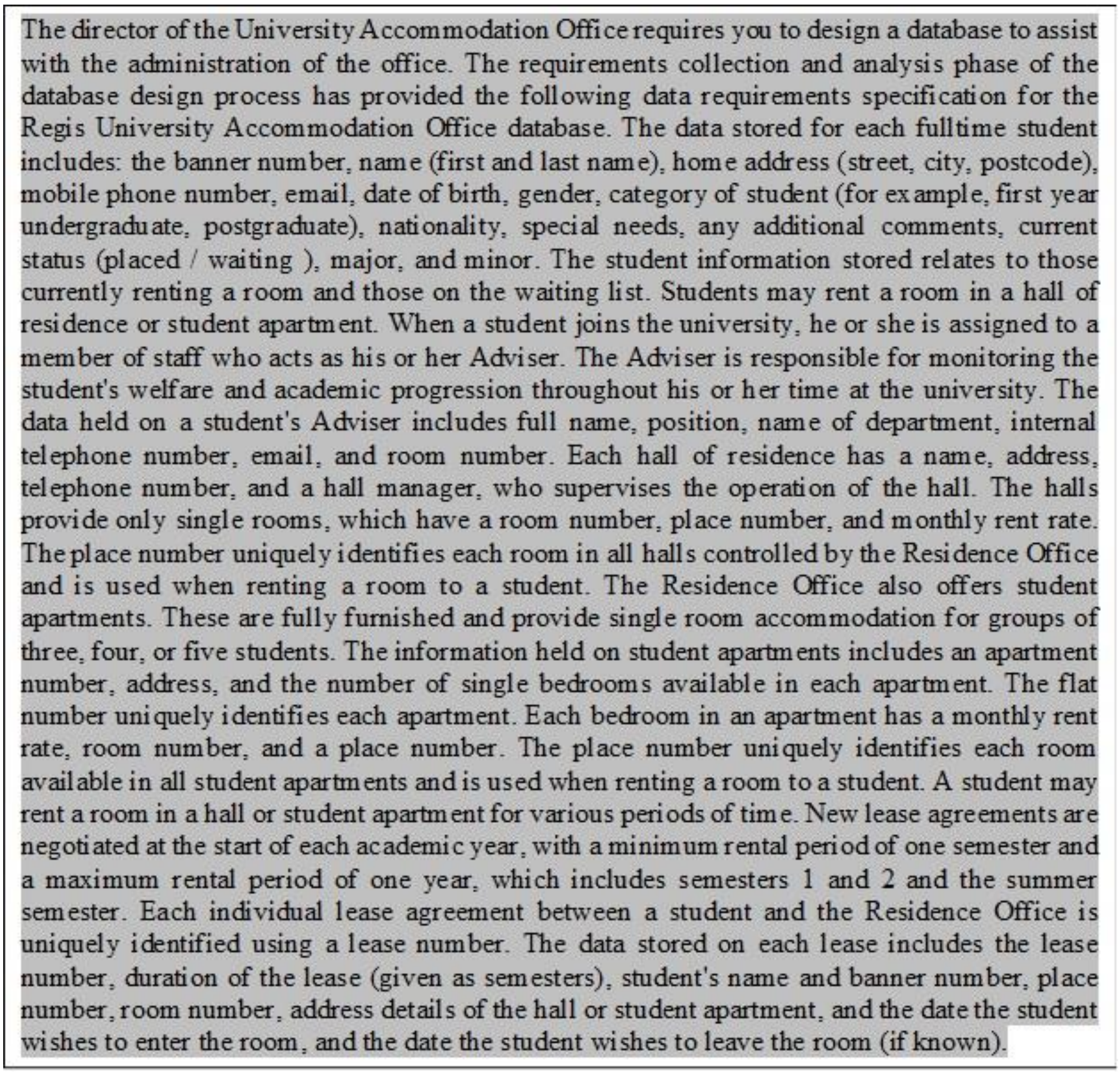
Appendix Figure 71 Conference Review Database Case Study (Elmasri & Navathe, 2017, p. 134)


Read the following case study, which describes the date requirements for a DVD rental company. The DVD rental company has several branches throughout the United States. The data held on each branch is the branch address made up of street, city, state, and zip code, and the telephone number. Each branch is given a branch number, which is unique throughout the company. Each branch is allocated staff, which includes a Manager. The Manager is responsible for the day to day running of a given branch. The data held on a member of staff is his or her name, position and salary. Each member of staff is given a staff number, which is unique throughout the company. Each branch has stock of DVDs. The data held on a DVD is the catalogue number, DVD number, title, category, daily rental, cost, status, and the names of the main actors and the director. The catalogue number uniquely identifies each DVD. However, in most cases, there are several copies of each DVD at a branch, and the individual copies are identified using the DVD number. A DVD is given a category such as Action, Adult, Children, Drama, Horror, or SciFi. The status indicates whether a specific copy of a DVD is available for rent. Before borrowing a DVD from the company, a customer must first register as a member of a local branch. The data held on a member is the first and last name, address, and the date that the member registered at a branch. Each member is given a member number, which is unique throughout all branches of the company. Once registered, a member is free to rent DVDs, up to a maximum of ten at any one time. The data held on each DVD rented is the rental number, the full name and number of the member, the DVD number, title, and daily rental, and the dates the DVD is rented out and returned. The DVD number is unique throughout the company.

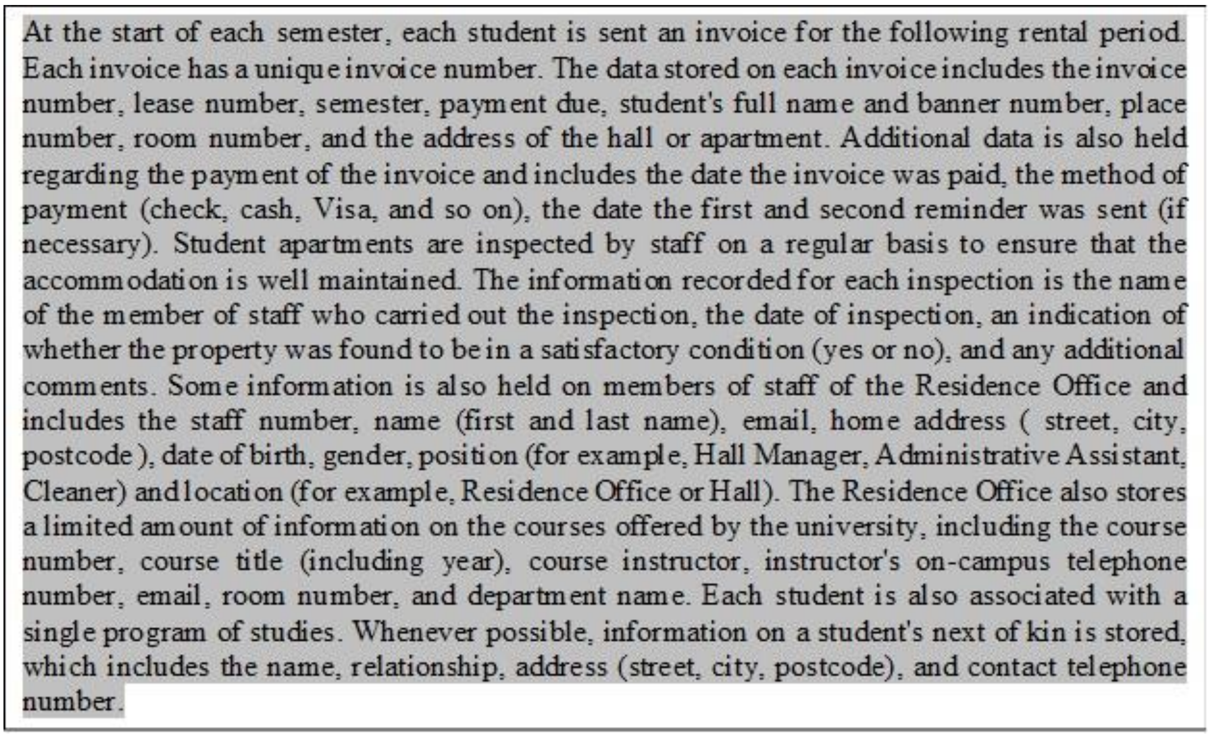
Appendix Figure 72 DVD Database Case Study (Connolly & Begg, 2015, p. 431)


Consider the ER schema for the movies database. Assume that movies is a populated database. Actor is used as a generic term and includes actresses. There are no actors in this database that have been in no movies. There are some actors who have acted in more than ten movies. A movie can have only a maximum of two lead actors. Every director has been an actor in some movie. No producer has ever been an actor. There are movies with more than a dozen actors. Most movies have one director and one producer. No movie has a director who also acted in that movie.

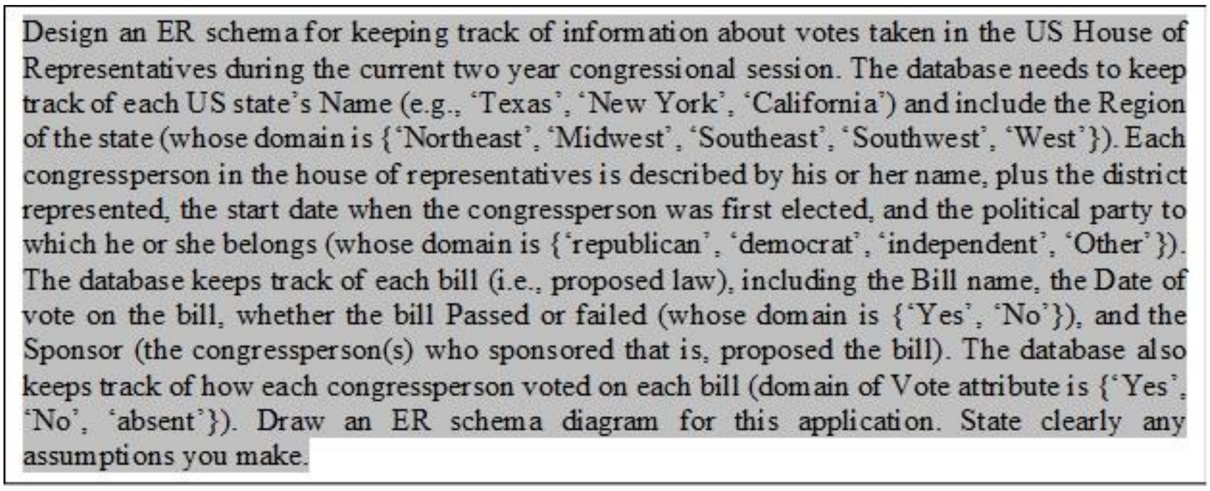
Appendix Figure 73 Movie Database Case Study (Elmasri & Navathe, 2017, p. 132)


The director of the University Accommodation Office requires you to design a database to assist with the administration of the office. The requirements collection and analysis phase of the database design process has provided the following data requirements specification for the Regis University Accommodation Office database. The data stored for each fulltime student includes: the banner number, name (first and last name), home address (street, city, postcode), mobile phone number, email, date of birth, gender, category of student (for example, first year undergraduate, postgraduate), nationality, special needs, any additional comments, current status (placed / waiting), major, and minor. The student information stored relates to those currently renting a room and those on the waiting list. Students may rent a room in a hall of residence or student apartment. When a student joins the university, he or she is assigned to a member of staff who acts as his or her Adviser. The Adviser is responsible for monitoring the student's welfare and academic progression throughout his or her time at the university. The data held on a student's Adviser includes full name, position, name of department, internal telephone number, email, and room number. Each hall of residence has a name, address, telephone number, and a hall manager, who supervises the operation of the hall. The halls provide only single rooms, which have a room number, place number, and monthly rent rate. The place number uniquely identifies each room in all halls controlled by the Residence Office and is used when renting a room to a student. The Residence Office also offers student apartments. These are fully furnished and provide single room accommodation for groups of three, four, or five students. The information held on student apartments includes an apartment number, address, and the number of single bedrooms available in each apartment. The flat number uniquely identifies each apartment. Each bedroom in an apartment has a monthly rent rate, room number, and a place number. The place number uniquely identifies each room available in all student apartments and is used when renting a room to a student. A student may rent a room in a hall or student apartment for various periods of time. New lease agreements are negotiated at the start of each academic year, with a minimum rental period of one semester and a maximum rental period of one year, which includes semesters 1 and 2 and the summer semester. Each individual lease agreement between a student and the Residence Office is uniquely identified using a lease number. The data stored on each lease includes the lease number, duration of the lease (given as semesters), student's name and banner number, place number, room number, address details of the hall or student apartment, and the date the student wishes to enter the room, and the date the student wishes to leave the room (if known).

Appendix Figure 74 University Accommodation Office Case Study Part One (Connolly & Begg, 2015, p. B-1)


At the start of each semester, each student is sent an invoice for the following rental period. Each invoice has a unique invoice number. The data stored on each invoice includes the invoice number, lease number, semester, payment due, student's full name and banner number, place number, room number, and the address of the hall or apartment. Additional data is also held regarding the payment of the invoice and includes the date the invoice was paid, the method of payment (check, cash, Visa, and so on), the date the first and second reminder was sent (if necessary). Student apartments are inspected by staff on a regular basis to ensure that the accommodation is well maintained. The information recorded for each inspection is the name of the member of staff who carried out the inspection, the date of inspection, an indication of whether the property was found to be in a satisfactory condition (yes or no), and any additional comments. Some information is also held on members of staff of the Residence Office and includes the staff number, name (first and last name), email, home address (street, city, postcode), date of birth, gender, position (for example, Hall Manager, Administrative Assistant, Cleaner) and location (for example, Residence Office or Hall). The Residence Office also stores a limited amount of information on the courses offered by the university, including the course number, course title (including year), course instructor, instructor's on-campus telephone number, email, room number, and department name. Each student is also associated with a single program of studies. Whenever possible, information on a student's next of kin is stored, which includes the name, relationship, address (street, city, postcode), and contact telephone number.

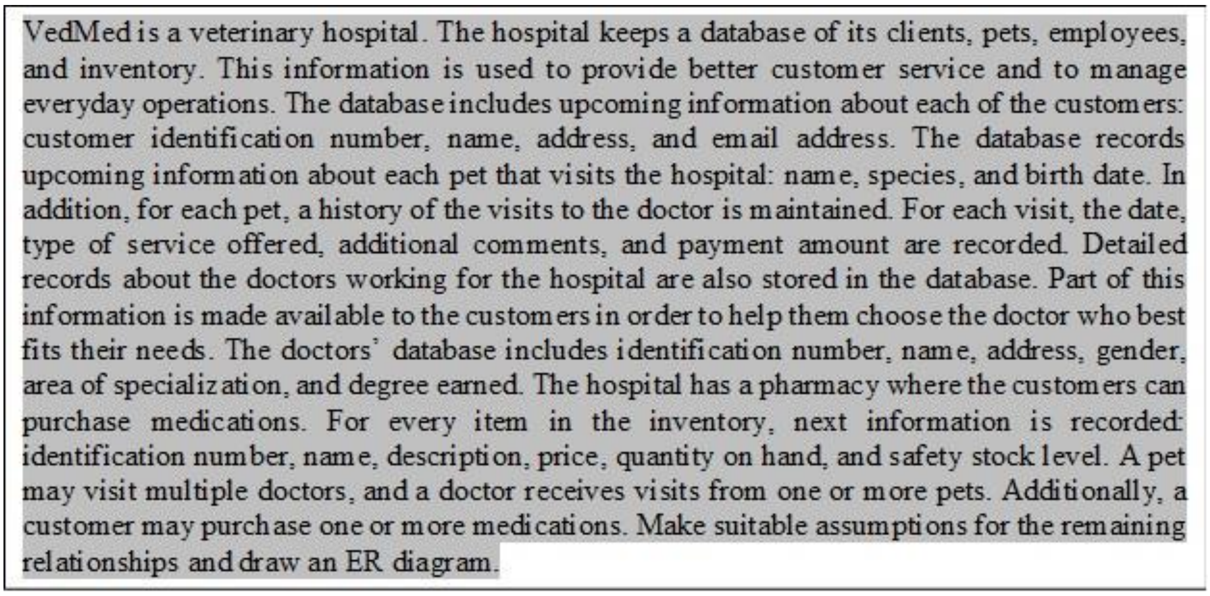
Appendix Figure 75 University Accommodation Office Case Study Part 2 (Connolly & Begg, 2015, p. B-1)


Design an ER schema for keeping track of information about votes taken in the US House of Representatives during the current two year congressional session. The database needs to keep track of each US state's Name (e.g., 'Texas', 'New York', 'California') and include the Region of the state (whose domain is {'Northeast', 'Midwest', 'Southeast', 'Southwest', 'West'}). Each congressperson in the house of representatives is described by his or her name, plus the district represented, the start date when the congressperson was first elected, and the political party to which he or she belongs (whose domain is {'republican', 'democrat', 'independent', 'Other'}). The database keeps track of each bill (i.e., proposed law), including the Bill name, the Date of vote on the bill, whether the bill Passed or failed (whose domain is {'Yes', 'No'}), and the Sponsor (the congressperson(s) who sponsored that is, proposed the bill). The database also keeps track of how each congressperson voted on each bill (domain of Vote attribute is {'Yes', 'No', 'absent'}). Draw an ER schema diagram for this application. State clearly any assumptions you make.

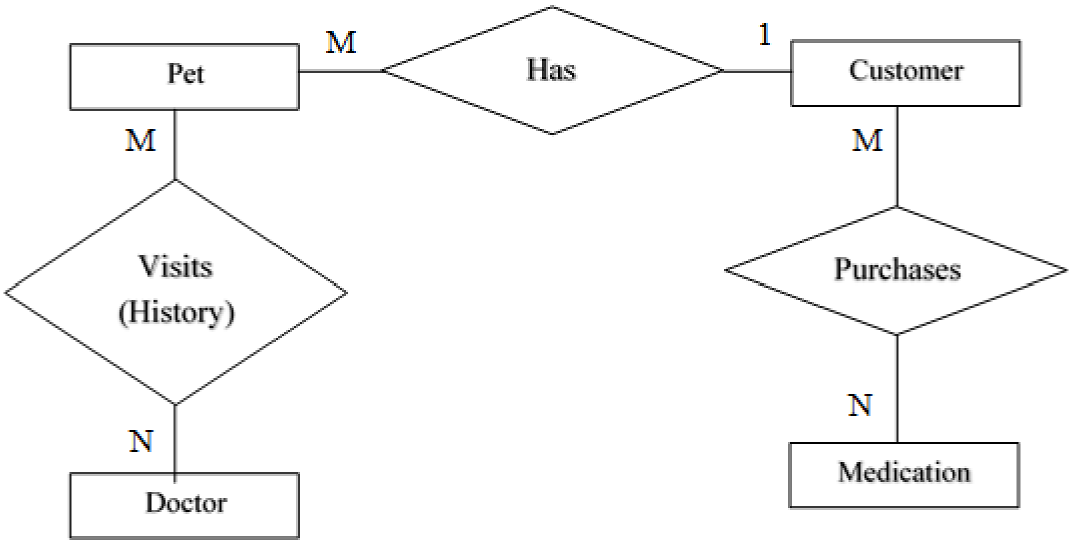
Appendix Figure 76 Votes Database Case Study (Elmasri & Navathe, 2017, p. 127)

Appendix 5:

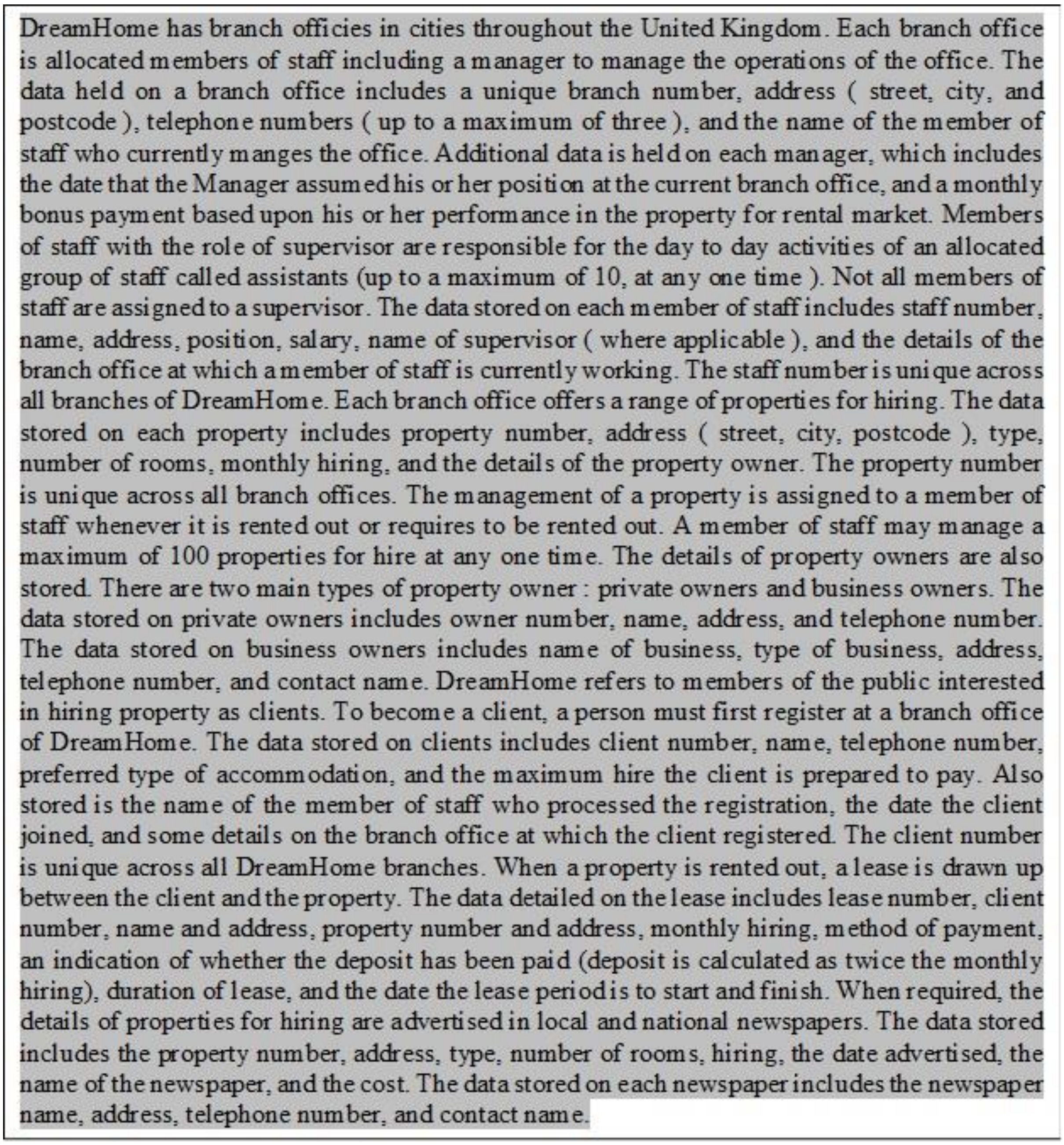
Test set with its model answers used for Experimental Two.


VedMed is a veterinary hospital. The hospital keeps a database of its clients, pets, employees, and inventory. This information is used to provide better customer service and to manage everyday operations. The database includes upcoming information about each of the customers: customer identification number, name, address, and email address. The database records upcoming information about each pet that visits the hospital: name, species, and birth date. In addition, for each pet, a history of the visits to the doctor is maintained. For each visit, the date, type of service offered, additional comments, and payment amount are recorded. Detailed records about the doctors working for the hospital are also stored in the database. Part of this information is made available to the customers in order to help them choose the doctor who best fits their needs. The doctors' database includes identification number, name, address, gender, area of specialization, and degree earned. The hospital has a pharmacy where the customers can purchase medications. For every item in the inventory, next information is recorded: identification number, name, description, price, quantity on hand, and safety stock level. A pet may visit multiple doctors, and a doctor receives visits from one or more pets. Additionally, a customer may purchase one or more medications. Make suitable assumptions for the remaining relationships and draw an ER diagram.

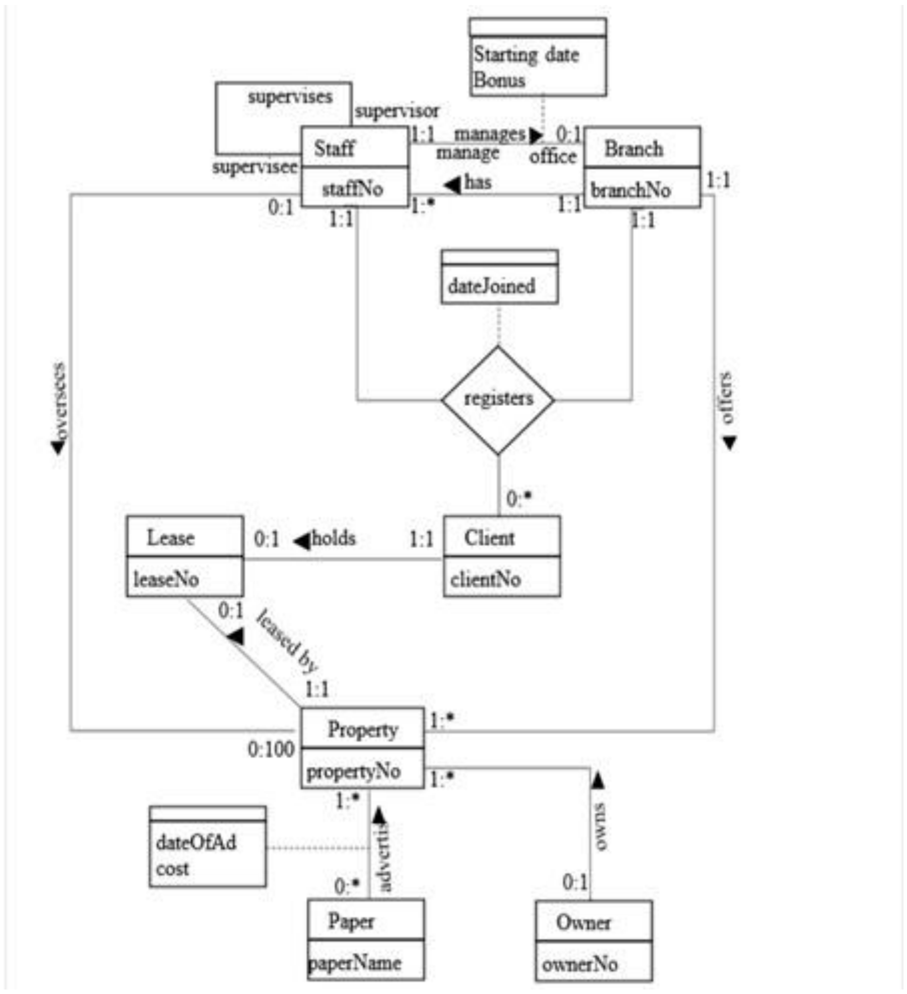
Appendix Figure 77 Veterinary Hospital Case Study (Pol & Ahuja, 2007, p. 76)



Appendix Figure 78 Model Answer for Veterinary Hospital provided by Database Designer[21]

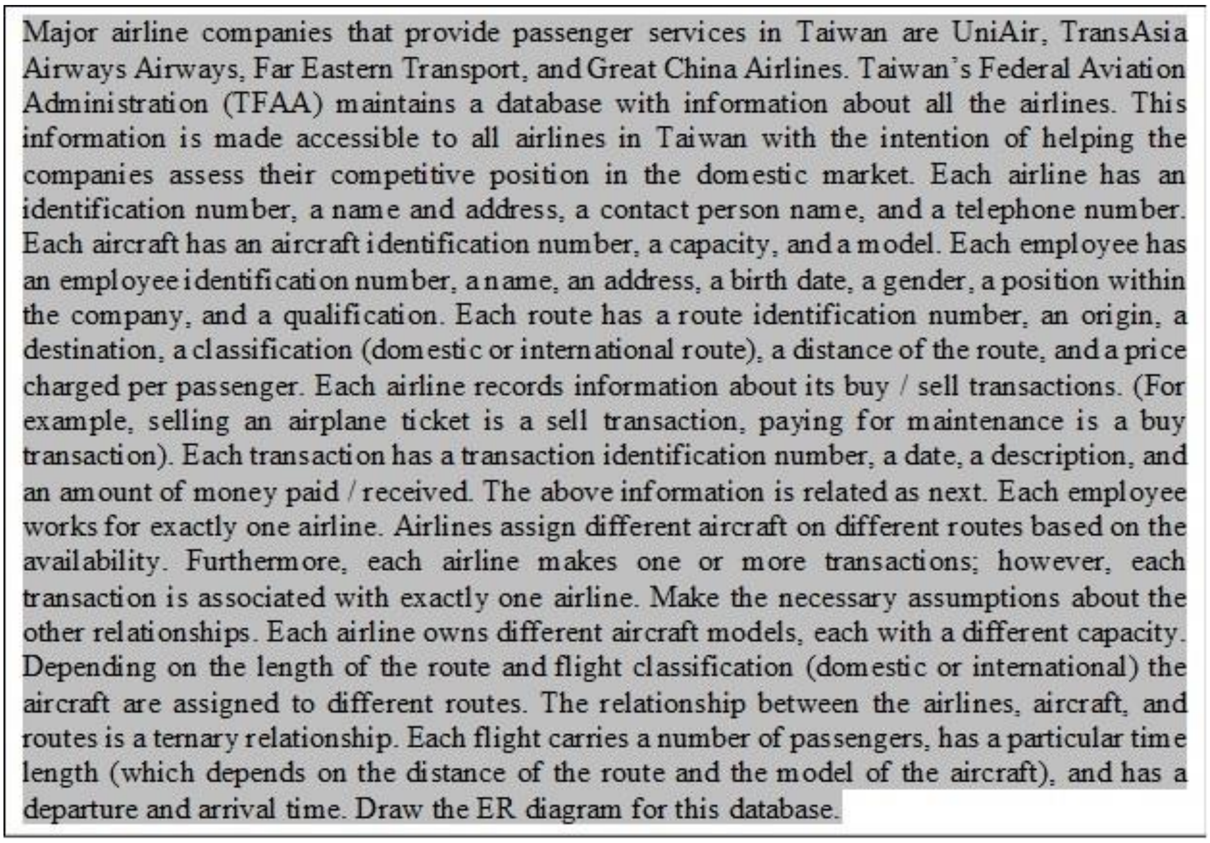

DreamHome has branch officies in cities throughout the United Kingdom. Each branch office is allocated members of staff including a manager to manage the operations of the office. The data held on a branch office includes a unique branch number, address (street, city, and postcode), telephone numbers (up to a maximum of three), and the name of the member of staff who currently manges the office. Additional data is held on each manager, which includes the date that the Manager assumed his or her position at the current branch office, and a monthly bonus payment based upon his or her performance in the property for rental market. Members of staff with the role of supervisor are responsible for the day to day activities of an allocated group of staff called assistants (up to a maximum of 10, at any one time). Not all members of staff are assigned to a supervisor. The data stored on each member of staff includes staff number, name, address, position, salary, name of supervisor (where applicable), and the details of the branch office at which a member of staff is currently working. The staff number is unique across all branches of DreamHome. Each branch office offers a range of properties for hiring. The data stored on each property includes property number, address (street, city, postcode), type, number of rooms, monthly hiring, and the details of the property owner. The property number is unique across all branch offices. The management of a property is assigned to a member of staff whenever it is rented out or requires to be rented out. A member of staff may manage a maximum of 100 properties for hire at any one time. The details of property owners are also stored. There are two main types of property owner : private owners and business owners. The data stored on private owners includes owner number, name, address, and telephone number. The data stored on business owners includes name of business, type of business, address, telephone number, and contact name. DreamHome refers to members of the public interested in hiring property as clients. To become a client, a person must first register at a branch office of DreamHome. The data stored on clients includes client number, name, telephone number, preferred type of accommodation, and the maximum hire the client is prepared to pay. Also stored is the name of the member of staff who processed the registration, the date the client joined, and some details on the branch office at which the client registered. The client number is unique across all DreamHome branches. When a property is rented out, a lease is drawn up between the client and the property. The data detailed on the lease includes lease number, client number, name and address, property number and address, monthly hiring, method of payment, an indication of whether the deposit has been paid (deposit is calculated as twice the monthly hiring), duration of lease, and the date the lease period is to start and finish. When required, the details of properties for hiring are advertised in local and national newspapers. The data stored includes the property number, address, type, number of rooms, hiring, the date advertised, the name of the newspaper, and the cost. The data stored on each newspaper includes the newspaper name, address, telephone number, and contact name.

Appendix Figure 79 DreamHome Case Study (Connolly & Begg, 2015, p. A-1)

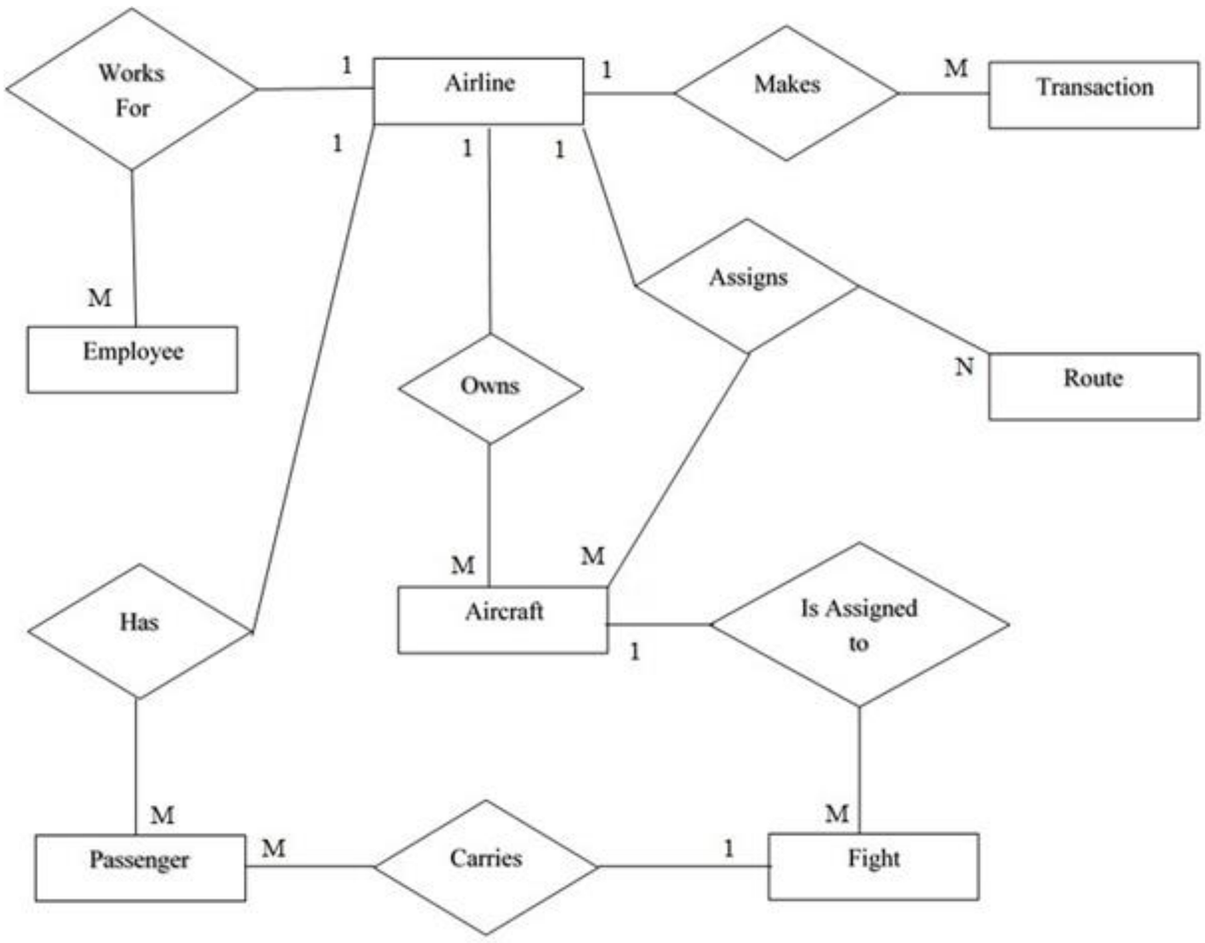


Appendix Figure 80 Model Answer for DreamHome Case Study[27]

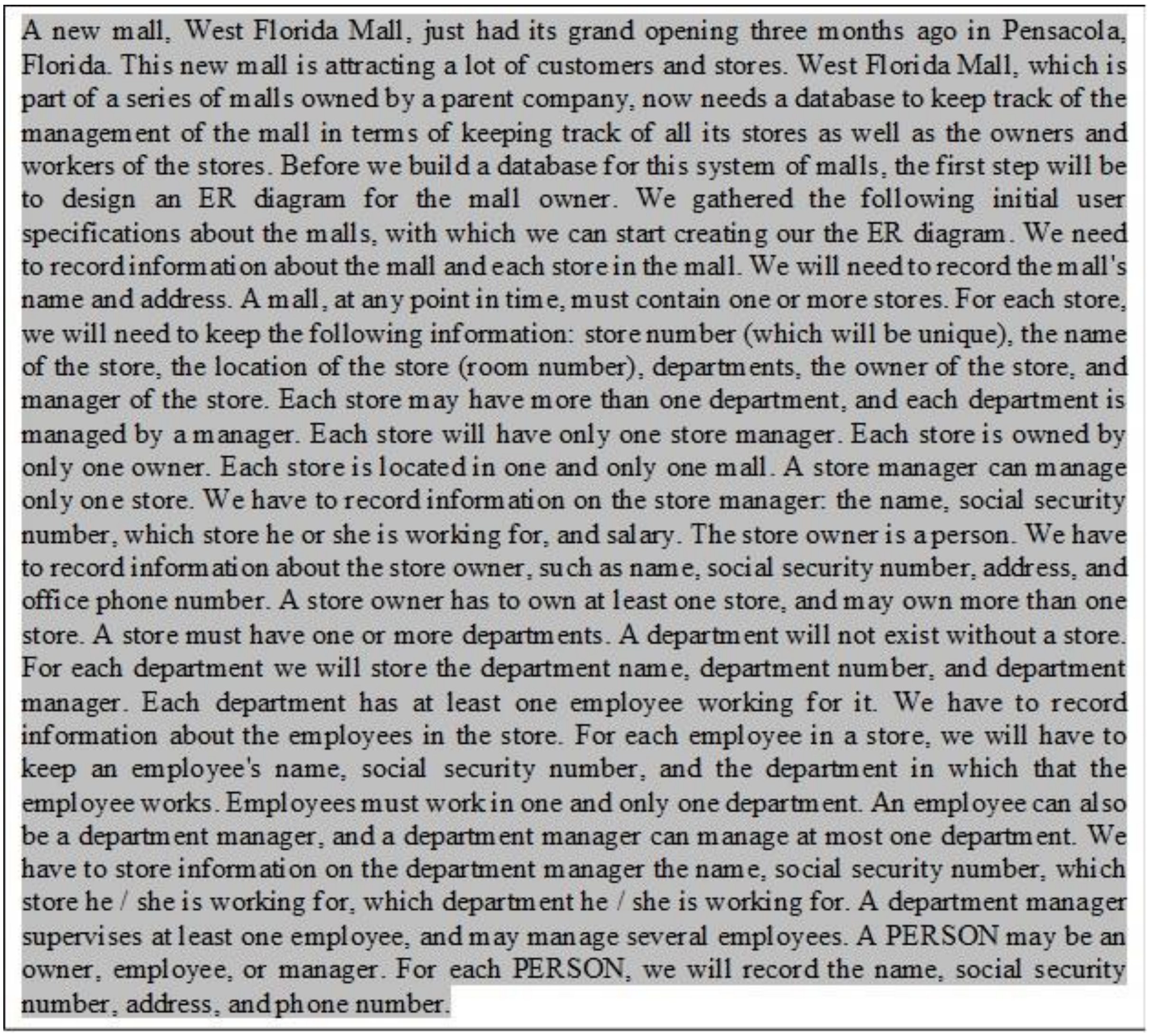
[27] http://www.chegg.com/homework-help/dreamhome-case-studycreate-relational-schema-branch-user-vie-chapter-17-problem-9e-solution-9780321523068-exc


Major airline companies that provide passenger services in Taiwan are UniAir, TransAsia Airways Airways, Far Eastern Transport, and Great China Airlines. Taiwan's Federal Aviation Administration (TFAA) maintains a database with information about all the airlines. This information is made accessible to all airlines in Taiwan with the intention of helping the companies assess their competitive position in the domestic market. Each airline has an identification number, a name and address, a contact person name, and a telephone number. Each aircraft has an aircraft identification number, a capacity, and a model. Each employee has an employee identification number, a name, an address, a birth date, a gender, a position within the company, and a qualification. Each route has a route identification number, an origin, a destination, a classification (domestic or international route), a distance of the route, and a price charged per passenger. Each airline records information about its buy / sell transactions. (For example, selling an airplane ticket is a sell transaction, paying for maintenance is a buy transaction). Each transaction has a transaction identification number, a date, a description, and an amount of money paid / received. The above information is related as next. Each employee works for exactly one airline. Airlines assign different aircraft on different routes based on the availability. Furthermore, each airline makes one or more transactions; however, each transaction is associated with exactly one airline. Make the necessary assumptions about the other relationships. Each airline owns different aircraft models, each with a different capacity. Depending on the length of the route and flight classification (domestic or international) the aircraft are assigned to different routes. The relationship between the airlines, aircraft, and routes is a ternary relationship. Each flight carries a number of passengers, has a particular time length (which depends on the distance of the route and the model of the aircraft), and has a departure and arrival time. Draw the ER diagram for this database.

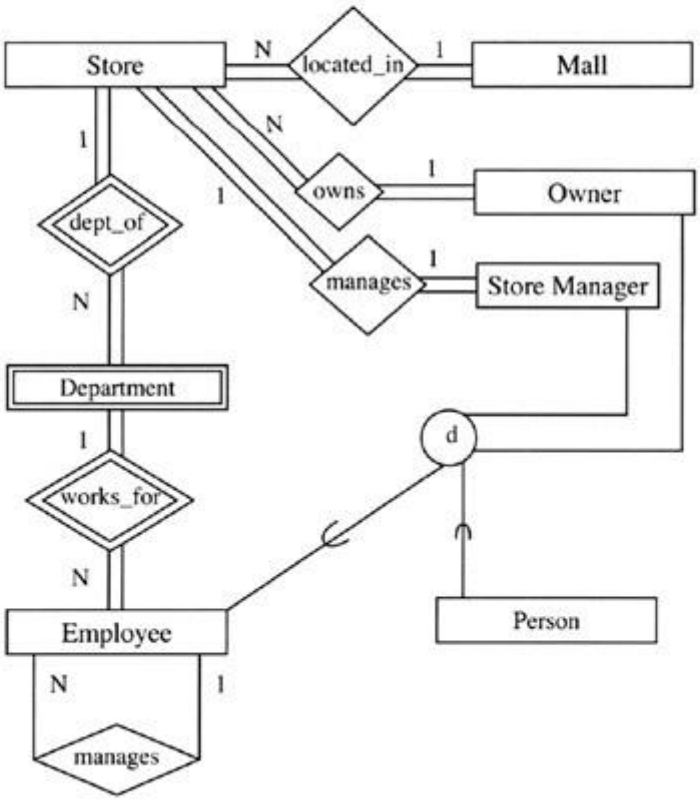
Appendix Figure 81 Airline Case Study (Pol & Ahuja, 2007, p. 74)



Appendix Figure 82 A Model Answer for Airlines Case Study Provided by Database Designer[21]

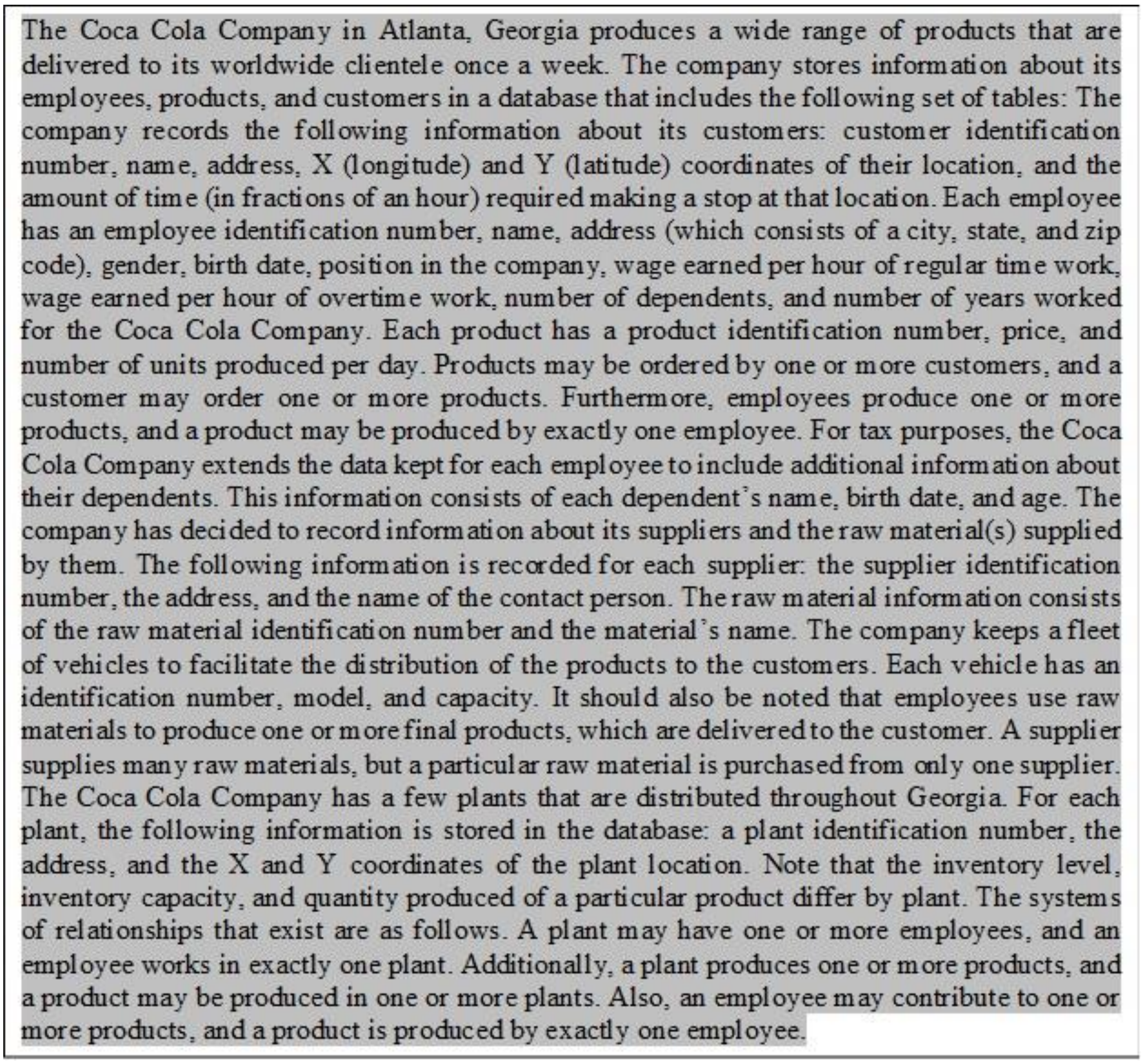

A new mall, West Florida Mall, just had its grand opening three months ago in Pensacola, Florida. This new mall is attracting a lot of customers and stores. West Florida Mall, which is part of a series of malls owned by a parent company, now needs a database to keep track of the management of the mall in terms of keeping track of all its stores as well as the owners and workers of the stores. Before we build a database for this system of malls, the first step will be to design an ER diagram for the mall owner. We gathered the following initial user specifications about the malls, with which we can start creating our the ER diagram. We need to record information about the mall and each store in the mall. We will need to record the mall's name and address. A mall, at any point in time, must contain one or more stores. For each store, we will need to keep the following information: store number (which will be unique), the name of the store, the location of the store (room number), departments, the owner of the store, and manager of the store. Each store may have more than one department, and each department is managed by a manager. Each store will have only one store manager. Each store is owned by only one owner. Each store is located in one and only one mall. A store manager can manage only one store. We have to record information on the store manager: the name, social security number, which store he or she is working for, and salary. The store owner is a person. We have to record information about the store owner, such as name, social security number, address, and office phone number. A store owner has to own at least one store, and may own more than one store. A store must have one or more departments. A department will not exist without a store. For each department we will store the department name, department number, and department manager. Each department has at least one employee working for it. We have to record information about the employees in the store. For each employee in a store, we will have to keep an employee's name, social security number, and the department in which that the employee works. Employees must work in one and only one department. An employee can also be a department manager, and a department manager can manage at most one department. We have to store information on the department manager the name, social security number, which store he / she is working for, which department he / she is working for. A department manager supervises at least one employee, and may manage several employees. A PERSON may be an owner, employee, or manager. For each PERSON, we will record the name, social security number, address, and phone number.

Appendix Figure 83 Florida Mall Case Study (Bagui & Earp, 2012, pp. 96-99)

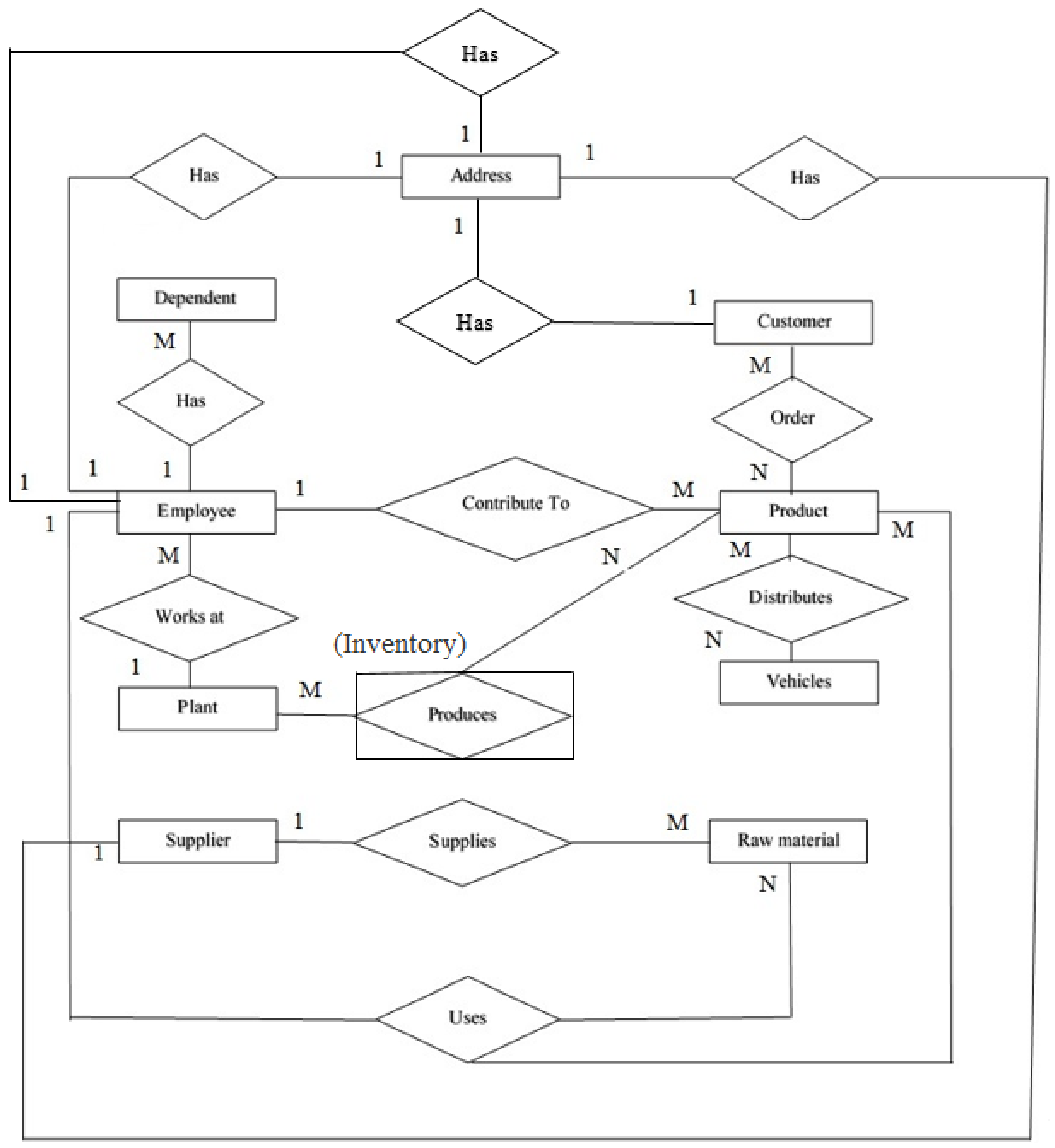


Appendix Figure 84 Model Answer for Florida Mall Case Study[28]

[28] http://dbgroup.eecs.umich.edu/timber/mct/er10.html


The Coca Cola Company in Atlanta, Georgia produces a wide range of products that are delivered to its worldwide clientele once a week. The company stores information about its employees, products, and customers in a database that includes the following set of tables: The company records the following information about its customers: customer identification number, name, address, X (longitude) and Y (latitude) coordinates of their location, and the amount of time (in fractions of an hour) required making a stop at that location. Each employee has an employee identification number, name, address (which consists of a city, state, and zip code), gender, birth date, position in the company, wage earned per hour of regular time work, wage earned per hour of overtime work, number of dependents, and number of years worked for the Coca Cola Company. Each product has a product identification number, price, and number of units produced per day. Products may be ordered by one or more customers, and a customer may order one or more products. Furthermore, employees produce one or more products, and a product may be produced by exactly one employee. For tax purposes, the Coca Cola Company extends the data kept for each employee to include additional information about their dependents. This information consists of each dependent's name, birth date, and age. The company has decided to record information about its suppliers and the raw material(s) supplied by them. The following information is recorded for each supplier: the supplier identification number, the address, and the name of the contact person. The raw material information consists of the raw material identification number and the material's name. The company keeps a fleet of vehicles to facilitate the distribution of the products to the customers. Each vehicle has an identification number, model, and capacity. It should also be noted that employees use raw materials to produce one or more final products, which are delivered to the customer. A supplier supplies many raw materials, but a particular raw material is purchased from only one supplier. The Coca Cola Company has a few plants that are distributed throughout Georgia. For each plant, the following information is stored in the database: a plant identification number, the address, and the X and Y coordinates of the plant location. Note that the inventory level, inventory capacity, and quantity produced of a particular product differ by plant. The systems of relationships that exist are as follows. A plant may have one or more employees, and an employee works in exactly one plant. Additionally, a plant produces one or more products, and a product may be produced in one or more plants. Also, an employee may contribute to one or more products, and a product is produced by exactly one employee.

Appendix Figure 85 Coca Cola Case Study (Pol & Ahuja, 2007, p. 71)



Appendix Figure 86 Model Answer for Coca Cola Case Study provided by Database Designer[21]

References

Abdouli, M., Karaa, W. B. A., & Ghezala, H. B. (2016). Survey of works that transform requirements into UML diagrams. In *IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA), Towson, MD, USA,* 107-105.

Abdullah, T. N., & Saleem, N. N. (2013). Design a Data Model Diagram from Textual Requirements. *International Journal of Computer Science and Information Security, 11*(6), 7-12.

Aguilera, D., Gómez, C., & Olivé, A. (2012). A method for the definition and treatment of conceptual schema quality issues. In *International Conference on Conceptual Modelling, Florence, Italy.* Retrieved from https://link.springer.com/chapter/10.1007/978-3-642-34002-4_39.

Aguilera, D., Gómez, C., & Olivé, A. (2013). A complete set of guidelines for naming UML conceptual schema elements. *Data & Knowledge Engineering, 88*, 60-74.

Agustini, A., Gamallo, P., & Lopes, G. P. (2003). Selection restrictions acquisition for parsing improvement. In *Proceedings of the Applications of Prolog, 14th international conference on Web knowledge management and decision support, Tokyo, Japan*. Retrieved from https://link.springer.com/book/10.1007/3-540-36524-9.

Al-Btoush, A. A.-S. (2015). Extracting Entity Relationship Diagram (ERD) from English Sentences. *International Journal of Database Theory and Application, 8*(2), 235-244.

Al-Masree, H. K. (2015). Extracting Entity Relationship Diagram (ERD) from relational database schema. *International Journal of Database Theory and Application, 8*(3), 15-26.

Al Balushi, T. H., Sampaio, P. R. F., & Loucopoulos, P. (2013). Eliciting and prioritizing quality requirements supported by ontologies: a case study using the ElicitO framework and tool. *Expert Systems, 30*(2), 129-151.

Al Omran, F. N. A., & Treude, C. (2017). Choosing an NLP library for analyzing software documentation: A systematic literature review and a series of experiments. In *14th International Conference on Mining Software Repositories.* 187-197.

Alexander, C. (1979). *The timeless way of building*. New York: Oxford University Press.

Ambriola, V., & Gervasi, V. (2003). *The Circe approach to the systematic analysis of NL requirements*. Pisa: Università di Pisa.

Ambriola, V., & Gervasi, V. (2006). On the systematic analysis of natural language requirements with circe. *Automated Software Engineering, 13*(1), 107-167.

Anthony, S., & Mellarkod, V. (2009). Data modelling patterns: a method and evaluation. In *Proceedings of the Americas Conference on Information Systems (AMCIS), SanFrancisco, California, USA.* Retrived From: https://pdfs.semanticscholar.org/9692/bc23fb8b0fa9f8a4c2d4928505e1341c08cd.pdf.

Antony, S. R., & Batra, D. (2002). CODASYS: a consulting tool for novice database designers. *ACM SIGMIS Database, 33*(3), 54-68.

Anwer, S., & Ikram, N. (2008). A process for goal oriented requirement engineering. In *Proceedings of the IASTED International Conference on Software Engineering, Innsbruck, Austria*, 255-261.

Arora, C., Sabetzadeh, M., Briand, L., & Zimmer, F. (2016). Extracting domain models from natural-language requirements: approach and industrial evaluation. In P*roceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, St Malo, France*, 250-260.

Assawamekin, N., Sunetnanta, T., & Pluempitiwiriyawej, C. (2010). Ontology-based multiperspective requirements traceability framework. *Knowledge and Information Systems, 25*(3), 493-522.

Athenikos, S. J., & Song, I. Y. (2013). CAM: A Conceptual Modelling Framework based on the Analysis of Entity Classes and Association Types. *Journal of Database Management (JDM)*, *24*(4), 51-80.

Atzeni, P. (1999). *Database systems: Concepts, languages & architectures*. London: McGraw-Hill.

Bagui, S. S., & Earp, R. (2012). *Database design using entity-relationship diagrams.* Boca Raton, Fla.: Auerbach.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., & Etzioni, O. (2007). Open Information Extraction from the Web. In *IJCAI International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India*, 2670-2676.

Baroni, M., & Bernardini, S. (2004). BootCaT: Bootstrapping Corpora and Terms from the Web. *In Proceedings of LREC, Lisbon, Portugal*, 1313-1316.

Batra, D. (2005). Conceptual data modelling patterns: Representation and validation. *Journal of Database Management, 16*(2), 84-106.

Batra, D. (2007). Cognitive complexity in data modelling: causes and recommendations. *Requirements Engineering, 12*(4), 231-244. doi: 10.1007/s00766-006-0040-y.

Batra, D., & Antony, S. R. (1994). Novice errors in conceptual database design. *European Journal of Information Systems, 3*(1), 57-69.

Ben Abdessalem Karaa, W., Ben Azzouz, Z., Singh, A., Dey, N., S Ashour, A., & Ben Ghazala, H. (2016). Automatic builder of class diagram (ABCD): an application of UML generation from functional requirements. *Software: Practice and Experience, 46*(11), 1443-1458.

Berendt, B., Hotho, A., Mladenic, D., Van Someren, M., Spiliopoulou, M., & Stumme, G. (2004). A roadmap for web mining: From web to semantic web. In B. Berendt, A. Hotho, D. Mladenič, M. van Someren, M. Spiliopoulou & G. Stumme (Eds). *Web Mining: From Web to Semantic Web. Lecture Notes in Computer Science, vol 3209, Berlin, Heidelberg: Springer,* 1-22.

Bicchierai, I., Bucci, G., Nocentini, C., & Vicario, E. (2012). An ontological approach to systematization of SW-FMEA. In F. Ortmeier & P. Daniel (Eds). *Computer Safety, Reliability, and Security. Lecture Notes in Computer Science, vol 7612, Berlin, Heidelberg: Springer,* 173-184.

Bird, S. (2006). NLTK: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions, Sydney, Australia,* 69-72.

Blaha, M. (2010). *Patterns of data modelling*. Boca Raton, Florida: CRC Press.

Boehm, B. W. (1981). *Software engineering economics (Prentice-Hall Advances in Computing Science and Technology Series)*. Englewood Cliffs: Prentice-Hall.

Bogatyrev, M., & Nuriahmetov, V. (2011). Application of conceptual structures in requirements modelling. In *Proc. of the International Workshop on Concept Discovery in Unstructured Data (CDUD) at the Thirteenth International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing-RSFDGrC, Moscow, Russia,* 11-19.

Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M. A., Maynard, D., & Aswani, N. (2013). TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP), Hissar, Bulgaria,* 83-90.

Bordag, S. (2008). A comparison of co-occurrence and similarity measures as simulations of context. In *Proceedings of the 9th international conference on Computational linguistics and intelligent text processing, Haifa, Israel,* 52-63.

Boukhari, I., Bellatreche, L., & Jean, S. (2012). An ontological pivot model to interoperate heterogeneous user requirements. In *Proceedings of the 5th international conference On Leveraging Applications of Formal Methods, Verification and Validation: Applications and Case studies- Part II, Springer, Berlin, Heidelberg,* 344-358.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (1997). Extensible markup language (XML). *World Wide Web Journal, 2*(4), 27-66.

Brdjanin, D., & Maric, S. (2013). Towards the automated business model-driven conceptual database design. In T. Morzy, T. Härder, & R. Wrembel (Eds). *Advances in Databases and Information Systems. Advances in Intelligent Systems and Computing, vol 186*. Berlin, Heidelberg: Springer.

Breaux, T. D., Antón, A. I., & Doyle, J. (2008). Semantic parameterization: A process for modelling domain descriptions. *ACM Transactions on Software Engineering and Methodology (TOSEM), 18*(2), 5.

Brewster, C., Ciravegna, F., & Wilks, Y. (2002). User-centred ontology learning for knowledge management. In *International Conference on Application of Natural Language to Information Systems(NLDB), Springer, Berlin, Heidelberg,* 203-207.

Brickley, D., & Guha, R. V. (2004). *RDF vocabulary description language 1.0: RDF schema*. Retrieved from https://www.w3.org/TR/rdf-schema/.

Brill, D. (1993). *LOOM reference manual version 2.0.* Los Angeles, California, USA: University of Southern California.

Brill, E. (1992). A simple rule-based part of speech tagger. In *Proceedings of the third conference on applied natural language processing, Trento, Italy,* 152-155.

Btoush, E. S., & Hammad, M. M. (2015). Generating ER Diagrams from Requirement Specifications Based On Natural Language Processing. *International Journal of Database Theory and Application, 8*(2), 61-71.

Buchholz, E., Cyriaks, H., Düsterhöft, A., Mehlan, H., & Thalheim, B. (1995). Applying a natural language dialogue tool for designing databases. In *Proceedings of the First International Workshop on Applications of Natural Language to Databases (NLDB), Versailles, France,* 119-133.

Budanitsky, A. (1999). *Lexical semantic relatedness and its application in natural language processing.* Technical Report Computer Systems Research Group (CSRG): University of Toronto.

Buitelaar, P., Cimiano, P., & Magnini, B. (2005). Ontology learning from text: An overview. In P. Buitelaar, P. Cimiano, & B. Magnini (Eds.). *Ontology learning from text: Methods, evaluation and applications. Frontiers in Artificial Intelligence and Applications, Vol 123 Amsterdam: IOS Press,* 3-12..

Burg, J., & Van de Riet, R. (1996). Analyzing informal requirements specifications: a first step towards conceptual modelling. In R.P. van de Riet. J.F.M. Burg & A.J. van der Vos (Eds.). *Applications of Natural Language to Information Systems: Proceedings of the Second International Workshop (NLDB), Amsterdam, The Netherlands,* 15-27.

Burg, J., & van de Riet, R. (1998). Color-x: Using knowledge from wordnet for conceptual modelling. In C. Fellbaum & G. Miller (Eds.). *WordNet, An Electronic Lexical Database, Cambridge, MA: MIT Press,* 353-377.

Cardei, I., Fonoage, M., & Shankar, R. (2008). Model based requirements specification and validation for component architectures. In *IEEE 2nd Annual Systems Conference, Montreal, Canada*, 1-8.

Carter, J. (2003). *Database design and programming with Access, SQL, Visual Basic and ASP* (2nd ed.). London: McGraw-Hill Education.

Castañeda, V., Ballejos, L. C., & Caliusco, M. L. (2012). Improving the Quality of Software Requirements Specifications with Semantic Web Technologies. *WER*. Retrieved from http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER12/paper_4.pdf.

Castro, L., Baiao, F., & Guizzardi, G. (2009). A survey on Conceptual Modelling from a Linguistic Point of View. *Technical Reports of the Applied Informatics Department of UNIRIO, 19,* 3-12.

Castro, L., Baião, F., & Guizzardi, G. (2010). A linguistic approach to conceptual modelling with semantic types and ontoUML. In *14th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW), Vitória, Brazil,* 215-224.

Chaiyasut, P., & Shanks, G. (1994). Conceptual data modelling process: a study of novice and expert data modellers. In *1st International Conference on Object-Role Modelling, University of Queensland, Australia*, 310-323.

Chan, K. Y. (2017). *An algorithm for Finding a Relationship Between Entities: Semi-Automated Schema Integration Approach* (PhD thesis). Seoul National University Graduate School, Seoul.

Chen, P. P. S. (1976). The entity-relationship model-toward a unified view of data. *ACM Trans. Database Syst., 1*(1), 9-36. doi: 10.1145/320434.320440.

Chen, P. P. S. (1983). English sentence structure and entity-relationship diagrams. *Information Sciences, 29*(2), 127-149.

Chen, P. P. S. (1997). English, Chinese and ER diagrams. *Data & Knowledge Engineering, 23*(1), 5-16.

Chen, Z. Y. (2006). *Formalization and classification of product requirements using axiomatic theory of design modelling* (Masters thesis). Concordia University, Montreal, Canada.

Chicaiza, J., López, J., Piedra, N., Martínez, O., & Tovar, E. (2010). Usage of social and semantic web technologies to design a searching architecture for software requirement artefacts. *IET software, 4*(6), 407-417.

Chioasca, E.-V. (2015). *Automatic Construction of Conceptual Models to Support Early Stages of Software Development* (PhD thesis). University of Manchester, Manchester.

Choobineh, J., & Lo, A. W. (2004). CABSYDD: Case-based system for database design. *Journal of management information systems, 21*(3), 281-314.

Christopher, A. (1979). *The Timeless Way of Building.* New York: Oxford University Press.

Cimiano, P., Pivk, A., Schmidt-Thieme, L., & Staab, S. (2005). Learning taxonomic relations from heterogeneous sources of evidence. In Buitelaar P, Cimiano P, Magnini B, (eds). *Ontology Learning from Text: Methods, Evaluation and Applications.* Frontiers in Artificial Intelligence, *Amsterdam: IOS Press,* 59-73.

Conesa, J., Storey, V. C., & Sugumaran, V. (2010). Usability of upper level ontologies: The case of ResearchCyc. *Data & Knowledge Engineering, 69*(4), 343-356.

Connolly, T. M., & Begg, C. (2015). *Database Systems: practical approach to design, implementation, and management* (6th ed.). Harlow: Pearson Education Limited.

Corcho, O., Fernández-López, M., & Gómez-Pérez, A. (2003). Methodologies, tools and languages for building ontologies. Where is their meeting point? *Data & Knowledge Engineering, 46*(1), 41-64.

Cunningham, H. (2002). GATE, a general architecture for text engineering. *Computers and the Humanities, 36*(2), 223-254.

Currim, S. (2008). *Towards improving conceptual modelling: An examination of common errors and their underlying reasons* (PhD thesis). The University of Arizona, Arizona, USA.

Dalianis, H. (1992). Natural language discourse generation in a support tool for conceptual modelling. In *Third Nordic Conference on Text Comprehension in Man and Machine, NOTEX-92, Link ping, Sweden,* 21-23.

Daramola, O., Sindre, G., & Moser, T. (2012). Ontology-based support for security requirements specification process. *Lecture Notes in Computer Science (LNCS), 7567,* 194-206.

Daramola, O., Stålhane, T., Omoronyia, I., & Sindre, G. (2013). Using ontologies and machine learning for hazard identification and safety analysis. In: W. Maalej & A. K. Thurimella (Eds.). *Managing requirements knowledge , Heidelberg: Springer,* 117-141.

Davis, C., Jajodia, S., Ng, P., & Yeh, R. (1983). ER—A historical perspective and future directions. In *Proceedings of the Third International Conference on the Entity-Relationship Approach to Software Engineering, Anaheim, California, USA,* 71-77.

De Marneffe, M.-C., & Manning, C. D. (2008). *Stanford typed dependencies manual: Technical report, Stanford University*. Retrieved from https://nlp.stanford.edu/software/dependencies_manual.pdf.

Dehne, F., Steuten, A., & van de Riet, R. P. (2001). WordNet++: A lexicon for the Color-X-method. *Data & Knowledge Engineering, 38*(1), 3-29.

Dermeval, D., Vilela, J., Bittencourt, I. I., Castro, J., Isotani, S., Brito, P., & Silva, A. (2016). Applications of ontologies in requirements engineering: a systematic review of the literature. *Requirements Engineering, 21*(4), 405-437.

Dey, D., Storey, V. C., & Barron, T. M. (1999). Improving database design through the analysis of relationships. *ACM Transactions on Database Systems (TODS), 24*(4), 453-486.

Du, S. (2008). *On the use of natural language processing for automated conceptual data modelling* (PhD thesis). University of Pittsburgh. Retrieved from http://d-scholarship.pitt.edu/8965/1/du-siqing.pdf.

Du, S., & Metzler, D. P. (2006). An automated multi-component approach to extracting entity relationships from database requirement specification documents. In *International conference on application of natural language to information systems (NLDB), Klagenfurt, Austria*. Retrieved from https://link.springer.com/book/10.1007/11765448.

Dullea, J., Song, I.-Y., & Lamprou, I. (2003). An analysis of structural validity in entity-relationship modelling. *Data & Knowledge Engineering, 47*(2), 167-205. doi: https://doi.org/10.1016/S0169-023X(03)00049-1.

Elbendak, M., Vickers, P., & Rossiter, N. (2011). Parsed use case descriptions as a basis for object-oriented class model generation. *Journal of Systems and Software, 84*(7), 1209-1223.

Elbendak, M. E. (2011). *Requirements-driven Automatic Generation of Class Models* (PhD thesis). Northumbria Univeristy, Newcastle upon Tyne.

El-Ghalayini, H., Odeh, M., & McClatchey, R. (2006). Engineering conceptual data models from domain ontologies: A critical evaluation. In *4th International Conference on Computer Science and Information Technology (CSIT), Amman, Jordan*. Retrieved from https://arxiv.org/abs/cs/0601119.

Elmasri, R., & Navathe, S. (2017). *Fundamentals of database systems* (7th ed.). Boston: Pearson.

Farquhar, A., Fikes, R., & Rice, J. (1997). The ontolingua server: A tool for collaborative ontology construction. *International Journal of Human-Computer Studies, 46*(6), 707-727.

Faure, D., & Nédellec, C. (1998). Asium: Learning subcategorization frames and restrictions of selection. In *Proceedings of the 10th Conference on Machine Learning (ECML), Chemnitz, Germany*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.4927.

Fayad, M. S., D., & Johnson, R. (1997). *Object-oriented Application Frameworks: Problem and Perspectives.* NY: Wiley Publishing.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database.* London: The Mitt Press.

Fliedl, G., Kop, C., Mayerthaler, W., Mayr, H. C., & Winkler, C. (2000). Guidelines for NL-Based requirements specifications in NIBA (NLDB). In *International Conference on Application of Natural Language to Information Systems, Versailles, France,* 251-264.

Fliedi, G., Kop, C., Mayerthaler, W., Mayer, H. C., & Winkler, C. (1996). NTS-based derivation of KCPM cardinalities: From natural language to conceptual predesign. In R.P. van de Riet. J.F.M. Burg & A.J. van der Vos (Eds.). *Applications of Natural Language to Information Systems: Proceedings of the Second International Workshop (NLDB), Amsterdam, The Netherlands,* 222-233.

Fowler, M. (1997). *Analysis patterns: reusable object models*. Menlo Park, Calif: Addison Wesley.

Fürst, F., & Trichet, F. (2006). Heavyweight ontology engineering. In *OTM Confederated International Conferences: On the Move to Meaningful Internet Systems, Montpellier, France.* 38-39.

Gamma, E. (1995). *Design patterns: elements of reusable object-oriented software*. Reading, Mass: Addison-Wesley.

Gandhi, R. A., & Lee, S. W. (2011). Discovering multidimensional correlations among regulatory requirements to understand risk. *ACM Transactions on Software Engineering and Methodology (TOSEM), 20*(4), 16.

Gaševic, D., Djuric, D., & Devedžic, V. (2006). *Model driven architecture and ontology development.* Berlin, Heidelberg: Springer Science & Business Media.

Gašević, D., Kaviani, N., & Milanović, M. (2009). Ontologies and software engineering. In S. Staab & R. Studer (Eds.), *Handbook on Ontologies, Berlin: Springer*, 593-615

Gehrke, R. R. J. (2002). *Database Management Systems Solutions Manual.* Retrieved from http://www.cs.princeton.edu/courses/archive/spr00/cs425/soln_from_text_midterm.pdf.

Genesereth, M. R., & Fikes, R. E. (1992). *Knowledge interchange format-version 3.0: reference manual*. Computer Science Department, Stanford University.

Ghaisas, S., & Ajmeri, N. (2013). Knowledge-assisted ontology-based requirements evolution. In W. Maalej & A. K. Thurimella (Eds.), *Managing requirements knowledge, Berlin, Heidelberg: Springer*, 143-167.

Gilberg, R. F. (1985). A Schema Methodology For Large Entity-Relationship Diagrams. In *Proceedings of the Fourth International Conference on Entity-Relationship Approach, Chicago, Illinois, USA*, 320-325.

Giunchiglia, F., & Zaihrayeu, I. (2009). Lightweight ontologies. In L. Liu & M. T. Özsu (Eds.), *Encyclopedia of Database Systems, Berlin, Heidelberg: Springer,* 1613-1619.

Gómez-Pérez, A., & Manzano-Macho, D. (2003). *Deliverable 1.5: A survey of ontology learning methods and tools*. doi: 10.1.1.93.3714.

Gomez, F., Segami, C., & Delaune, C. (1999). A system for the semiautomatic generation of E-R models from natural language specifications. *Data & Knowledge Engineering, 29*(1), 57-81. doi: https://doi.org/10.1016/S0169-023X(98)00032-9.

Grefenstette, G. (1999). Tokenization. In H. van Halteren (Ed.), *Syntactic Wordclass Tagging, Berlin, Heidelberg: Springer*, 117-133.

Grishman, R., & Sundheim, B. (1996). Message understanding conference-6: A brief history. In *Proceedings of the 6th International Conference on Computational Linguistics (COLING), Copenhagen, Denmark,* 466-471.

Gruber, T. R. (1992). *Ontolingua: A mechanism to support portable ontologies*. Stanford University, Knowledge Systems Laboratory Stanford.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition, 5*(2), 199-220.

Han, T., Purao, S., & Storey, V. C. (2008). Generating large-scale repositories of reusable artifacts for conceptual design of information systems. *Decision Support Systems, 45*(4), 665-680.

Harmain, H. M., & Gaizauskas, R. (2003). CM-Builder: A Natural Language-Based CASE Tool for Object-Oriented Analysis. *Automated Software Engineering, 10*(2), 157-181. doi: 10.1023/A:1022916028950.

Hartmann, S., & Link, S. (2007). English sentence structures and EER modelling. In *Proceedings of the fourth Asia-Pacific conference on conceptual modelling - Volume 67, Ballarat, Australia*, 27-35.

Hasegawa, R., Kitamura, M., Kaiya, H., & Saeki, M. (2009). Extracting conceptual graphs from Japanese documents for software requirements modelling. In *Proceedings of the Sixth Asia-Pacific Conference on Conceptual Modelling-Volume 96, Wellington, New Zealand*, 87-96.

Herchi, H. & Abdessalem, W. B. (2012). From user requirements to UML class diagram. In *International Conference on Computer Related Knowledge, Sousse, Tunisia*. Retrieved from http://arxiv.org/abs/1211.0713.

Hay, D. C. (1996). *Data model patterns: conventions of thought*. New York: Dorset House Publishing.

Hendler, J. (2001). Agents and the semantic web. *IEEE Intelligent systems, 16*(2), 30-37.

Hjelm, H., & Volk, M. (2011). Cross-language ontology learning. In W. Wong, W. Lu, & M. Bennamoun (Eds.), *Ontology Learning and Knowledge Discovery Using the Web: Challenges and Recent Advances, Hershey, PA: IGI Global*. 272-297.

Hoffer, J., Prescott, M., & Mcfadden, F. (2004). *Modern database management* (7th ed.). Upper Saddle River, New Jersey: Prentice Hall Press.

Hwang, C. H. (1999). Incompletely and imprecisely speaking: using dynamic ontologies for representing and retrieving information. In *Proceedings of the 6th International Workshop on Knowledge Representation meets Databases (KRDB), Linköping, Sweden*, 14-20.

Ide, N., & Véronis, J. (1998). Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics, 24*(1), 2-40.

Jackson, H. (1982). *Analysing English*. Oxford: Pergman Press.

Jacobson, I. (1992). *Object-oriented software engineering: a use case driven approach*. Wokingham: ACM Press.

Jiang, T., Tan, A.-H., & Wang, K. (2007). Mining generalized associations of semantic relations from textual web content. *IEEE Transactions on Knowledge and Data Engineering, 19*(2), 164-179.

Jiang, X., & Tan, A. H. (2010). CRCTOL: A semantic-based domain ontology learning system. *Journal of the Association for Information Science and Technology, 61*(1), 150-168.

Johannesson, P., & Wohed, P. (1999). The deontic pattern–a framework for domain analysis in information systems design. *Data & Knowledge Engineering, 31*(2), 135-153.

Johnson, R. E., & Foote, B. (1988). Designing reusable classes. *Journal of object-oriented programming, 1*(2), 22-35.

Karp, P. D., Chaudhri, V. K., & Thomere, J. (1999). *XOL: An XML-based ontology exchange language.* Menlo Park, California: SRI International.

Kern, V. M., & Ramos, A. L. (2002). Bridging the gap between natural and information modelling languages: an informal approach to information modelling learning. In *Seventh International Conference on Engineering and Technology Education (INTERTECH), Santos-SP, Brasil*. Retrieved from http://eprints.rclis.org/25202/.

Kifer, M., Lausen, G., & Wu, J. (1995). Logical foundations of object-oriented and frame-based languages. *Journal of the ACM (JACM), 42*(4), 741-843.

Kim, N., Lee, S., & Moon, S. (2008). Formalized Entity Extraction Methodology for Changeable Business Requirements. *Journal of Information Science & Engineering, 24*(3), 649-671.

Kimball, R., & Ross, M. (2002). *The data warehouse toolkit: the complete guide to dimensional modelling*. Hoboken, New Jersey: John Wiley & Sons.

Kof, L., & Pizka, M. (2005). Validating Documentation with Domain Ontologies. In *Proceedings of the fourth conference on New Trends in Software Methodologies, Tools and Techniques (SoMeT),* 126-143.

Kop, C. (2008). Conceptual modelling tool for novice designers. *International Journal of Metadata, Semantics and Ontologies, 3*(2), 151-165.

Kop, C., Fliedl, G., & Mayr, H. C. (2010). From Natural Language Requirements to a Conceptual Model. In *Proceedings of the First International Workshop on Evolution Support for Model-Based Development and Testing (EMDT), Ilmenau, Germany,* 67-73.

Kop, C., & Mayr, H. C. (1998). Conceptual predesign bridging the gap between requirements and conceptual design. In *Proceedings of the Third International Conference on Requirements Engineering, Colorado Springs, Colorado, USA.* Retrieved from https://ieeexplore.ieee.org/document/667813/.

Kotonya, G., & Sommerville, I. (1998). *Requirements engineering: processes and techniques.* New York: Wiley.

Kroha, P., Janetzko, R., & Labra, J. E. (2009). Ontologies in checking for inconsistency of requirements specification. In *Third International Conference on.Advances in Semantic Processing (SEMAPRO), Sliema, Malta,* 32-37.

Kwartler, T. (2017). Text mining in practice with R. New York: John Wiley & Sons, Incorporated.

La-Ongsri, S., & Roddick, J. F. (2015). Incorporating ontology-based semantics into conceptual modelling. Information Systems, 52, 1-20.

Neill, C. J., & Laplante, P. A. (2003). Requirement engineering: the state of the practice. *IEEE Software, 20(6),* 40-45.

Larman, C. (2001). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Lassila, O., & Swick, R. R. (1999). *Resource description framework (RDF) model and syntax specification*. W3C Recommendation. Retrieved from https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

Lavrac, N. & Dzeroski, S. (1994). *Inductive logic programming: techniques and applications.* New York: E. Horwood.

Lecoeuche, R. (2000). Finding comparatively important concepts between texts. In *Proceedings of the Fifteenth IEEE International Conference on Automated Software Engineering (ASE), Grenoble, France,* 55-60.

Lee, S. (2009). Automated Enterprise Data Model by Formulating Requirements. *Journal of Information Technology Applications & Management, 16*(4), 263-283.

Lee, S., & Shin, K.-s. (2010). Requirement-Oriented Entity Relationship Modelling. *Journal of Information Technology Applications and Management, 17*(3), 1-24.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., . . . Christian, B. (2015). DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web, 6*(2), 167-195.

Li, G., Jin, Z., Xu, Y., & Lu, Y. (2011). An engineerable ontology based approach for requirements elicitation in process centered problem domain. *Knowledge science, engineering and management, 7091,* 208-220.

Liao, C., & Palvia, P. C. (2000). The impact of data models and task complexity on end-user performance: an experimental investigation. *International Journal of Human-Computer Studies*, *52*(5), 831-845.

Lima, J. F., Garcia, B. P., Amaral, C. M. G., & Caran, G. M. (2011). Building an ontological model for software requirements engineering. In *International Conference on Enterprise Information Systems, Vilamoura, Portugal*, 228-237.

Lin, D. (1994). PRINCIPAR: an efficient, broad-coverage, principle-based parser. In *Proceedings of the 15th conference on Computational linguistics (COLING), Kyoto, Japan* - Volume 1, 482-488.

Lin, D. (2003). Dependency-based evaluation of MINIPAR. In A Abeille (Ed.), *Treebanks: Building and Using Parsed Corpora, Alphen aan den Rijn, Netherlands: Kluwer*, 317-329.

Lindberg, D. A. B., Humphreys, B. L., & McCray, A. T. (1993). The Unified Medical Language System. *Methods Inf Med, 32(04), 281-291.* doi: 10.1055/s-0038-1634945.

Lindén, K., & Piitulainen, J. O. (2004). Discovering synonyms and other related words. In *Proceedings of 3rd International Workshop on Computational Terminology (COLING), Geneva, Switzerland*, 63-70.

Liu, W., Weichselbraun, A., Scharl, A., & Chang, E. (2005). Semi-automatic ontology extension using spreading activation. *Journal of Universal Knowledge Management*, *1*(1), 50-58.

López, C., Astudillo, H., & Cysneiros, L. M. (2008). Semantic-aided interactive identification of reusable NFR knowledge fragments. In R. Meersman, Z. Tari, & P. Herrero (Eds.), *On the Move to Meaningful Internet Systems: OTM 2008 Workshops. Lecture Notes in Computer Science*, *5333*. Berlin, Heidelberg: Springer.

Luisa, M., Mariangela, F., & Pierluigi, N. I. (2004). Market research for requirements analysis using linguistic tools. *Requirements Engineering*, *9*(1), 40-56. doi: 10.1007/s00766-003-0179-8.

Luk, W. (1989). Building natural language interface to an er database. In *Proceedings of the Eighth International Conference on Enity-Relationship Approach to Database Design and Querying, Toronto, Canada*, 345-360.

Luke, S., & Heflin, J. (2000). SHOE 1.01. *Proposed specification: Shoe Project.* Retrieved from http://www.cs.umd.edu/projects/plus/SHOE/spec.html.

Maedche, A., & Staab, S. (2001). Ontology learning for the semantic web. *IEEE Intelligent systems, 16*(2), 72-79.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). The stanford core nlp natural language processing toolkit. In *Proceedings of 52nd Annual*

Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, Maryland USA, 55-60.

Martínez, P., & García-Serrano, A. (2000). On the automatization of database conceptual modelling through linguistic engineering. In *Proceedings of the 5th International Conference on Application of Natural Language to Information Systems (NLDB), Versailles, France,* 276-287.

Mascardi, V., Cordì, V., & Rosso, P. (2007). A Comparison of Upper Ontologies. *WOA, 2007*, 55-64.

Matuszek, C., Cabral, J., Witbrock, M. J., & DeOliveira, J. (2006). An Introduction to the Syntax and Content of Cyc. In *AAAI Spring Symposium: Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering, Stanford CA*, USA, 44-49.

Medelyan, O., & Witten, I. H. (2005). Thesaurus-based index term extraction for agricultural documents. In *Proceedings of EFITA/WCCA Joint Congress on IT in Agriculture, Vila Real, Portugal*, 1122-1129.

Ménard, P. A., & Ratté, S. (2016). Concept extraction from business documents for software engineering projects. *Automated Software Engineering, 23*(4), 649-686.

Métais, E. (2002). Enhancing information systems management with natural language processing techniques. *Data & Knowledge Engineering, 41*(2), 247-272.

Meyer, B. (1985). On formalism in specifications. *IEEE Software, 1(2), 6-26.*

Meziane, F. (1994). *From English to Formal Specifications* (PhD thesis). University of Salford, Salford.

Meziane, F., Athanasakis, N., & Ananiadou, S. (2008). Generating Natural Language specifications from UML class diagrams. *Requirements Engineering, 13*(1), 1-18.

Meziane, F., & Vadera, S. (2004). Obtaining ER diagrams semiautomatically from natural language specifications. *In Sixth International Conference on Enterprise Information Systems (ICEIS 2004). Porto, Portugal,* 638-642.

Mich, L., & Garigliano, R. (1999). The NL-OOPS project: object oriented modelling using the natural language processing system LOLITA. *In the Proceedings of the 4th International Conference on the Applications of Natural Language to Information Systems (NLDB'99), Klagenfurt*, 215-218.

Mich, L., Giuliani, M. (1995). *From Natural Language to Object Oriented Requirements: an Annotated Bibliography.* Retrieved from http://www.academia.edu/22429523/From_Natural_Language_to_Object_Oriented_requirements_An_annotated_bibliography.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography, 3*(4), 235-244. doi: 10.1093/ijl/3.4.235.

Miller, G. A., Leacock, C., Tengi, R., & Bunker, R. T. (1993). A semantic concordance. In *Proceedings of the workshop on Human Language Technology, Princeton, New Jersey,* 303-308.

Missikoff, M., Navigli, R., & Velardi, P. (2002). Integrated approach to web ontology learning and engineering. *Computer, 35*(11), 60-63.

Miyoshi, H., Sugiyama, K., Kobayashi, M., & Ogino, T. (1996). An overview of the EDR electronic dictionary and the current status of its utilization. In *Proceedings of the 16th conference on computational linguistics (COLING), Copenhagen, Denmark*-Volume 2, 1090-1093.

Monarchi, D. E., & Smith, J. R. (1992). The representation of rules in the ER model. Data & *Knowledge Engineering, 9*(1), 45-61.

Moody, D. L. (2004). Cognitive load effects on end user understanding of conceptual models: An experimental analysis. In *East European Conference on Advances in Databases and Information Systems (ADBIS), Budapest, Hungary*, 129-143.

Moody, D. L., & Shanks, G. G. (1994). What makes a good data model? Evaluating the quality of entity relationship models. In P. Loucopoulos (Ed.), *Entity-Relationship Approach — ER '94 Business Modelling and Re-Engineering: 13th International Conference on the Entity-Relationship Approach, Manchester, United Kingdom, Berlin, Heidelberg: Springer*, 94-111.

Motta, E. (1999). *Reusable Components for Knowledge Modelling: Case Studies in Parametric Design Problem Solving*. Amsterdam, Netherlands: IOS Press.

Niles, I., & Pease, A. (2001). Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS), Ogunquit, ME, USA*, 2-9.

Njike-Fotzo, H., & Gallinari, P. (2004). Learning 'Generalization/Specialization' Relations between Concepts–Application for Automatically Building Thematic Document Hierarchies. In *RIAO '04 Coupling approaches, coupling media and coupling languages for information retrieval, Vaucluse, France*, 143-155.

North, D., Mayfield, M., & Coad, P. (1995). *Object Models: Strategies, Patterns and Applications*. Englewood Cliffs, NJ: Yourdon Press.

O'Hara, T., Mahesh, K., & Nirenburg, S. (1998). Lexical acquisition with WordNet and the Mikrokosmos Ontology. In *Proceedings of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems, Montreal, Canada*, 94-101.

Oliveira, A., Pereira, F. C., & Cardoso, A. (2001). Automatic reading and learning from text. In *Proceedings of the international symposium on artificial intelligence (ISAI), Kolhapur, India,* 302-310.

Omar, N., Hanna, J. R. P, & McKevitt, P. (2004). Heuristic-based entity-relationship modelling through natural language processing. In *Proc. of the 15th Artificial Intelligence and Cognitive Science Conference (AICS), Galway-Mayo Institute of Technology (GMIT), Castlebar, Ireland*, 302-313.

Omar, N., Hanna, P., & McKevitt, P. (2006). Semantic analysis in the automation of ER modelling through natural language processing. In *International Conference on Computing & Informatics (ICOCI), Kuala Lumpur, Malaysia*, 441-446.

Omar, N., Muhammad, N. A & Yahya, Y. (2007). The Use of Semantic Heuristics in the Automation of ER Modelling. In *Proceedings of the International Conference on Electrical Engineering and Informatics, Institut Teknologi Bandung, Indonesia*. Retrieved from http://publication.gunadarma.ac.id/bitstream/123456789/655/1/C-15.pdf.

Omer, M., & Wilson, D. (2015). Implementing a Database from a Requirement Specification. World Academy of Science, Engineering and Technology. *International Journal of Computer, Electrical, Automation, Control and Information Engineering, 9*(1), 33-41.

Osborne, M., & MacNish, C. (1996). Processing natural language software requirement specifications. In *Proceedings of the Second International Conference on Requirements Engineering, Colorado Springs, Colorado, USA*, 229-233.

Overmyer, S. P., Lavoie, B., & Rambow, O. (2001). Conceptual modelling through linguistic analysis using LIDA. In *Proceedings of the 23rd international conference on Software engineering, Eden Roc Renaissance, Miami Beach, USA*, 401-410.

Paek, Y.-K., Seo, J., & Kim, G.-C. (1996). An expert system with case-based reasoning for database schema design. *Decision Support Systems, 18*(1), 83-95.

Pan, J. Z., Staab, S., Aßmann, U., Ebert, J., & Zhao, Y. (2012). *Ontology-driven software development*. Berlin, Heidelberg: Springer Science & Business Media.

Parsons, J., & Saunders, C. (2004). Cognitive heuristics in software engineering applying and extending anchoring and adjustment to artifact reuse. *IEEE Transactions on Software Engineering, 30*(12), 873-888.

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). WordNet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at HLT-NAACL, Boston, Massachusetts,* 38-41.

Pinto, A., Gonçalo Oliveira, H., & Oliveira Alves, A. (2016). Comparing the performance of different NLP toolkits in formal and social media text. Paper presented at the *5th Symposium on Languages, Applications and Technologies (SLATE).* doi: 10.4230/OASIcs.SLATE.2016.3.

Pires, P. F., Delicato, F. C., Cóbe, R., Batista, T., Davis, J. G., & Song, J. H. (2011). Integrating ontologies, model driven, and CNL in a multi-viewed approach for requirements engineering. *Requirements Engineering, 16*(2), 133-160.

Pohl, K. (1993). The three dimensions of requirements engineering. In *International Conference on Advanced Information Systems Engineering, Paris, France,* 275-292.

Pol, A. A., & Ahuja, R. K. (2007). *Developing Web-Enabled Decision Support Systems Using Access VB.NET and ASP.NET*. Belmont, Mass., USA: Dynamic Ideas Llc.

Polpinij, J. (2009). An ontology-based text processing approach for simplifying ambiguity of requirement specifications. In *IEEE Asia-Pacific.Services Computing Conference (APSCC), Singapore*, 219–226.

Pree, W. (1994). *Design patterns for object-oriented software development*. Wokingham: Addison-Wesley.

Presland, S. G. (1986). *The analysis of natural language requirements documents* (PhD thesis). University of Liverpool, Liverpool.

Pulido, J., Ruiz, M., Herrera, R., Cabello, E., Legrand, S., & Elliman, D. (2006). Ontology languages for the semantic web: A never completely updated review. *Knowledge-Based Systems, 19*(7), 489-497.

Purao, S. (1998). APSARA: a tool to automate system design via intelligent pattern retrieval and synthesis. *ACM SIGMIS Database, 29*(4), 45-57.

Purao, S., Storey, V. C., & Han, T. (2003). Improving analysis pattern reuse in conceptual design: Augmenting automated processes with supervised learning. *Information Systems Research, 14*(3), 269-290.

Reinhartz-Berger, I., Sturm, A., & Wand, Y. (2011). External variability of software: classification and ontological foundations. In M. Jeusfeld, L. Delcambre, & T. W. Ling (Eds.), *Conceptual Modelling – ER 2011.Lecture Notes in Computer Science, 6998, Berlin, Heidelberg: Springer,* 275-289.

Riechert, T., & Berger, T. (2009). Leveraging semantic data wikis for distributed requirements elicitation. In *Workshop on Wikis for Software Engineering (WIKIS4SE), at 31st International Conference on Software Engineering (ICSE), IEEE Computer Societ,y Vancouver, Canada*, 7-13.

Rob, P., & Coronel, C. (2009). *Database systems: design, implementation, and management* (8th ed.). Boston, Massachusetts: Course Technology Cengage Learning.

Rolland, C. (2013). Conceptual Modelling and Natural Language Analysis. In J. Bubenko, J, Krogstie, Ó. Pastor, B. Pernici, C. Rolland, & A. Sølvberg (Eds.), *Seminal Contributions to Information Systems Engineering, Berlin, Heidelberg: Springer,* 57-61.

Roussey, C., Pinet, F., Kang, M. A., & Corcho, O. (2011). An Introduction to Ontologies and Ontology Engineering. In G. Falquet, C. Métral, J. Teller, & C. Tweed (Eds.), *Ontologies in Urban Development Projects, London: Springer*, 9-38.

Saeki, M., Hayashi, S., & Kaiya, H. (2013). Enhancing goal-oriented security requirements analysis using common criteria-based knowledge. *International Journal of Software Engineering and Knowledge Engineering, 23*(05), 695-720.

Sanderson, M., & Croft, B. (1999). Deriving concept hierarchies from text. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, Californi*a, *USA*. doi: 10.1145/312624.312679.

Santorini, B. (1990). Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision). *University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-47*. Retrieved from https://repository.upenn.edu/cgi/viewcontent.cgi?article=1603&context=cis_reports.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. *In Proceedings of the International Conference on New Methods in Language Processing, Manchester, UK*. Retrieved from http://www.aclweb.org/anthology/A92-1021.

Shahbaz, D. M., Ahsan, S., Shaheen, M., Nawab, R. M. A., & Masood, S. A. (2011). Automatic generation of extended er diagram using natural language processing. *Journal of American Science, 7*(8), 1-10.

Shamsfard, M., & Barforoush, A. A. (2003). The state of the art in ontology learning: a framework for comparison. *The Knowledge Engineering Review, 18*(4), 293-316.

Shamsfard, M., & Barforoush, A. A. (2004). Learning ontologies from natural language texts. *International Journal of Human-Computer Studies, 60*(1), 17-63.

Shinde, R., Kulkarni, R., Patwardhan, M., Sarda, S., & Mantri, P. (2015). Conceptual schema extraction using POS annotations and weighted edit distance algorithm. In *International Conference on Information Processing (ICIP), Pune, India*, 719-724.

Shoval, P., & Shiran, S. (1997). Entity-relationship and object-oriented data modelling—an experimental comparison of design quality. *Data & Knowledge Engineering, 21*(3), 297-315.

Silva, M. J., & Carlson, C. R. (1995). MOODD, a method for object-oriented database design. *Data & Knowledge Engineering, 17*(2), 159-181.

Silverston, L., Inmon, W. H., & Graziano, K. (2001). *The data model resource book: Vol. 2, A library of data models by industry types* (Rev. ed.). New York; Chichester: Wiley.

Simsion, G. (2007). *Data Modelling: Theory and Practice*. Bradley Beach, N.J: Technics Publications.

Slankas, J. (2013). Implementing database access control policy from unconstrained natural language text. In *Proceedings of the 2013 International Conference on Software Engineering (ICSE), San Francisco, CA, USA*, 1357-1360.

Slankas, J. B. (2015). *Implementing Database Access Control Policy from Unconstrained Natural Language Text* (PhD thesis). North Carolina State University, Raleigh, NC, USA.

Song, I.-Y., Yano, K., Trujillo, J., & Luján-Mora, S. (2004). A taxonomic class modelling methodology for object-oriented analysis. *IGI Global*, 216-240. doi:10.4018/978-1-59140-375-3.ch011.

Song, I.-Y., Zhu, Y., Ceong, H., & Thonggoom, O. (2015). Methodologies for Semi-automated Conceptual Data Modelling from Requirements. In *34th International Conference on Conceptual Modelling, Stockholm, Sweden,* 18-31.

Srikant, R., & Agrawal, R. (1995). Mining generalized association rules. In *Proceedings of 1995 International Conference on Very Large Data Bases (VLDB), Zurich, Switzerland,* 407-419.

Storey, V. C. (1993). Understanding semantic relationships. *The VLDB Journal 2(4), 455-488*.

Storey, V. C., Chiang, R. H., Dey, D., Goldstein, R. C., & Sudaresan, S. (1997). Database design with common sense business reasoning and learning. *ACM Transactions on Database Systems (TODS), 22*(4), 471-512.

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web, Banff, Alberta, Canada*, 697 -706.

Suchanek, F. M., Kasneci, G., & Weikum, G. (2008). Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web, 6*(3), 203-217.

Sugumaran, V., & Storey, V. C. (2002). Ontologies for conceptual modelling: their creation, use, and management. *Data & Knowledge Engineering, 42*(3), 251-271. doi: https://doi.org/10.1016/S0169-023X(02)00048-4.

Sugumaran, V., & Storey, V. C. (2006). The role of domain ontologies in database design: An ontology management and conceptual modelling environment. *ACM Trans. Database Syst., 31*(3), 1064-1094. doi: 10.1145/1166074.1166083.

Šuman, S., Jakupović, A., & Kuljanac, F. G. (2016). Knowledge-Based Systems for Data Modelling. *International Journal of Enterprise Information Systems (IJEIS), 12*(2), 1-13.

Szyperski, C. (1997). *Component software: beyond object-oriented programming* (1st ed.). Reading, Mass, USA: ACM Press.

Teorey, T. J. (1999). *Database modelling & design* (3rd ed.). San Francisco: Morgan Kaufmann.

Teorey, T. J., Lightstone, S. S., Nadeau, T., & Jagadish, H. V. (2005). *Database Modelling and Design: Logical Design* (5th ed.). Burlington, USA: Elsevier Science.

Thalheim, B. (2000). *Entity-Relationship Modelling: Foundations of Database Technology*. New York: Springer-Verlag, Inc.

Thonggoom, O. (2011). *Semi-automatic Conceptual Data Modelling Using Entity and Relationship Instance Repositories* (PhD thesis). Drexel University, Philadelphia, PA, USA.

Thonggoom, O., Song, I.-Y., & An, Y. (2011a). EIPW: A Knowledge-Based Database Modelling Tool. In C. Salinesi & O. Pastor (Eds), *Advanced Information Systems Engineering Workshops. CAiSE 2011. Lecture Notes in Business Information Processing*, 83. Berlin, Heidelberg: Springer.

Thonggoom, O., Song, I.-Y., & An, Y. (2011b). Semi-automatic conceptual data modelling using entity and relationship instance repositories. In M. Jeusfeld, L. Delcambre, & T. W. Ling (Eds), *Conceptual Modelling – ER 2011. Lecture Notes in Computer Science,* 6998, *Berlin, Heidelberg: Springer*, 219-232.

Tjoa, A. M., & Berger, L. (1994). Transformation of requirement specifications expressed in natural language into an EER model. In R. Elmasri, V. Kouramajian & B. Thalheim (Eds.), *Entity-Relationship Approach — ER '93. Lecture Notes in Computer Science, 823, Berlin, Heidelberg: Springer,* 206-217.

Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003, Edmonton Canada* -Volume 4, 142-147.

Topi, H. R., V. (2002). Human factors research on data modelling: A review of prior research. *Journal of Database Management, 13(2),* 3-15.

Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Canada,* 173-180.

Tseng, F. S., Chen, A. L., & Yang, W.-P. (1992). On mapping natural language constructs into relational algebra through ER representation. *Data & Knowledge Engineering, 9*(1), 97-118.

Tseng, F. S., & Chen, C.-L. (2006). Extending the UML concepts to transform natural language queries with fuzzy semantics into SQL. *Information and Software Technology, 48*(9), 901-914.

Tseng, F. S., & Chen, C.-L. (2008). Enriching the class diagram concepts to capture natural language semantics for database access. *Data & Knowledge Engineering, 67*(1), 1-29.

Turcato, D., Popowich, F., Toole, J., Fass, D., Nicholson, D., & Tisher, G. (2000). Adapting a synonym database to specific domains. In *Proceedings of the ACL-2000 workshop on Recent advances in natural language processing and information retrieval, Hong Kong,* Volume 11, 645.

Velasco, J. L., Valencia-García, R., Fernández-Breis, J. T., & Toval, A. (2009). Modelling reusable security requirements based on an ontology framework. *Journal of Research and Practice in Information Technology, 41*(2), 119-133.

Sagar, V. B. R. V., & Abirami, S. (2014). Conceptual modelling of natural language functional requirements. *Journal of Systems and Software, 88,* 25-41. doi: http://dx.doi.org/10.1016/j.jss.2013.08.036.

Völker, J., Fernandez Langa, S., & Sure, Y. (2008). Supporting the construction of Spanish legal ontologies with Text2Onto. In P. Casanovas, G. Sartor, N. Casellas & R. Rubino (Eds), *Computable Models of the Law. Languages, Dialogues, Games, Ontologies, Lecture Notes in Artificial Intelligence, 4884, Berlin, Heidelberg: Springer,* 105-112.

Wang, M. (2013). *Requirements Modelling: From Natural Language to Conceptual Models Using Recursive Object Model (ROM) Analysis* (PhD thesis). Concordia University, Montreal, Canada.

Wohed, P. (2000). Conceptual patterns for reuse in information systems analysis. In *International Conference on Advanced Information Systems Engineering. Springer, Berlin, Heidelberg,* 157-175.

Wong, W., Liu, W., & Bennamoun, M. (2007). Tree-traversing ant algorithm for term clustering based on featureless similarities. *Data Mining and Knowledge Discovery, 15*(3), 349-381.

Wong, W., Liu, W., & Bennamoun, M. (2012). Ontology learning from text: A look back and into the future. *ACM Computing Surveys (CSUR), 44*(4), 20. doi:10.1145/2333112.2333115.

Wong, W. Y. (2009). *Learning lightweight ontologies from text across different domains using the web as background knowledge* (PhD thesis). University of Western Australia, Perth, Australia.

Yangarber, R., Grishman, R., Tapanainen, P., & Huttunen, S. (2000). Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th conference on computational linguistics, Saarbrücken, Germany, Volume 2,* 940-946.

Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., & Soderland, S. (2007). Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, Morristown, NJ,* 25-26.

Zapata, C., & Cardona, D. (2008). Heuristic rules for transforming preconceptual schemas into uml 2.0 diagrams: a C# implementation. *Revista Facultad de Ingeniería Universidad de Antioquia* (44), 119-136.

Zapata, C. M., & Arango, F. (2007). An environment for automated UML diagrams obtaining from a controlled language. *DYNA, 74*(153), 223-236.

Zeng, Y., Kim, K.-Y., Raskin, V., Fung, B., & Kitamura, Y. (2013). Modelling, extraction, and transformation of semantics in computer aided engineering systems. *Advanced Engineering Informatics, 27*(1), 1-3.

Zhang, W. (2012). *A Suite of Case Studies in Relational Database Design*. (Masters thesis). McMaster University, Hamilton, Ontario. Retrieved from https://macsphere.mcmaster.ca/bitstream/11375/11862/1/fulltext.pdf.

Zhou, L. (2007). Ontology learning: state of the art and open issues. *Information Technology and Management, 8*(3), 241-252.

Zhou, N., & Zhou, X. (2004). Automatic acquisition of linguistic patterns for conceptual modelling. *INFO 629: Concepts in Artificial Intelligence*. Philadelphia, PA, USA: Drexel University.