META-MODELING FOR MULTI-MODEL INTEGRATION

by

Muhammad Faraz Rafi
Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial Fulfillment of
The Requirements for the Degree
of
Master of Science
Software Engineering

Committee:

____________________	Dr. Alexander H. Levis, Thesis Director
____________________	Dr. Jeff Offutt, Committee Member
____________________	Dr. Sam Malek, Committee Member
____________________	Dr. Hassan Gomaa, Department Chair
____________________	Dr. Lloyd J. Griffiths, Dean, The Volgenau School of Information Technology and Engineering
Date: ____________________	Summer Semester 2010 George Mason University Fairfax, VA

Meta-modeling for Multi-Model Integration

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Muhammad Faraz Rafi
Bachelor of Engineering
NED University of Engineering & Technology, 2006

Director: Alexander H. Levis, Professor
Department of Electrical and Computer Engineering

Summer Semester 2010
George Mason University
Fairfax, VA

ACKNOWLEDGEMENTS

First of all I would thank Allah, the most gracious and most merciful who has enabled me to reach up to this point in life.

I would like to thank my advisor, Prof. Alexander H. Levis, for his patience, guidance and support throughout this thesis. His criticism has always been constructive and has allowed me refine my mistakes, improve my work and taught me how to pursue quality in my work. I would like to thank him for the time and effort he spent on correcting my drafts.

I am also thankful to Dr. Zaidi who has been very helpful throughout this period, whose suggestions and directions have contributed a lot in this thesis. He has always been there to listen to my ideas and help them mature.

I would like to thank my committee members Prof. Jeff Offutt and Dr. Sam Malek for being part of the committee and taking out time to review my thesis during their busy schedules.

I would also like to thank Dr. Lee W. Wagenhals, from the System Architectures Laboratory for the discussions and advices which helped me improve my research work.

TABLE OF CONTENTS

Page

ABSTRACT ... ix

CHAPTER 1: INTRODUCTION ... 1

1.1. Motivation ... 1

1.2. Problem Hypothesis ... 2

1.3. Semantic Integration of Modeling Languages ... 2

1.4. Contributions ... 3

1.5. Thesis Layout ... 4

CHAPTER 2: CONCEPT MAPS ... 5

2.1. What is a Concept Map? ... 5

2.2. Concept Map Construction ... 7

2.3. Using Concept Maps for Knowledge Elicitation ... 8

CHAPTER 3: BAYESIAN NETS AND INFLUENCE NETS ... 9

3.1. Bayesian Networks ... 9

3.2. Influence Nets ... 10

3.3. Concept Mapping Influence Nets ... 12

CHAPTER 4: SOCIAL NETWORKS ... 22

4.1 Social Networks ... 22

4.2 Concept Mapping Social Networks ... 23

4.3 Social Network Analysis ... 28

CHAPTER 5: MODELS AND META MODELS ... 31

5.1. Model ... 31

5.2. Modeling Technique ... 33

5.3. Meta-Model ... 34

5.4. Meta-Modeling ... 35

5.5. Influence Net and Social Network Meta-Models 37
5.6. Multi-Modeling 43
5.7. Meta-Models and Ontologies 44
CHAPTER 6: ONTOLOGY 46
6.1 Ontology 46
6.2 Web Ontology Language (OWL) 48
6.3 Why Construct Ontology? 56
6.4 Use of Ontologies for Semantic Extraction 56
6.5 Ontology and Schema Matching 57
6.6 Steps of Ontology Construction 57
6.7 Influence Net Ontologies 61
6.8 Social Network Ontologies 79
6.9 Construction of the Enriched Ontology 85
CHAPTER 7: WORKFLOW 90
7.1. Overview 90
7.2. Conceptual Modeling Level 93
7.3. Meta-Modeling Level 94
7.4. Ontological Modeling Level 94
7.5. Domain Ontology Instantiation 102
7.6. Definitions 102
CHAPTER 8: CASE STUDY – IRAQI INVASION OF KUWAIT 107
8.1. Scenario (Corpus of Data) 107
8.2. Domain Enriched Ontology 111
CHAPTER 9: APPLICATION - SAF TO MEASURES OF CENTRALITY 117
9.1. Overview 117
9.2. Steps 120
CHAPTER 10: CONCLUSION AND FUTURE WORK 129
10.1. Conclusion 129
10.2. Future Extension 131
REFERENCES 132

LIST OF TABLES

Table Page

TABLE 1: INFLUENCE NET META-MODEL ELEMENTS 40

TABLE 2: SOCIAL NETWORK META-MODEL ELEMENTS 42

TABLE 3: PSEUDO ONTOLOGY & INFLUENCE NET META MODEL ELEMENTS 64

TABLE 4: PSEUDO ONTOLOGY AND SOCIAL NETWORK META-MODEL ELEMENTS 82

TABLE 5: ENRICHED ONTOLOGY 88

TABLE 6: INFLUENCE NET REFACTORED ONTOLOGY ELEMENTS (CONCEPT MAP IMPORTS) 97

TABLE 7: EXPLICIT INFLUENCE NET CONCEPTS IN REFACTORED ONTOLOGY 97

TABLE 8: SOCIAL NETWORK REFACTORED ONTOLOGY ELEMENTS (CONCEPT MAP IMPORTS) 98

TABLE 9: SOCIAL NETWORK ENTITIES 110

TABLE 10: INFLUENCE NET PROPOSITIONS 110

TABLE 11: ACTION PROPOSITION CLASS ELEMENTS IN ENRICHED ONTOLOGY 113

TABLE 12: BELIEF PROPOSITION CLASS ELEMENTS IN ENRICHED ONTOLOGY................. 113

TABLE 13 DECISION PROPOSITION CLASS ELEMENTS IN ENRICHED ONTOLOGY.............. 114

TABLE 14: TOTAL DEGREE CENTRALITY BEFORE AND AFTER SAF.................................. 125

TABLE 15: BETWEENNESS CENTRALITY BEFORE AND AFTER SAF 126

TABLE 16: CLOSENESS CENTRALITY BEFORE AND AFTER SAF 127

LIST OF FIGURES

Figure Page
FIGURE 1: A SAMPLE CONCEPT MAP FOR WHAT IS AN INFLUENCE NET FOCUS QUESTION . 6
FIGURE 2: CONCEPTUAL MODELING LEVEL 8
FIGURE 3: SAMPLE BAYESIAN NETWORK 10
FIGURE 4: SAMPLE INFLUENCE NET 11
FIGURE 5: WHAT AN INFLUENCE NET IS 14
FIGURE 6: WHAT THE CONSTRUCTS OF AN INFLUENCE NET ARE 15
FIGURE 7: CONCEPT MAP OF TOOLS AVAILABLE FOR INFLUENCE NETS 16
FIGURE 8: INFLUENCE NET MODEL FOR IRAQ-KUWAIT CASE STUDY 17
FIGURE 9: PROBABILITY PROFILE FOR FINAL EFFECT NODE 17
FIGURE 10: SAF ALGORITHM RESULTS FOR IRAQ-KUWAIT SCENARIO 18
FIGURE 11: INFLUENCE NET ANALYSES 20
FIGURE 12: PEOPLE INVOLVED IN INFLUENCE NET CONSTRUCTION 21
FIGURE 13: SAMPLE SOCIAL NETWORK 23
FIGURE 14: WHAT A SOCIAL NETWORK IS 24
FIGURE 15: WHAT THE CONSTRUCTS OF A SOCIAL NETWORK ARE 26
FIGURE 16: CONCEPT MAP OF TOOLS AVAILABLE FOR SOCIAL NETWORKS 27
FIGURE 17: SOCIAL NETWORK ANALYSES 28
FIGURE 18: INFLUENCE NET MODEL FOR IRAQ-KUWAIT SCENARIO 32
FIGURE 19: SOCIAL NETWORK MODEL FOR IRAQ-KUWAIT SCENARIO 33
FIGURE 20: META-MODEL OF *ELEMENT* IN UML 35
FIGURE 21: OMG'S FOUR LAYER ARCHITECTURE FOR META-MODELING 36
FIGURE 22: META-MODELING LEVEL 37
FIGURE 23: INFLUENCE NET META MODEL 39
FIGURE 24: SOCIAL NETWORK META MODEL 41
FIGURE 25: CONCEPT OF MULTI-MODELING 43
FIGURE 26: PIZZA ONTOLOGY TBOX 47
FIGURE 27: PIZZA ONTOLOGY 47
FIGURE 28: DATA PROPERTY FOR CLASS *PIZZASIZE* 53
FIGURE 29: DOMAIN AND RANGE OF *HASTOPPING* OBJECT PROPERTY 54
FIGURE 30: ASSERTED CLASS HIERARCHY OF INFLUENCE NET PSEUDO ONTOLOGY 55

FIGURE 31: INFERRED CLASS HIERARCHY OF INFLUENCE NET PSEUDO ONTOLOGY 55
FIGURE 32: FORMALISM SHIFTING 61
FIGURE 33: INFLUENCE NET PSEUDO ONTOLOGY 63
FIGURE 34: MEMBER INSTANCES OF INFLUENCE CLASS 70
FIGURE 35: VALUE OF AN INFLUENCE INSTANCE "MAXIMUM INHIBITION" 70
FIGURE 36: REFACTORED ONTOLOGY CONSTRUCTION 72
FIGURE 37: INFLUENCE NET REFACTORED ONTOLOGY (ADDITIONAL CONCEPTS ONLY).... 73
FIGURE 38: ABILITY CLASS 74
FIGURE 39: ACTION CLASS 75
FIGURE 40: BELIEF CLASS 75
FIGURE 41: DECISION CLASS 76
FIGURE 42: EQUIVALENT CLASSES 76
FIGURE 43: EVENT CLASS 76
FIGURE 44: INTENT CLASS 77
FIGURE 45: PROPOSITION TYPE 77
FIGURE 46: SUBJECT AND OBJECT INDIVIDUALS 79
FIGURE 47: VERB INDIVIDUAL 79
FIGURE 48: SOCIAL NETWORK PSEUDO ONTOLOGY 80
FIGURE 49: SOCIAL NETWORK REFACTORED ONTOLOGY (ADDITIONAL CONCEPTS ONLY) 81
FIGURE 50: ENRICHED ONTOLOGY CLASSES 85
FIGURE 51: SUBJECT, OBJECT CLASSES MAPPED TO AGENT CLASS 87
FIGURE 52: REASONER INFERRED EQUIVALENCES 87
FIGURE 53: SUBJECT, OBJECT, ORGANIZATION AND AGENT AS EQUIVALENT CLASSES..... 87
FIGURE 54: ONTOLOGICAL MODELING LEVEL 89
FIGURE 55: THE WORKFLOW 91
FIGURE 56: PROTÉGÉ 4.0 CLASSES & PROPERTIES FOR THE INFLUENCE NET PSEUDO ONTOLOGY 95
FIGURE 57: GRAPHVIZ DIAGRAM - INFLUENCE NET PSEUDO ONTOLOGY 95
FIGURE 58: PROTÉGÉ 4.0 CLASSES & PROPERTIES FOR THE SOCIAL NETWORK PSEUDO ONTOLOGY 96
FIGURE 59: GRAPHVIZ DIAGRAM - SOCIAL NETWORK PSEUDO ONTOLOGY 96
FIGURE 60: GRAPHVIZ DIAGRAM - INFLUENCE NET ASSERTED REFACTORED ONTOLOGY 98
FIGURE 61: GRAPHVIZ DIAGRAM - INFLUENCE NET INFERRED REFACTORED ONTOLOGY 98
FIGURE 62: GRAPHVIZ DIAGRAM - ASSERTED/INFERRED REFACTORED SOCIAL NETWORK ONTOLOGY 99
FIGURE 63: DETERMINED MAPPINGS IN THE ENRICHED ONTOLOGY 99
FIGURE 64: GRAPHVIZ DIAGRAM - ASSERTED ENRICHED ONTOLOGY 100
FIGURE 65: GRAPHVIZ DIAGRAM - INFERRED ENRICHED ONTOLOGY 101

FIGURE 66: DOMAIN SPECIFIC ONTOLOGY FOR IRAQ-KUWAIT SCENARIO 115
FIGURE 67: ACTION PROPOSITION POPULATED IN A DOMAIN ENRICHED ONTOLOGY 116
FIGURE 68: SUBJECT, OBJECT, AGENT, ORGANIZATION, VERB AND TASKS IN A DOMAIN ENRICHED ONTOLOGY 116
FIGURE 69: SAF TO MEASURES OF CENTRALITY PROCESS FLOW 118
FIGURE 70: INFLUENCE NET MODEL FOR IRAQ-KUWAIT WAR SCENARIO 119
FIGURE 71: SOCIAL NETWORK MODEL FOR IRAQ-KUWAIT SCENARIO 120
FIGURE 72: SAF ALGORITHM EXECUTION 121
FIGURE 73: UPDATED SOCIAL NETWORK MODEL FOR IRAQ-KUWAIT SCENARIO 124

ABSTRACT

META-MODELING FOR MULTI-MODEL INTEGRATION

Muhammad Faraz Rafi, MS

George Mason University, 2010

Thesis Director: Alexander H. Levis

This thesis work proposes a technique based on concept maps, meta-models, and ontologies to develop a theoretical foundation for the use of multiple interacting models in order to determine the valid interaction between different modeling techniques. It can assist a modeler to find out if a model can provide partial answers to queries generated by another one. The modeling techniques considered in this thesis are Social network and Influence net.

A phase by phase approach using concept maps, meta-models, and ontologies based on comparing the ontologies (for each modeling technique) is employed which helps identify the similarities, overlaps, and/or mapping across the models. In the first phase, concept maps are developed for each technique by identifying a set of focus questions. This concept map representation is then formalized using meta-models. The aim of constructing the meta-model is to reveal the structural aspects of the modeling

technique and lay down the foundation for its ontology. A basic ontology is constructed which mirrors the meta-model and serves as the foundation ontology; it does not contain any explicit concepts (related to the modeling technique). As the next step, explicit concepts and relationships are added to this foundation ontology to make it complete. Once the individual ontologies are completed, mapping of concepts across the ontologies is performed. The resulting ontology which contains these explicit concepts and relationships within and across both ontologies is called as *enriched ontology*. The mappings determined with the help of this enriched ontology can then pave the way towards ensuring consistency and exchanging information among the two different types of models.

An example based on an actual event is used for the construction of a Social network and an Influence net model. The construction of both Social network and the Influence net is based on the same corpus of data, but they are constructed in isolation. A domain specific ontology for this case study is instantiated from the enriched ontology. The mappings determined with the help of the enriched ontology help translate the analysis output of the Influence net model and incorporate it into the corresponding Social network. As a result of this incorporation changes made in the Social network can then be analyzed and studied for any revealing information about the network.

CHAPTER 1: INTRODUCTION

1.1. Motivation

Various modeling techniques such as Bayesian networks, Influence nets, Social networks, and Colored Petri nets (CPN) have been very effective in their respective domains; each one has an associated modeling language and an underlying procedure for model construction. Each of the models constructed is derived from or uses the same body of data and the queries answered by these models not only complement each other but also supplement the knowledge required to refine them. Each model can be analyzed with the help of different analysis techniques; for instance, course of action analyses can be performed on Influence nets, structural/behavioral analyses can be performed on CPN models, and measures of centrality (degree, betweenness, and closeness) provide information about each associated node in a social network. The nexus between these models can be explored with the following set of questions which are also the focus of this thesis:

i. What connections or translations exist among the <u>outputs</u> of these analyses?

ii. What <u>overlaps</u> exist among the models?

iii. How to identify any inconsistencies or incompleteness?

Results obtained from analyzing each model provide information about that model only; it would be useful if this information can be utilized to improve other types of models as well. To do that, we need to know the relationships which might exist between the two types of models. For that purpose, we need to know which semantics (concepts) of each modeling technique map/overlap with which concepts of the other. If we can somehow identify those mapped concepts, we might be able to identify inconsistencies or incompleteness using those overlaps. The proposed solution to these questions is based on using an ontological approach originating from concept maps and meta-models.

1.2. Problem Hypothesis

Concepts-maps, meta-models and ontologies can be employed to extract the semantics of various modeling techniques and to define mappings among them. These mappings can then be utilized to determine inconsistencies, incompleteness, ensure accuracy, and exchange information among models constructed with these modeling techniques using the same corpus of data.

1.3. Semantic Integration of Modeling Languages

From the software perspective, integration of modeling languages at both syntactic and semantic levels has been a very active research area, specially the use of meta-models and ontologies to achieve model integration at the semantic level [1]. Most of the research work in this area has a focus on integration of software modeling languages while - in this thesis the objective is to enable the semantically correct interoperation of modeling techniques. Kappel et al. [1] make use of meta-models such as Ecore and Ontology Definition Meta-model (ODM) and various levels of ontologies to determine the mapping between them. They also use ER and UML meta-models to construct the ontologies and perform the matching using an ontology matching tool. They have introduced a lifting procedure used in this thesis as well, in which a meta-model (abstract syntax) is transformed into an ontology representing the concepts covered by the modeling language. This thesis makes use of their lifting strategy as well as the concepts of various types of ontologies such as *pseudo*, *refactored*, and *enriched* ontologies (see Section 7.4) but for a pair of modeling techniques with a slight variation in refactored ontology where refactoring is performed using the concept maps and the knowledge of the ontology designer about the techniques instead of any of the refactoring patterns.

1.4. Contributions

This thesis lays down the theoretical foundation for the use of multiple interacting models in order to determine semantic equivalences between Influence net and Social network modeling techniques. A workflow has been devised which facilitates the determination of overlaps, mappings, and relationships between these techniques. Concept maps, meta-models, and ontologies were developed. These mappings were ultimately used to introduce analysis results from the Influence net model into the Social network model constructed for a domain. This proposed workflow is repeatable for any model constructed using another modeling technique that needs to inter-operate with these two (Influence net and Social network) types of models. All that is required is for the same steps to be performed as explained in later chapters.

1.5. Thesis Layout

The next chapter provides necessary background for Concept map construction and its use for knowledge elicitation. Chapter 3 explains Bayesian nets, Influence nets and the constructed concept maps. Chapter 4 explains Social networks and the constructed concept maps. Chapter 5 describes concepts of models, meta-models, meta-modeling and multi-modeling along with the developed meta-models for both modeling techniques. Chapter 6 introduces ontologies, the need for ontologies, an ontology construction process, the Web Ontology Language as well as the constructed ontologies for both techniques. Chapter 7 describes the actual workflow and puts all the pieces together to explain how the process starts from concept mapping and continues to meta-modeling and to the different levels of ontology construction until an enriched ontology is obtained. Chapter 8 covers a case study about the *Iraqi Invasion of Kuwait* and explains the domain ontology instantiation process. Chapter 9 describes an application of this workflow by utilizing Pythia's[1] (Influence net) analysis results obtained from the *Sequence of Actions Finder (SAF)* algorithm and using them to update the ORA [2] (Social network) model. Chapter 10 concludes the thesis by suggesting future extension in this research area.

---

[1] Pythia is an Influence net modeling tool developed by System Architectures Laboratory.
[2] ORA is a social network construction and analysis tool by Center for Computational Analysis of Social and Organizational Systems (CASOS).

CHAPTER 2: CONCEPT MAPS

2.1. What is a Concept Map?

A concept, as defined by Novak and Cañas [2] in the technical report about concept maps, is a *perceived regularity (or pattern) in events or objects, or records of events or objects, designated by label*. In a concept map, each concept is represented using some type of geometrical shape (rectangular, circular, elliptical etc.) and is connected with other concepts using a directed link. This link can be tagged with a description of the relationship between the two concepts. Concepts connected together with a relationship referring to a meaningful entity define a proposition. Figure 1 is an example of a concept map. It shows a sub-part of the complete concept map constructed for a focus question, *What is an Influence net?*

Concept mapping is a representation technique to organize knowledge about a specific domain. It is based on the learning theories proposed by cognitive psychologists, such as David Ausubel and his Assimilation Learning theory [3] which distinguishes between rote and meaningful learning techniques; it is argued that concept mapping directly facilitates meaningful learning.

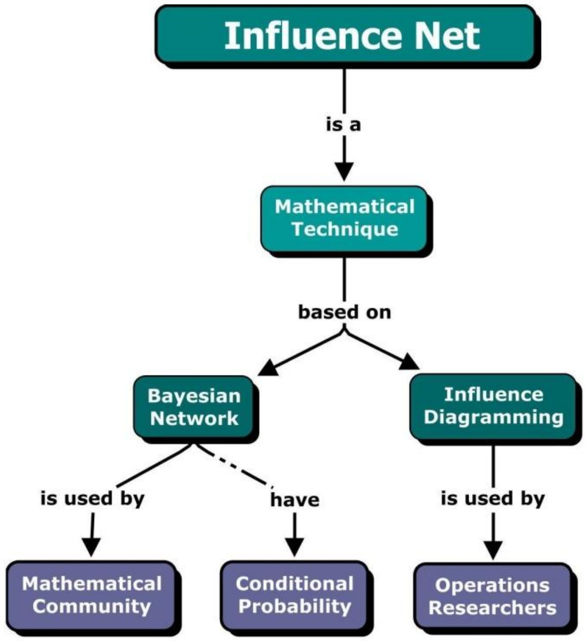


Figure 1: A Sample Concept Map for What is an Influence Net Focus Question

In order to understand the utility of concept maps, we need to consider first the origin of the initial concepts:

a. **Discovery Learning:** A process of learning when an individual learns by figuring out patterns or regularities in the events or objects and starts joining these with the same regularities as identified by other persons with specific words or symbols. Children aging from birth to three go through the phase of discovery learning.

b. **Reception Learning:** A process of learning which begins after discovery learning, in which new concepts are learned by inquiring the old concepts and propositions using a language to extract new meanings and understandings.

The learning process can be meaningful or rote. According to Ausubel [3], in rote learning new knowledge and concepts hardly integrate with the existing ones. As a consequence of this, the bearer of such knowledge quickly forgets the learned concepts unless excessive rehearsals are done. Concept maps fulfill most of the conditions of meaningful learning and can be used as an effective tool that facilitates it.

2.2. Concept Map Construction

The construction of a concept map is an iterative process that requires brainstorming and frequent revisions until a final and concrete concept map is developed for a specific domain. The following step by step method can be used for concept map construction:

a. **Identification of Focus Questions:** The identification of focus questions is the starting point of constructing a concept map. A list of focus questions should be prepared for which concept map needs to be developed. For instance, *what is an Influence net?* can be a good focus question as it would prevent any distracting ideas during the construction and will also allow the person constructing the concept map to concentrate or re-focus on the question, if focus is lost.

b. **Construction of Parking Lot:** For each focus question, a pool of concepts is generated. The term "parking lot" refers to the idea that only concepts relevant to the focus question reside together. No relationships or links have been defined between the concepts at this point. The constructor thinks about the focus question and puts a relevant concept in the lot. For instance, the parking lot for an Influence net focus question will have concepts like *mathematical technique, Bayesian network,* *Influence diagramming, Conditional probability, Operations researchers, Prediction, Situation, Crisis,* etc.

c. **Establishing cross-links between concepts:** Once the parking lot has an adequate number of concepts in it, links can be established between related concepts. For instance, Influence net *is a* mathematical technique *based on* Bayesian networks.

2.3. Using Concept Maps for Knowledge Elicitation

Concept Mapping has been used to construct concept maps for Influence net and Social network modeling techniques. The aim is to gain a syntactic and semantic insight into both modeling techniques to reveal aspects which will ultimately facilitate the ontology construction process later on. This level serves as the conceptual modeling level as shown in Figure 2. A set of five focus questions were identified for each technique:

a. What is an Influence (Social) Network?
b. What are the constructs of an Influence (Social) Network?
c. What tools are available to construct Influence (Social) Network?
d. What analyses can be performed using Influence (Social) Network?
e. Who are the people involved in Influence (Social) Network construction?

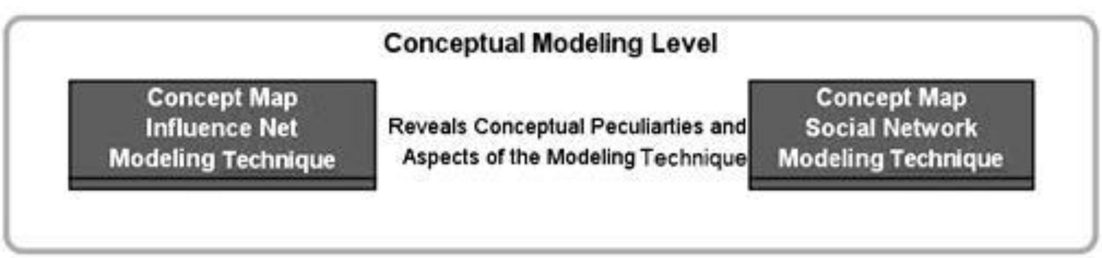


Figure 2: Conceptual Modeling Level

CHAPTER 3: BAYESIAN NETS AND INFLUENCE NETS

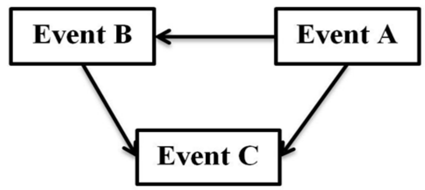
3.1. Bayesian Networks

Bayesian networks is a probabilistic modeling technique that uses directed acyclic graphs as the modeling language for the modeling of an uncertain domain. Bayesian networks have been used in many areas of applications such as medical expert systems, diagnosis of failures, pattern matching, speech recognition, and software testing [4].

In a Bayesian network model, random variables of interest are represented as nodes (vertices) along with conditional interdependencies as arcs (edges) of the directed graph. Each node has an associated conditional probability table that models the uncertainty between itself and its parent node(s). For a complete Bayesian network, all conditional probabilities have to be specified; they can be elicited either from historical data or knowledge of subject matter experts.

An example of a Bayesian network is shown in Figure 3. There are three random variables of interest, Event A, Event B, and Event C. Event A has no parents, its conditional probability table has only one entry each for being True and False. Event B is dependent upon Event A only; therefore its conditional probability table is 2x2. Similarly, the conditional probability table for Event C is dependent upon two parents, Event A and Event B, and is 2x4.

Event B	Event A: True	Event A: False
True	0.4	0.8
False	0.6	0.2



	True	False
Event A	0.5	0.5

	Event A: True		Event A: False	
	Event B: True	Event B: False	Event B: True	Event B: False
Event C				
True	1.0	0.25	0.55	0.7
False	0.0	0.75	0.45	0.3

Figure 3: Sample Bayesian Network

For a node having *n* parents, there are 2^n possible entries in the conditional probability table. Specification of the conditional probabilities can become a tedious task, since the size of the conditional probability table grows exponentially with the number of parents of a node. Influence nets discussed next simplify this issue to some extent.

3.2. Influence Nets

Rosen and Smith [5] proposed a formalism called *Influence net* which utilizes directed acyclic graphs as the modeling language as in Bayesian networks. The nodes represent random variables (propositions) such as beliefs, actions, events, etc., whereas an edge represents a causal relationship (influence) between two nodes (propositions). The parent and child nodes are often called *cause* and *effect*, respectively. The causal relationship between a cause and an effect can either be promoting or inhibiting as identified by the edge terminator (arrow head or filled circle) as shown in Figure 4.

Influence nets make use of the CAST Logic algorithm by Chang et al. [6] which defines influence parameters *h* and *g* assigned to each link (edge) between the two nodes. These parameters define the causal strength of the influence between the cause and effect nodes. Parameter *h* models the case when occurrence of a cause would influence the likelihood of occurrence of the effect whereas parameter *g* models the case when non-occurrence of cause would influence the likelihood of occurrence of the effect.

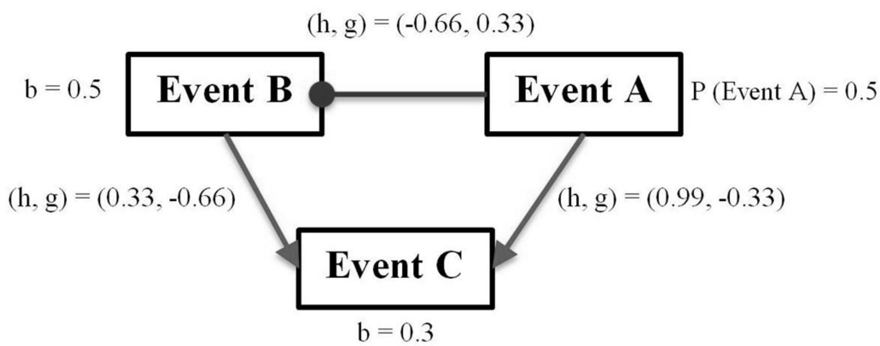


Figure 4: Sample Influence Net

The CAST algorithm generates from the influence parameters the required conditional probabilities thus reducing substantially the data specification effort. The details of CAST Logic algorithm can be found in Rosen and Smith [5] and K.C. Chang et al. [6].

The following characteristics define an Influence net:

a. A set of random variables defined by nodes.
b. A set of directed links that connect pairs of nodes.
c. Each link has an associated pair of *h* and *g* (CAST Logic) parameter values which define the causal strength of the influence between the two nodes.
d. Each non-root node has an associated CAST Logic parameter called the baseline probability (given by *b*), whereas a prior probability is associated with each root node (given by *P (event)*).

3.3. Concept Mapping Influence Nets

a. What are the constructs of an Influence net?

An Influence net is an acyclic graph composed of nodes and links. Nodes represent *propositions* and are connected by links which represent *influences*. The nodes can either be *input* or *non-input*. An input node has no parents and is specifically an actionable (controllable) event which can be assigned to a course of action [7]. A course of action has an associated *time* and *status* and is explained later in section 3.3c (i). The *status* can either be *true* or *false* representing the occurrence or non-occurrence of that event, whereas *time* describes an instance of time when that event is either *true* or *false*. A probability profile [7] is a graph which plots the probability of an event against time and can be generated for all non-input nodes. The non-input nodes can either be either objective or intermediate nodes. An objective node is the final effect or desired event in the network and does not influence (it is not a cause of) any other event, whereas an intermediate node serves as the influencing proposition somewhere between the actionable event node and the objective node. The input nodes have a marginal probability, whereas non-input nodes have a baseline probability. Two concept maps, Figs. 5 and 6, show what an Influence net is and what its basic constructs are.

The influence between the cause and effect propositions is defined by the *CAST Logic* algorithm [5], which associates with each influence the two parameters '*h*' and '*g*'. Both of these parameters can have values between 1 and -1. To have maximum promoting influence, we set *h=1*, which means occurrence of the cause promotes the occurrence of the effect. If g=1, the absence of cause will promote the occurrence of the effect. To have an inhibiting influence, h is set to have a value less than zero which implies that the likelihood of the effect occurring is inversely affected by the presence of this influence. The no-influence condition is specified by $h=0$ and $g=0$ values, i.e., occurrence or non-occurrence of cause has no influence on the effect.

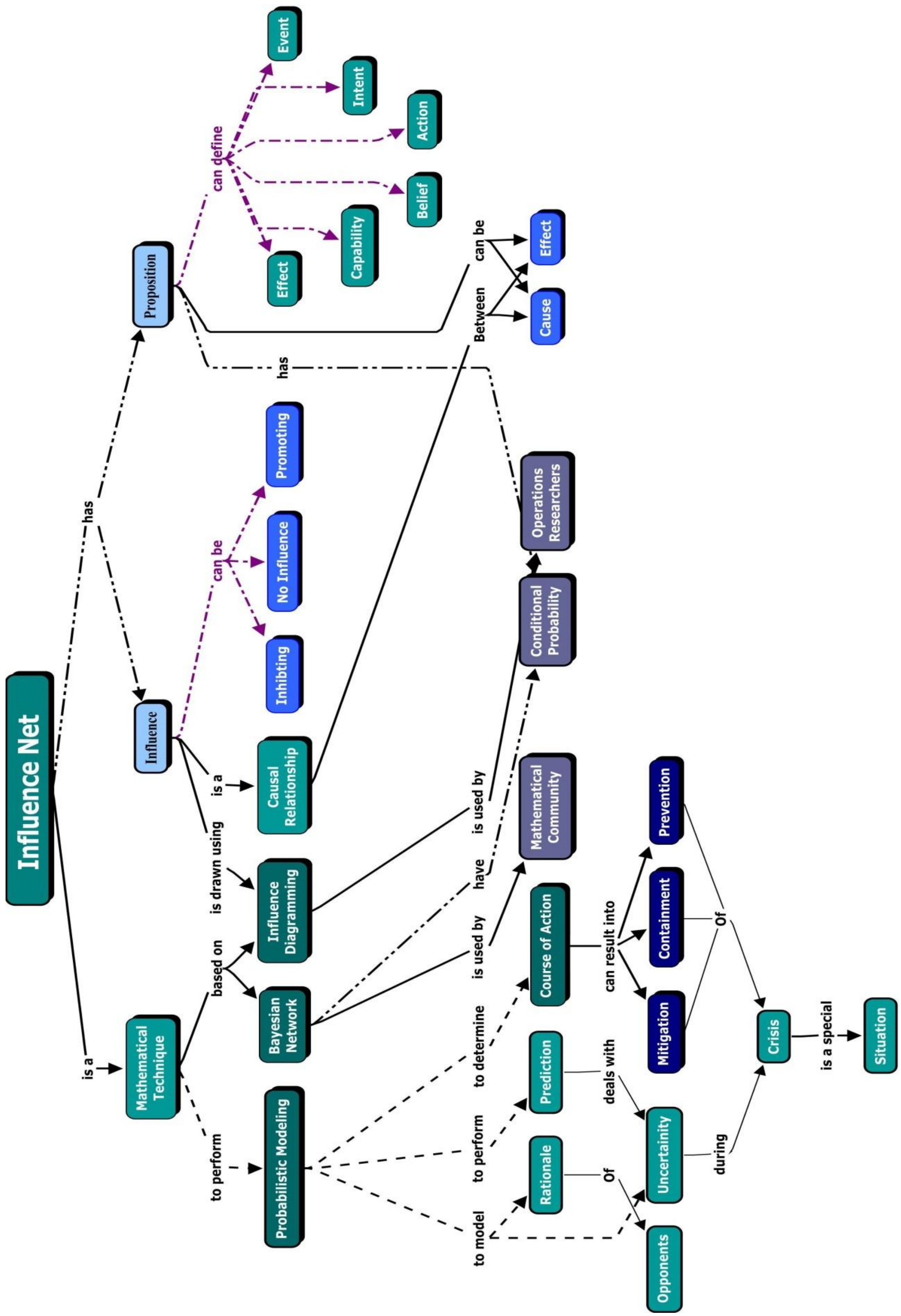


Figure 5: What an Influence net is

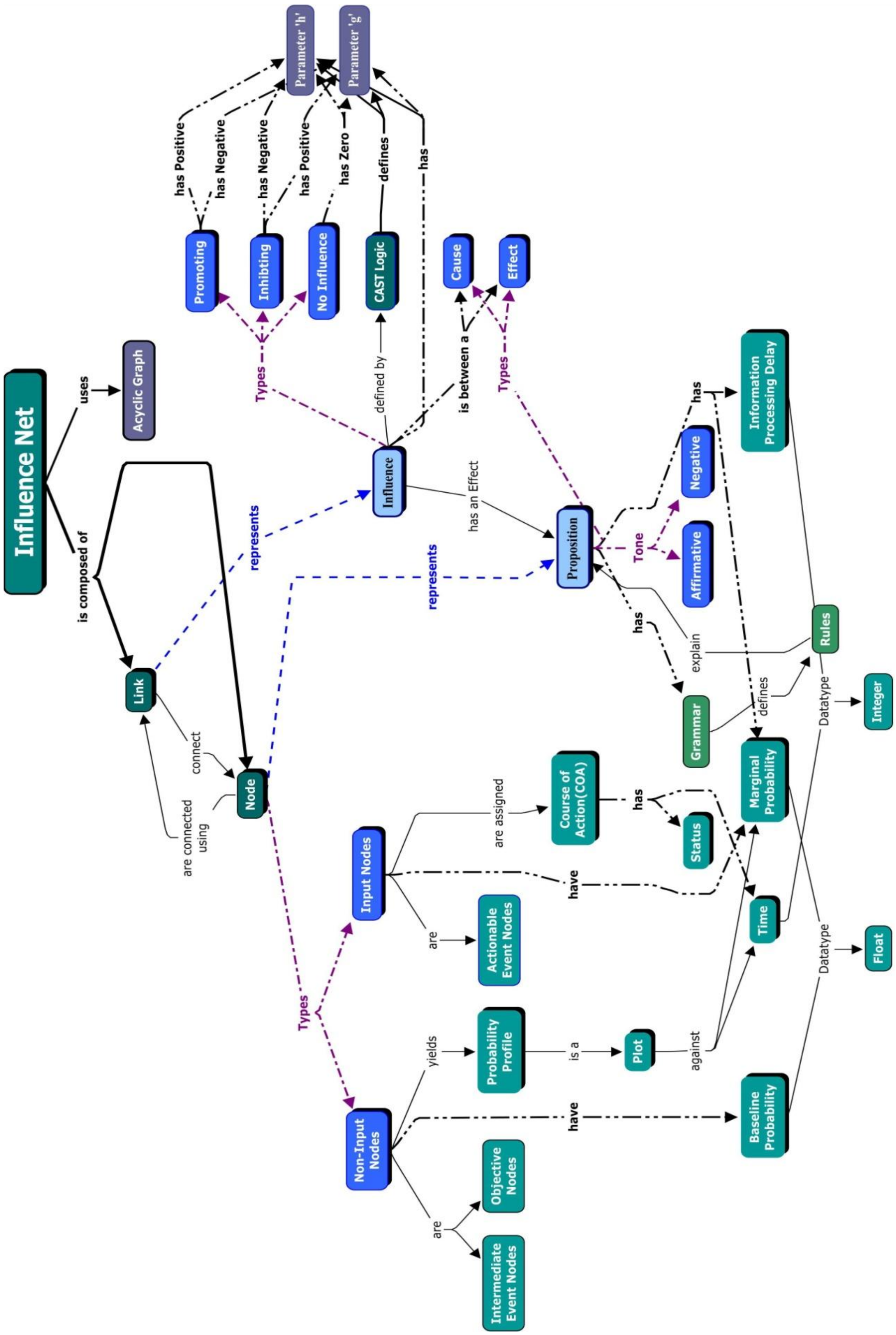


Figure 6: What the constructs of an Influence Net are

b. What tools are available to construct Influence nets?

The UNIX based application *SIAM* [3] supports the development of Influence net models (non-timed version). In order to overcome issues with Bayesian network models and the complete assignment of conditional probability matrices, SAIC[4] staff and the collaborating George Mason University's research team developed an approach that uses *Causal Strengths (CAST)* [5] which is implemented in *SIAM.* Another tool called *Pythia* [8], developed by the research team at George Mason University, supports modeling of the timed version of Influence nets (TINs). Additionally it also supports analysis techniques such as *Sensitivity Analysis, Course of Action Analysis,* and *Sequence of Actions Finder (SAF) algorithm*. The concept map for this focus question is given in Figure 7.

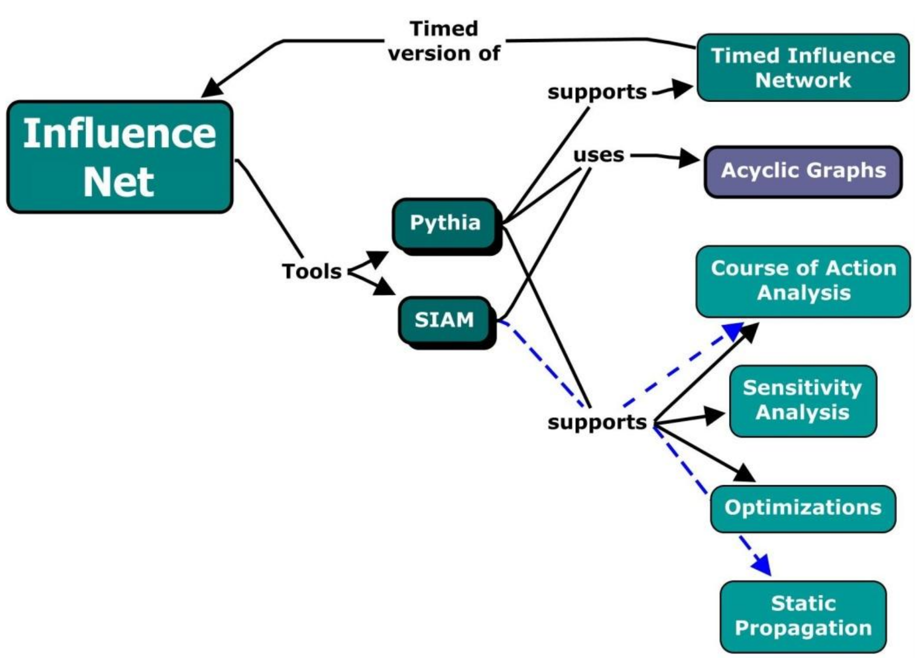


Figure 7: Concept Map of tools available for Influence Nets

[3] SIAM: Situational Influence Assessment Module.
[4] SAIC: Scientific Applications International Corporation

c. What analyses can be performed on Influence Nets?

i. Course of Action Analysis

Wagenhals and Levis [7] describe a course of action as a composition of a timed sequence of actionable events. A course of action is designed by setting the *time* and *status* of the input nodes (actionable events). After the actionable events are assigned these parameters, a probability profile can be generated for the desired effect or intermediate nodes. A probability profile is a plot of a non-input node's varying probability of occurrence of the event it represents against time. It shows how the probability of a non-input node varies with time because of other influencing propositions over time. Figure 8 shows an Influence net model for the *Iraq-Kuwait* scenario discussed later in this thesis and Figure 9 shows the probability profile for the objective node *Saddam decides to withdraw from Kuwait peacefully.* These were developed using Pythia. Figure 9 shows that the probability of this effect occurring increases from 0 to 1 over time.

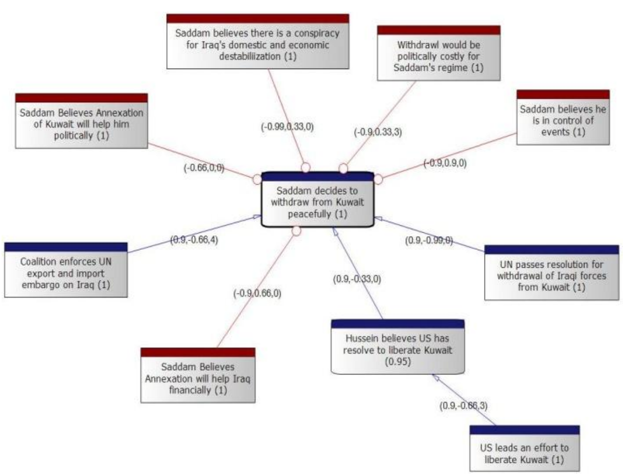


Figure 8: Influence Net Model for Iraq-Kuwait Case Study

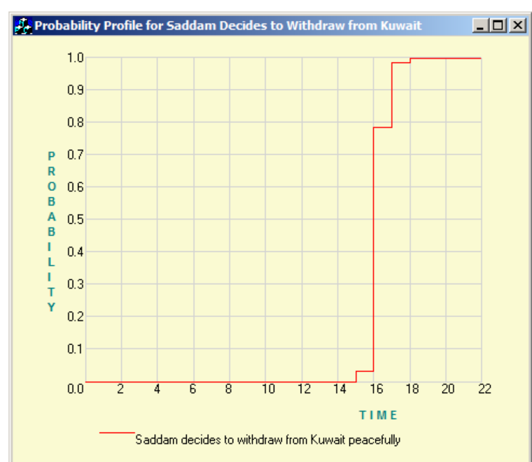


Figure 9: Probability Profile for Final Effect Node

ii. Sequence of Actions Finder (SAF) Algorithm:

Pythia also implements the heuristic approach by Haider et al. [9] for finding the best sets of actions for achieving the maximum likelihood of a desired effect. This approach requires a threshold probability for the desired effect/objective node and outputs all possible combinations of best actions which yield the probability of the effect node being equal to or more than the specified threshold probability. This approach can be beneficial in circumstances where one needs to find out the possible sets of actions that will result into an acceptable probability level for a desired effect. Figure 10 shows the results of the SAF Algorithm for the *Iraq-Kuwait* case study.

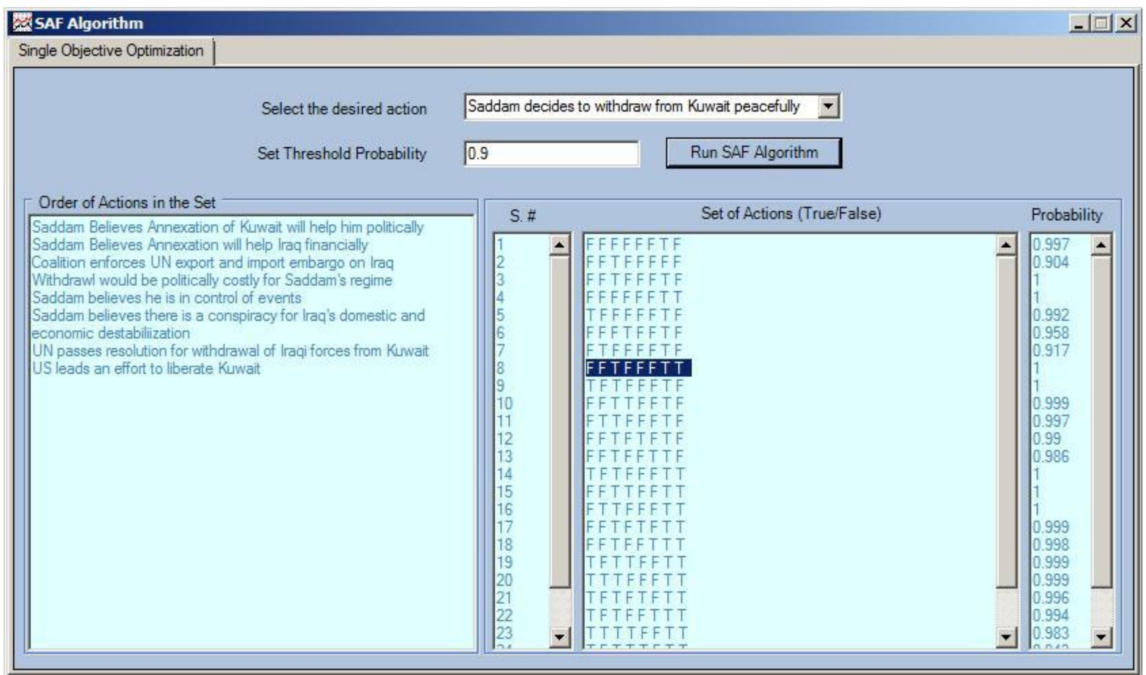


Figure 10: SAF Algorithm Results for Iraq-Kuwait scenario

The threshold probability of the desired effect *Saddam decides to withdraw from Kuwait peacefully* can be specified by selecting a desired effect from the first drop down menu and setting the threshold value 0.9 in the textbox beneath it (Fig. 10). The left frame of this window shows the available actions and the order in which they should be executed. The right frame shows a list where each entry from left to right corresponds to each of the available actions, (where T = true for occurrence and F = false). The right most frame shows the resultant probability value for the selected effect as a result of using this set of actions.

The algorithm produces the best sets of actions which will result in the desired probability of the effect. Figure 11 includes the segment of the concept map for the *SAF Algorithm*, the lower part of the figure shows how the occurrence or non-occurrence of actions (events) can be used to update the links among the entities of a corresponding social network. This point is elaborated further in Chapter 9.

iii. Sensitivity Analysis

Sensitivity Analysis is another analysis technique provided in *Pythia* that allows one to observe how sensitive the likelihood of the occurrence of an effect node is with respect to the occurrence/non-occurrence of an actionable event or with respect to the maximum and minimum influences. Figure 11 shows the segment of concept map for *Sensitivity Analysis*.

d. Who constructs Influence nets?

Typically, domain experts and analysts supporting decision makers are involved in the construction of Influence nets. Influence net facilitates collaborative analysis among domain experts. These domain experts can also be decision makers and are responsible for the construction of the nodes and links in a model. Decision makers are responsible for identifying or examining courses of actions to make sure a crisis situation can be effectively mitigated, contained, or prevented. The concept map for people who are involved in Influence net construction is given in Figure 12.

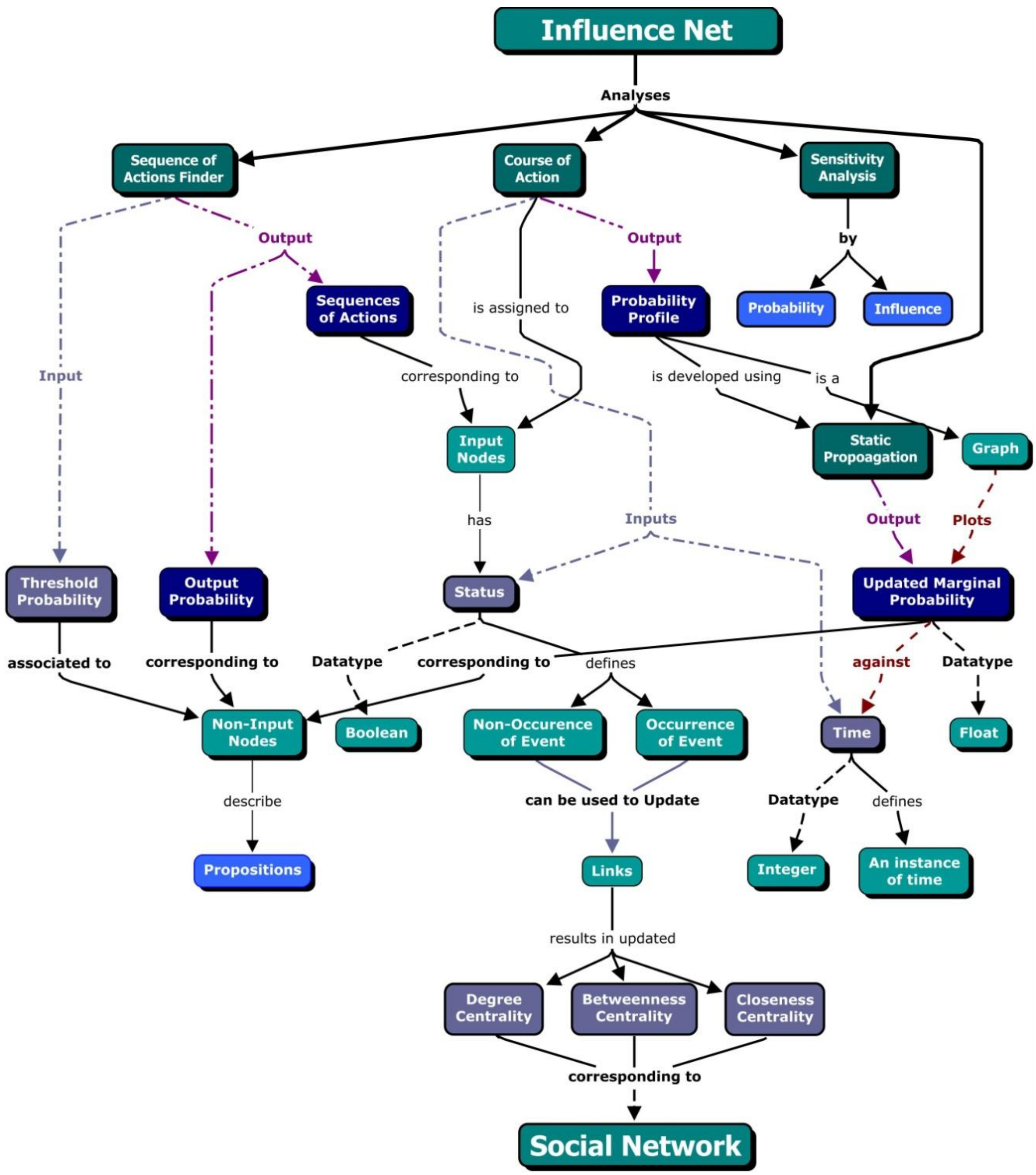


Figure 11: Influence Net Analyses

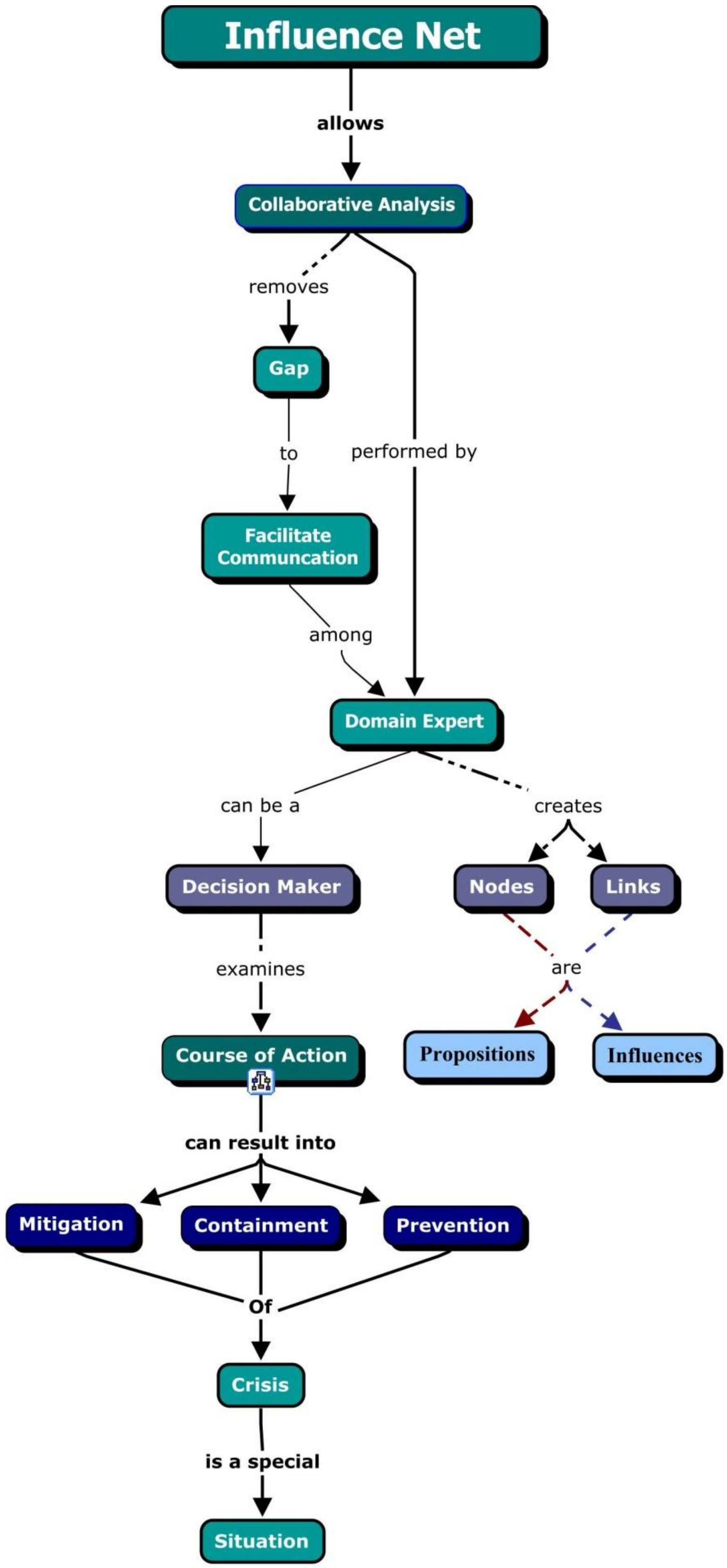


Figure 12: People involved in Influence Net construction

CHAPTER 4: SOCIAL NETWORKS

4.1 Social Networks

A social network is a structure composed of real world entities (e.g. human beings, organizations, actions, tasks, etc.) and associations or interdependencies (e.g., interaction, relationship, kinship, etc.) among those entities. The term structure more specifically refers to social structure which is a concept in sociology and which refers to an enduring relationship between real world entities. The resultant structure yields a graph-like formalism as shown in Figure 13 which has the properties of a typical graph but also allows other set of measures such as density, betweenness, closeness, and degree centralities.

In the sample social network given in Figure 13 the circular nodes represent entities such as human beings and the edges connecting these entities represent associations between them. These associations can be an interaction between *Albert* and *Cynthia* in the form of mutual discussions; it can also be a kinship relationship between *Branden* and *Debby* or a friendship relationship between *Francis* and *Hammond*. The graphical form of the social network can also have a matrix representation in which the entities are represented in the matrix rows and columns and the matrix entries indicate their interaction. *ORA* makes use of the matrix form to represent social networks. Single-mode matrices represent networks containing only one type of entities (e.g., persons only) while multi-modal matrices consider networks with multiple types of entities (e.g., persons, tasks, knowledge, resources etc.).

Carley and Reminga [10] have implemented in ORA a large set of social network measures.

a. Network Level Measures provide information about the complete network such as network density or diameter.
b. Node Level Measures provide information about a specific node in the network such as degree, betweenness, and closeness measures of centrality.

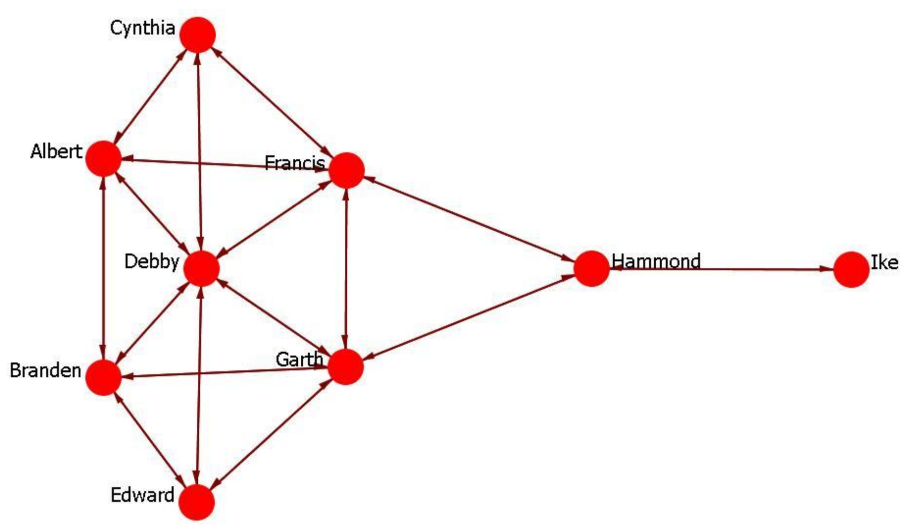


Figure 13: Sample Social Network

4.2 Concept Mapping Social Networks

Figure 14 shows the concept map that describes what a social network is. It is a graph composed of nodes representing entities, which are connected together by links (or ties). The links represent interdependencies of different types. A quantitative attribute can be associated with interdependencies that reflect the strength of relationship between the connected entities as shown in Figure 15, a concept map that describes the constructs of a social network.

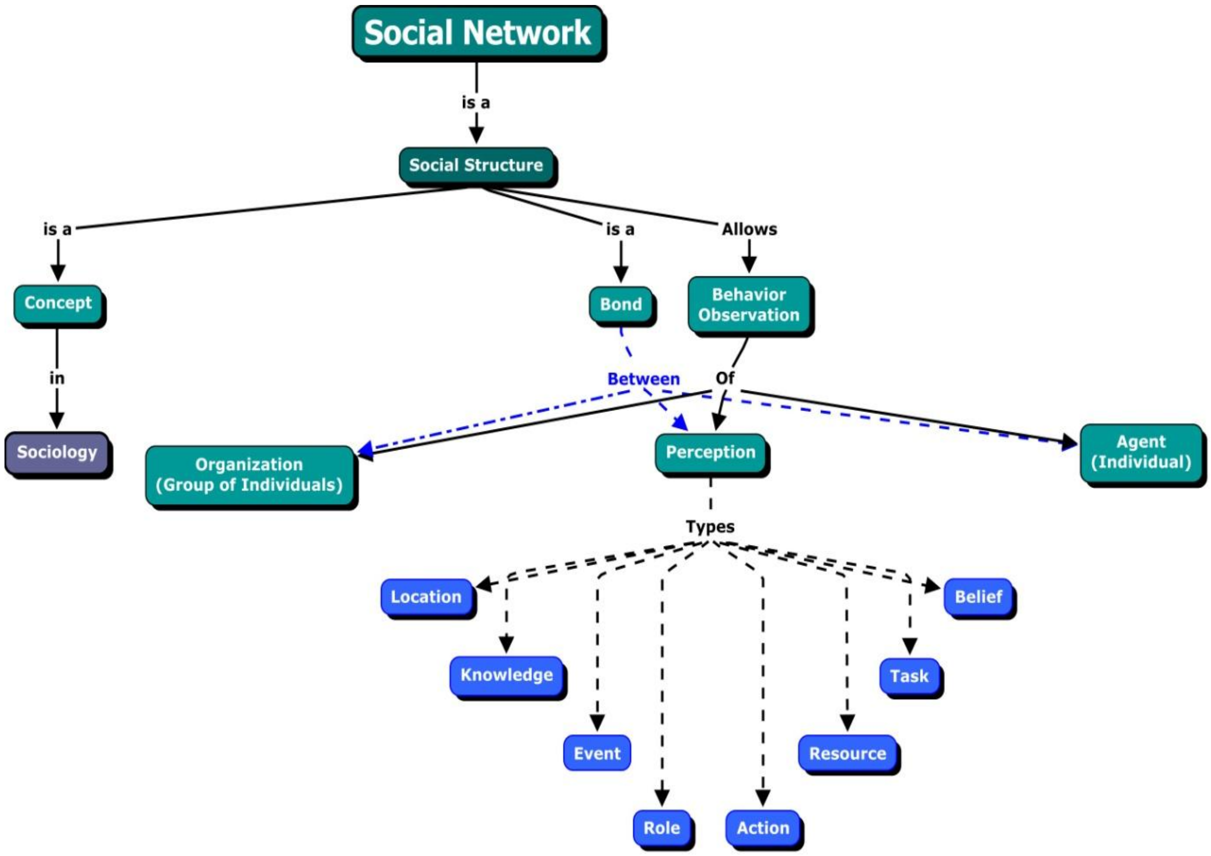


Figure 14: What a Social network is

The different types of interdependencies depend upon the types of nodes being connected by the links. For instance, a link connecting two persons could represent kinship, specific role (*boss of, friend of*), interactions (*talks to, advices*), and affiliations (*belongs to, corresponds to*), whereas a link connecting a person with a perceptual concept (*knowledge, resource*) could represent ownership, need, or possession (*Person A possesses knowledge* or *owns/needs* a *resource* to *perform* a *certain task.*)

A graph representing a social network can have directed or undirected links. Directed links show interdependencies only in one direction, e.g., *Person A delivered equipment to Person B*; whereas undirected links show interdependencies in both directions, e.g., *Person A and Person B have discussions regularly*.

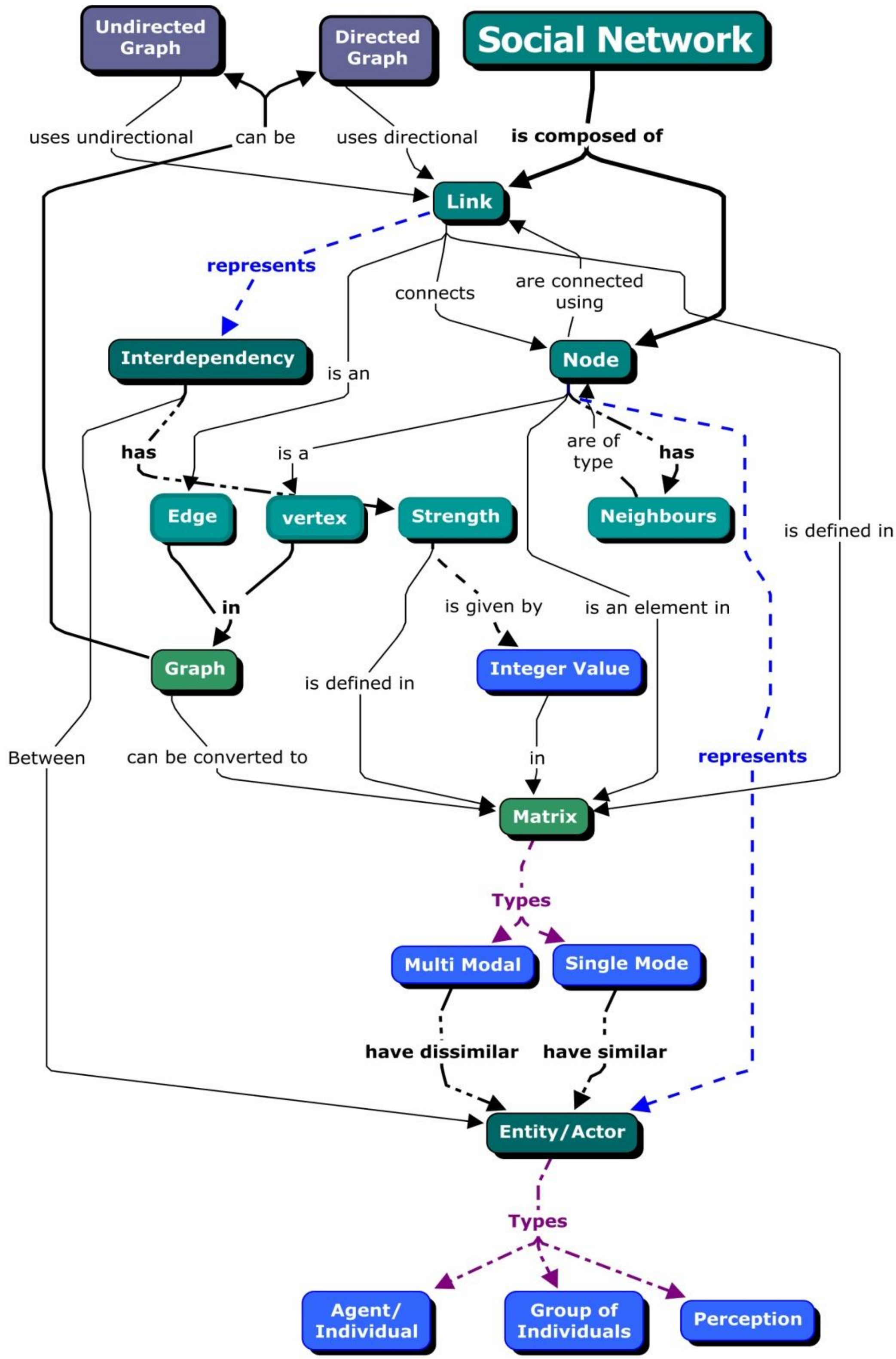


Figure 15: What the constructs of a Social network are

CASOS [11] provides a set of useful tools for social network analysis. *Automap* [12] supports textual analysis and allows parsing and editing of text after which you can specify the meta-matrix elements which can subsequently be used in *ORA*. A meta-matrix is a framework for analyzing complex networks and is the conceptual representation of a meta-network (or set of networks). The meta-matrix links multiple matrices together to form a larger matrix structure. These matrices help analysts or modelers define the interdependencies between the entities. If a certain network has *n* entities, then the interdependencies between the entities can be defined using an *nxn* matrix. In *ORA*, a network is represented using a matrix as shown in Figure 16.

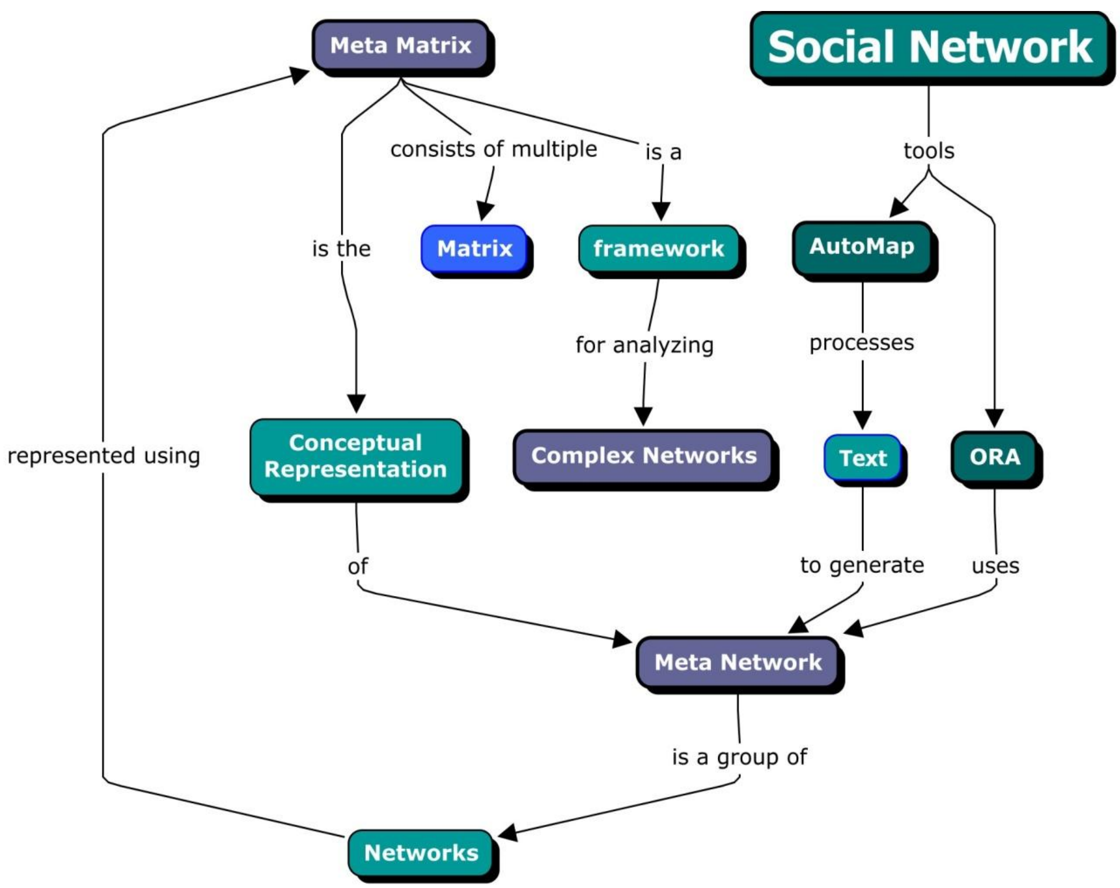


Figure 16: Concept Map of tools available for Social networks

4.3 Social Network Analysis

Some of the many different measures associated with social network analysis (SNA) are given in what follows. All these measures can be calculated with the help of *ORA* tool very easily for any social network. Figure 17 illustrates the concept map for SNA.

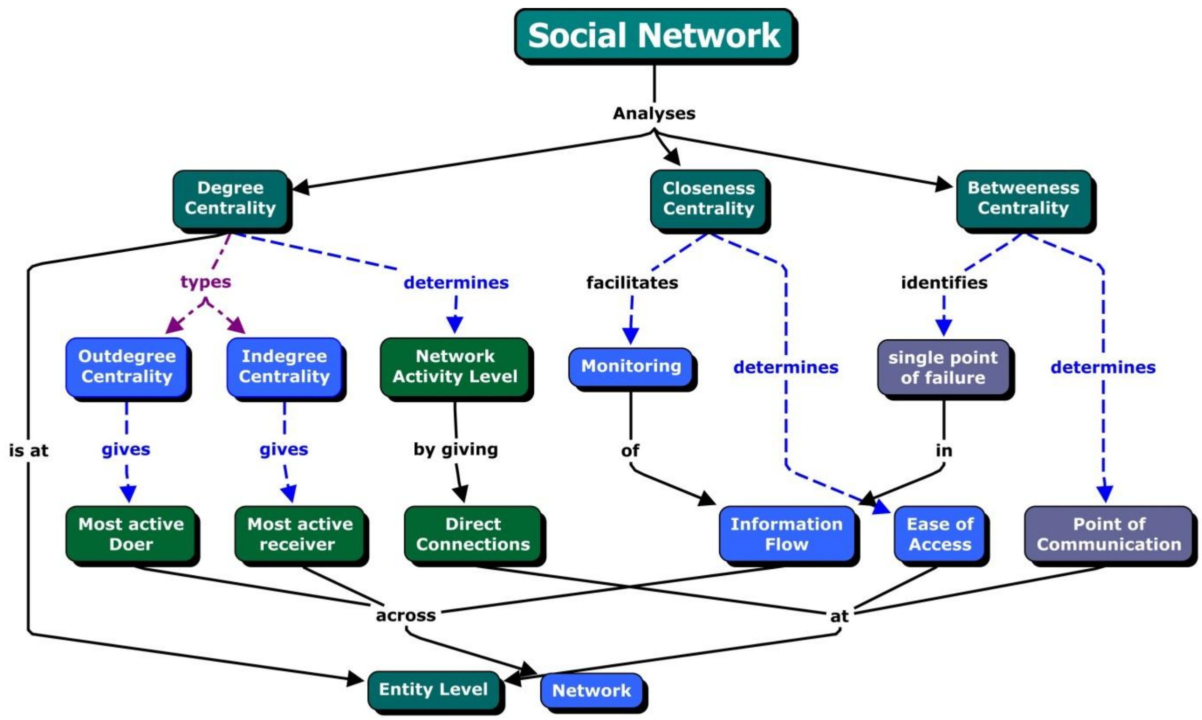


Figure 17: Social Network Analyses

i. *Degree Centrality* is associated with each entity (node) in a social network. It measures the network activity level by measuring the number of direct connections a node has in the network. For a Graph G = (V, E), (where V = vertices and E = edges) having total *n* vertices, the (total) degree centrality for a vertex *v* can be calculated using the following formula:

Total Degree Centrality: $DC_{total}\ (v) = \frac{degree\ (v)}{n-1}$ (i) where $degree(v)$ gives the total number of direct connections that node v has, e.g., $degree(Debby) = 6$, and $degree(Francis) = 5$. For the sample Social network given in Figure 13 as described earlier, *Debby* has the highest total-degree centrality and is considered the most active entity (agent) in the network, followed by *Francis* and *Garth* and the rest.

ii. *Betweenness Centrality* is also associated with each entity (node) in a social network. It depicts the point of communication among nodes in the network. A node with the highest betweenness centrality can become a single point of failure of communication for the whole network. For a Graph G = (V, E), (where V = vertices and E = edges) having total *n* vertices, then closeness centrality for a vertex *v* can be calculated using the following formula:

$$BC(v) = \sum_{\substack{s \neq v \neq t \in V \\ s \neq t}} \frac{\sigma_{st}(v)}{\sigma_{st}} \qquad \text{(ii)}$$

where σ_{st} is the number of shortest paths from s to t and $\sigma_{st}(v)$ represents the number of shortest paths from s to t that pass through a vertex *v*.

For the sample Social network given in Figure 13, *Hammond* has the highest betweenness centrality, followed by *Francis*, *Garth*, *Debby* and the rest. The importance of *Hammond's* role is evident from the figure as he is the key communicator for the rest of the entities in the network. If *Hammond* is removed, the communication mechanism is disrupted, since *Ike* is disconnected from the rest of the network and won't be able to communicate with anyone. If *Ike* and *Hammond* hold some information which is required by everyone, this could be quite disruptive for the whole network.

iii. *Closeness Centrality* is also associated with each entity (node) in a social network. It determines ease of access across the network for an entity, i.e., how easily a node can reach the rest of the nodes throughout the network. It also facilitates monitoring of information flow in the network. A node with the highest closeness centrality will have a key property of monitoring the information flow across the network. For a Graph G = (V, E), (where V = vertices and E = edges) having total *n* vertices, and where $d_G(v, i)$ is the distance between vertex *v* and *i*, where $v, i \in V$, then closeness centrality for the vertex *v* can be calculated using the following formula:

$$CC(v) = \frac{(n-1)}{\sum_{i \in V} d_G(v,i)} \qquad \text{(iii)}$$

For the sample Social network given in Figure 13, Debbie has the highest closeness centrality, followed by *Francis* and *Garth* and the rest. The importance of *Debby's* role is also evident from the figure as she acts as the bridge for the some of the entities in the network. If *Debby* is removed, the communication mechanism is also disrupted, as *Cynthia* would need to go through *Albert* and *Branden* to reach *Edward*, whereas before she could reach him through *Debby* only.

This chapter provided a brief description of the Social network modeling technique.

CHAPTER 5: MODELS AND META MODELS

5.1. Model

A model is a simplified abstraction of a real world phenomenon. The aim of modeling is to construct a formal structure which interprets reality. It can also be defined as a physical, mathematical or logical representation of a system, phenomenon, or process. DOD[5] classifies models into three basic categories, i.e. mathematical, physical and procedural [13]:

a. Mathematical Model:

A mathematical model is a representation which makes use of procedures (algorithms) and mathematical equations. Models developed using Influence and Social network modeling techniques would come under the heading of mathematical models.

b. Physical Model:

A physical representation of real world objects is a physical model. These models consist of scaled down versions of the real world objects, e.g., airfoils and ship contours for use in wind tunnels.

---

[5] DOD: US Department of Defense

c. Procedural Model:

It is an expression of dynamic relationships of a situation given by mathematical and logical processes and is commonly referred to as simulation.

Following are examples of Influence net and Social network models developed for the *Iraq-Kuwait* scenario (Figure 18 & 19) which are discussed in detail in Chapter 8 and 9.

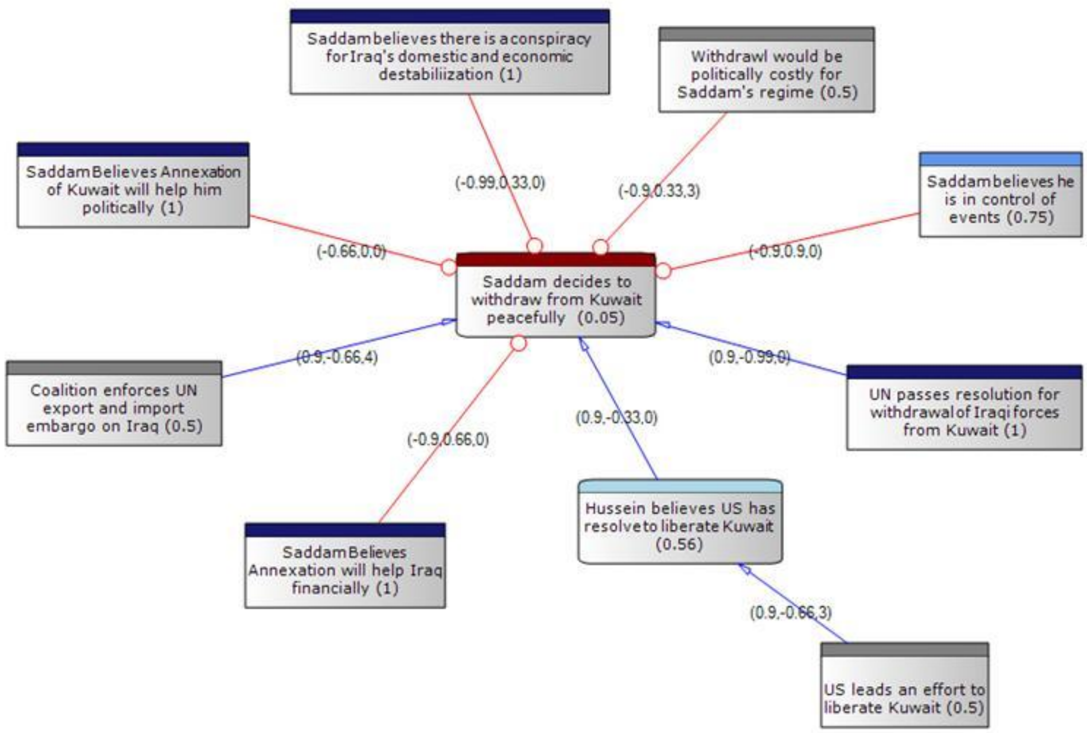


Figure 18: Influence Net Model for Iraq-Kuwait Scenario

As already explained earlier (Section 3.2), Influence net models an uncertain domain. This domain may include real world events, actions, beliefs, and effects that are represented by rectangular boxes while the influences between them are given by links (with arrowheads or circles) as shown in Figure 18. Similarly, as explained in Section

4.1, Social networks model real world entities and the associations between them, where the entities include human beings, their beliefs, knowledge, resources, etc. and are given by circular nodes, while the associations between them may include relationships and interactions and are depicted by directed or undirected links.

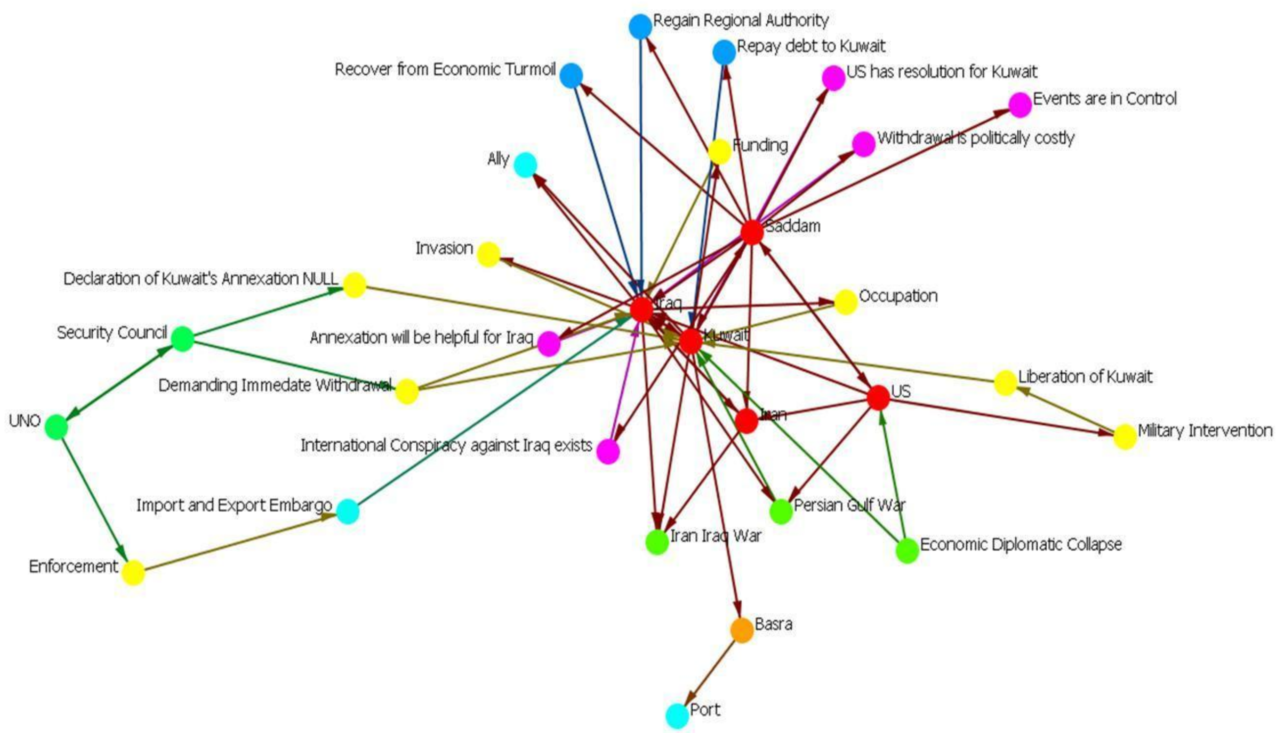


Figure 19: Social Network Model for Iraq-Kuwait Scenario

5.2. Modeling Technique

A modeling technique is composed of a modeling language and a modeling procedure. Every modeling language contains the elements with which a model can be described and has syntax and notation. The modeling procedures describe how the syntax and notation should be used in constructing a model to generate results by utilizing certain mechanism and algorithms [14]. Bayesian, Influence and Social network modeling techniques make use of graphs as the modeling language, more specifically Influence nets use acyclic graphs, and Social networks use directed and undirected graphs, and each technique has a modeling procedure of its own which defines how the elements of the graphs, i.e. nodes and links, should be used to construct an Influence net model and a Social network model.

The modeling procedure also makes use of certain algorithms in order to generate results from the models. As already described, an Influence net makes use of *CAST Logic Algorithm* to calculate the estimate for the likelihood of the occurrence of effect nodes whereas Social networks make use of various measures of centrality from Graph theory such as betweenness, closeness and degree centrality.

5.3. Meta-Model

A meta-model is an abstraction layer above the actual models. As the name implies, a meta-model provides information about the model itself. A model conforms to its meta-model exactly the way a piece of code written in a programming language should conform to the grammar for correct compilation and execution.

The typical role of a meta-model is to define the semantics for how model elements are instantiated. Consider a meta-model given in Figure 20 which deals with the naming and typing of *elements* in UML. As it can be seen, this meta-model provides information about an artifact *Element* which is a constituent of every UML model [15]. A UML *Element* can be of two types, i.e. a *NamedElement* which corresponds to an element with a *name* (string) attribute, or a *Comment* which is a textual description about the element and has an attribute *body* (string).

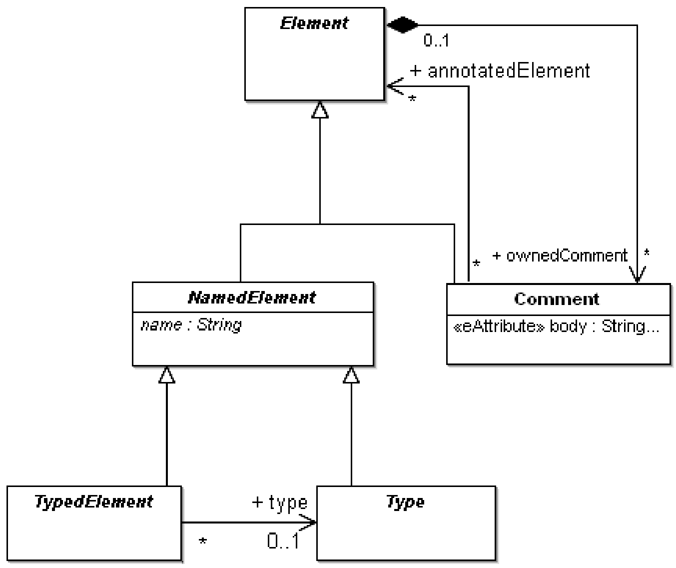


Figure 20: Meta-Model of *Element* in UML

5.4. Meta-Modeling

Meta-Modeling is the process of constructing a meta-model which may include the analysis, construction and development of rules and constraints in order to model a pre-defined class of problems within a certain domain.

There is a large number of software and non-software modeling languages and tools available [16] such as UML, *Web*ML[6], and EER[7]. OMG[8] defines the architecture for meta-modeling called as MOF (Meta Object Facility) (Figure 21). MOF is a four layered architecture with each successive layer labeled from M_3 to M_0, i.e. Meta-meta-

[6] *WebML:* Web Modeling Language
[7] EER: Enhanced Entity Relationship
[8] OMG: Object Management Group

model (M_3 layer), Meta-model (M_2 layer), User-defined model (M_1 layer), and Run-time instances (M_0 layer). An M_i layer model is an instance of the M_{i+1} layer model and must conform to its formalism.

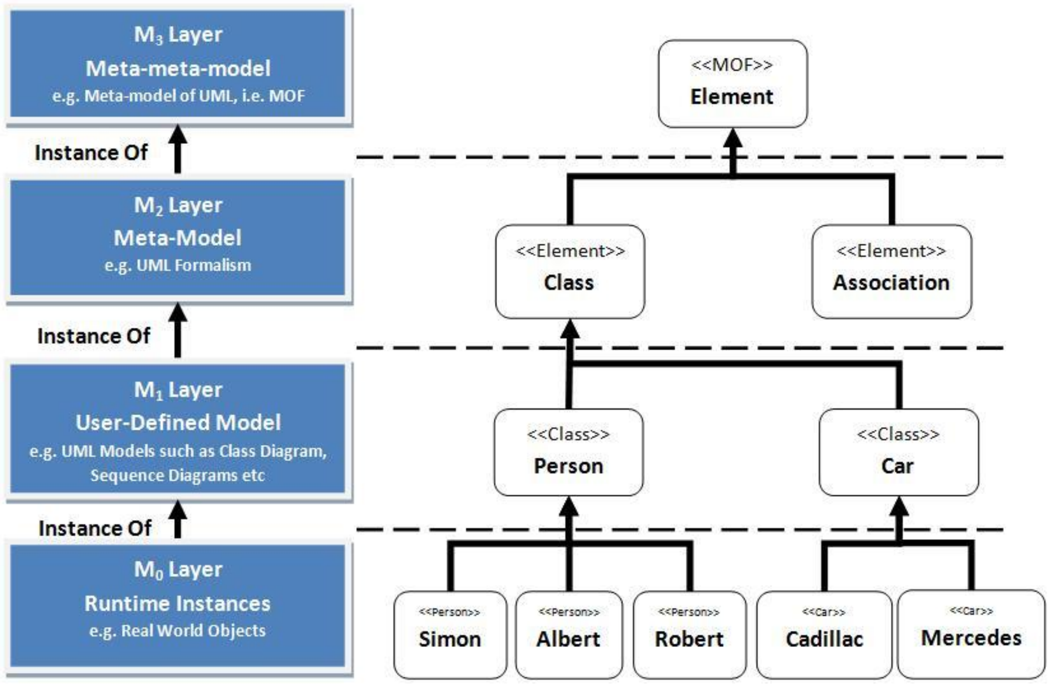


Figure 21: OMG's Four Layer Architecture for Meta-modeling

At the M_3 layer which forms the foundation of the meta-modeling hierarchy, the meta-model for the meta-model (at the M_2 layer) is defined (e.g. Meta-model for UML-Meta-model, i.e. MOF). As shown in Figures 20 and 21, *Element* is an artifact defined at the M_3 layer whose instances can be the *Class* and *Association* (at the layer M_2). At the M_2 layer, meta-model which describes the model itself (at the M_1 layer) is defined (e.g. UML meta-model elements like *Class* and *Association*). At the M_1 layer, the model describes the real-world objects (e.g. domain specific models written in UML such as classes for *Person* or *Car*). The M_0 layer describes the run-time instances of real-world objects such as *Simon*, *Albert* and *Robert* as instances of the class *Person*, and *Cadillac* and *Mercedes* as instances of the class *Car*.

The work in this thesis employs the $M_{2,}$ $M_{1,}$ M_0 layers of modeling, which are the Meta-modeling, User-Defined Modeling, and Run-time instances of the user-defined models. Meta-models describing the Influence and Social network modeling techniques were constructed. Instances of these meta-models would be user-defined models such as the *Iraq-Kuwait* scenario discussed in this thesis; instances of each element of these models would be the run-time instances of real world objects.

5.5. Influence Net and Social Network Meta-Models

The meta-model construction was initiated with the help of concept maps from the conceptual modeling level, using only those concepts, which reveal the structural aspects of the techniques. Since concept mapping is an informal representation of concepts, notation formalization was needed as shown in Figure 22.

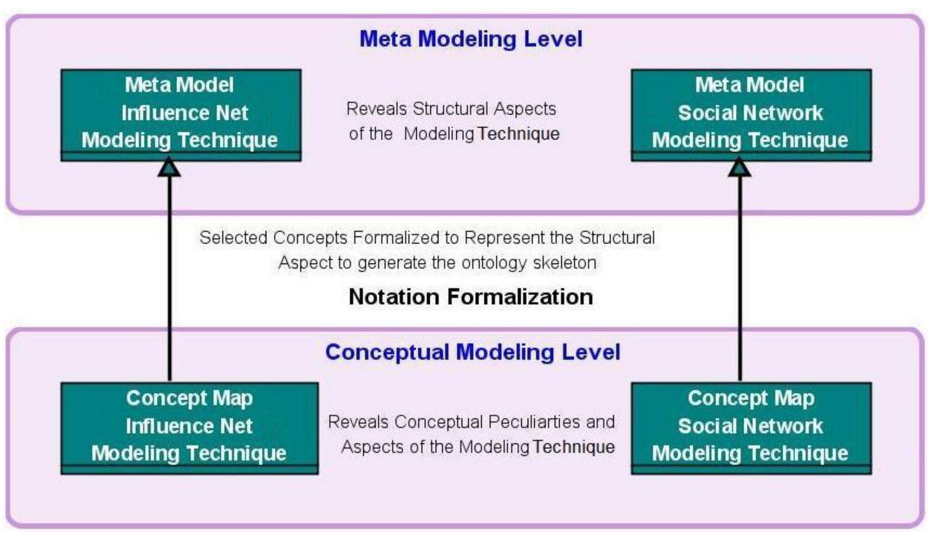


Figure 22: Meta-Modeling Level

The meta-models for Influence nets and Social networks are given in Figure 23 and 24. Every element in the meta-model has an associated description which provides an informal definition of this element. This description, along with *attributes* and *associations*, provides an abstract syntax of this element. The *attributes* define the properties of this element and the *associations* define its relationships with other meta-model elements as given in Table 1 and 2.

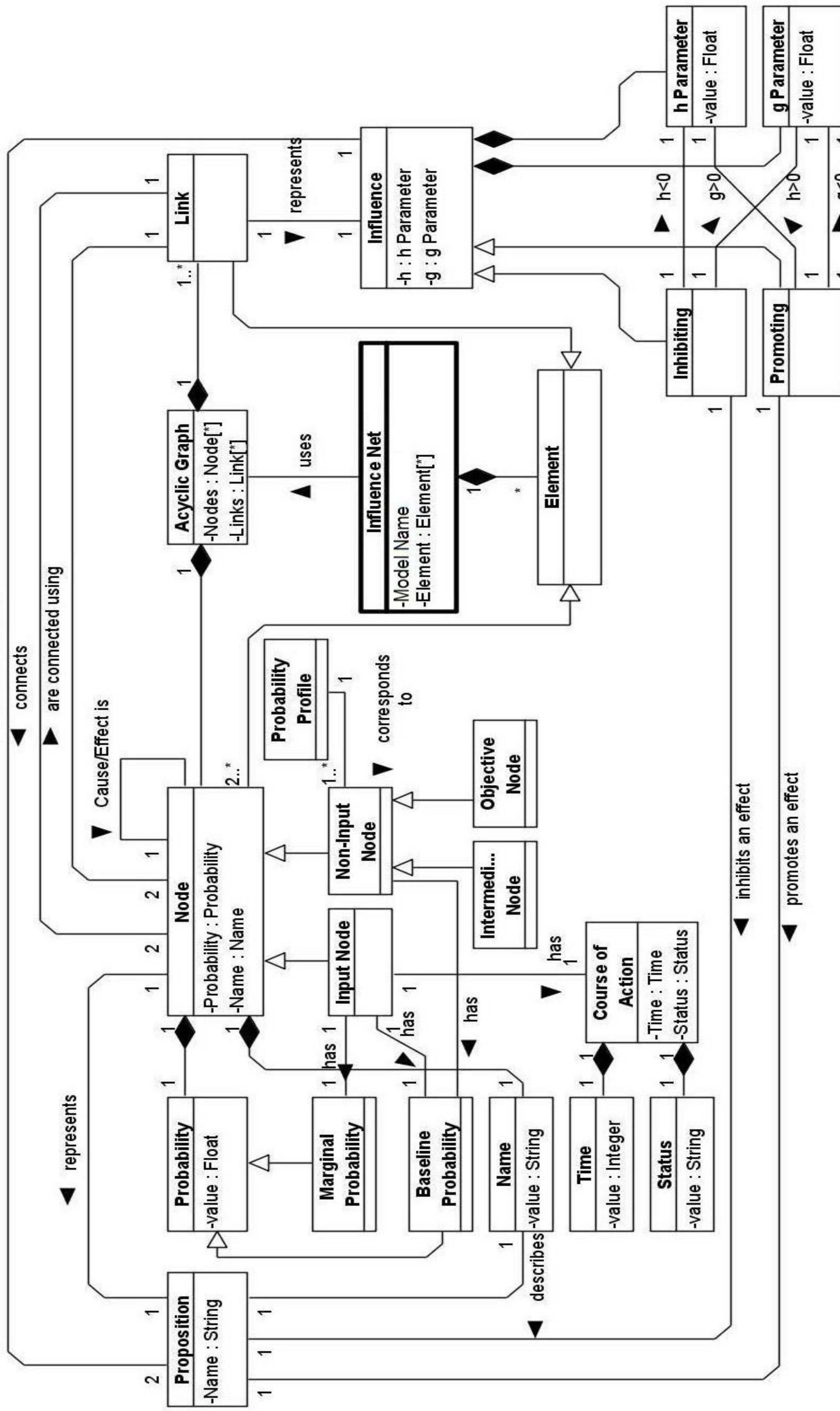


Figure 23: Influence net Meta Model

Table 1: Influence net Meta-model Elements

Element Name	Description	Attributes	Association	Generalization
Influence net	It represents an *Influence net* model instance developed for a specific domain	Model Name	*has modeling language* [1:1] with element *Acyclic Graph*	None
Acyclic Graph	It represents *Acyclic Graph* as the modeling language of an *Influence net* model	No additional attributes	No additional associations	None
Element	It is a constituent of every *Influence net* model.	No additional attributes	No additional associations	None
Node	It is an *Element* and a constituent of an *Acyclic Graph.*	*Name* and *Probability*	*represents* [1:1] with *Proposition*, *are connected using* [2:1] with *Link*, *Cause/Effect is Node* [1:1] with itself	*Element*
Input Node	It is a *Node* which has a *Course of Action.*	No additional attributes	*has* [1:1] with *Course of Action, has* [1:1] with *Marginal* and *Baseline Probability*	*Node*
Course of Action	It represents *Course of Action* as explained in Section 3.3c (i).	*Time* and *Status.*	No additional associations.	None
Time	It is a primitive type representing integer values for *Time.*	No additional attributes	No additional associations	None
Status	It is a primitive type representing Boolean values for *Status*	No additional attributes	No additional associations	None
Non-Input Node	It is a *Node* and does not have a *Course of Action*	No additional attributes	*has* [1:1] with *Baseline Probability*	*Node*
Intermediate Node	It is a *Non-Input Node* and does not have a *Course of Action*	No additional attributes	No additional associations	*Non-Input Node*
Objective Node	It is a *Non-Input Node* and does not have a *Course of Action*	No additional attributes	No additional associations	*Non-Input Node*
Probability Profile	It represents *Probability Profile* for each *Non-Input Node* as explained in Section 3.3c (i).	No additional attributes	*corresponds to* [1:1] with *Non-Input Node*	None
Probability	It is a primitive type representing floating point values for *Probability* between 0 and 1	No additional attributes	No additional associations	None
Baseline Probability	It represents Baseline probability	No additional attributes	No additional associations	*Probability*
Marginal Probability	It represents Marginal probability	No additional attributes	No additional associations	*Probability*
Name	It is a primitive type representing string values for the name of the *Node*	No additional attributes	*describes* [1:1] with *Proposition*	None
Proposition	It represents a proposition represented by a *Node*	*Name* (string)	No additional associations	None
Link	It is an *Element* and a constituent of an *Acyclic Graph*	No additional attributes	*connects* [1:2] with *Node*, *represents* [1:1] with *Influence*	*Element*
Influence	It represents *Influence* between two *Propositions* as explained in Section 3.2	*h Parameter*, *g Parameter*	No additional associations	None
Inhibiting	It is an *Influence* which inhibits a *Proposition*	No additional attributes	*inhibits* [1:1] with *Proposition*	*Influence*
Promoting	It is an *Influence* which promotes a *Proposition*	No additional attributes	*promotes* [1:1] with *Proposition*	*Influence*

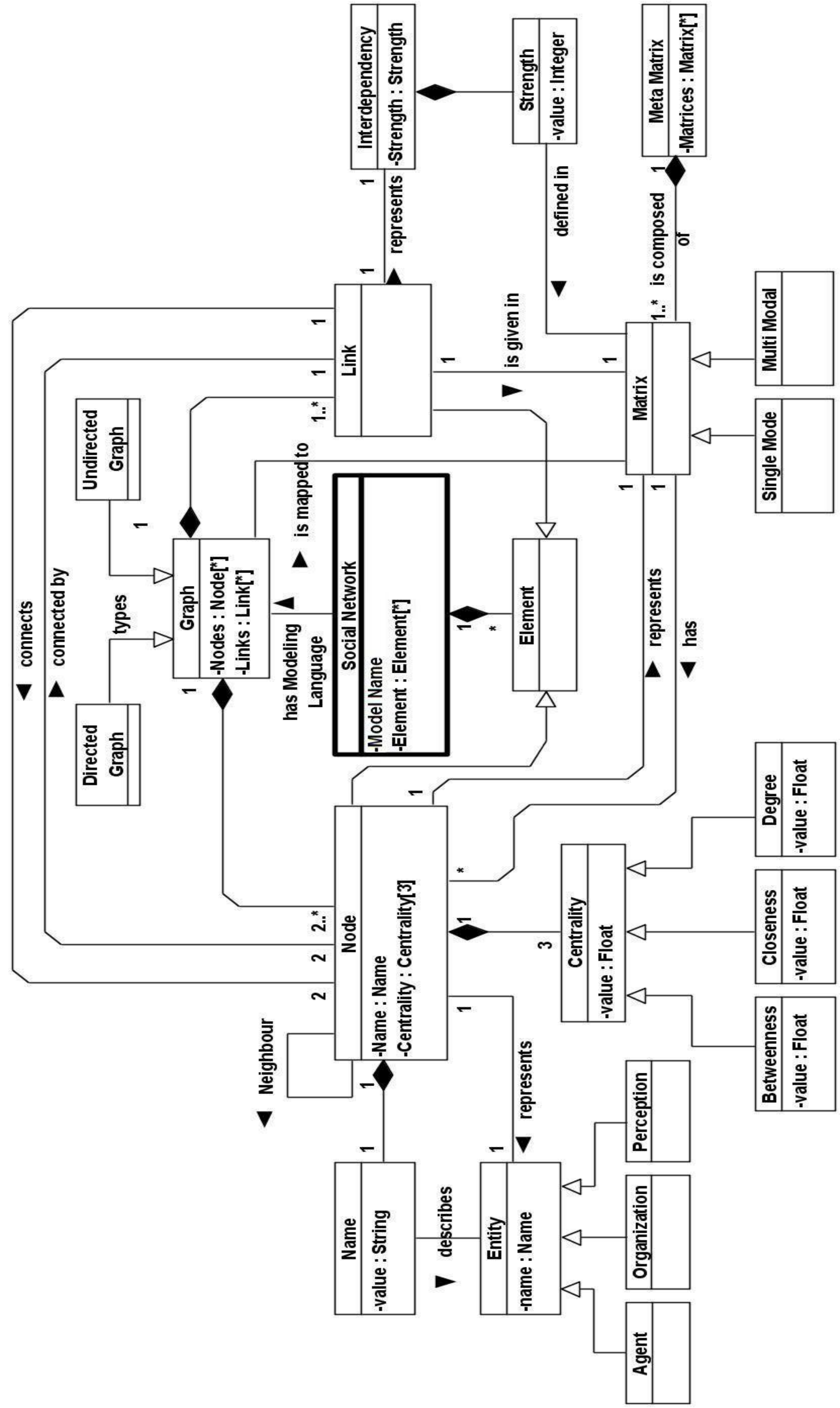


Figure 24: Social Network Meta Model

Table 2: Social network Meta-model Elements

Element Name	Description	Attributes	Association	Generalization
Social Network	It represents a *Social network* model instance developed for a specific domain	Model name	*has Modeling Language* [1:1] with element *Graph*	None
Graph	It represents *Graph* as the modeling language of a *Social Network* model instance	No additional attributes	*is mapped to* [1:1] with element *Matrix*	None
Directed/ Undirected Graph	They represent directed and undirected graphs as the types of element *Graph*	No additional attributes	No additional associations	*Graph*
Element	It is a constituent of every *Social network* model	No additional attributes	No additional associations	None
Node	It is an *Element* and a constituent of element *Graph*	*Name* and *Centrality*	*neighbor* [1:1] with itself, *represents* [1:1] with element *Entity*, *connected by* [2:1] with element *Link*	*Element*
Entity	It represents *Entity* represented by a *Node*	*Name* (string)	No additional associations	None
Agent	It represents an individual (agent) as a type of *Entity*	No additional attributes	No additional associations	*Entity*
Organization	It represents a group of individuals as a type of *Entity*	No additional attributes	No additional associations	*Entity*
Perception	It represents a perception (action, belief, event etc) as a type of *Entity*	No additional attributes	No additional associations	*Entity*
Name	It is a primitive type representing string values for the name of the *Node*	No additional attributes	*describes* [1:1] with *Entity*	None
Centrality	It represents the *Centrality* property that each social network node has	No additional attributes	No additional associations	None
Betweenness, Closeness and **Degree**	They represent betweenness, closeness and degree centralities as the types of element *Centrality*	No additional attributes	No additional associations	*Centrality*
Link	It is also an *Element* and a constituent of *Graph*	No additional attributes	*is given in* [1:1] with *Matrix, connects* [1:2] with *Node represents*[1:1] with *Interdependency*	*Element*
Interdependency	Interdependency represented by a *Link*.	No additional attributes	No additional associations	None
Strength	It represents the actual *strength* that each interdependency has between two nodes	No additional attributes	*defined in* [1:1] with element *Matrix*	None
Meta Matrix	It represents the *Meta-Matrix* as explained in Section 4.2 which is composed of other matrices	No additional attributes	No additional associations	None
Matrix	It represents the structure *Matrix* whose rows and columns constitute the nodes. It is also the constituent of *Meta-Matrix*	No additional attributes	*has* [1:*] with element *Node*	None
Single Mode, Multi-Modal	These represent the single and multi-modal matrices as explained in Section 4.1 as the type of element *Matrix*	No additional attributes	No additional associations	*Matrix*

5.6. Multi-Modeling

So far we have talked about the two types of modeling techniques and the models developed using them. Each of these models provides details about the domain it models only. It would be interesting, if these models can inter-operate with each other facilitating exchange of information, or more specifically, analysis results among them which could reveal new insights. This process of *combining diverse domain specific models in order to address a complex problem is called Multi-Modeling*. This concept of multi-modeling between Social and Influence net modeling techniques is presented in Figure 25.

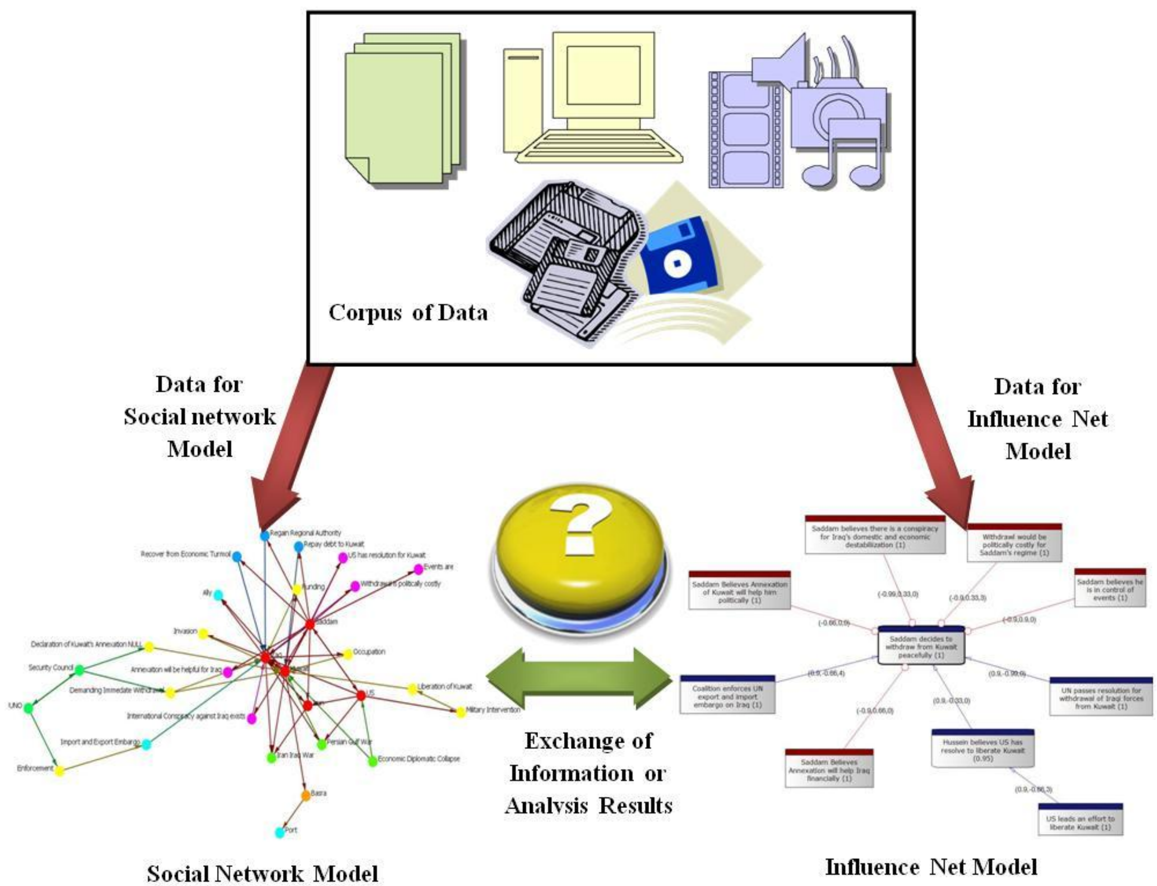


Figure 25: Concept of Multi-Modeling

The need for multi-modeling arises from the fact that using multiple models helps gain insights about the problem not offered by each model itself. Consider the Social network model constructed for a domain with the help of a corpus of data, for which an Influence net model is also constructed as shown in Figure 25. Since both models are constructed for the same domain, there can be certain aspects of the Social network model which might help refine or update the Influence net model or vice versa, the interactions of the two models might subsequently provide some new information or insight. For instance, analysis results from the Influence net model can be utilized to update the Social network model and vice versa. This inter-operation between models developed using the two techniques for the same domain to answer new multi-modeling queries is the process of multi-modeling.

5.7. Meta-Models and Ontologies

Meta-modeling plays a key role in the development of model description languages suitable for certain domains by defining the abstract syntax. For instance, UML specifications [15] define the abstract syntax for all of its UML models; similarly, meta-models define the abstract syntax for Influence and Social network models. These meta-models, however, only express the syntactic structures of the models, due to their implementation oriented focus. Meta-models are used to define languages to describe real world domains or systems, which implies that instances of meta-models are models instead of instances of models. However, ontologies contribute mostly to the modeling of real world domains or systems and describe real world entities as *Individuals*[9] and also capture certain semantics of the system which meta-models can't capture.

Since ontologies are used to effectively capture knowledge of a certain domain; they can be used to determine semantic equivalences between Influence and Social network modeling techniques as well. Due to the implementation oriented focus of meta-models, there might be certain *modeling technique specific* concepts hidden inside them, which can be revealed only with the help of ontologies by adding these concepts explicitly. However, the knowledge contained in the meta-models at this level can only be utilized to lay down the foundation of ontology only. This initial ontology, whose structure resembles that of the meta-models, is called *pseudo ontology*. In order to add the explicit concepts, knowledge from the concept maps can be utilized to feed into the ontologies to construct a *refactored* and then an *enriched ontology* which will contain both the structural aspects (from meta-models) and the semantic aspects (from concept maps and knowledge of the designer about the techniques). The terms *pseudo* and *enriched ontology* will be explained in detail in Chapter 7.

[9] Individual is an OWL construct representing the instance of a class (concept).

CHAPTER 6: ONTOLOGY

6.1 Ontology

An ontology defines a common vocabulary for researchers who need to share information in a domain and the relationships between the elements of that vocabulary. This information may include machine readable definitions of basic concepts in a domain [17]. An ontology can be considered as a thesaurus of words and inferences rules, where the words in the thesaurus represent concepts and the inference rules operate on the relationships on the words. It can also be defined as a knowledge base K = (TBox, ABox), where:

a. TBox is a finite set of concepts and relationships between the concepts.
b. ABox is a finite set of instances, relationships between instances, and the relationships between instances and concepts.

In the Web Ontology Language (OWL) ontology, classes are used to represent the concepts. Figure 26 shows the TBox of the *Pizza* ontology [18]. The TBox contains ontology concepts in hierarchical order only; *Meat Topping*, *Vegetable Topping* and *Cheese Topping* are sub-classes of the class *Pizza Topping*, which is a sub-class of class *Thing*. The ABox contains the instances of these concepts and the relationships between concepts and instances represented by the grey shaded boxes in Figure 27. For example, *American Hot Pizza* is an instance of *Non-Vegetarian Pizza*, *Cheesy Pizza* and *Named Pizza* whereas; *Has Topping* is the relationship between *Non-Vegetarian Pizza* and *Meat Topping*, *Has Topping* is also the relationship between *Cheesy Pizza* and *Cheese Topping* and between *Vegetarian Pizza* and *Vegetable Topping.*

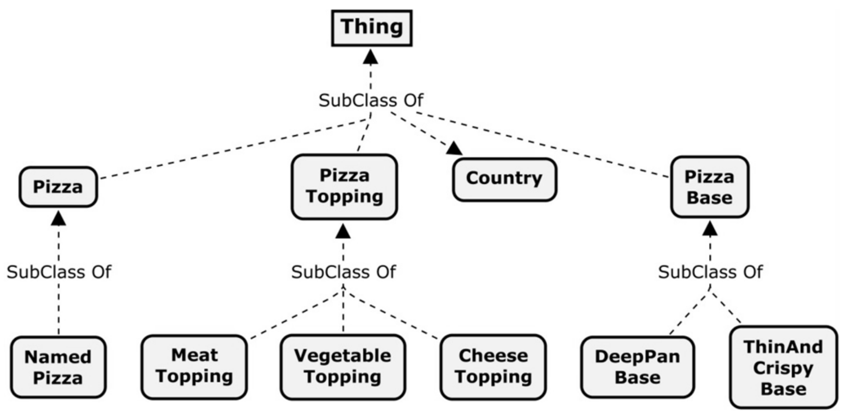


Figure 26: Pizza Ontology TBox

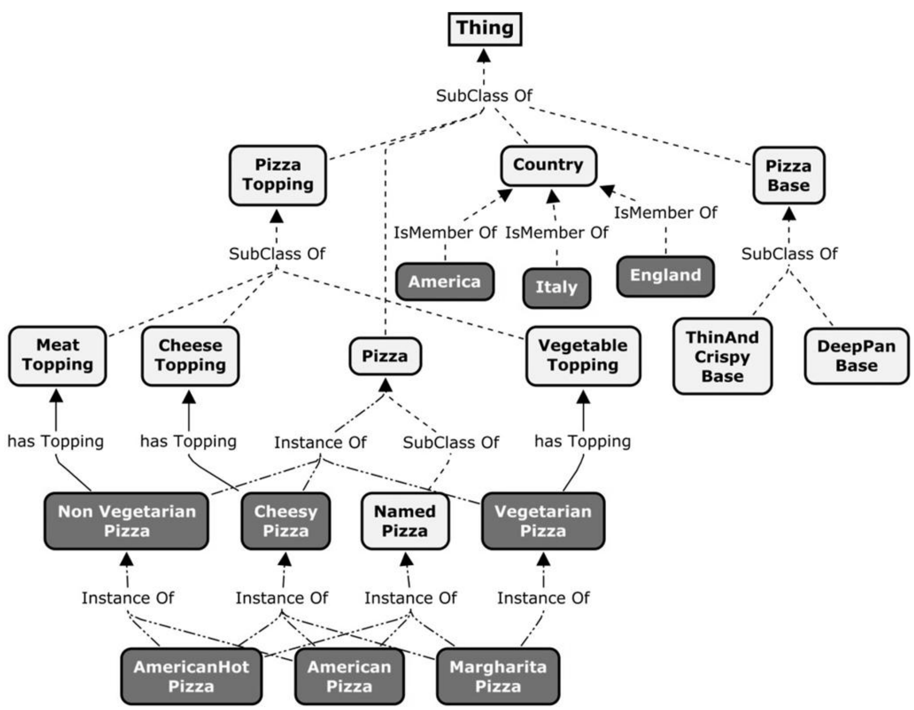


Figure 27: Pizza Ontology

There are different kinds of ontology languages available such as *Ontology Interchange Language* (OIL), *Resource Description Framework* (RDF) *Schema*, and *Web Ontology Language* (OWL).

6.2 Web Ontology Language (OWL)

OWL makes use of XML to encode the knowledge domain. This section discusses some aspects of OWL; refer to Mcguinness and Harmelen [19] for complete description of the language. The *Protégé-OWL editor* is one of the tools within the *Protégé* suite of applications [20] which allows construction of OWL based ontologies. OWL has been used to construct all of the ontologies in this thesis.

The components of the OWL ontology include a *class* which represents a concept and an *individual* which represents the instance of a class (or concept). Classes and individuals are the main focus of ontologies. There are *Object* and *Data Properties* which represent binary relationships on *individuals.*

The *Protégé-OWL editor* makes use of reasoners such as *Pellet* [21] or *Fact++* [22]. A reasoner can verify if all of the definitions in the ontology are mutually consistent with each other and can also recognize which concepts fall under which definitions. If an ontology is classified by a reasoner correctly, it means that the ontology is correct and all the definitions inside it are mutually consistent with each other. The components of the OWL ontology are as follows:

a. Class:

A class is a concrete representation of a concept and the main building block of the OWL ontology. The term *concept* is also used in place of *class*; you may encounter this throughout this document. An OWL class can also be interpreted as a set which contains individuals (instances of a class) that are described using formal (mathematical) descriptions that state precisely the requirements for membership of the class. Consider the example of the *Pizza Ontology* described earlier. A class *Country* would contain only those countries which are in our domain of interest, i.e., those countries where *pizza* is mostly eaten, such as *America*, *England*, and *Italy*. These will be the *individuals*, the instances of the class *Country*. The sub-classes specialize (or are subsumed by) their super-classes. For instance, *Meat Topping* is a sub-class of *Pizza Topping* which is a sub-class of *Thing*. Other ways of looking at this would be for *Pizza Topping* to be the super-class of *Meat Topping* which implies that any kind of *Meat Topping* is a *Pizza Topping* and all members of the class *Meat Topping* are members of the class *Pizza Topping* also.

b. Individual:

Individuals, also called *instances*, are instances of classes. For example, the instances of class *Country* would be *America*, *England* and *Italy*.

c. Properties:

Properties are the binary relationships between individuals (instances of classes), i.e., a property connects two individuals together in the form of a relationship. There are two types of properties in the OWL ontology as described below:

i. Object Properties:

A relationship between two individuals is defined by an *object property*. In OWL, an object property can have different characteristics, some of which are explained below and are illustrated in the *Characteristics* tab in Figures 28 & 29. For example, *has Topping* is an object property between *Pizza* and *Pizza Topping* classes.

- *Functional property*: It is the property exhibited by mathematical functions. Consider equation (iv) below which shows variable y as a function of variable x. It can be implied that for every unique value of x, there is a unique value of y, which means that via $f(x)$ only a unique value of x would result into a unique value of y.

$$y = f(x) = x + 1 \qquad \text{(iv)}$$

Similarly, in a functional property between two individuals there can be at most one individual that is related to the other one via the property. For example, individuals *James* and *Robert* are related via the object property *has Birth Father*, i.e., *James has Birth Father Robert*. In this case, the object property has to be functional, because a person can have only one birth father. If *James* is associated with another individual using this relationship, the ontology will not be classified when the reasoner *Pellet* is executed since it is not consistent with its definition of having a functional object property.

The following are the *equivalence properties of equality* in mathematics.

- *Transitive property*: It is a property exhibited by a transitive relationship and can be defined as a binary relation *R* over a set *X*, and holds for all *a*, *b*, and c in *X*, such that if a is related to b and b is related to c then a is related to c and can be expressed by the following notation:

$$\forall\ a, b, c \in X, aRb \wedge bRc \Rightarrow aRc \qquad \text{(v)}$$

For example, in the context of OWL consider a relationship *has Ancestor* between two individuals *Albert* and *Simon*. If *Simon* also has this property with another individual *Peter*, such that *Albert has Ancestor Simon* and *Simon has Ancestor Peter*, this implies that *Albert* also has an ancestor who is *Peter*.

- *Symmetric/Anti-symmetric property*: A symmetric property is exhibited by a symmetric relationship and can be defined as a binary relation *R* over a set *X* and holds for all *a* and *b* in *X*, such that if *a* is related to *b* then *b* is related to *a* and can be expressed by the following notation:

$$\forall\ a, b \in X, aRb \Rightarrow bRa \qquad \text{(vi)}$$

An anti-symmetric property is exhibited by an anti-symmetric relationship and can be defined as a binary relation *R* over a set *X* and holds for all *a* and *b* where *a* is not equal to *b* in *X*, such that if *a* is related to *b* and *b* is related to *a* then *a* is equal to *b*. It can be expressed by the following notation:

$$\forall\ a, b \in X,\ a \neq b,\ \text{if}\ \ aRb \wedge bRa \Rightarrow a = b \quad \text{(vii)}$$

For example, in the context of OWL consider *has Sibling* between two individuals *Simon* and *Samantha* i.e., *Simon has Sibling Samantha*, the property *has Sibling* can be symmetric property because it is also valid for *Samantha has Sibling Simon*. Similarly, an anti-symmetric property such as *is a Child Of* will only be valid from an individual representing a child to an individual representing the father and not the other way around. The reasoner will not be able to classify this ontology as consistent if the latter case is incorporated into the ontology.

- *Reflexive/Irreflexive property*: A reflexive property is exhibited by reflexive relation *R* on set *X* where for all a in *X*, a is *R*-related to itself and can be expressed by the following notation:

$$\forall\, a \in X, aRa \qquad \text{(viii)}$$

An irreflexive property is exhibited by an irreflexive relation *R* on set *X* where for all a in *X*, a is never *R*-related to itself and can be expressed by the following notation:

$$\forall\, a \in X, a\not{R}a \qquad \text{(ix)}$$

For example, an individual *Simon knows* himself. The reflexive property *knows* relates *Simon* to himself.

ii. Data Properties:

A relationship between an individual and data values is defined by *data properties*. These properties link an individual to an XML Schema Datatype value. Most of the object property characteristics don't apply to data properties except for the functional property as it associates an individual to a value. The domain represents individuals but the range can only have XML Schema Datatype values such as *XMLLiteral*, *byte*, *Boolean*, *int*, *float* etc. This is illustrated in Figure 28 which shows an object property *hasDiameter* for a *PizzaSize* class which represents different sizes of the class *Pizza* (sizes are not shown in the figure). The datatype of *hasDiameter* is specified to be *int.*

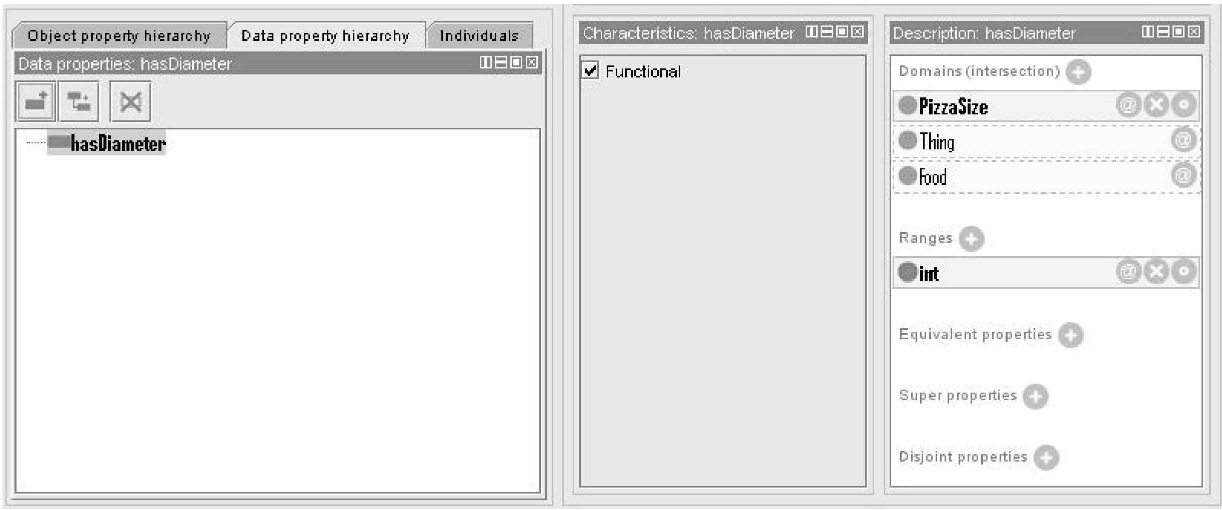


Figure 28: Data Property for class *PizzaSize*

d. Domain & Range:

The *domain* of a mathematical function is the set of all possible input values for which the function is defined i.e., domain of $f(x) = \{\forall x \mid y = f(x)$ is defined$\}$. For example, consider the function given in equation (iv), domain of this function constitutes all real numbers.

The *range* whereas is the set of all possible output values as a result of using all possible input values in the domain. i.e., range of $f(x) = \{\forall y \mid \exists\, x$ in the domain of $f(x)$ such that $y = f(x)\}$. For example, consider the function given in equation (iv), range of function $f(x)$ would be real numbers as a result of using the input values in the domain.

In the context of OWL, properties also have specified *Domains* and *Ranges*. If specified, properties link an individual from the *Domain* to the individuals in the *Range.* Both of them represent a collection of possible classes which constitute *Domain* and

Range. For example, the property *has Topping* can have a domain *Pizza* and range *Pizza Topping*, which implies that *has Topping* will only relate individuals from the *Pizza* domain to individuals in the *Pizza Topping* range as shown in Figure 29. Domain and ranges are used as axioms while reasoning. In addition, domain and range can be empty, if an object property relates all of the classes defined in the ontology.

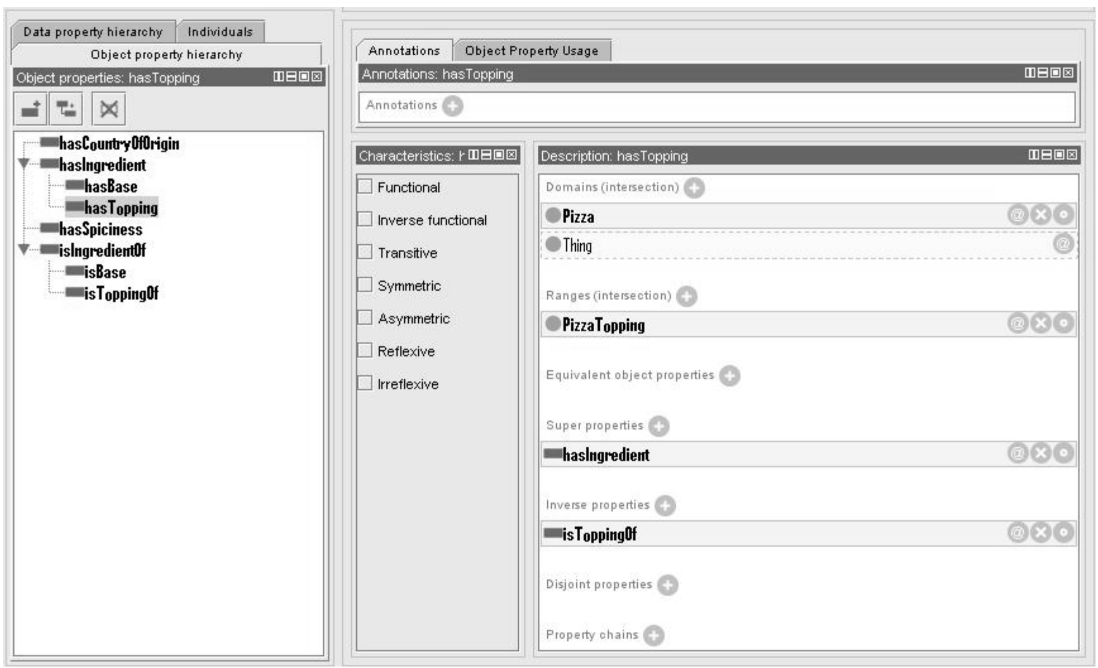


Figure 29: Domain and Range of *hasTopping* Object Property

e. Asserted Class Hierarchy:

In the Protégé 4.0 editor for OWL, the view that shows the class hierarchy in exactly the way the classes are created is the *Asserted class hierarchy;* it can also be called the manually constructed class hierarchy without executing the reasoner. Any addition or deletion of classes is directly reflected on the view as shown in Figure 30.

f. Inferred Class Hierarchy:

The class hierarchy which is automatically computed by the reasoner is called *inferred class hierarchy*. If the reasoner executes successfully, it means that the ontology is classified correctly, and all of the definitions are mutually consistent with each other. It also recognizes which classes (concepts) fall under which classes as sub-classes. For instance, as shown in Figure 31, the class "-_Adam_completes_work_-" is classified as a sub-class of *Proposition.* It has an object property *propositionName* which is the object property of class *Proposition.* This makes this new class a sub-class of *Proposition*. When the reasoner is executed, it identifies this new class as a sub-class of *Proposition* and classifies it under its super-class as shown by the directed arrow between Figures 30 and 31.

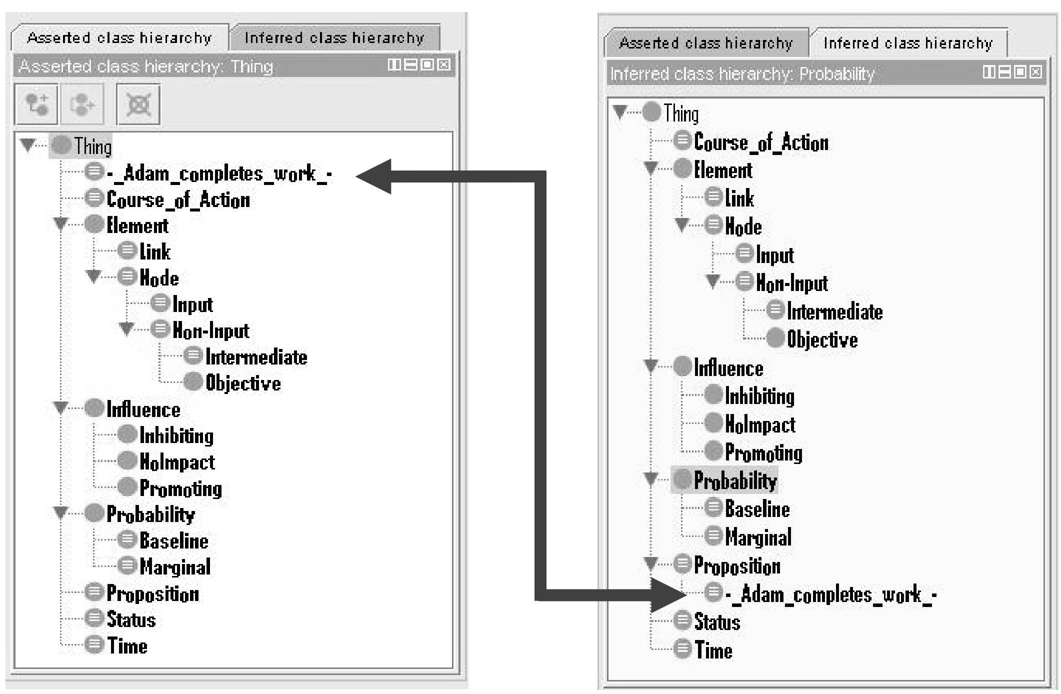


Figure 30: Asserted Class Hierarchy of Influence net Pseudo Ontology

Figure 31: Inferred Class Hierarchy of Influence net Pseudo Ontology

6.3 Why Construct Ontology?

Many different disciplines have been developing ontologies which can be used by domain experts to share and annotate information in their fields. Some of the important reasons for constructing ontology could be:

a. To share common understanding of information regarding a domain.

b. To enable reuse of domain knowledge.

c. To analyze domain knowledge.

One of the most important things to remember about ontologies is that there is no perfect ontology for a specific domain. Ontology designing is a creative process and construction is an iterative process. The potential applications and the level of understanding possessed by the designer certainly affect the design decisions, as a result of which no two ontologies developed for the same domain (by different designers) would be the same.

6.4 Use of Ontologies for Semantic Extraction

Ontologies are used to capture knowledge about a domain. Use of ontologies for semantic extraction from models has been an active research area. Saeki and Kaiya [23] have proposed using the inference rules of ontologies for domain specific semantics of models and semantics for meta-models in order to detect semantic inconsistencies included in models and meta-models. They point out how model engineers and meta-model engineers can try mapping their model elements to the respective ontology classes (concepts) to ensure consistency in their constructed models.

6.5 Ontology and Schema Matching

There are various tools available that perform schema and ontology matching using different kinds of matching strategies. COMA++ [24], [25], [26] is one such tool that can be used to identify semantic correspondences and mappings between different ontologies. It provides a comprehensive graphical user interface with different matching strategies. However, the mappings between refactored ontologies of both modeling techniques in this thesis have been determined manually; use of tools such as COMA++ is a suggested extension of this work.

6.6 Steps of Ontology Construction

There is no specific process for constructing an ontology that can be claimed as being the correct one. The construction process depends upon the knowledge and objective of the ontology. Ontology construction in this thesis utilizes the iterative approach by Noy and McGuinness [17] as follows.

a. Determination of Domain[10] of Ontology:

A number of questions need to be answered while starting constructing ontology for a domain such as:

a. *What is the domain of the ontology?* The domain of both ontologies is the containment of knowledge regarding Influence nets and Social networks (for a specific scenario).

[10] This domain is a bit different than the one explained in Section 6.2c (ii).

b. *What is the use of it?* The ontologies will be used to construct an enriched ontology which contains the mapped concepts of both types of ontologies. Since the knowledge contained in this enriched ontology is in machine readable form, it can be further re-used by other applications to execute model-specific queries such as retrieving all agents, or knowledge elements (from Social network related concepts) or all action and belief propositions (from Influence net concepts).

c. *Who is the user and maintainer of it*? The modeler of (Influence net and Social networks) will be the user of this ontology and the ontology designer would be the maintainer.

d. *What kind of questions will it answer?* The ontologies are to answer most of the questions identified in the concept mapping phase (Chapter 3 and 4) such as the constructs and their meanings and the kinds of analyses supported by each technique.

b. Reusing Existing Ontologies:

Reusing an ontology constructed by someone else can be helpful as it can be refined or modified according to the needs of the designer and the domain. There are different libraries of ontologies available online for specific domains which can be used as needed [27], [28], [29].

c. Enumerating Important Terms:

It is useful to start writing down on paper the terms that would become part of the ontology. In order to do that, one can ask such questions as *what terms we have to talk about*, and *what properties do they have*.

d. Define Class, Class Hierarchy and Individuals:

Once the terms are enumerated, they can be defined as classes and individuals in the ontology. To define the class hierarchy, approaches like top-down, bottom-up or a combination of both can be used. In a top-down approach, most general classes are defined first leading towards specialization. For instance, the most general concepts of an Influence net would be *proposition* and *influence* leading towards specialized concepts such as propositions types *action, belief, event* and influence types *inhibiting, promoting, no impact*.

e. Disjoint Classes:

When defining the classes and class hierarchy, it is important to define which of the classes are disjoint with which other classes. For instance, the classes representing the types of influences such as *Inhibiting*, *Promoting* and *No Impact* all are disjoint with each other, i.e., either an influence is inhibiting, promoting or no impact.

f. Define Properties, Domain, and Range:

Once classes, individuals, and a class hierarchy are defined, we need to describe the internal structure of each of these classes which can be done by defining their properties. For instance, in Influence nets, every proposition is composed of certain elements such as *subject*, *object* and a *verb.* For the *Proposition* class we can define an object property *has Elements* with a domain value *Proposition*. The class *Inhibiting* will have a data property called *has Inhibiting Value*. Since an inhibiting influence has a certain data value, we can define it by a data property *has Inhibition Value* with range *floating point* and domain *Inhibiting*.

g. Define Necessary and Sufficient conditions

Conditions which are necessary to be fulfilled to be a member (sub-class) of a class are called *necessary* conditions. However, by fulfilling necessary conditions alone we can't say that something can be a member of this class. In order to be sure of this we need to define *necessary and sufficient* conditions. For instance, in the *Pizza* ontology described earlier, *necessary* condition for a class to be a *Cheesy Pizza* would be to be the sub-class of *Pizza* or to have at least one *Cheese Topping*. Each one is a *necessary* condition. To be certain, both of these conditions need to be fulfilled and this constitutes the *necessary and sufficient* conditions.

When we define the *necessary* conditions, we actually specify the necessary conditions for the membership of a class. In Protégé 4.0, *necessary* conditions are simply called *Superclasses*. These *necessary* conditions are displayed under *Superclass* header in the class description view (Figure 38). This header also displays those classes which subsume this class based on the defined properties.

In order to convert *necessary* conditions to *necessary and sufficient* conditions, the conditions must be moved from under the *Superclass* header to be under *Equivalent class* header. In Protégé 4.0, *necessary and sufficient* conditions are called *Equivalent classes* and all *necessary and sufficient* conditions are displayed under its header (Figure 38). Under this header also are shown all those classes which are inferred by the reasoner to be equivalent based on their defined properties.

6.7 Influence Net Ontologies

The basic structure of the Influence net ontology was established with the help of the meta-model given in Section 5.5. This initial ontology is called *Pseudo Ontology* whose structure resembles that of the meta-model. The idea is to have a basic ontology which can be enhanced by adding explicit concepts later on. These explicit concepts can either be added by the designer or from the concepts identified in the concept mapping phase given in Sections 2.3 and 3.3 as shown in Figure 32. The enhanced ontology is called *Refactored Ontology*. It is a complete ontology for the Influence net modeling technique containing its relevant concepts and relationships.

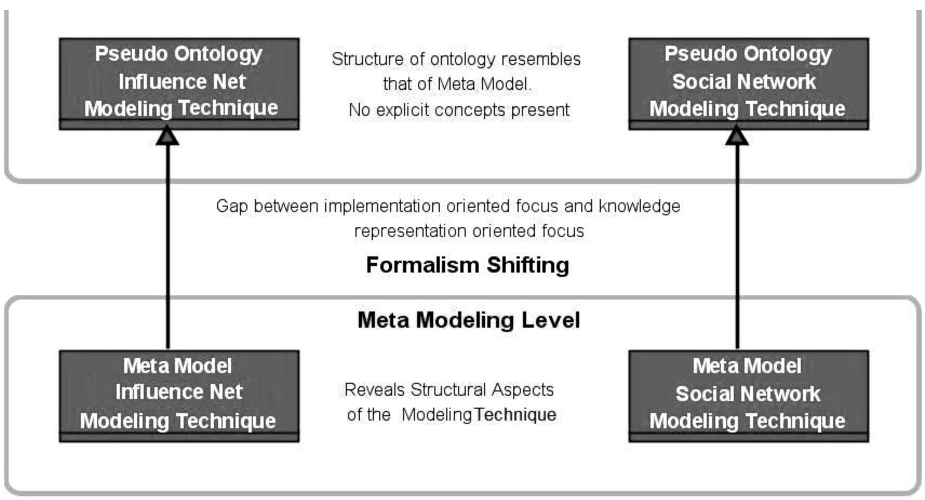


Figure 32: Formalism Shifting

a. Influence net Pseudo Ontology

The concept of formalism shifting as given by Kappel et al. [1] aims at eliminating the gap between the implementation oriented focus of meta-models and the knowledge representation focus of ontologies. Figure 33 illustrates the pseudo ontology for the Influence net modeling technique. In ontology, every concept (class) is fundamentally the concept *Thing*. In other words, all sub-sequent concepts (classes) are the sub-classes of the concept *thing.* In the figure, each class (box) is stereotyped using its own type and its inherited types. For instance, an *Input Node* is stereotyped as a *Node*, *Element*, and *Thing* which means that every *Input Node* is a *Thing* of type *Element* which is of type *Node*.

Table 3 shows a comparison of the pseudo ontology classes and properties with the meta-model elements and associations. The grey shaded cells represent the common entities (concepts in ontology and elements in meta-modeling). The table is followed by a description of the ontology elements.

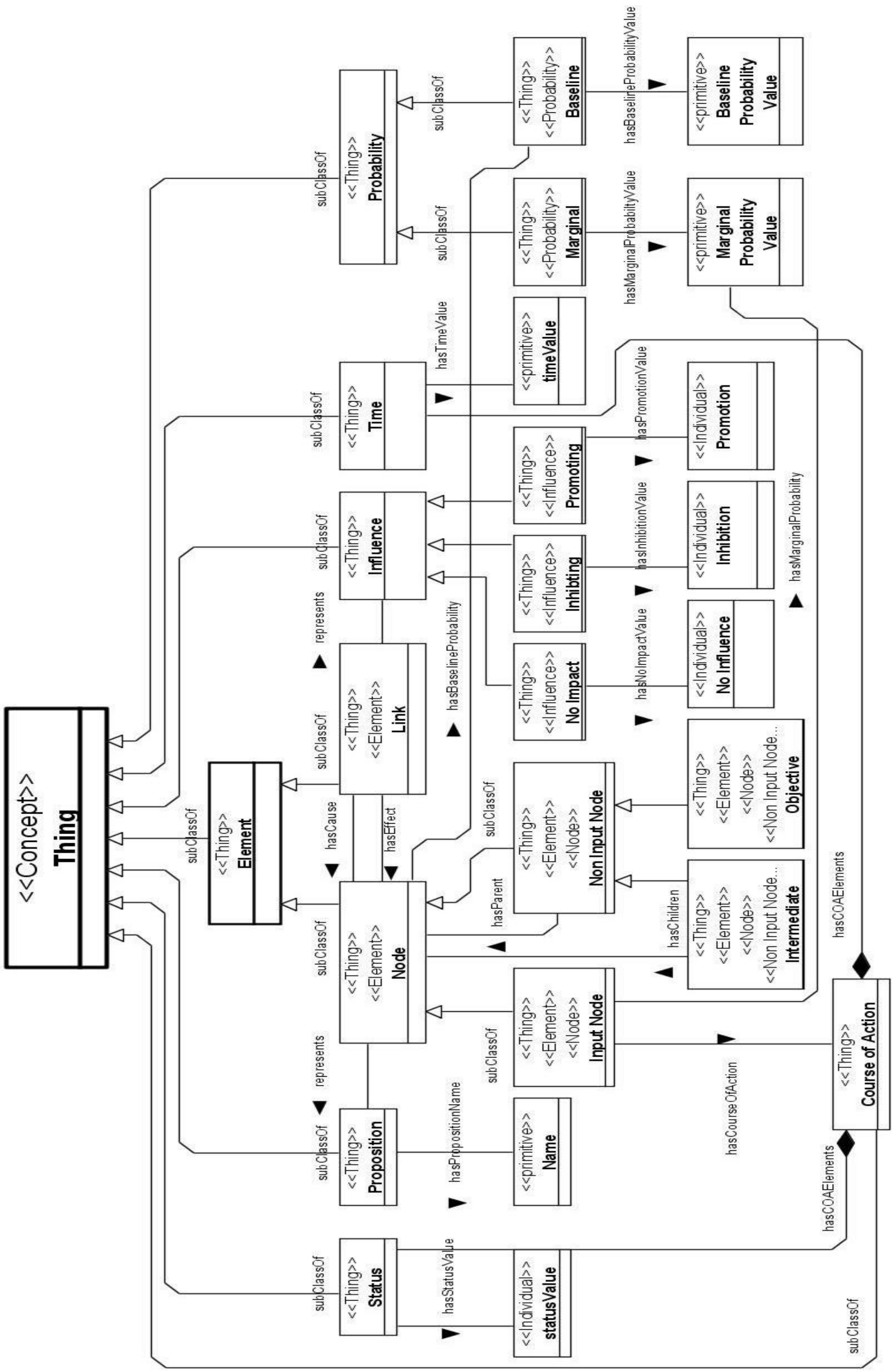


Figure 33: Influence net Pseudo Ontology

Table 3: Pseudo Ontology & Influence Net Meta Model Elements

Ontology Domain Class	Ontology Object Property	Ontology Range Class	Meta-Model Element	Association	Meta-Model Element
Element	superClassOf	Node	**Element**	Types	Node
		Link			
	subClassOf	Thing			Link
Node	superClassOf	Input Node	**Node**	Types	Input Node
		Non Input Node			Non-Input Node
	subClassOf	Element		Type of	Element
	representsProposition	Proposition		Represents	Proposition
	hasBaselineProbability	Baseline		is composed of	Probability
					Name
Link	subClassOf	Element	**Link**	Type of	Element
	hasCause	Node		connects	Node
	hasEffect	Node			
	representsInfluence	Influence		Represents	Influence
Proposition	subClassOf	Thing	**Proposition**	-	-
	hasPropositionName	Name [Individual]			
Influence	subClassOf	Thing	**Influence**	Types	Inhibiting
	superClassOf	Inhibiting			
		Promoting			Promoting
Input Node	subClassOf	Node	**Input Node**	Type of	Node
				Has	Baseline Probability
	hasMarginalProbability	Marginal			Marginal Probability
	hasCourseOfAction	Course of Action			Course of Action
Non Input Node	superClassOf	Intermediate	**Non Input Node**	Types	Intermediate
		Objective			Objective
	subClassOf	Node		Type of	Node
	hasParent			Has	Baseline Probability
Intermediate	hasChildren	Node	**Intermediate**	-	-
Course of Action	subClassOf	Thing	**Course of Action**	is composed of	Time
	hasCOAElements	Time			Status
		Status			
Time	hasTimeValue	Integer [Datatype]	**Time**	-	-

Ontology Domain Class	Ontology Object Property	Ontology Range Class	Meta-Model Element	Association	Meta-Model Element
Status	hasStatusValue	True	**Status**	-	-
		False			
Inhibiting	hasInhibitionValue	Inhibition [Individual]	**Inhibiting**	Inhibits an effect	Proposition
Promoting	hasPromotionValue	Promotion [Individual]	**Promoting**	Promotes an effect	Proposition
No Impact	hasNoImpactValue	No Influence [Individual]	**-**	-	-
Probability	superClassOf	Marginal	**Probability**	Type	Marginal
		Baseline			Baseline
	subClassOf	Thing			
Marginal	hasMarginalProbability Value	Marginal ProbabilityValue [Individual]	**Marginal Probability**	-	-
Baseline	hasBaselineProbability Value	Baseline ProbabilityValue [Individual]	**Baseline Probability**	-	-
-	-	-	**Probability Profile**	CorrespondsTo	Non-Input

Following are the classes and properties of the Influence net pseudo ontology.

i. **Element**: An *element* is a sub-class of *thing*. Its corresponding meta-model element is also called *element*. Following are its sub-classes:

- ***Node:*** It is a sub-class of *element* and is the super-class of *Input Node* and *Non-Input Node* classes. The corresponding meta-model element of *Node* is also *Node* with the same classification of its types as *Input Node* and *Non Input Node*. This class has the following object properties:

representsProposition A *Node* represents a *Proposition*. The domain of this property is *Node* and the range is *Proposition*. This property is functional as a node can only represent one proposition.

hasBaselineProbability Every node (*Input* or *Non-Input)* has a baseline probability. The domain of this property is *Node* and the range is *Baseline.* This property is also functional as every node can have only one baseline probability.

The sub-classes of class *Node* are explained next.

- *Input Node:* It is a sub-class of *Node* representing only those nodes which don't have any parents. It has the following object properties in addition to the properties inherited from its super-class *Node*:

hasCourseOfAction Course of action is assigned only to input nodes. The domain of this property is *Input Node* and the range is *Course of Action*. *Course of Action* is a separate concept (class) in the ontology. This property is functional, considering that input nodes can have only one course of action.

hasMarginalProbability Every input node has an associated marginal probability. The domain of this property is *Input Node* and the range is *Marginal* which is a separate concept (class) in the ontology. This property is functional since each input node can have only one marginal probability.

- *Non Input Node:* It is a sub-class of *Node* representing those nodes which have parents. It is also a super-class of *Intermediate* and *Objective* classes. It has the following object properties in addition to the properties inherited from its super-class *Node*:

hasParent Every non input node has an associated parent (cause). The domain of this property is *Non Input Node* and the range is *Node.* This property is not functional, as any node can have more than one parent (cause).

There are further two sub-classes of *Non Input Nodes* as given below; their corresponding meta-model elements are also same, i.e., *Intermediate* and *Objective Node*:

<u>*Intermediate*</u>: It is a sub-class of *Non Input Node* representing only those nodes which have both parents (causes) and children (effects) and have the following object property in addition to the properties inherited from its super-class *Non Input Node*:

hasChildren This property is similar to *has Parent* property explained earlier. The domain of this property is *Input Node* or *Intermediate* and the range is *Node.* It is not functional as any node can have more than one child (effect).

<u>*Objective*</u>: It is a sub-class of *Non Input Node* representing only object nodes that don’t have any children (effects). They don’t have any additional object properties besides the ones inherited from their parents (causes).

- ***Link:*** It is a sub-class of *element* and has the following object properties. Its corresponding meta-model element is also the same.

represents	A *Link* represents *Influence* which is a separate concept (class) in the ontology. The domain of this property is *Link* and the range is *Influence*. This property is functional, as a link can represent only one influence.
hasCause	Every link associates a cause to an effect. The domain of this property is *Link* and the range is *Node.* Since a link can only have one cause, this property is functional.
hasEffect	Similarly, the domain of this property is *Link* and the range is *Node.* Since a link can only have one effect, this property is also functional.

ii. **Proposition:** A *Proposition* is a sub-class of *Thing* and its corresponding meta-model element is also *Proposition.* It has the following datatype property:

propositionName	Every proposition is described by a textual description. The domain of this property is *Proposition* and the range is datatype *string*, since it is a datatype property. This property is functional, as only a proposition can be described using only one string.

iii. **Influence:** It is a sub-class of *Thing* with the same corresponding meta-model element. It doesn't have any object properties but has sub-classes representing the three kinds of influences. There are total nine defined instances of class *Influence*. Four for each of *Inhibiting* and *Promoting* classes, and one for *No Impact.* The instances have pre-defined levels and values of influences as shown in Figures 34 and 35.

- ***Inhibiting:*** This class represents the inhibiting influences having four member instances (*individuals),* i.e. *MaximumInhibition*, *ModerateInhibition, ModeratelyLessInhibition, and MinimumInhibition.* Since influence is defined by an influence value, a datatype property would be suitable to define it. The *Inhibiting* class has the following datatype property.

hasInhibitionValue	Its range is between 0 and -0.99. All corresponding inhibiting influence values are pre-defined, such as -0.99 for *MaximumInhibition,* -0.33 for *MinimumInhibtion* and so on, as shown in Figure 35.

- ***Promoting:*** This class represents the promoting influences and has four member instances (*individuals),* i.e. *MaximumPromotion*, *ModeratePromotion, ModeratelyLessPromotion* and *MinimumPromotion*. It has the following datatype property:

hasPromotionValue	Its range is between 0 and 0.99. All corresponding promoting influence values are pre-defined, such as 0.99 for *MaximumPromotion,* 0.33 for *MinimumPromotion* and so on.

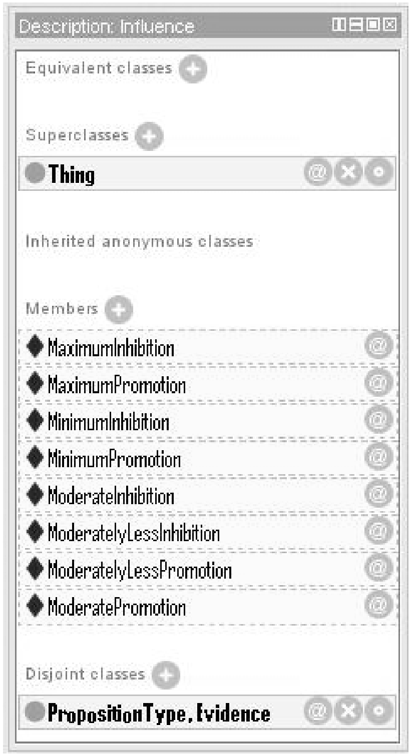


Figure 34: Member Instances of Influence Class

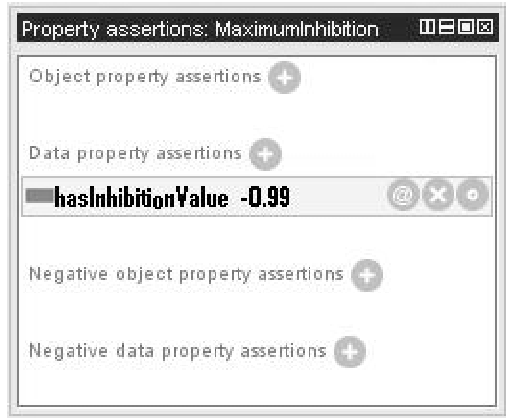


Figure 35: Value of an Influence Instance "Maximum Inhibition"

- ***No Impact:*** This class was explicitly added into the ontology to complete the concept of influence. This class has a member instance (*individual)* called *No Influence*. The pre-defined values for *No Influence* are specified using both datatype properties *hasInhibitionValue* and *hasPromotionValue* having value exactly 0 each.

iv. **Status:** It is a sub-class of *Thing* and is one of the constituents of class *Course of Action*. It corresponds to the same meta-model element and has two member instances (individuals) *TRUE* and *FALSE.*

v. **Time:** It is a sub-class of *Thing* and is the second constituent of the class *Course of Action.* It also corresponds to the same meta-model element and has the following datatype property:

hasTimeValue Instance of class *Time* will define an integer value of time using this property. The domain of this property is *Time* and range is datatype integer.

vi. Probability: This class was created considering the two distinct types of probabilities used in Influence nets. *Probability* is a sub-class of *Thing*, and super-class of *Marginal* and *Baseline* classes as follows:

- ***Marginal:*** It is a sub-class of *Probability* representing the marginal probability of a node. It has the following datatype property:

  hasMarginalProbabilityValue The domain of this property is *Node* as only sub-classes of *Node* can have marginal probabilities, and range is the real closed set [0, 1].

- ***Baseline:*** It is a sub-class of *Probability* representing the baseline probability of a node. It has the following datatype property:

  hasBaselinePropertyValue The domain of this property is also *Node*, and range is the real closed set [0, 1].

b. Influence net Refactored Ontology:

Once the pseudo ontology construction is completed, we can use the knowledge obtained from the concept maps, as well as the knowledge of the designer about the modeling technique, to define and feed new concepts (classes) and relationships (properties) explicitly into the pseudo ontology to construct a completely specified

Refactored ontology for Influence nets. This process is shown in Figure 36. For clarity, the refactored ontology illustrated in Figure 37 shows only the new concepts that are added to the pseudo ontology.

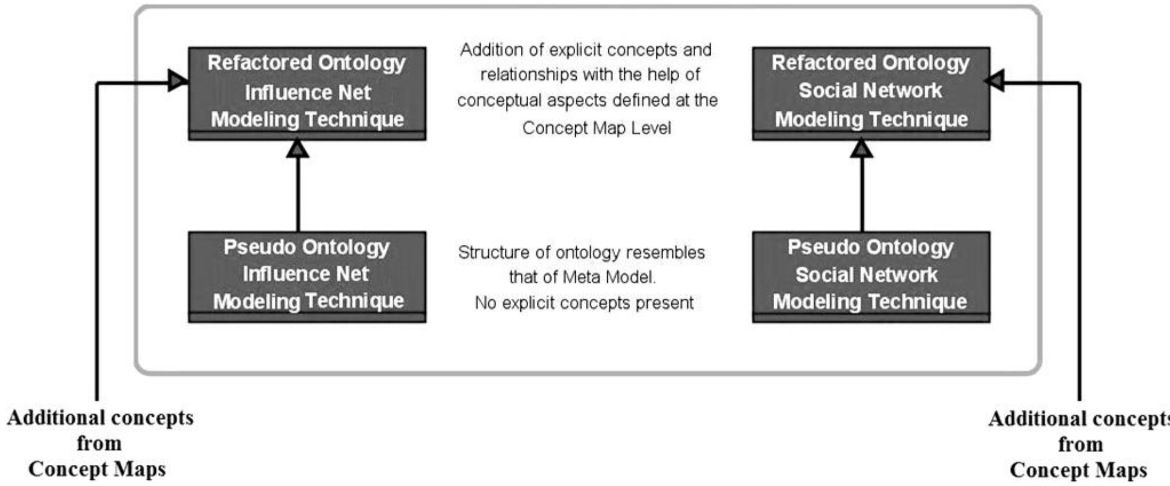


Figure 36: Refactored Ontology Construction

The classification of propositions is a critical step. There can be many types of propositions besides the ones mentioned earlier. The actual aim is to come up with an initial set of proposition types that are the most recurrent among the two models studied. Since ontology construction is an iterative process, its evolution can continue over time. As new types emerge, they can be incorporated as new classes and properties into the existing ontologies.

Based on the definitions of propositions given in Section 7.6, the constituents of each proposition were considered as new concepts (classes), if not already present in the ontology. The explicit concepts (classes) added to the ontology are *subject*, *object*, *verb*, *outcome*, *quality*, *PropositionType* etc.

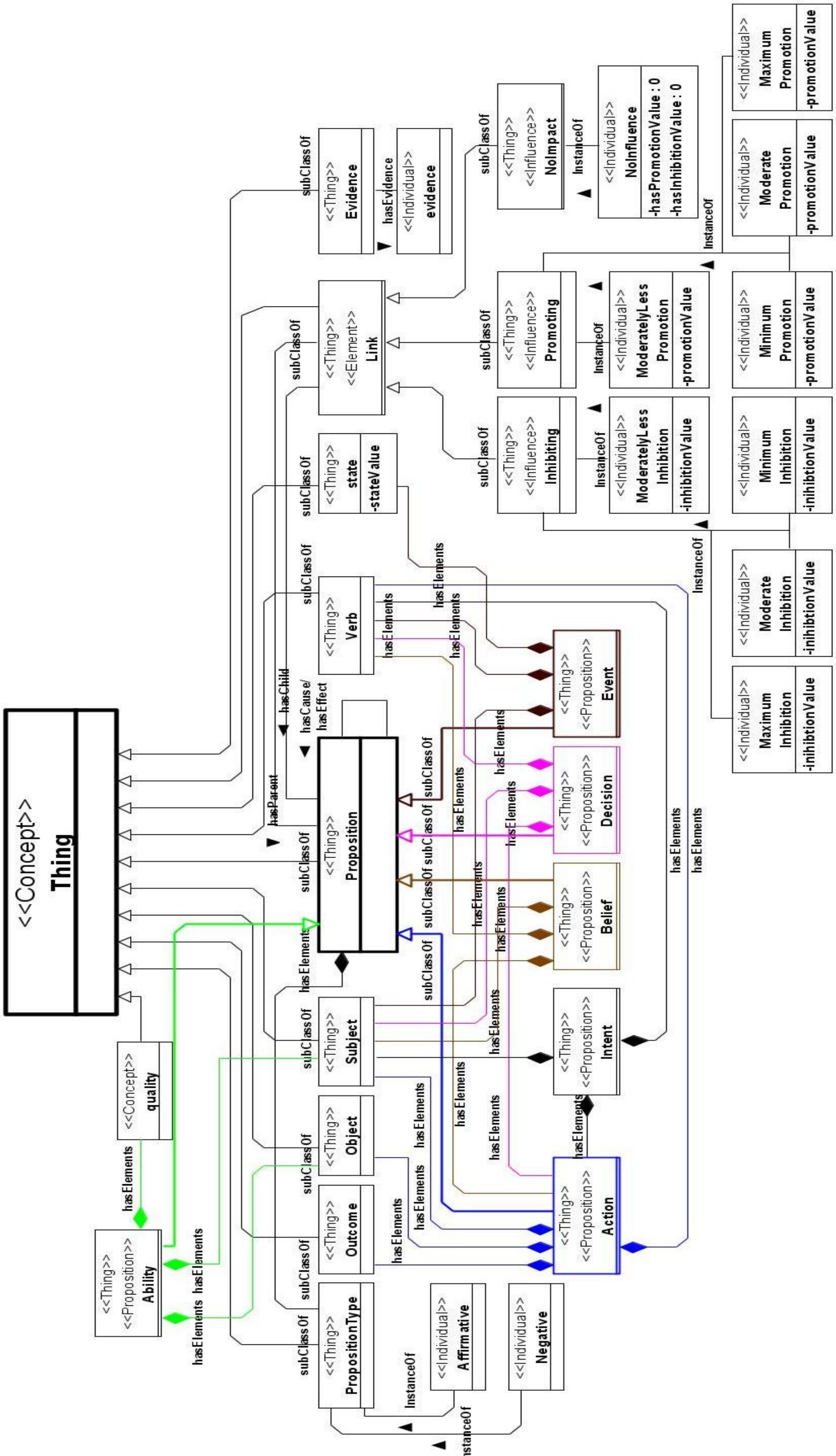


Figure 37: Influence Net Refactored Ontology (additional concepts only)

The following classes and properties were explicitly added into the refactored ontology in addition to the ones already present in the pseudo ontology. The first six are the sub-classes of the *Proposition* class and each one of them has the same object property *hasElements*, the same domain as *Proposition*, but different range, as each sub-class of *Proposition* class may be related to other similar or different classes (see Section 7.6a for definitions and examples). Some sub-classes are disjoint with others, e.g., *Action* is disjoint with *Intent, Belief,* and *Event* based on their properties. This is displayed at the bottom of each figure (Figures 38-45). The constituent classes *subject*, *object*, *verb*, *quality* are described later in this section. All *Proposition* sub-classes have necessary and sufficient conditions defined, which are shown under *Equivalent classes* in each figure below. The *necessary conditions* are shown under *Superclasses:*

i. **Class *Ability*:** It is the power or practical ability required to do something. The constituents are *subject*, *object* and *quality*. The class *quality* is explained ahead in this section and represents a *subject*'s quality of being able to do an action such as "can complete", "can perform", "can deliver" etc. *subject*, *object* and *quality* form the range of *hasElements* as shown in Figure 38. It implies that any class having a subject, object and quality will become the sub-

Figure 38: Ability Class

class of class *Ability*.

ii. **Class *Action:*** It refers to doing something towards achieving an outcome. The range of *hasElements* for this class is *subject*, *object*, *verb* and *outcome* as shown in Figure 39. It follows that any class which has a subject, object, verb and an optional outcome becomes the sub-class of *Action* class. The class *outcome* is optional as not every action necessarily has to have an outcome.

Figure 39: Action Class

iii. **Class *Belief*:** It refers to the degree of conviction of truth based on some evidence. Its constituents are *subject*, *verb*, *proposition (Ability*, *Action*, *Decision* or *Event)*, and an optional *evidence* as shown in Figure 40 which also form the range of *hasElements* for this class. From this definition, it follows that any class which has a subject, verb, optional evidence, and one of the mentioned propositions will become the sub-class of *Belief* class.

Figure 40: Belief Class

iv. Class *Decision*: It refers to a choice that one makes after thinking about several possibilities. Its constituents are *subject, verb*, and an *Action* as shown in Figure 41. This class shall have a specific *individual* called *decides to* as the instance of the *verb* class. This specific individual shows a *subject* making a decision about an action. From this definition, it follows that any class which has a subject, verb *(decides to)* and an Action becomes the sub-class of the *Decision* class. The way the definition is set up, the reasoner classifies *Decision* and *Intent* as equivalent classes as shown in Figure 42. The only difference between them is the individual for class *verb*, which is *intends to* in the case of the *Intent* class.

Figure 41: Decision Class

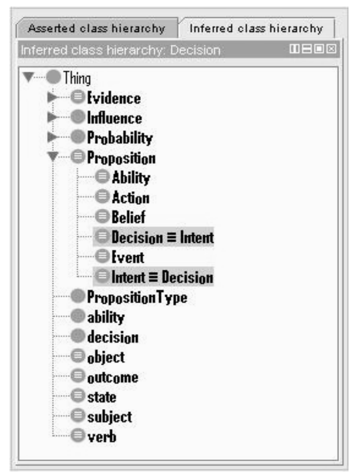


Figure 42: Equivalent Classes

v. Class *Event*: It refers to something that happens and its constituents are *subject*, *verb* and *state* as shown in Figure 43. These are the range of *hasElements* for this class. It follows that any class which has a subject, object and a state will become the sub-class of the *Event* class.

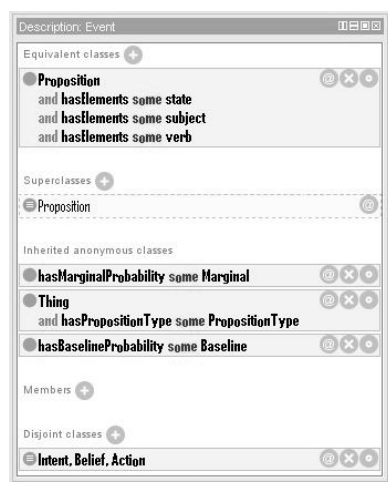


Figure 43: Event Class

vi. Class *Intent*: This class is similar to the *Decision* class. It refers to when one wants and plans to do something. Its constituents are also *subject*, *verb* (*intends to)*, and *an Action* which are the range of *hasElements* for this class as shown in Figure 44. It follows that any class having a subject, verb and an Action becomes the sub-class of the *Intent* class.

Figure 44: Intent Class

vii. Class *Proposition Type*: This class is associated with every *Proposition* and was created to give the concept of being affirmative or negative, as a proposition can make either an affirmative or a negative statement. A new object property *hasPropositionType* was also created whose domain is *Proposition.* This class contains two member instances (individuals) *affirmative* and *negative*. As an instance of a proposition is created, its type can be defined easily using this object property and its range is either *affirmative* or *negative*. Figure 45 shows an action proposition *Military withdraw troops* that includes a proposition type *Affirmative.*

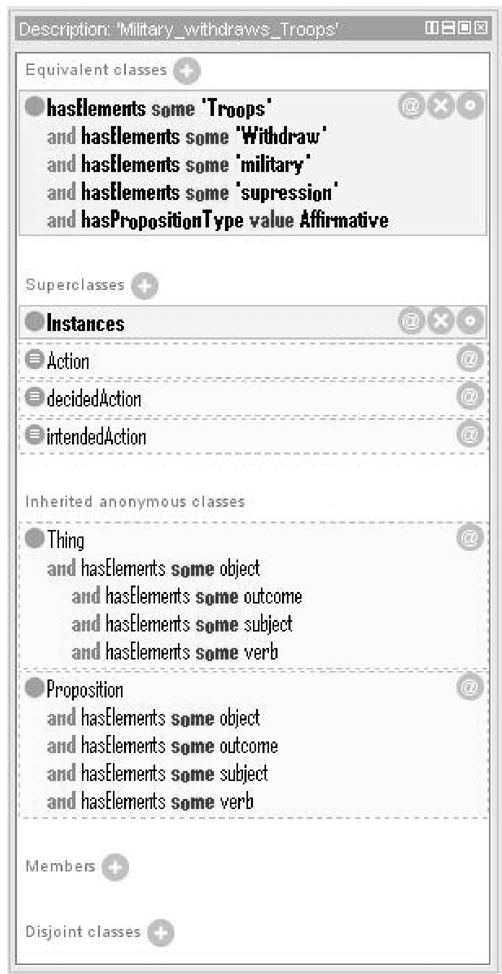


Figure 45: Proposition Type

viii. **Class *quality*:** This class represents one of the constituents of the *Ability* proposition class. It has an object property *hasQualityValue* where the domain is the quality class and the range has the member instances (individuals) of this class. In order to use its instances, the ontology designer has to create a library of member instances such as "can complete", "can perform", "can do", etc. Any newly created class which has an object property *hasQualityValue* with a range of any one of these member instances becomes a sub-class of *quality.*

ix. **Class *Evidence*:** This class represents one of the constituents of *Belief*. This class has an object property *hasEvidence* whose domain is class *Evidence* and range has the member instances of this class.

x. **Classes *subject, object, verb, outcome*** and ***state*:** These classes are the most used one throughout the ontology. The class *subject* has object property *hasSubjectValue,* class object has *hasObjectValue*, class verb has *hasVerbValue*, class outcome has *hasOutcomeValue* and state has *hasStateValue* object property.

For instance, in the action proposition class *Military withdraw troops* the designer would create individuals named *Military*, *Withdraws* and *Troops*. Each of these individuals, are then assigned their types (i.e., classes of which these individuals are instances). For example, individuals, *Military* and *Rebels* are the instances of classes *subject* or *object* as shown in the right frame of Figure 46, and *Withdraw* is the instance of class *verb* as shown in the right frame of Figure 47.

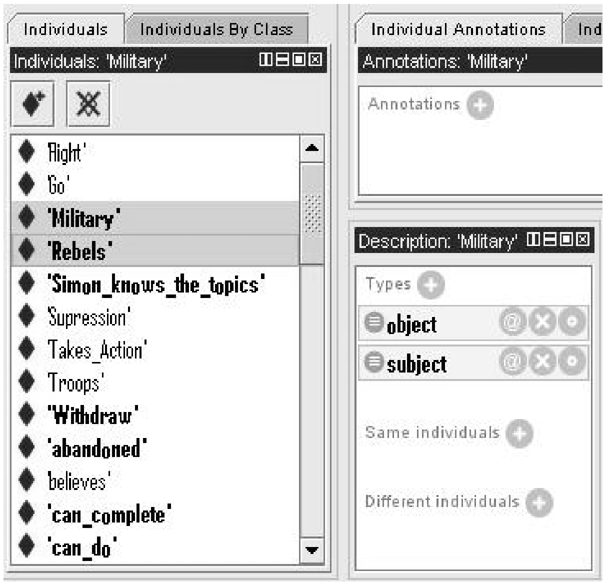


Figure 46: Subject and Object Individuals

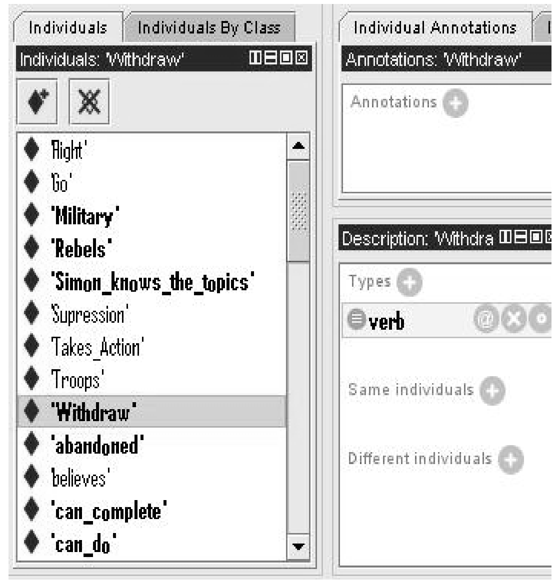


Figure 47: Verb Individual

6.8 Social Network Ontologies

Since the Social network formalism is not as rich as that of Influence nets, pseudo and refactored ontology construction for social networks was easier.

a. Social Network Pseudo and Refactored Ontologies

The pseudo ontology for social networks contains the same classes as the corresponding meta-model elements; a few ones were omitted for the sake of keeping this ontology simple. There isn't much difference between the pseudo and refactored ontologies for Social networks (Figures 48 and 49).

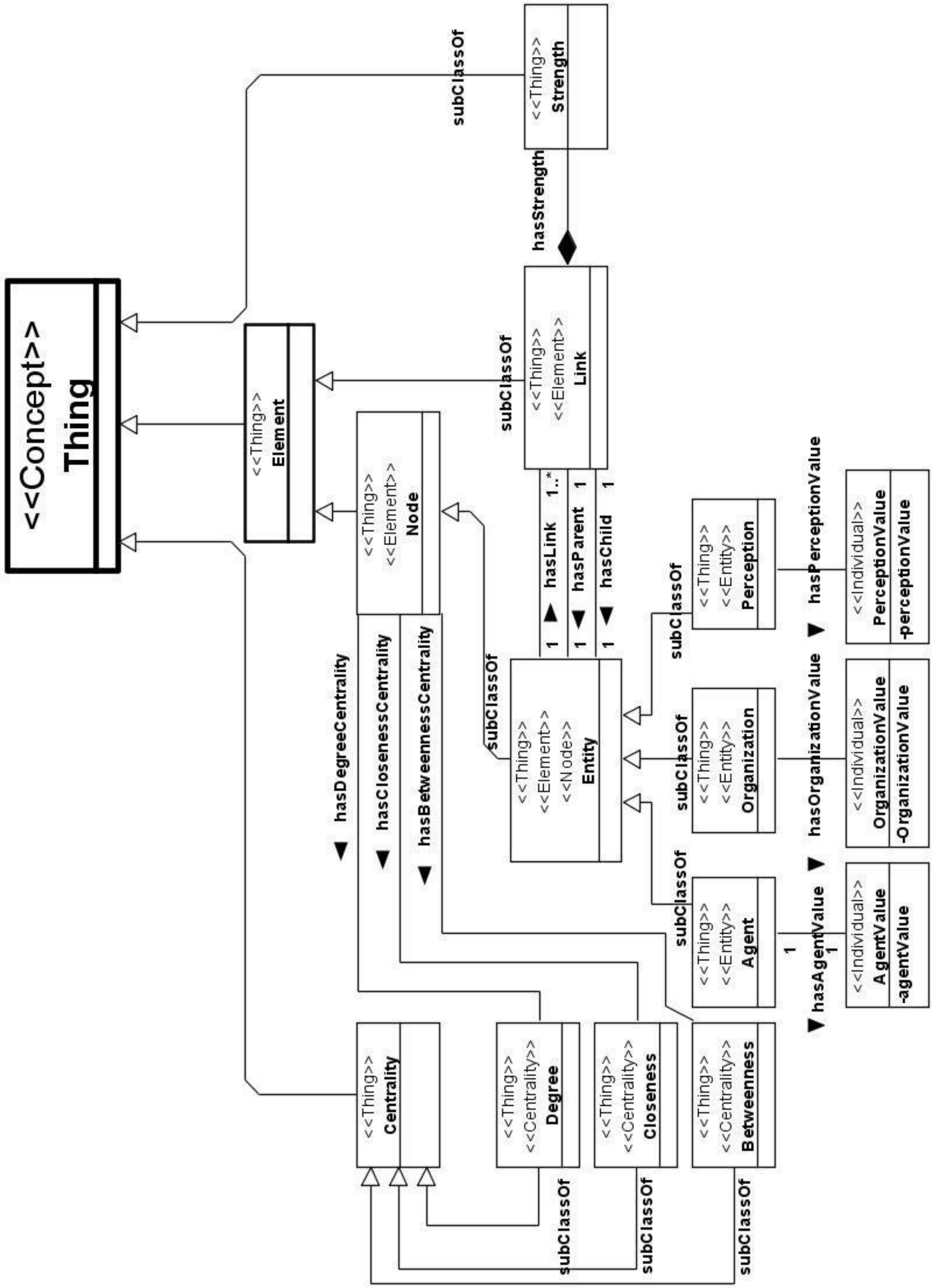


Figure 48: Social Network Pseudo Ontology

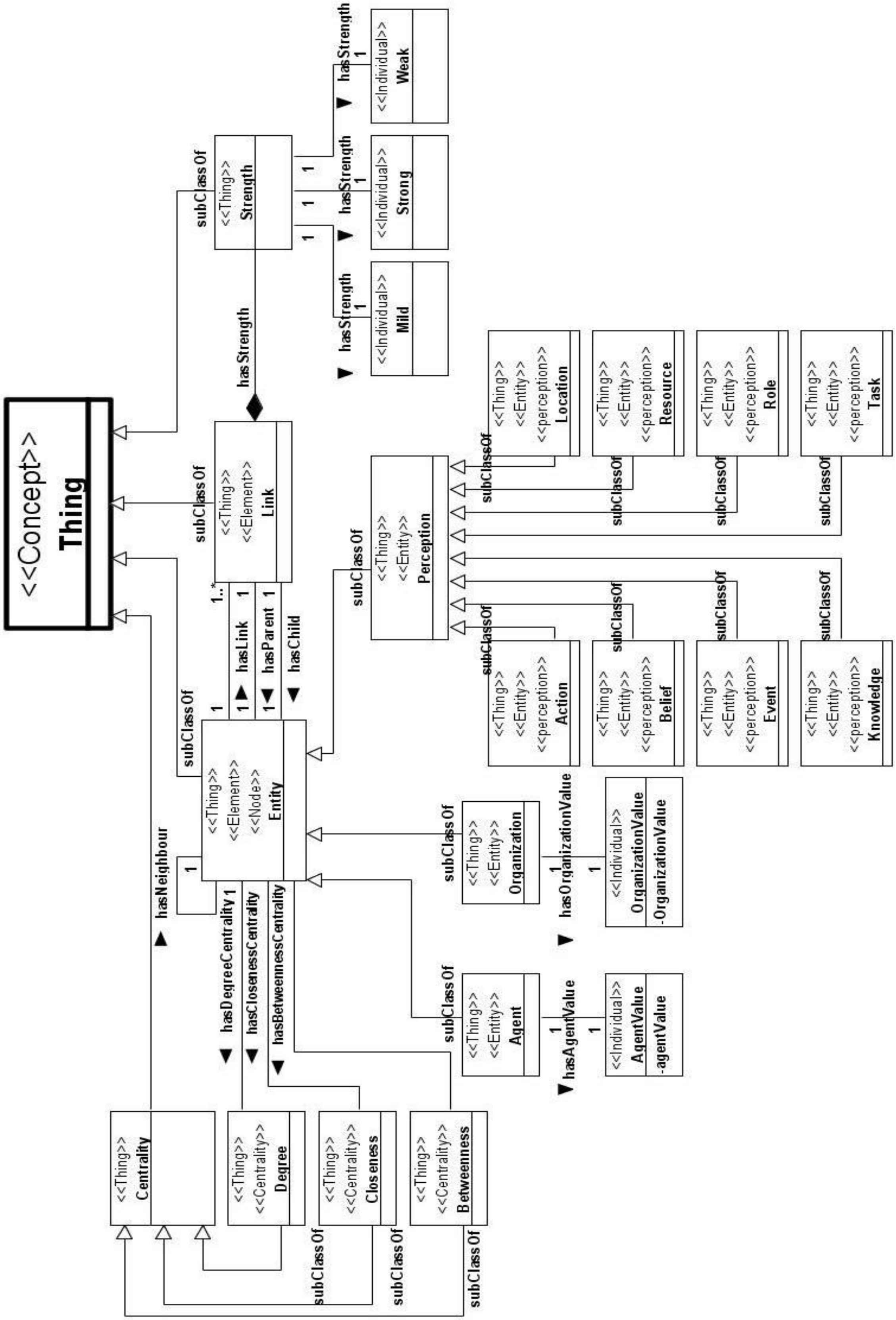


Figure 49: Social Network Refactored Ontology (additional concepts only)

Table 4 shows a comparison of the pseudo ontology classes and properties with the meta-model elements and associations. The grey shaded cells represent the common entities.

Table 4: Pseudo Ontology and Social Network Meta-Model Elements

Ontology Domain Class	Ontology Object Property	Ontology Range Class	Meta-Model Element	Association	Meta-Model Element
Element	SuperClassOf	Node, Link	**Element**	Types	Node, Link
Node	SuperClassOf	Entity	**Node**	Represents	Entity
	hasDegreeCentrality hasBetweennessCentrality hasClosenessCentrality	Betweenness, Closeness, Degree, Link,		is composed of	Centrality
					Name
Link	Has	Strength	**Interdependency**	Has	Strength
Entity	SuperClassOf	Agent, Organization, Perception	**Entity**	Types	Agent, Organization, Perception Name
	hasLink	Link			
Agent	hasAgentValue	Agent	**Agent**	-	-
Organization	hasOrganizationValue	Organization	**Organization**	-	-
Perception	hasPerceptionValue	Perception	**Perception**	-	-
Strength	hasStrength	Mild [Individual], Strong[Individual], Weak[Individual]	**Strength**	Defined in	Matrix
Centrality	SuperClassOf	Betweenness, Closeness, Degree	**Centrality**	Types	Betweenness, Closeness, Degree
Betweenness	betweenessValue	Floating point	**Betweenness**	-	-
Closeness	closenessValue	Floating point	**Closeness**	-	-
Degree	degreeValue	Floating point	**Degree**	-	-

Following are the classes and properties of the Social network refactored ontology.

a. ***Entity*****:** It is a sub-class of *Thing* and corresponds to a meta-model element with the same name. This class has the following object properties:

i. ***hasBetweenessCentrality*:** Every entity has a value of betweenness centrality captured by the class *Betweenness* which forms the range of this object property; its domain is *Entity*.

ii. ***hasClosenessCentrality*:** Every entity has a value for closeness centrality as well captured by the class *Closeness* which forms the range of this object property; its domain is *Entity*.

iii. ***hasDegreeCentrality*:** Every entity has a value for degree centrality also captured by the class *Degree* which forms the range of this object property; its domain is *Entity*.

This Entity class has the following three sub-classes. The definitions of each of these concepts (classes) are given in Section 7.6.

- ***Agent*:** This class represents the concept of an entity being a human being and has an object property *hasAgentValue* with domain *Agent* and range as the instances (individuals) of this class.
- ***Organization*:** This class represents the concept of a group of people in the form of an organization. It has an object property *hasOrganizationValue* with domain *Organization* and range as the instances (individuals) of this class.
- ***Perception*:** This class represents those concepts whose description is formed by our perceptions about them as described by the following sub-classes. The following sub-classes were added into the pseudo ontology in order to complete the refactored

ontology. These concepts were imported from the concept maps constructed for the social network.

- *Action*: This class represents an activity performed by an agent and has an object property *hasActionValue*.
- *Belief:* This class represents a belief held by an agent and has an object property *hasBeliefValue*.
- *Event:* This class represents the occurrence of an event and has an object property called *hasEventValue*.
- *Knowledge:* This class represents knowledge possessed by an agent. It has an object property called *hasKnowledgeValue*.
- *Location, Resource and Role:* These classes represent the social network specific concepts having object properties *hasLocationValue*, *hasResourceValue* and *hasRoleValue*.
- *Task:* Task is considered as an un-executed action and has an object property *hasTaskValue*.

b. ***Link*:** This class represents the link between two social network entities and is the same as its meta-model equivalent element. Every link has an optional child and a parent and has some strength of association between both and these are captured by the object properties *hasChild*, *hasParent*, and *hasStrength.*

c. ***Strength*:** It represents the strength of the link class and has three member instances *Mild*¸ *Strong* and *Weak.*

d. ***Centrality (Betweenness, Closeness, and Degree)*****:** These classes capture the three types of measures of centrality exhibited by every social network entity. Their data properties are *betweennessValue*, *closenessValue*, and *degreeValue* having domain *Centrality* and range as floating point values.

6.9 Construction of the Enriched Ontology

The focus of the effort so far has been to construct an enriched ontology filled with the concepts (classes) and relationships (properties) of Influence net and Social network modeling techniques within and across them. The motive behind the construction of an enriched ontology was to identify mappings between the semantically equivalent concepts of both modeling techniques so that the exchange of information or analysis results between models constructed using both techniques can take place. The refactored ontologies serve as the basis for the enriched ontology as shown in Figure 50.

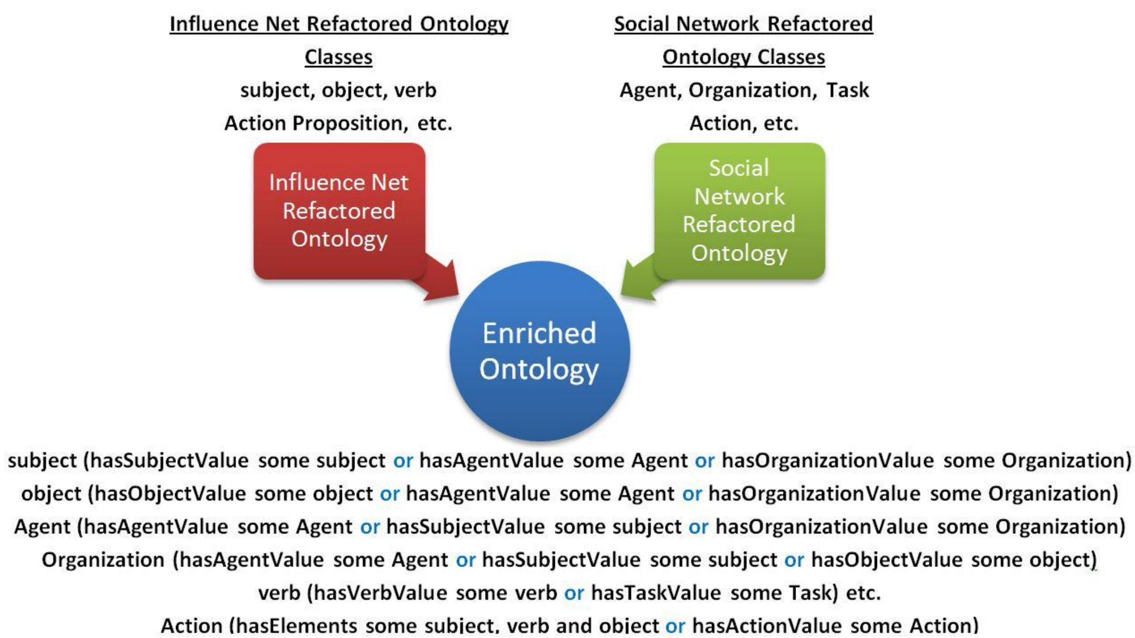


Figure 50: Enriched Ontology Classes

The diagram in Figure 50 illustrates the fusing of concepts from both types of refactored ontologies inside the enriched ontology. This is achieved by defining additional object properties in related classes and is done manually. For instance, the *Agent* and *Organization* classes from the Social network refactored ontology can be mapped to the *subject* and *object* classes of the Influence net refactored ontology by adding *hasSubjectValue* and *hasObjectValue* object properties to the existing object property of *Agent* and *Organization* classes as shown in Figure 51. Once, the reasoner is executed, it figures out and classifies *Agent*, *Subject*, *Object*, and *Organization* as equivalent classes by inferring that the new object properties added to the *Agent* class map *Subject*, *Object*, and *Organization* to itself as shown in Figure 52. The inferred class hierarchy shows these classes with the equivalence sign (Figure 53). Similarly, *Belief* class in Social network refactored ontology can be mapped to Influence net's *Belief* class. The *Event* class of Social network refactored ontology maps to the *state* class which is the constituent of the *Event* class in Influence net refactored ontology also Social network's *Knowledge* class maps to *quality* class which is the constituent of *Ability* class in Influence net refactored ontology. The class *Task* of Social network can be mapped to the class *verb* of Influence net refactored ontology. Table 5 summarizes these mapped concepts between both the refactored ontologies. The ultimate result of this mapping is an *enriched ontology* which is the knowledge container of both Influence net and Social network modeling techniques.

Figure 51: Subject, Object Classes mapped to Agent Class

Figure 52: Reasoner Inferred Equivalences

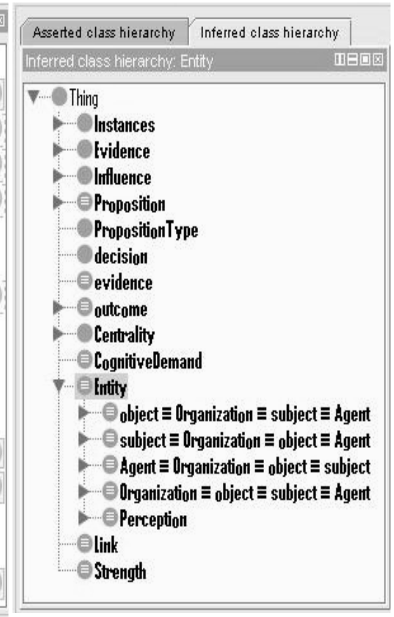


Figure 53: Subject, Object, Organization and Agent as Equivalent Classes

This whole process of ontology construction is shown in Figure 54. The objective of this approach is the identification of the mapping between the modeling techniques based on the semantic equivalences identified after constructing the enriched ontology.

This enriched ontology acts as the (template) knowledge container for a specific domain and it can be re-used, since this knowledge is in machine-readable form. For example, by querying an *agent* or a *subject* will pull all relevant instantiated subjects and agents associated with both types of models.

Table 5: Enriched Ontology

Enriched Ontology (Influence Net & Social Network Refactored Ontology Mapped Concepts)					
Influence Net Refactored Ontology Elements			**Social Network Refactored Ontology Elements**		
Domain Class	**Object Property**	**Range Class**	**Domain Class**	**Object Property**	**Range Class**
subject	hasSubjectValue	subject	**Agent**	hasSubjectValue	Subject
	hasAgentValue	Agent		hasAgentValue	Agent
	hasOrganizationValue	Organization		hasObjectValue	Object
object	hasObjectValue	Object	**Organization**	hasObjectValue	Object
	hasAgentValue	Agent		hasAgentValue	Agent
	hasOrganizationValue	Organization		hasSubjectValue	Subject
verb	hasVerbValue	verb	**Task**	hasVerbValue	verb
Intent/ Decision	hasTaskValue	Task		hasTaskValue	Task
	hasElements some Action and hasElements some subject and hasElements some verb	Action, subject, verb		hasElements some Action and hasElements some subject and hasElements some verb	Action, subject, verb
Action	hasElements some subject and hasElements some verb and hasElements some object	Action, subject, verb	**Action**	hasElements some subject and hasElements some verb and hasElements some object	Action, subject, verb
	hasActionValue	Action		hasActionValue	Action
Belief	hasElements some subject and hasElements some verb and hasElements some (Ability or Decision or Action or Event) and hasEvidence some Evidence	subject, object, verb, Ability or Decision or Action or Event, Evidence	**Belief**	hasElements some subject and hasElements some verb and hasElements some (Ability or Decision or Action or Event) and hasEvidence some Evidence	subject, object, verb, Ability or Decision or Action or Event, Evidence
	hasBeliefValue some Belief	Belief		hasBeliefValue some Belief	Belief
state	hasStateValue	State	**Event**	hasStateValue	State
	hasEventValue	Event		hasEventValue	Event
quality	hasQualityValue	quality	**Knowledge**	hasQualityValue	quality
	hasKnowledgeValue	Knowledge		hasKnowledgeValue	Knowledge

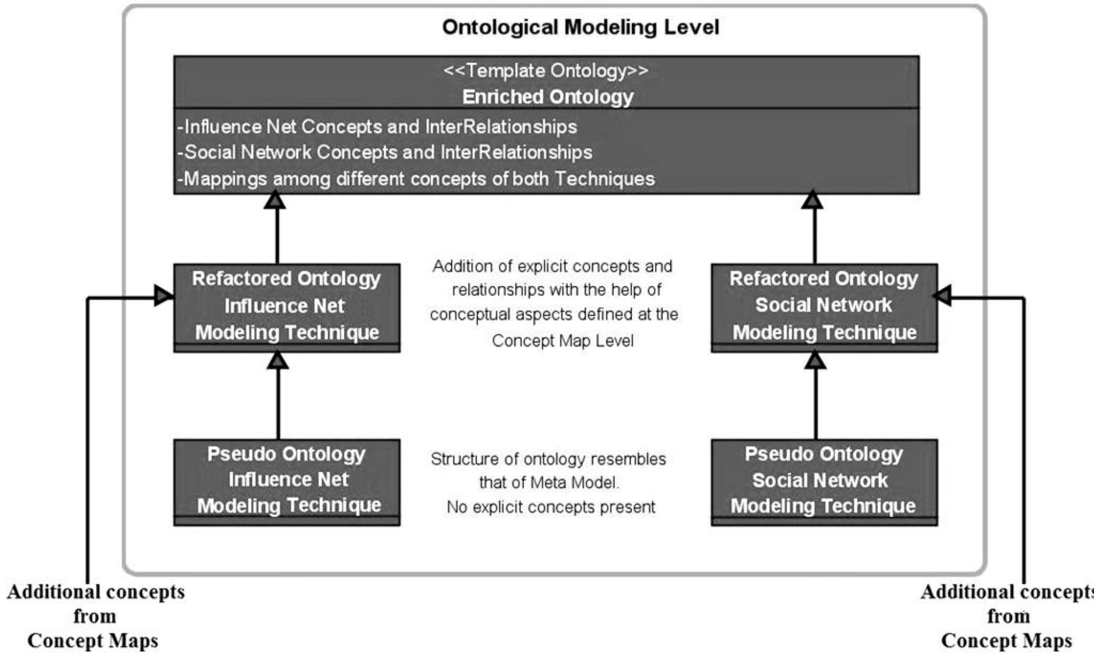


Figure 54: Ontological Modeling Level

CHAPTER 7: WORKFLOW

7.1. Overview

The previous chapters provided information about concept maps and concept-mapping, meta-models, meta-modeling and multi-modeling, ontologies and ontology construction. In this chapter, the pieces are put together to effect the extraction of semantic equivalences between the two modeling techniques, Influence nets and Social networks, in the form of a complete workflow.

The technique suggested by Kappel et al. [1] has been used but for a different purpose and for a different set of modeling languages. The first difference is the additional level of modeling introduced in this thesis: the first level of concept mapping called the *Conceptual Modeling Level*. The second difference is the manual determination of semantic equivalences between both types of refactored ontologies; Kappel et al. use the COMA++ tool. The third difference is the use of concept maps and the knowledge of the ontology designer about both techniques in constructing the refactored ontology rather than using refactoring patterns.

Figure 55 shows the complete workflow of the technique this thesis proposes.

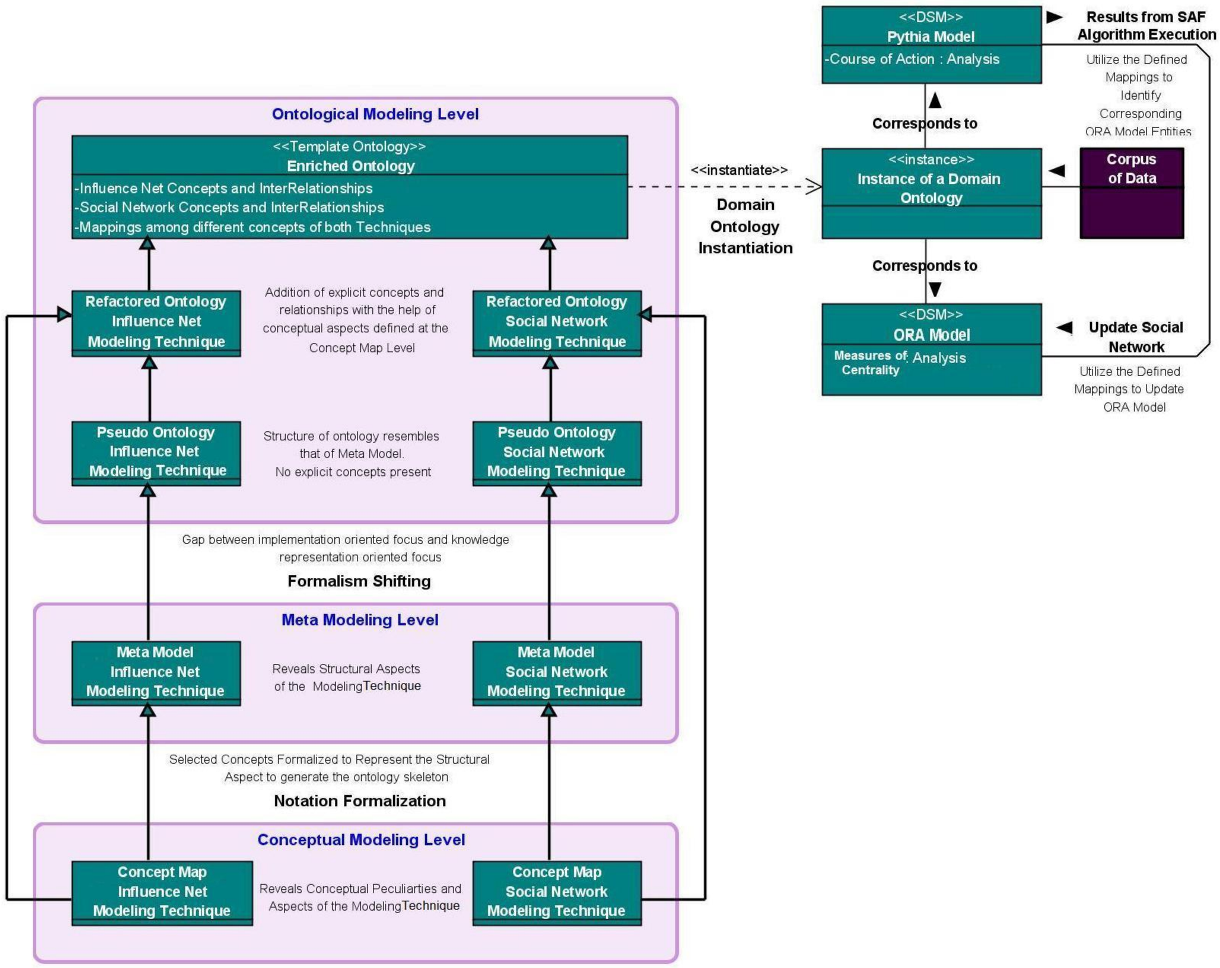


Figure 55: The Workflow

The foundation is laid at the very abstract *conceptual modeling level* with the help of concept maps. Selected concepts from this level are used in the development of the *meta-models* at the *meta-modeling level* to reveal the structural aspects, formalize the notation, and create a skeleton for the *ontological modeling* level. Before entering into the *ontological modeling level*, a formalism shift is required to reduce the gap between the implementation specific focus of *meta-models* and the knowledge representation oriented focus of *ontologies*. The *meta-models* act as the skeleton for constructing the first ontologies of both modeling techniques.

The first ontology that is constructed is called *pseudo ontology* and resembles its meta-model equivalent. The construction of this ontology utilizes the meta-model in such a way that each meta-model element becomes an ontology class and the association between the meta-model elements becomes either an object or a data property in the ontology as expressed in Tables 3 and 4 where grey shaded cells represent the common entities in both the meta-model and the pseudo ontology. Explicit concepts and relationships of the modeling technique are added into the pseudo ontology to construct a *refactored ontology*. Mapping of concepts between the refactored ontologies of both techniques is done to construct an *enriched ontology*. This enriched ontology contains the individual and mapped concepts and relationships of both modeling techniques. It can be considered as a template ontology which contains the *intra* and *inter-modeling technique* concepts and relationships. A domain ontology can be instantiated for a specific domain which will serve as the knowledge container for that domain.

The right portion of the figure shows the instantiation process of a domain specific ontology driven by a corpus of data. This domain specific ontology contains information corresponding to both types of models. The arrow coming out from the *Pythia Model* box shows that the SAF Algorithm is executed and its results are used with the help of the defined mappings to identify corresponding entities in the *ORA model* and update them.

The process described in this chapter is repeatable for any sets of modeling techniques. For instance, to extract semantic knowledge about CPN (Colored Petri Net), similar concept maps (for the defined focus questions) should be developed, followed by its meta-model, and then pseudo and refactored ontologies. Since ontology construction is an intense brainstorming activity, by the time the refactored ontology is completed, enough insight into the modeling technique should have been achieved that the ontology designer would easily be able to map CPN concepts to the related Social network or Influence net concepts (if there exist any). These newly mapped concepts can then be incorporated into the enriched ontology and an updated enriched ontology can be constructed which would serve as the knowledge container for Influence net, Social network and CPN modeling techniques altogether.

The rest of this chapter provides additional detail and describes how a domain specific ontology can be instantiated. The next chapter illustrates how the mappings defined in the enriched ontology can be used to update a Social network model using the analysis results from a corresponding Influence net model.

7.2. Conceptual Modeling Level

The *concept mapping* process is conducted in the *Conceptual Modeling Level*. The basic motive at this level is to extract syntactic or semantic concepts and reveal the conceptual characteristics of both modeling techniques at a very abstract level. Since a concept map is an informal and abstract representation of concepts, a formalization of notation is needed; this is done in the meta-modeling level.

7.3. Meta-Modeling Level

After the conceptual modeling level, only selected concepts from it are formalized to represent the structural aspects of the modeling techniques in the form of a meta-model. This level is called the *Meta-Modeling Level* and the objective at this level is to generate a basic ontology skeleton. Meta-models do not contain any detailed concepts and relationships of the domains they model, but their structure can be used as the basis for the first ontology to be constructed at the next level.

7.4. Ontological Modeling Level

Kappel et al. [1] refer to the process of formalism shifting as reducing the gap between the implementation oriented focus of meta-models and the knowledge representation oriented focus of ontologies. This formalism shift is led by the meta-model developed in the previous level. There are three sub-levels of this modeling level that ultimately yield an ontology enriched with concepts and relationships from both modeling techniques.

a. Pseudo Ontologies

In the first sub-level, the pseudo ontology resembling that of its corresponding meta-model has no explicit concepts present except the ones related to the structure of the modeling technique similar to what is in the meta-model. The source of information for the pseudo ontology is the meta-model. . Figure 56 shows the asserted class hierarchy and object properties of Influence net pseudo ontology.

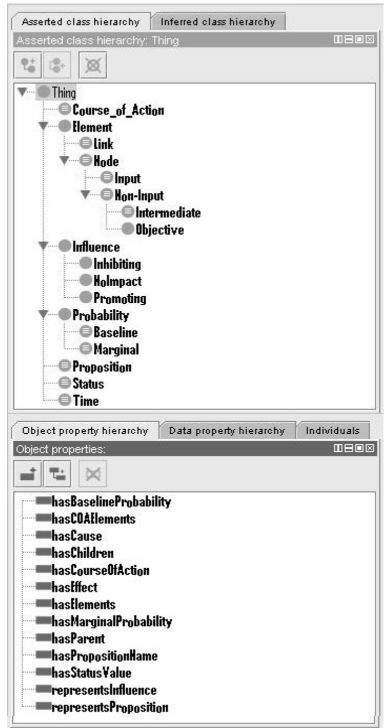


Figure 56: Protégé 4.0 Classes & Properties for the Influence Net Pseudo Ontology

Figure 57 shows the GraphViz diagram of the super-class/sub-class hierarchical view of this ontology [30].

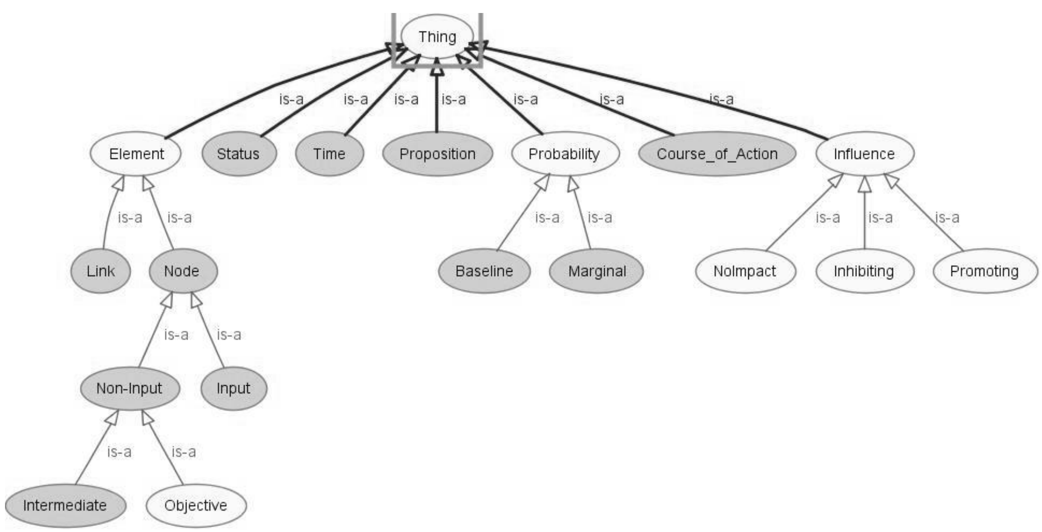


Figure 57: GraphViz diagram - Influence Net Pseudo Ontology

The asserted class hierarchy of the Social network pseudo ontology with object properties shown in Figure 58 and its GraphViz diagram is shown in Figure 59.

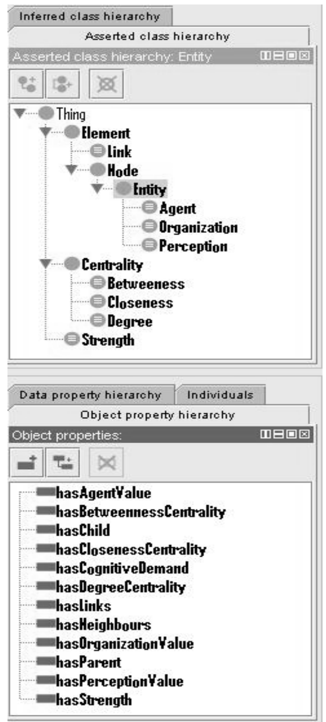


Figure 58: Protégé 4.0 Classes & Properties for the Social Network Pseudo Ontology

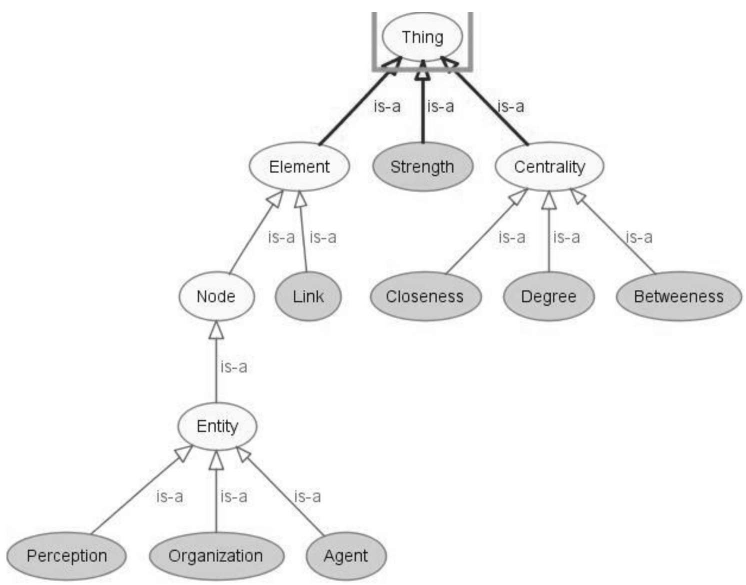


Figure 59: GraphViz diagram - Social Network Pseudo Ontology

b. Refactored Ontologies

Our approach adds additional concepts into the refactored ontology from the concept mapping phase and the domain knowledge of the ontology designer. Table 6 shows the concepts imported into the Influence net refactored ontology from the concept maps. Table 7 shows the explicit concepts added into the refactored ontology. Figures 60 and 61 show the asserted and inferred GraphViz diagrams for this refactored ontology.

Table 6: Influence Net Refactored Ontology Elements (Concept Map Imports)

Ontology Domain Class	Ontology Object Property	Ontology Range Class	Concept Map - Propositions		
			Concept 1	Relationship	Concept 2
Proposition	SuperClassOf	Action	Proposition	can define	Action
		Ability			Ability
		Belief			Belief
		Decision			Decision
		Event			Event
		Intent			Intent

Table 7: Explicit Influence Net Concepts in Refactored Ontology

Category	Ontology Domain Class	Ontology Object Property	Ontology Range Class
Explicit Concepts	subject	hasSubjectValue	subject
	verb	hasVerbValue	verb
	object	hasObjectValue	object
	outcome	hasOutcomeValue	outcome
	Evidence	HasEvidence	Evidence
	quality	hasQualityValue	quality
	state	hasStateValue	state
	PropositionType	hasPropositionType	Affirmative Individual, Negative Individual

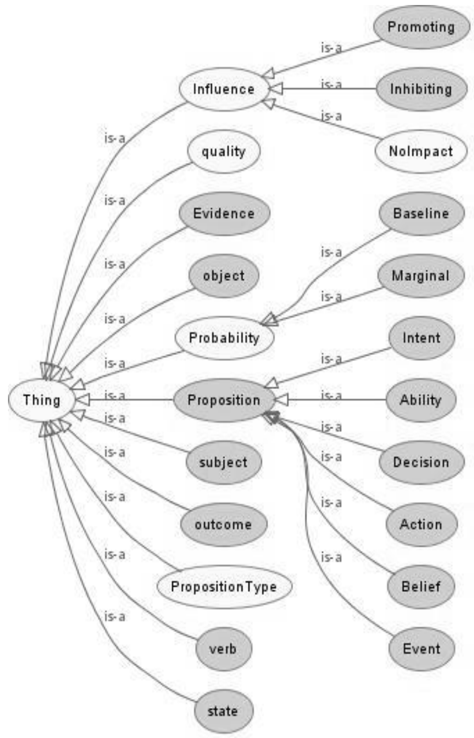


Figure 60: GraphViz Diagram - Influence Net Asserted Refactored Ontology

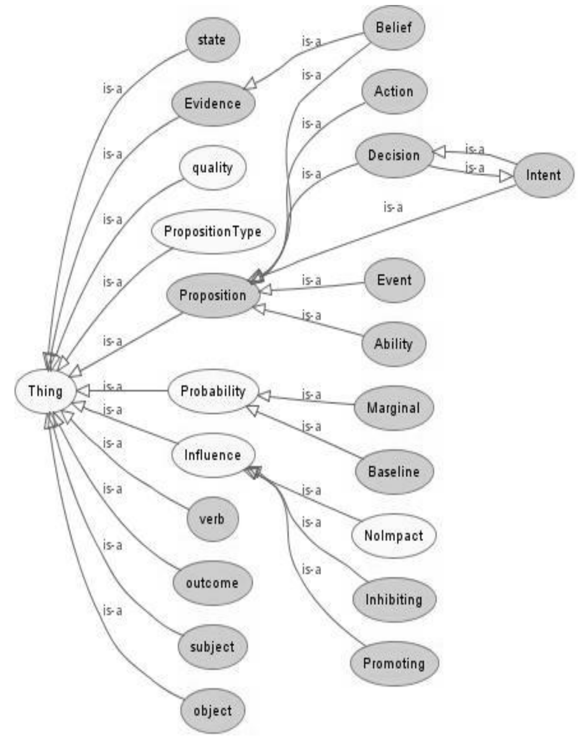


Figure 61: GraphViz Diagram - Influence Net Inferred Refactored Ontology

Table 8 shows the concepts imported from the concept maps into the Social network refactored ontology. Figure 62 shows the GraphViz diagram of the refactored ontology for Social networks.

Table 8: Social Network Refactored Ontology Elements (Concept Map Imports)

Category	Ontology Domain Class	Ontology Object Property	Ontology Range Class	Concept Map - Propositions		
				Concept 1	Relationship	Concept 2
Concepts from Concept Maps	Perception	hasPerceptionValue	Action	Perception	Types	Action
			Belief			Belief
			Event			Event
			Knowledge			Knowledge
			Location			Location
			Resource			Resource
			Role			Role
			Task			Task

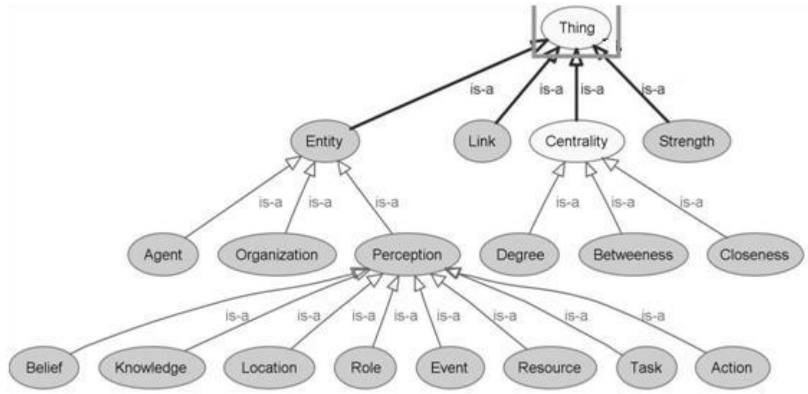


Figure 62: GraphViz Diagram - Asserted/Inferred Refactored Social Network Ontology

c. Enriched Ontology

Figure 63 shows a high level view of the mappings between the relevant concepts of both modeling techniques. The definitions of all elements shown in the figure are given in Section 7.6.

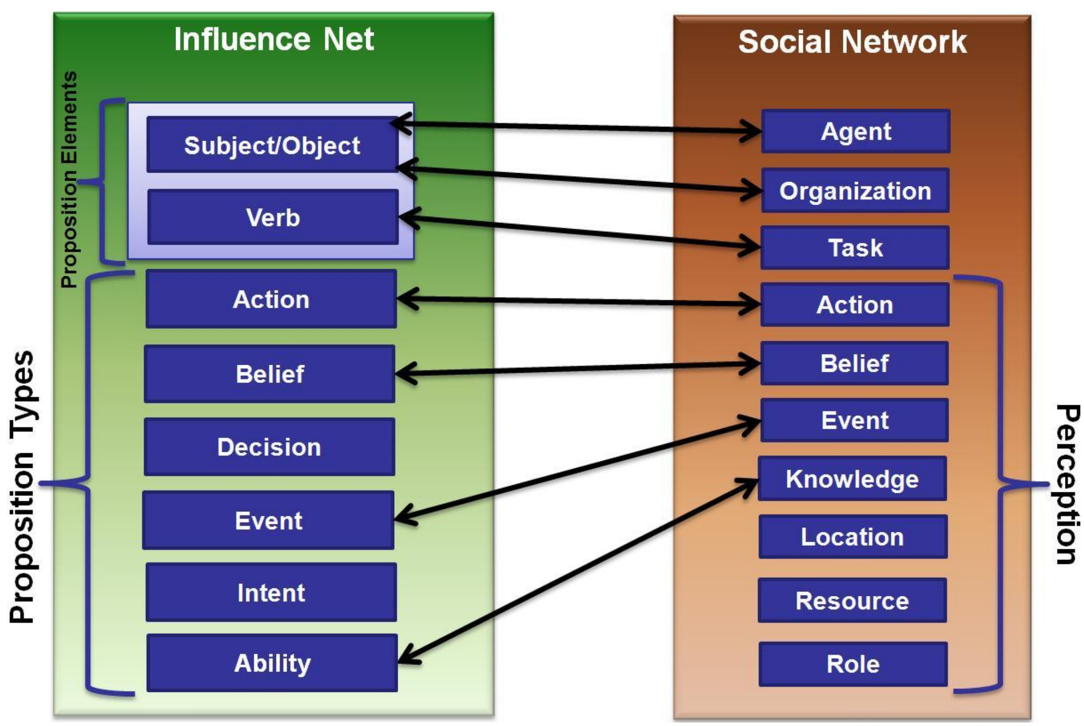


Figure 63: Determined Mappings in the Enriched Ontology

The asserted and inferred versions of the enriched ontology are shown in Figures 64 and 65.

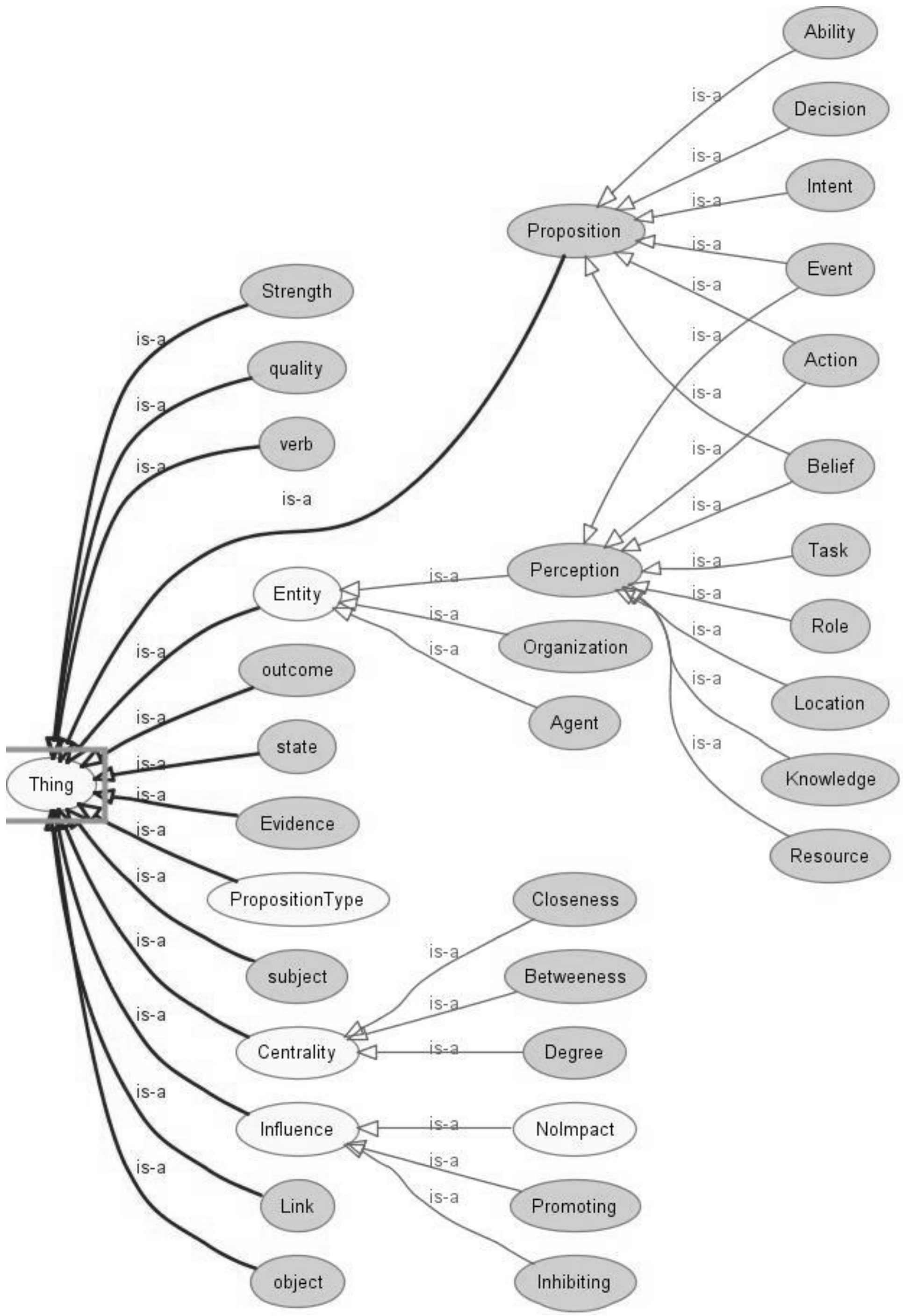


Figure 64: GraphViz Diagram - Asserted Enriched Ontology

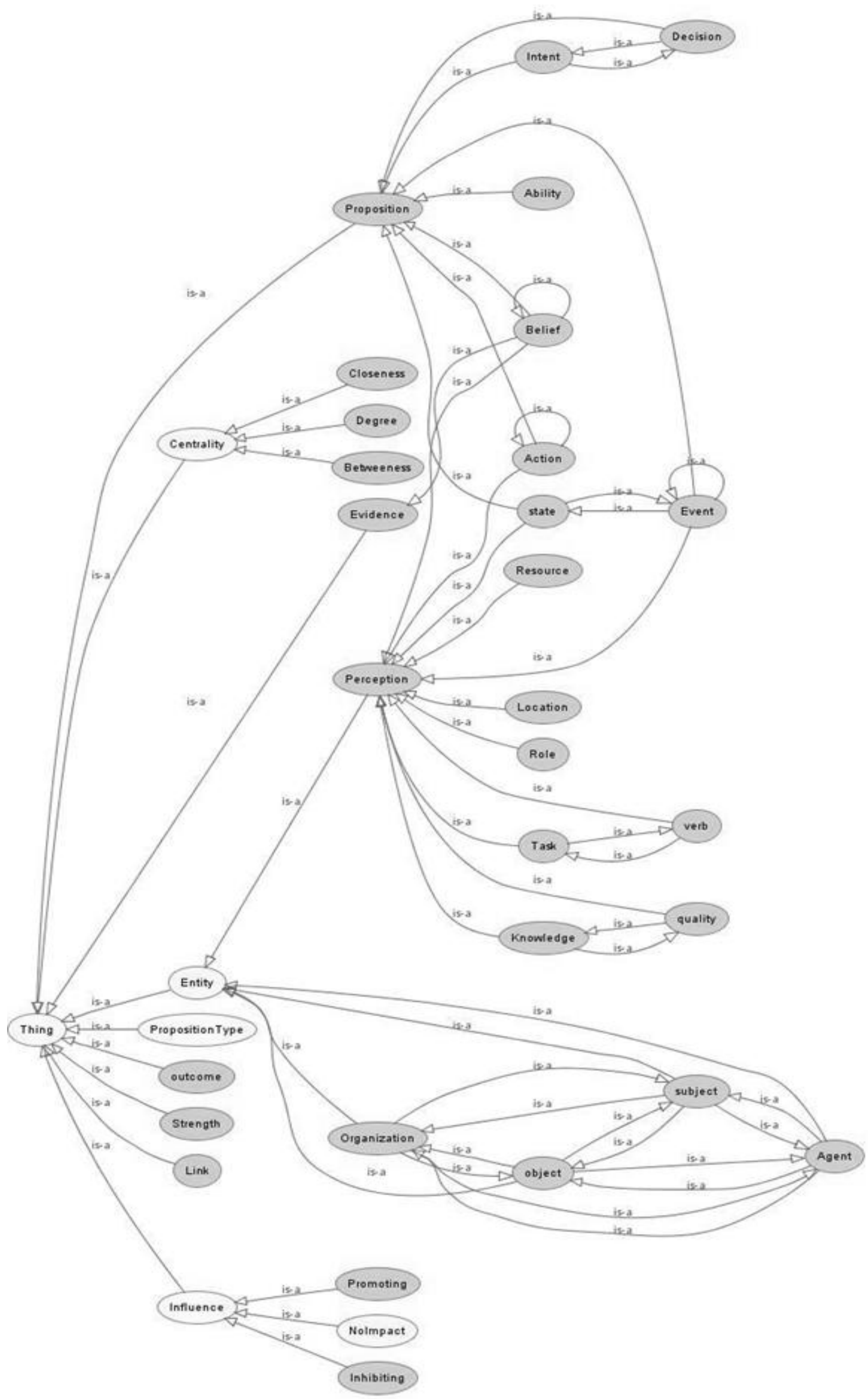


Figure 65: GraphViz Diagram - Inferred Enriched Ontology

As you can see in the inferred version, the reasoner computes the relations based on the defined properties and maps the related classes together. For example, subject, object, agent, and organization are mapped as equivalent classes.

7.5. Domain Ontology Instantiation

The enriched ontology acts as template ontology for constructing any domain specific ontologies. In the final steps of the process, as shown in Figure 55, the domain ontology can be instantiated that corresponds for a specific domain. A corpus of data drives this instantiation process.

7.6. Definitions

This section contains the definitions of all propositions, their constituents, and social network elements (see Section 6.8b):

a. **Proposition[11]:** A proposition is a statement that affirms or denies something and is either true or false.

Following are some typical types of propositions that are used in Influence net construction. This classification is not final and it can have many other types of propositions as well. A proposition will typically have:

- Subject: The person or entity involved in performing the activity given in the proposition is referred to as *Subject*.
- Verb: It refers to the actual *Verb* used in the proposition describing that activity.

[11] Webster's Online Dictionary

- Object: An entity (person or thing) on which the activity is performed is referred to as *Object*.

The first six propositions have been defined as follows:

i. Action[12]: An action refers to doing something towards a goal or the process of doing something in order to achieve a purpose. An action will have:

  - Action performer (Subject)
  - Action (Verb)
  - Action receiver (Object)
  - Purpose of Action (Outcome) (optional)

e.g., $\underbrace{\text{Military}}_{\textbf{Subject}}\ \underbrace{\text{fights}}_{\textbf{Verb}}\ \underbrace{\text{the rebels}}_{\textbf{Object of Action}}.$

ii. Belief[13]: Belief is a degree of conviction of the truth of something especially based on a consideration or examination of the evidence. A belief will have:

  - Believing Subject (Subject)
  - Verb (verb "believe")
  - Actual belief about an action, decision, or ability (Action, Decision, or Ability)
  - Evidence (optional)

[12] MSN-Encarta Online Dictionary
[13] Dictionary.COM

e.g.,

$$\underbrace{\text{Simon}}_{\textbf{Subject}}\ \underbrace{\text{believes}}_{\textbf{Verb}}\ \overbrace{\underbrace{\text{he}}_{\textbf{Subject}}\ \underbrace{\text{can complete}}_{\textbf{Quality}}\ \underbrace{\text{his work}}_{\textbf{Object of Action}}}^{\textbf{Ability Proposition}}$$

iii. **Ability**[14]**:** The quality of being able to perform; a quality that permits or facilitates achievement or accomplishment.

Ability will have:

- The subject being able of doing something (Subject)
- Quality of being able to do an action. It refers to *quality* class explained in 6.8b (viii).
- Object of action (Object)

e.g., $\underbrace{\text{Simon}}_{\textbf{Subject}}\ \underbrace{\text{can complete}}_{\textbf{Quality}}\ \underbrace{\text{his work}}_{\textbf{Object of Action}}$

iv. **Event**[15]**:** An event refers to anything that happens, especially something important or unusual. An event will have:

- Event subject (Subject)
- Verb
- Actual occurrence of event (State)

[14] MSN-Encarta Online Dictionary
[15] Webster's Online Dictionary

e.g., $\underbrace{\text{Flight}}_{\textbf{Subject}}\ \underbrace{\text{will}}_{\textbf{Verb}}\ \text{be}\ \underbrace{\text{delayed}}_{\textbf{State}}$

v. **Decision**[16]**:** It refers to a choice that you make about something after thinking about several possibilities. A decision will have:

- The subject making decision (Subject)
- Verb (verb "decide")
- Decision about an action (Action)

e.g.,

$$\underbrace{\text{Military}}_{\textbf{Subject}}\ \underbrace{\text{decides}}_{\textbf{Verb}}\ \overbrace{\text{to}\ \underbrace{\text{withdraw}}_{\textbf{Verb}}\ \underbrace{\text{troops}}_{\textbf{Object}}}^{\textbf{Action Proposition}}$$

The above action proposition has an implicit action subject which is *military* itself.

vi. **Intent:** Intent refers to when you want and plan to do something. Intent will have:

- Subject intending an action (Subject)
- Verb (verb "intend")
- Intended action (Action)

e.g.,

$$\underbrace{\text{Alfred}}_{\textbf{Subject}}\ \underbrace{\text{intends}}_{\textbf{Verb}}\ \text{to}\ \overbrace{\underbrace{\text{travel}}_{\textbf{Verb}}\ \underbrace{\text{the world}}_{\textbf{Object}}}^{\textbf{Action Proposition}}$$

[16] Webster's Online Dictionary

The above action proposition has an implicit action subject which is *Alfred* himself.

b. **Agent:** It represents an individual or an entity performing an action. Identifying agents refers to gathering all possible information about the WHO's involved in a domain, e.g., person, leadership etc.

c. **Organization:** It represents individuals in the form of an organized group, e.g. Team, Gang, UNO etc.

d. **Knowledge:** The possession of vital information by an agent is knowledge, e.g., knowing how to drive a vehicle, flying an airplane, etc.

e. **Location:** It represents a place of interest in a domain where an event occurs, e.g., any country or city.

f. **Task:** This thesis interprets a task as an un-executed action which is delegated to an agent for completion, e.g., delivery of something, making of a car, etc. Tasks are exclusively associated with agents; one can't have a *location* or *event* perform a task.

CHAPTER 8: CASE STUDY – IRAQI INVASION OF KUWAIT

8.1. Scenario (Corpus of Data)

The corpus of data used is a real world event describing the Iraqi invasion of the state of Kuwait. The Influence net model for this scenario already existed and is taken from Julie A. Rosen and Smith [5] who constructed it from open source reference material investigating the influences on Saddam Hussein's decision making after Iraq's invasion of Kuwait in 1990 [31]. Three more propositions (input nodes) were added into the model including *US leading an effort to liberate Kuwait*, *UN passing resolution for withdrawal of Iraqi forces from Kuwait*, and *Saddam's belief about conspiracy for Iraq's domestic and economic destabilization*. The Social network model was constructed from the same narrative. This text was provided to AutoMap [12], the text processing tool which allows one to develop the meta-matrix for a social network by identifying elements from the text such as agents, actions, beliefs, knowledge, tasks, etc. Once the meta-matrix is constructed using AutoMap, it is provided to ORA [32] which visualizes the Social network graphically and lets one perform analyses on the model such as centrality, density and other measures. We have considered only betweenness, closeness and degree centrality measures.

The narrative of the Iraqi invasion of Kuwait event used in this illustrative example is given as follows. It outlines the reasons for the invasion, the actual invasion, and how the international community reacted to make the aggressors withdraw from the state of Kuwait. Elements of Social network and Influence net in the text are given in bold font and are grouped in Tables 9 and 10, respectively. This brief example is used to demonstrate the technical approach; it is neither an accurate nor a complete description of the events that led to the invasion and it consequences.

*The **invasion** of **Kuwait** was a major conflict between the Republic of **Iraq** and the State of **Kuwait** which resulted in the seven-month long Iraqi **occupation** of **Kuwait** which subsequently led to direct **military intervention** by United States-led forces in the **Persian Gulf War**. **Kuwait** was a close **ally** of **Iraq** during the **Iraq-Iran war** and functioned as the country's major **port** once **Basra** was shut down by the fighting. However, after the war ended, the friendly relations between the two neighboring Arab countries turned sour due to several **economic and diplomatic** reasons which finally culminated in an **Iraq**i **invasion** of **Kuwait**. **Kuwait** had heavily **funded** the 8 year long **Iraq**i war against **Iran**. By the time the war ended, **Iraq** was not in a financial position to **repay the 40$ billion** which it had borrowed from **Kuwait** to finance the war. However, Kuwait's reluctance to pardon the debt created strains in the relationship between the two Arab countries. After the **Iran-Iraq** war, the **Iraq**i economy was struggling to recover as its civil and military debt was higher than its state debt. Most of its ports were destroyed, oil fields mined. Saddam's regime clearly realized that seizing **Kuwait** could be the remedy of its financial problems and the **regaining of regional authority**. And*

annexation of Kuwait could be helpful *for Saddam's political motives.* ***Saddam*** *had a reason to believe that there was an alleged* ***international conspiracy going on*** *against* ***Iraq*** *to weaken its political and economic stability in the region. Also* ***Iraq*** *had accused* ***Kuwait*** *of flooding the world market with oil and had demanded compensation for oil produced from a disputed oil field on the border of the two countries. Following all these reasons* ***Saddam*** *invaded* ***Kuwait*** *with full might and kept its* ***occupation*** *for seven months. Right after the* ***invasion****, the* ***Security Council*** *voted 15-0 to* ***declare Iraq's annexation of Kuwait null and void*** *and* ***demanded an immediate withdrawal of Iraqi forces*** *from* ***Kuwait****. Several courses of actions were devised to determine how* ***Saddam*** *would withdraw his troops from* ***Kuwait*** *peacefully.* ***Saddam*** *could have been pressured more by the* ***coalition enforcing the UN export import embargo*** *on* ***Iraq****. Since the regime invaded* ***Kuwait*** *to* ***recover itself from the economic turmoil*** *it got into after the* ***Iran-Iraq war, a withdrawal could have been politically very costly****.* ***Saddam*** *would have continued the* ***occupation*** *had* ***events been in his control,*** *but the retaliation came from a multinational military force lead by the* ***US*** *and* ***Free Kuwaiti forces*** *for the* ***liberation of Kuwait****.*

Table 9: Social Network Entities

Entity	Value	Entity	Value
Action	Occupation	Location	Basra
	Invasion	Organization	UNO, Security Council
	Military Intervention	Resource	None
	Funding	Role	Ally, Port
	Declaration of Kuwait’s Annexation as Null	Task	Regaining Regional Authority
	Demanding Immediate Withdrawal		Repay debt to Kuwait
	Enforcement		Recover from Economic Turmoil
	Liberation of Kuwait		
Agent	Iraq, Iran, Kuwait, Saddam, US, Coalition		
Belief	International conspiracy against Iraq exists		
	Withdrawal is politically costly		
	US has resolution for Kuwait		
	Annexation will be helpful for Iraq		
	Events are in Control of Saddam		
Event	Persian Gulf War		
	Iran Iraq War		
	Economic Diplomatic Collapse		

Table 10: Influence Net Propositions

Proposition Type	Value
Action	US leads an effort to liberate Kuwait
	Coalition enforces UN export and import embargo on Iraq
	UN passes resolution for withdrawal of Iraqi forces from Kuwait
Belief	Saddam believes he is in control of events
	Saddam believes there is a conspiracy for Iraq’s domestic and economic destabilization
	Saddam believes annexation of Kuwait will help him politically
	Saddam believes annexation of Kuwait will Iraq financially
	Saddam believes US has resolve to liberate Kuwait
	Withdrawal would be politically costly for Saddam’s regime
Decision	Saddam decides to withdraw from Kuwait peacefully

8.2. Domain Enriched Ontology

To instantiate a domain specific ontology from the enriched ontology, the ontology designer will instantiate concepts (classes) associated with each type of proposition and social network entities. For instance, an action proposition has constituents including *subject, verb, object* and an optional *outcome* (see Section 7.6a (i)). The ontology designer would then create classes for each one of these along with a separate class with the name of this proposition and then would define the properties for each class accordingly. Upon executing the *Pellet* reasoner, this proposition class will automatically become a sub-class of the *Action* (Influence net) class and its constituents will become sub-classes of *subject, verb, object* and *outcome* classes. The constituent classes will also become sub-classes of the corresponding Social network classes in the enriched ontology such as *Agent, Organization, Action, Task,* etc. Similarly, a belief proposition has *subject, verb,* actual *belief* about an *Action, Ability* or *Decision* and optional *Evidence* (see Section 7.6a (ii)). Upon executing the reasoner, the class for this belief proposition will become a sub-class of *Belief* (Influence net) class and its constituent belief will become sub-class of *Belief* (Social network) class in the enriched ontology.

As a result, whenever this domain enriched ontology is queried for either *Action* or *Belief* classes, it will also return those actions and beliefs that were created as the sub-classes under Social network related classes. Tables 11, 12 and 13 show the elements of *Action, Belief,* and *Decision* propositions to be created as classes. Figure 66 shows the *Action* (Influence net) class *US leads an effort to liberate Kuwait*. - Only one proposition is shown. It is a sub-class of the *Action* (Influence net) class and it can be seen that its constituents (*subject*, *object* and *verb*) are also the sub-classes of *Entity* and *Perception* (Social network) classes due to the defined mappings in the enriched ontology. Figures 67 and 68 show the *Action* class instantiated and populated in both types of classes, i.e., *Proposition* (Influence net) class and *Entity* (Social network) class, and how the constituents of this action proposition class map to the related classes of Social network. For example, the class *US* becomes a subclass of *subject*, *object* and *agent* classes and *leads* becomes a subclass of the *verb* and *task* classes as shown by the directed arrows.

These mappings can be utilized to exchange information between an Influence net model and a Social network model. The next chapter explains how this can be achieved by using the SAF Algorithm (Influence net) and using its results to update the corresponding Social network model. This exchange of analysis results from Influence nets to Social network is not incorporated in the ontology and is performed manually. Up to this point, the enriched ontology has enabled us to see the mapped concepts, i.e., which Influence net concepts and relationships map to which concepts and relationships of Social networks.

Table 11: Action Proposition Class Elements in Enriched Ontology

Proposition	Value	Subject	Verb	Object	Outcome	Proposition Type
Action	US leads an effort to liberate Kuwait	US	Leads	Effort	To Liberate Kuwait	Affirmative
	Coalition enforces UN export and import embargo on Iraq	Coalition	Enforces	UN export & Import Embargo, Iraq	-	Affirmative
	UN passes resolution for withdrawal of Iraqi forces from Kuwait	UN	Passes	Resolution	To Withdraw Iraqi forces from Kuwait	Affirmative

Table 12: Belief Proposition Class Elements in Enriched Ontology

Proposition	Value	Subject	Verb	Action, Ability, Decision, Event	Evidence	Proposition Type
Belief	Saddam believes he is in control of events	Saddam	Believes	Events are in Control	-	Affirmative
	Saddam believes there is a conspiracy for Iraq's domestic and economic destabilization			Conspiracy exists for Iraq's domestic and economic destabilization	-	Affirmative
	Saddam believes annexation of Kuwait will help him politically			Kuwait Annexation will help him	-	Affirmative
	Saddam believes annexation of Kuwait will Iraq financially			Kuwait Annexation will Iraq	-	Affirmative
	Saddam believes US has resolve to liberate Kuwait			US has Kuwait liberation resolution	-	Affirmative
	Withdrawal would be politically costly for Saddam's regime	Coalition		Withdrawal is politically costly for Saddam's regime	-	Affirmative

Table 13 Decision Proposition Class Elements in Enriched Ontology

Proposition	Value	Subject	Verb	Action			Proposition Type
				Subject	**Verb**	**Object**	
Decision	Saddam decides to withdraw from Kuwait peacefully	Saddam	Decides	Saddam	Withdraws	Troops, Kuwait	Affirmative

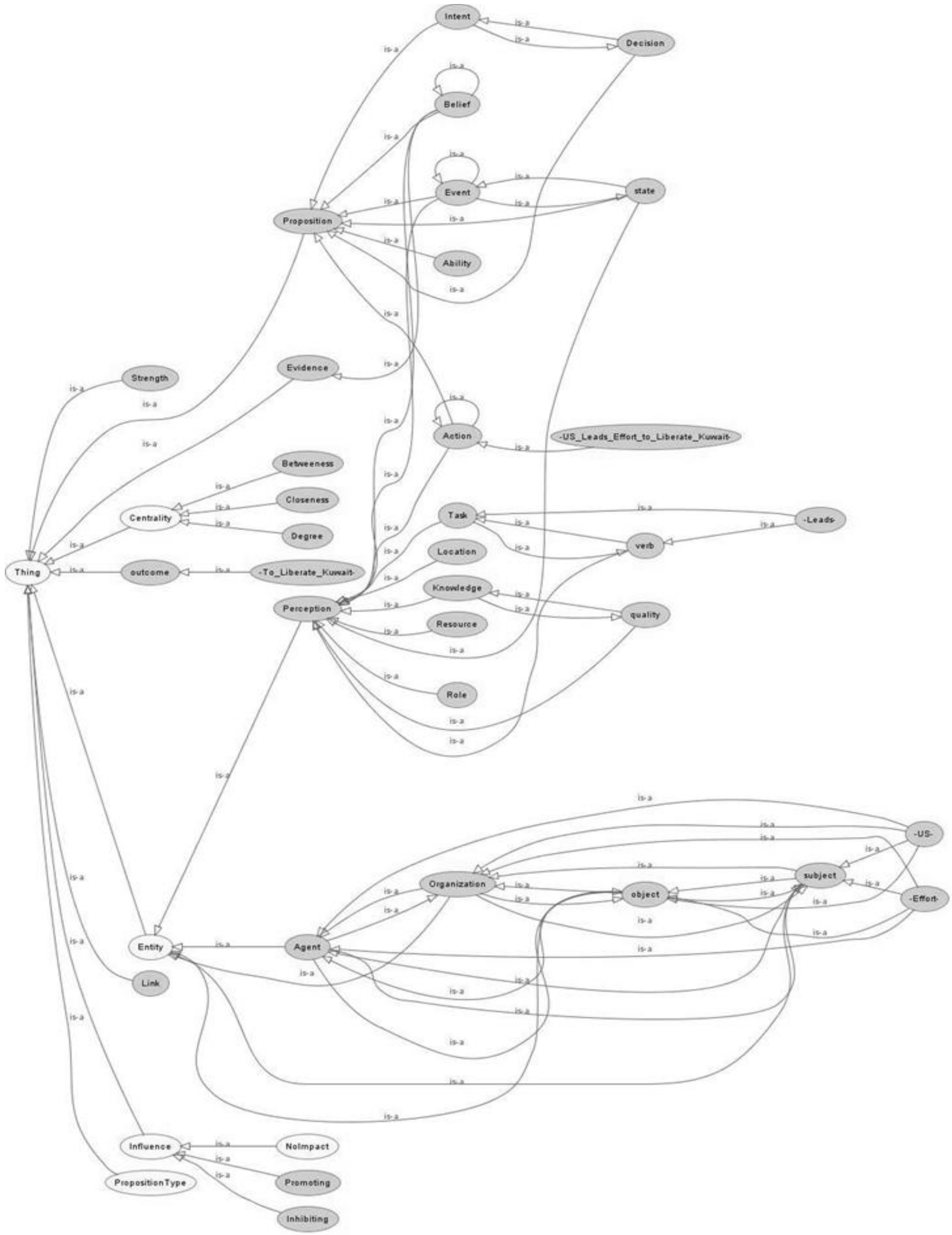


Figure 66: Domain Specific Ontology for Iraq-Kuwait Scenario

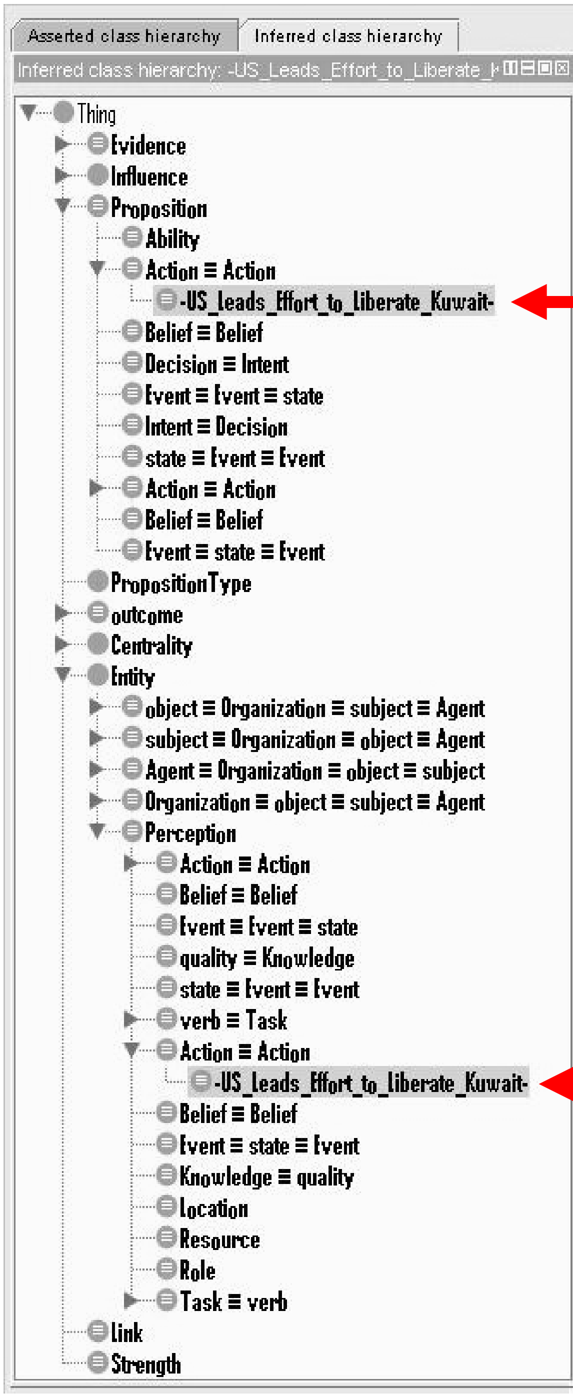


Figure 67: Action Proposition Populated in a Domain Enriched Ontology

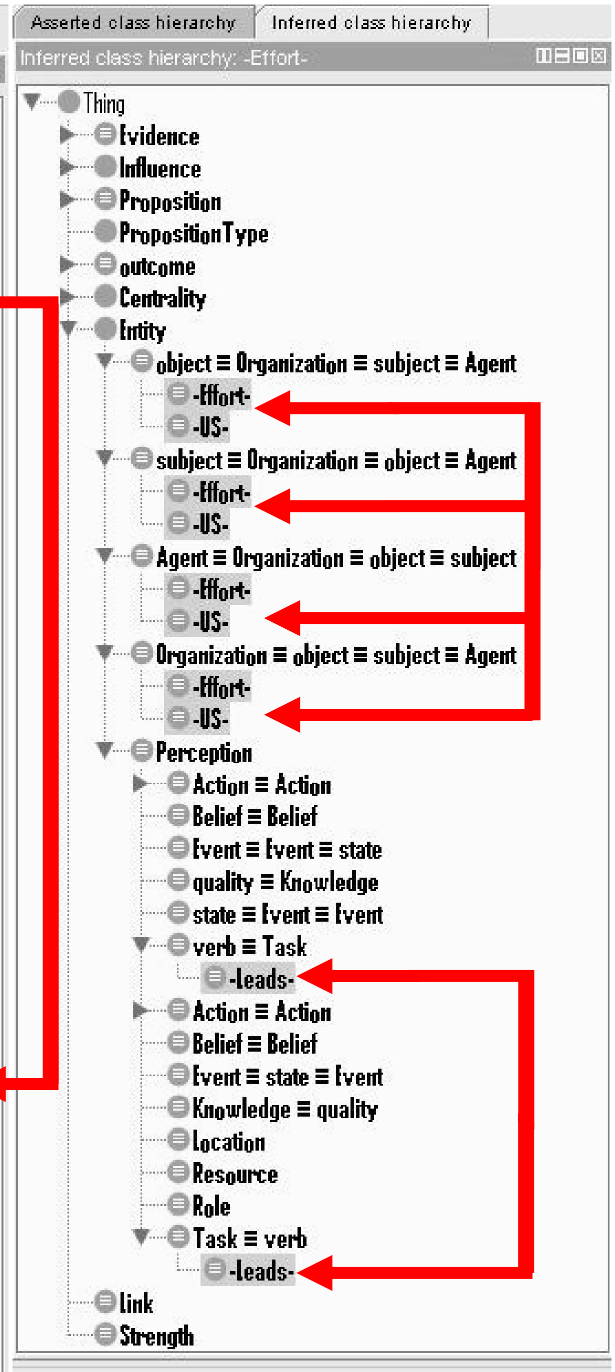


Figure 68: Subject, Object, Agent, Organization, Verb and Tasks in a Domain Enriched Ontology

CHAPTER 9: APPLICATION - SAF TO MEASURES OF CENTRALITY

9.1. Overview

In the previous chapter, we determined concepts of both modeling techniques which map to each other. If thoughtfully approached, we can utilize those mappings to exchange analysis results from an Influence net model into a corresponding Social network model. This step is also dependent upon the experience of the ontology designer regarding the two modeling techniques. This chapter describes the application of the workflow.

As a first step towards the application of multi-modeling, the SAF algorithm for the Influence net model implemented in Pythia and measures of centrality (degree, betweenness, closeness) for the Social network model implemented in ORA were used to exchange analysis results in one direction (from Pythia to ORA). Since models developed using both techniques use the same corpus of data, the objective behind this exchange is to ensure accuracy and consistency between both models and to observe the impact of changes suggested by the SAF Algorithm results. For instance, the results from the SAF algorithm may suggest removing certain links in the Social network which will alter the layout of the Social network and may have a drastic effect on the values of centrality measures. The updated social network analysis results would then provide key information regarding the network given the set of actions provided by SAF take place. Figure 69 shows the process.

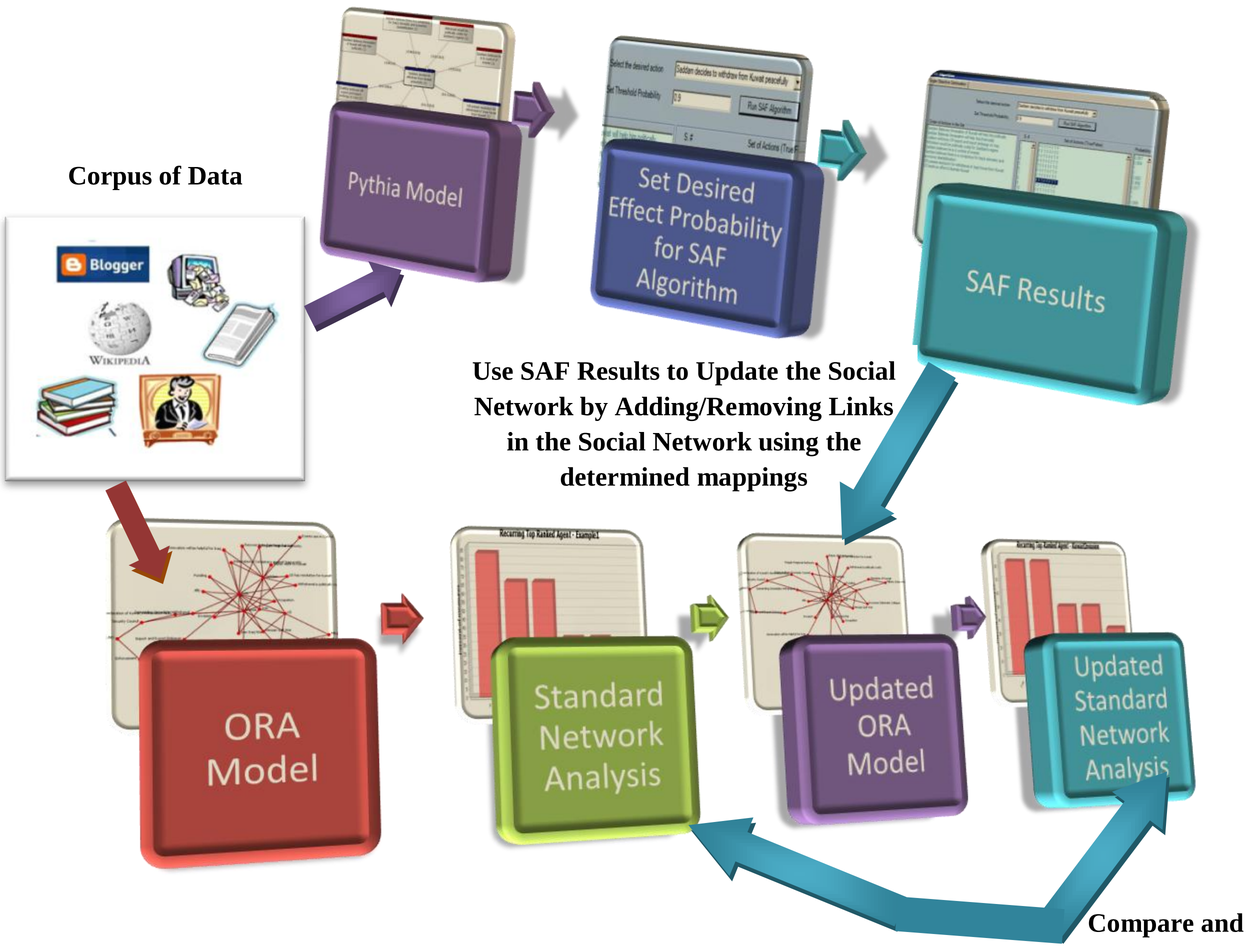


Figure 69: SAF to Measures of Centrality Process Flow

The SAF algorithm was discussed in Section 3.3c (ii); it outputs the best sets of actions which maximize the probability of an effect whereas measures of centrality provide details about network activity level, points of communication, and flow of information across the Social network, as explained in Section 4.3. Each of these measures of centrality changes when the network is modified and can provide meaningful information regarding the changes in the network structure. Examples of such changes are failure of communication, increase or decrease in information flow, and power possessed by entities [33].

The Influence net model developed for the example scenario (corpus of data) given in Section 8.1 is shown in Figure 70 and the Social network model is shown in Figure 71. Pythia was used to develop the Influence net, ORA was used to develop the Social network model with the help of meta-matrix generated using AutoMap.

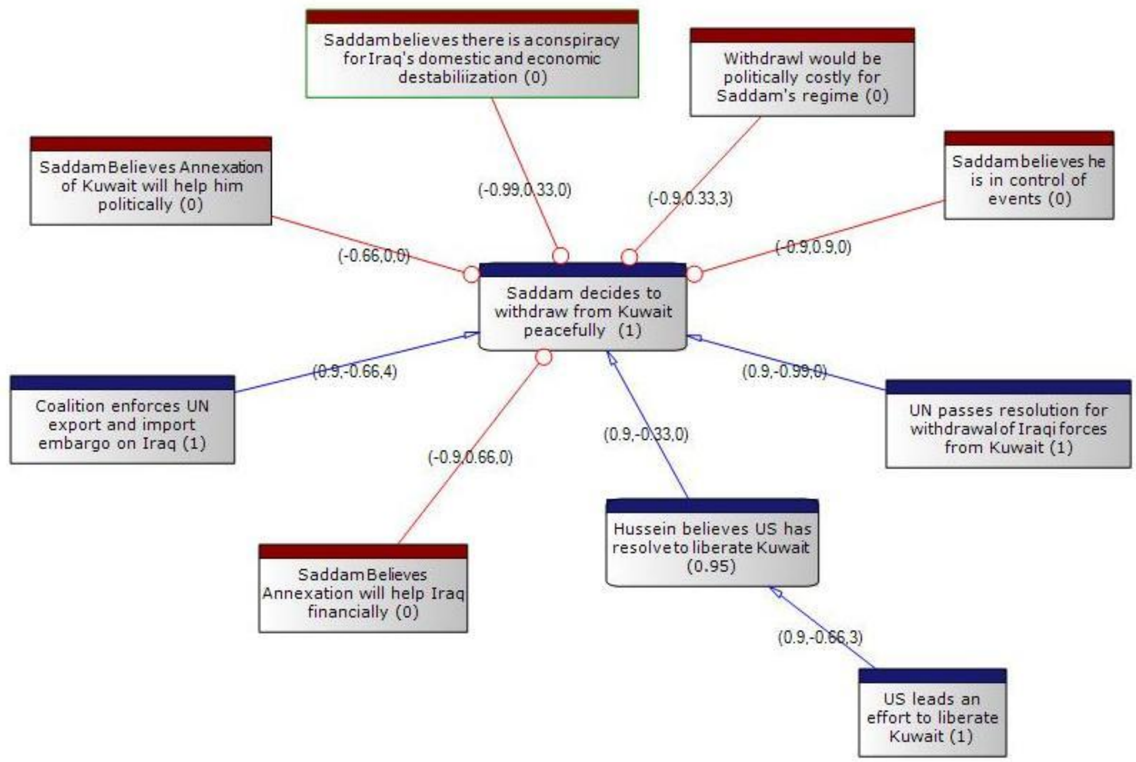


Figure 70: Influence net Model for Iraq-Kuwait War Scenario

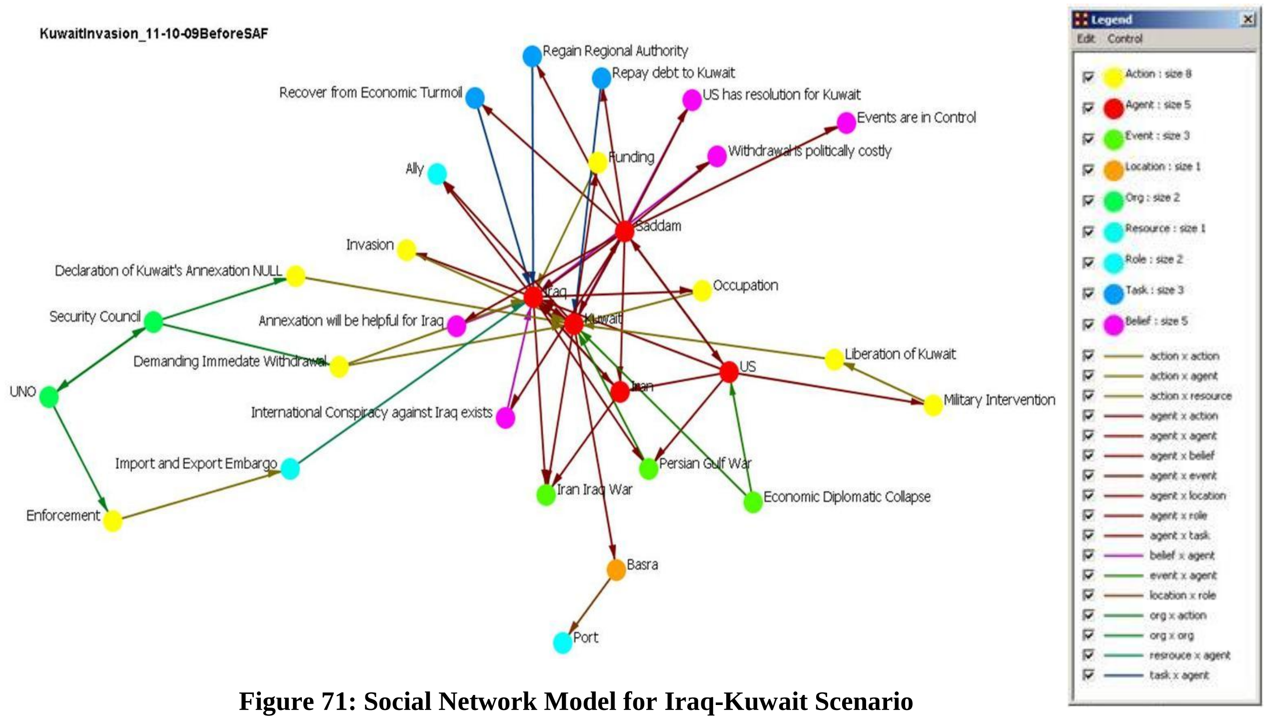


Figure 71: Social Network Model for Iraq-Kuwait Scenario

9.2. Steps

Following are the exact steps for using the SAF results from the Influence net model to update the Social network model:

a. Execute the SAF Algorithm for Pythia Model

The desired effect in the Influence net (Figure 70) whose probability should be maximum is *Saddam decides to withdraw from Kuwait peacefully*. This can be specified in Pythia's SAF Algorithm user interface as shown in Figure 72. Upon executing the algorithm, it yields the best sets of actions which maximize the likelihood of this desired effect.

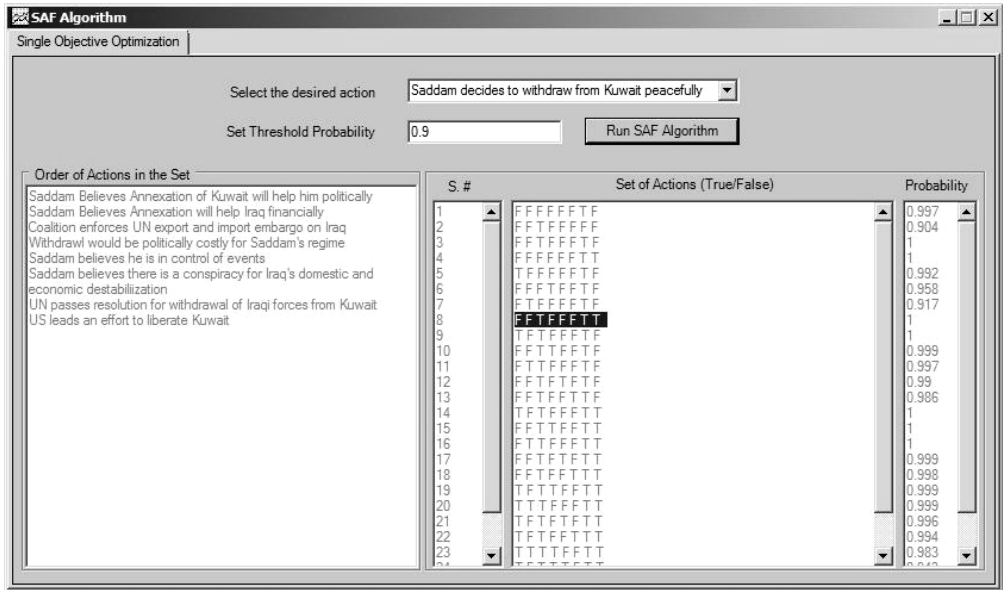


Figure 72: SAF Algorithm Execution

b. Select the best combination of actions

Select the best combination of actions which yields an acceptable probability of the desired effect. For instance, in this case, to achieve the desired effect the best set of actions suggests that the following propositions must be false as shown in Figure 72:

i. *Saddam believes annexation of Kuwait will help him politically*.

ii. *Saddam believes annexation of Kuwait will help Iraq financially*.

iii. *Withdrawal would be politically costly for Saddam's regime*.

iv. *Saddam believes he is in control of events*.

v. *Saddam believes there is a conspiracy for Iraq's domestic and economic destabilization*.

c. Use SAF Results to Update Social Network Model

Use this combination of actions to update the corresponding Social network (ORA) model by adding or removing links between the Social network entities which correspond to the concepts (classes) of the propositions and their constituents in the ontology. This should be simple now, since we already know the composition of a proposition, and also which concepts map between Influence nets and Social networks (with the help of our enriched ontology). We can start mapping the elements of these propositions with Social network entities as follows (Influence net = IN, Social network = SN):

i. **Saddam believes he is in control of events** (IN - Belief proposition)

- Subject/Agent = Saddam
- Verb = believes
- Belief perception = in Control of Events (i.e., SN Node: Events are in Control)

Link between agent *Saddam* and SN belief node *Events are in Control* should be removed as SAF suggests that *Saddam* shouldn't have this belief anymore.

ii. **Saddam believes there is a conspiracy for Iraq's domestic and economic destabilization** (IN – Belief proposition)

- Subject/Agent = Saddam
- Verb = believes
- Belief perception = International conspiracy against Iraq exists

Link between agent *Saddam* and belief node *International conspiracy against Iraq* exists should be removed, since this belief shouldn't exist anymore.

iii. **Saddam believes annexation of Kuwait will help him politically** (IN – Belief proposition)

- Subject/Agent = Saddam

- Verb = believes
- Belief perception = Annexation will be helpful for Iraq

Link between this agent *Saddam* and belief node *Annexation will be helpful for Iraq* should also be removed, as it shouldn't exist anymore.

iv. **Withdrawal would be politically costly for Saddam's regime** (IN – Belief proposition)

- Subject/Agent = Saddam
- Verb = believes
- Belief perception = Withdrawal is politically costly

Link between agent *Saddam* and belief node *Withdrawal is politically costly* should be removed also as it shouldn't exist anymore.

The updated Social network model is given in Figure 73.

d. ORA Standard Network Analysis before SAF

Execute ORA's standard network analyses (degree, betweenness, closeness measures of centrality) for the social network model before making changes suggested by the SAF Algorithm. ORA is a useful and rich social network analysis tool, it has a visualizer called ORA Visualizer which renders the conceptual images of social networks. ORA can generate different kinds of reports for the constructed social networks; we used the *Standard Network Analysis* report for both models before and after incorporating the SAF results. A standard network analysis report contains results of measures such as measures of centrality (degree, betweenness and closeness). All of these are calculated using the formulae given in Section 4.3.

e. ORA Standard Network Analysis after SAF

Execute standard (ORA) network analyses on the model after making the changes suggested by the SAF Algorithm.

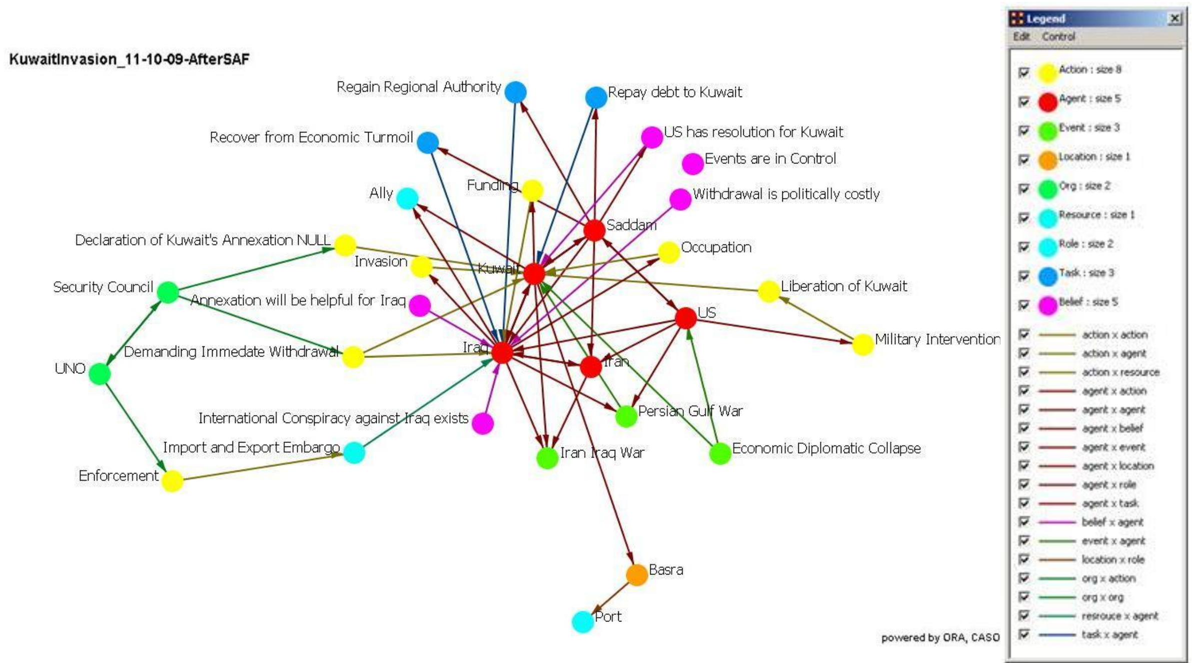


Figure 73: Updated Social Network Model for Iraq-Kuwait Scenario

f. Analyze the differences in the results

Once the standard network analysis reports are generated for both models, they can be compared and analyzed for any revealing information which might appear as a result of incorporating the SAF Algorithm results into the Social network model. ORA supports report generation for multiple models, so they can be compared in parallel as shown in Tables 14, 15 and 16. The first Social network model (without SAF results) has all the links intact among the entities, since SAF suggests that for *Saddam to withdraw from*

Kuwait peacefully, most of the beliefs held by him shouldn't exist or, in other words, if those beliefs don't exist the likelihood of him withdrawing from Kuwait would increase drastically. We need to keep in mind that we are trying to find out the possibilities which could lead to this effect. In this attempt, we also want to know the Social network analysis results which might give us information about the impact that the removal of these beliefs from the network would cause. Following is a comparative analysis of the three measures of centrality for the top ten entities in the network:

i. *Total Degree Centrality*

Table 14 shows the total degree centrality measure of the top 10 entities in the social network model.

Table 14: Total Degree Centrality before and after SAF

	Kuwait Invasion Before SAF Algorithm Results		**Kuwait Invasion After SAF Algorithm Results**		
Rank	**Nodes**	**Value**	**Nodes**	**Value**	**%Difference**
1	Iraq	0.3276	Iraq	0.3276	0%
2	Kuwait	0.2931	Kuwait	0.2931	0%
3	**Saddam**	**0.2414**	**Saddam**	**0.1724**	**-28.57%**
4	US	0.1207	US	0.1207	0%
5	Iran	0.0862	Iran	0.0862	0%
6	Security Council	0.069	Security Council	0.069	0%
7	Demanding Immediate Withdrawal	0.0517	Demanding Immediate Withdrawal	0.0517	0%
8	Persian Gulf War	0.0517	Persian Gulf War	0.0517	0%
9	Iran Iraq War	0.0517	Iran Iraq War	0.0517	0%
10	UNO	0.0517	UNO	0.0517	N/A

It is evident that since all the removed links were beliefs held by *Saddam*, his degree centrality would decrease (i.e., 0.2414 to 0.1724) by 28.6%, as it appears on the right side of the table. *Saddam* is the third most active entity in the network before and after the SAF results. Incorporation of the SAF results reduced this entity's activity level but retains its rank as one of the most active entities in the network. Degree centrality of all other entities remains unchanged.

ii. Betweenness Centrality

Betweenness centrality depicts the point of communication, and it can be seen from Table 15 that almost every other entity's betweenness centrality is affected negatively, with *Saddam*'s bearing the maximum change (i.e., 0.2997 to 0.1981) which is reduction by 33.9%, followed by *Kuwait* and *Security Council*.

Table 15: Betweenness Centrality before and after SAF

	Kuwait Invasion Before SAF Algorithm Results		Kuwait Invasion After SAF Algorithm Results		
Rank	**Nodes**	**Value**	**Nodes**	**Value**	**%Difference**
1	Kuwait	0.4657	Kuwait	0.3635	-21.95%
2	**Saddam**	**0.3126**	**Iraq**	**0.259**	**-17.14%**
3	**Iraq**	**0.2997**	**Saddam**	**0.1981**	**-33.90%**
4	US	0.0659	US	0.0634	-3.74%
5	Import and Export Embargo	0.032	Basra	0.0308	-3.85%
6	Demanding Immediate Withdrawal	0.0318	Military Intervention	0.0283	-10.97%
7	Basra	0.0308	Import and Export Embargo	0.0271	-12.00%
8	Military Intervention	0.0283	Demanding Immediate Withdrawal	0.0269	-5.07%
9	Security Council	0.0271	Security Council	0.0222	-18.18%
10	Liberation of Kuwait	0.0259	Liberation of Kuwait	0.0209	N/A

Saddam's rank dropped by one level because of incorporating the SAF results. The entities whose ranks improved by two levels were location *Basra* and the action *Military Intervention*, causing the ranks of *Import and Export Embargo* and *Demanding Immediate Withdrawal* to drop by two levels.

iii. Closeness Centrality

Closeness centrality determines how easily an entity can access others in the network; *Security Council* bears the maximum negative change (i.e., 0.2417 to 0.1295) as much as 46.43% followed by *UNO*, as shown in Table 16.

Table 16: Closeness Centrality before and after SAF

	Kuwait Invasion Before SAF Algorithm Results		Kuwait Invasion After SAF Algorithm Results		
Rank	**Nodes**	**Value**	**Nodes**	**Value**	**%Difference**
1	**Security Council**	**0.2417**	**Security Council**	**0.1295**	**-46.43%**
2	UNO	0.2028	UNO	0.1193	-41.15%
3	Economic Diplomatic Collapse	0.1229	Economic Diplomatic Collapse	0.0843	-31.40%
4	Demanding Immediate Withdrawal	0.1213	Demanding Immediate Withdrawal	0.0836	-31.12%
5	Declaration of Kuwait's Annexation NULL	0.1189	Enforcement	0.0831	-30.09%
6	Saddam	0.1189	Declaration of Kuwait's Annexation NULL	0.0824	-30.68%
7	Enforcement	0.1165	International Conspiracy against Iraq exists	0.0808	-30.64%
8	Kuwait	0.1155	Withdrawal is politically costly	0.0808	-30.08%
9	US	0.1146	Annexation will be helpful for Iraq	0.0808	-29.53%
10	Import and Export Embargo	0.1137	Import and Export Embargo	0.0808	N/A

Saddam is ranked sixth before the SAF results, and after incorporating the SAF results, he slips off the list. Other entities which don't remain in the top ten after the SAF results include *Kuwait* and *US.* The action *Enforcement* of import and export embargo improves its rank by two levels and other new entities such as the beliefs about international conspiracy against Iraq, politically costly withdrawal and annexation being helpful for Iraq, enter the list of the top ten ranked entities.

Changes in the Social network model driven by the SAF analysis results obtained from the Influence net model provided useful information about the Social network which can subsequently be utilized to perform further analyses and steps such as attempts to improve the betweenness centrality and closeness centrality of those entities which play key roles in the network and focusing on entities which have improved their ranks and have become more active in the network as compared to the situation prior to incorporating the SAF results.

This wouldn't have been possible without the mappings that were determined as a result of the enriched ontology. Making use of these mappings to exchange the results is a manual process but the possibility exists for future extensions where this process might be automated using the available automation tools.

CHAPTER 10: CONCLUSION AND FUTURE WORK

10.1. Conclusion

In this thesis, we presented a theoretical foundation using a workflow that facilitates semantic extraction, i.e., determination of mapped/overlapping concepts among different modeling techniques such as Influence net and Social network modeling techniques. The workflow is reusable and can be used for any other set of modeling techniques. These mapped concepts can then be utilized to exchange analysis results from a model constructed using one technique to a model constructed using another one.

The workflow starts at the conceptual modeling level, where *concept maps* were constructed for a set of identified focus questions for both techniques. Since concept maps are an informal representation of a domain, their notation was formalized by using specific concepts to construct the *meta-models* at the meta-modeling level. The meta-models reveal structural aspects of both techniques but are unable to capture knowledge about them. Therefore, they are used as the foundation for an initial ontology at the ontological modeling level. This initial ontology is called *pseudo ontology* and is similar to the meta-model and does not contain any specific concepts of the modeling technique. Using the additional concepts from concept maps, and the knowledge of the ontology designer about the modeling techniques, a *refactored ontology* is constructed in which specific concepts about the modeling technique are explicitly added. After both refactored ontologies are completed, an enriched ontology is constructed which contains the mappings between concepts (classes) of both techniques (refactored ontologies). A domain enriched ontology serves as the knowledge container about a specific domain and can be reused later, for instance, by querying it to get all *Agent* entities, *Actions*, or *Belief* propositions.

The application of this workflow was described using an illustrative scenario which served as the corpus of data from which both types of models were constructed. Pythia was used to construct the Influence net model, whereas AutoMap and ORA were used to construct the Social network models. The Sequence of Actions Finder (SAF) algorithm is one of the analysis techniques in Pythia that produces the best sets of actions for the maximum likelihood of occurrence of a desired effect. This best set of actions (propositions) is utilized to update the Social network model by adding/removing links between the Social network entities. This is facilitated by the mappings determined in the enriched ontology: subject and object classes of Influence net map to the agent and organization classes of Social networks. The impact of this change can be analyzed by comparing ORA's standard Social network analysis reports before and after the change in the network. The construction of the enriched ontology enabled us to determine the overlaps between both types of ontologies and those overlaps were used to exchange information from an Influence net model to a Social network model. They can also be used to ensure consistency between both types of models since we would know which concepts of each technique map to the other.

10.2. Future Extension

The proposed workflow in this thesis employing concept maps, meta-models and ontologies to extract semantics of different modeling techniques is the first attempt using two modeling techniques. There is enormous room for further work in this area such as enrichment of the ontologies which were constructed in this thesis either by adding more concepts, or by incorporation of other modeling techniques to further enrich the enriched ontology with new concepts. Since ontology construction is an iterative process, future researchers are encouraged to add more concepts (classes) related to Influence net and Social network modeling techniques, if they are not present in the current ontologies.

Furthermore, the proposed workflow can be repeated for other modeling techniques as well by constructing concept maps, meta-models, then the ontologies, which will facilitate information exchange between the newly included technique and the already existing ones.

In this thesis, the mappings between the refactored ontologies were determined manually. There are ontology and schema matching tools available such as COMA++ which perform matching of different ontologies and determining semantic equivalences among them. Automatic matching was not performed and is a suggested extension of this research work.

REFERENCES

REFERENCES

[1] Gerti Kappel, Elisabeth Kapsammer, Horst Kargl, Gerhard Kramler, Thomas Reiter, Werner Retschitzegger, Wieland Schwinger, and Manuel Wimmer, "Lifting Metamodels to Ontologies: A Step to the Semantic Integration of Modeling Languages", *Model Driven Engineering Languages and Systems*, Springer Berlin-Heidelberg, 2006, pp. 528-542.

[2] Joseph D. Novak & Alberto J. Cañas, "The Theory Underlying Concept Maps and How to Construct and Use Them" in *Technical Report IHMC CmapTools* 2006-01 Rev 01-2008.

[3] David P. Ausubel, "*The Psychology of Meaningful Verbal Learning*", Grune & Stratton, Orlando FL, 1963.

[4] Alan Page, Ken Johnston, Bj Rollison, "*How we test software at Microsoft*", Microsoft Press 1st edition, Redmond Washington, December 10, 2008, pp. 172-173.

[5] Julie A. Rosen and Wayne L. Smith, "Influence Net Modeling with Causal Strengths: An Evolutionary Approach", in *Proceedings of the Command and Control Research and Technology Symposium, Naval Post Graduate School*, Monterey CA, 1996, pp. 25-28.

[6] K.C. Chang, Paul E. Lehner, Alexander H. Levis, S. Abbas K. Zaidi, and Xinhai Zhao. "On Causal Influence Logic", in *Technical Report for Subcontract no. 26-940079-80, George Mason University, Center of Excellence for C3I,* December 3, 1994.

[7] Lee W. Wagenhals and Alexander H. Levis, "Course of Action Development and Evaluation", in *Proc. 5th Int'l Command and Control Research and Technology Symposium, Naval Postgraduate School,* Monterey, CA, June 2000.

[8] "Pythia: Timed Influence Net Modeler," http://sysarch.gmu.edu/main/software/, SAL-GMU.

[9] Sajjad Haider, Abbas K. Zaidi, and Alexander H. Levis, "A Heuristic Approach for Best Sets of Actions Determination in Influence Nets.", in *Proceedings 2004 IEEE International Conference on Information Reuse and Integration,* Las Vegas, NV, November 2004.

[10] Kathleen M. Carley, Jeff Reminga, “ORA: Organization Risk Analyzer”, *CASOS Technical Report*, CMU - Institute for Software Research International, July 2004
[11] “CASOS Social Networking Tools”, http://www.casos.cs.cmu.edu/computational_tools/tools.html.
[12] “CASOS AutoMap Tool”, http://www.casos.cs.cmu.edu/projects/automap/.
[13] “Department of Defense Training with Simulations Handbook”, http://www.strategypage.com/prowg/simulationshandbook/default.asp.
[14] Karagiannis, D., and Kuehn, H. 2002., “Meta-Modeling Platforms”, in *Proceedings of the IEEE International Conference on Information Reuse and Integration*, Las Vegas, NV, November 2004.
[15] “OMG Unified Modeling Language™ (OMG UML)”, Infrastructure Version 2.2, formal/2009-02-04, http://www.omg.org/spec/UML/2.2/Infrastructure
[16] “Eclipse Tools Project: Eclipse Modeling Framework (EMF)”, http://www.eclipse.org/emf/.
[17] Natalya F. Noy, Deborah L. McGuinness, “Ontology Development 101: A Guide to Creating Your First Ontology”, in *Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880*, Stanford, CA, March 2001.
[18] Matthew Horridge, Nick Drummond, Simon Jupp, Georgina Moulton, Robert Stevens, “A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools”, Edition 1.2, The University of Manchester, March 13, 2009.
[19] D. L. Mcguinness and F. van Harmelen, "OWL Web Ontology Language Overview," World Wide Web Consortium, 2004, http://www.w3.org/TR/owl-features/.
[20] “What is protégé”, http://protege.stanford.edu/overview/index.html.
[21] “Pellet OWL Reasoner”, http://clarkparsia.com/pellet/protege.
[22] “Fact++ OWL-DL Reasoner”, http://owl.man.ac.uk/factplusplus/.
[23] Motoshi Saeki and Haruhiko Kaiya, “On Relationships among Models, Meta Models and Ontologies”, in the *Proceedings of the 6th OOPSLA Workshop on Domain-Specific Modeling (DSM'06)*, University of Jyväskylä, Finland 2006.
[24] Gerti Kappel, Horst Kargl, Gerhard Kramler, Andrea Schauerhuber, Martina Seidl, Michael Strommer and Manuel Wimmer, “Matching Metamodels with Semantic Systems – An Experience Report”, in *Workshop Proceedings of Datenbanksysteme in Business, Technologie und Web (BTW 2007)*, Verlag Mainz, 2007, pp. 38-52.
[25] Daniel Engmann, Sabine Massmann, “Instance Matching with COMA++”, in *BTW 2007 Workshop: Model Management und Metadaten-Verwaltung*, Aachen, 2007, pp. 28-37.

[26] David Aumueller, Hong-Hai Do, Sabine Massmann, Erhard Rahm, "Schema and Ontology Matching with COMA++", in the *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, Baltimore, Maryland, 2005, pp. 906-908.

[27] "OpenCyc Ontologies", http://www.cyc.com/cycdoc/vocab/vocab-toc.html.

[28] "Swoogle, Semantic Web Search", http://swoogle.umbc.edu/.

[29] "DAML Ontology Library", http://www.daml.org/ontologies/.

[30] "Graphviz – Graph Visualization Software", http://www.graphviz.org/.

[31] Paul K. Davis and John Arquilla, "Deterring or coercing opponents in crisis: Lessons from the war with Saddam Hussein", *Rand Corporation*, Santa Monica, California, 1991.

[32] "CASOS ORA Tool", http://www.casos.cs.cmu.edu/projects/ora/.

[33] "Power in Networks", http://www.orgnet.com.

CURRICULUM VITAE

Muhammad Faraz Rafi received his Bachelor of Engineering in Computer & Information Systems in December 2005 from NED University of Engineering & Technology in Karachi, Pakistan. He worked for one and a half years as a Testing and Quality Assurance Engineer at EzValidation Pvt Ltd, and served as a part time consultant in another private software company in the area of Software Test Automation.

He started his graduate studies with a major in Software Engineering at George Mason University in fall 2007. He served the System Architectures Laboratory in George Mason University as a Graduate Research Assistant for two and a half years and co-authored two publications in the area of *Activation Timed Influence Nets* and has completed this master's thesis work in the area of meta-modeling and multi-modeling as per the degree requirement. He currently works as a J2EE Application Developer at *Science Applications International Corporation* (SAIC) in Mclean, VA.