UNIVERSITY OF TWENTE. BiZZdesign

Analysis of Indirect Influence Relations in Goal-Oriented Requirements Engineering

Thesis report for the degree of Master of Science in Computer Science

Abelneh Y. Teka
S1070592

Graduation Committee:

Dr. Nelly Condori Fernandez

Dr. Ivan Kurtev

Dr. Dick Quartel

Wilco Englesman, MSC

August 10, 2012

Abstract

Business environments nowadays are becoming increasingly more dynamic, demanding continuous adaptation in business process designs and realizations. Regardless of their causes, most changes in the business environment have often dramatic consequences upon business processes and supporting/enabling IT systems. In most cases, these changes manifest as alterations in one or more goals of stakeholders of the system. These goal changes will then propagate to the requirements, designs, implementations and test cases of a system development process.

Along with recent trends in using goal-oriented approaches for requirements engineering, various techniques for managing evolutionary goals and requirements are proposed and used by the software engineering community. Enterprise Architecture (EA) models which tie business goals, business processes and supporting IT systems are also expected to have a technique for analyzing changes in goals and requirements.

Unfortunately common Enterprise Architecture (EA) frameworks like The Open Group Architecture Framework (TOGAF) and EA modeling languages like ArchiMate lacks support for analyzing goal and requirement change impacts. This reduces the adaptability of EA in addition to limiting the dynamicity of the organization employing the EA. Furthermore, lack of reasoning support on influence relations on goal models limits the decision-making capability of EA users by reducing the amount of available information about goal change impacts.

This thesis endeavors to fill these gaps by extending a metamodel of an existing requirements modeling language called ARMOR. Our approach proposes well-defined semantics for goal influence relations that can support reasoning on indirect influence relations. Since ARMOR is now part of the motivation aspect of ArchiMate, our approach will be tailored to the context of ArchiMate modeling language.

To leverage existing change impact-analysis techniques, a literature review was conducted on existing goal-oriented requirements engineering techniques. Two types of reasoning techniques are selected from a comparative analysis performed on the results of the literature review: TROPOS-based Qualitative reasoning and Fuzzy-logic based Quantitative reasoning. These two techniques support different levels of reasoning abstractions and help in entertaining different types of users (technical and non-technical). This report also proposes a quantitative-reasoning based approach to model and simulate feedback loops of goal influences relations.

Adapted algorithms as well as tool support for the reasoning techniques are realized and validated on a test case study. The test case study shows that both approaches are applicable to analyze indirect influence relations and they generate reasonably consistent results.

Furthermore the test case study reveals that each approach has its own merits and demerits. The Fuzzy logic based quantitative approach is better in providing concrete values for more detailed goal analysis. But it tends to result undetermined (zero level) goal satisfaction values. The TROPOS-based qualitative approach is suitable in providing high level goal analysis and detecting conflicting goal contributions. It is also usable for non technical users since it uses more understandable textual specifications though its textual specification can be ambiguous.

Preface

This report is the finishing line of the six months I spent working hard on my thesis. It is also the benchmark to conclude the two years I spent in University of Twente to finish my MSC in Computer Science. But most importantly, this is the point I have been looking for so long.

Looking back the endeavor to reach here, I remember bumpy roads. But the bumps are history now. As the old saying goes "We acquire the strength we have overcome", I feel that those bumps make me a better person.

I am grateful to many people who help me in writing this thesis. Nelly condor Fernandez, my first supervisor, so many things were impossible without your support. Your comments were always invaluable not only in my thesis work, but also in the papers we write together. Ivan Kurtev, my second supervisor, I always taught I had two first supervisors because you have been like a first supervisor during my thesis work. Dick Quartel, finishing my project on time would have been out of question without your continuous support. Wilco Engelsman and Henry Franken, thanks for giving me the chance to work at BiZZdesign and for your remarks in guiding my thesis work.

I am also truly indebted and thankful to many people who help me to reach this point. Maya Daneva, you were by my side by every respect the past two years. Roel Wieringa, the conferences in Essen and Madrid were possible with your support.

Konjo, I am lucky to have you by my side. Our Skype talks were always mind boosters before I start crunching papers and slides. My friend Addisu, It was always good to have someone to call and visit when feeling down. Your comments were also invaluable to the success of my work.

My family, you were always supporting me in every step to reach this point. I am where I am because of you.

Fellow interns at BiZZdesign, you were the first Dutch students I really acquainted with. It was really fun to work with you.

And of course my Ethiopian friends in Enschede, It was always fun to hang out with you for a little chitchat out of our "prison cells". I am goanna miss Macandra, Wale's Doro, Exotic club, Abatu Street and so much more.

Abelneh Yirsaw Teka,

August 2012

Table of Contents

List of Figures

List of Tables

1. Introduction

Business environments nowadays are becoming increasingly more dynamic, demanding continuous adaptation in business process designs and realizations. Numerous events can trigger changes that are responsible for the dynamic aspects of business processes and supporting/enabling IT systems. In most cases, these changes manifest as alterations in one or more goals of stakeholders of the system. These goal changes will then propagate to requirements, designs, implementations and test cases of a system development process.

To analyze the impacts of dynamic stakeholder goals and requirements, various change impact analysis techniques are proposed and used by the software engineering community [1], [2]. Enterprise Architecture (EA) models which tie business goals, business processes and supporting IT systems are also expected to have a technique for analyzing changes in goals and requirements of the EA.

Unfortunately common EA frameworks like The Open Group Architecture Framework (TOGAF) [3] and EA modeling languages like ArchiMate[4] lacks support for analyzing impact of goal and requirement changes. The Architecture Development Method (ADM) of TOGAF tries to emphasize requirements management as a central process of EA development cycle but it does not provide an explicit technique to specify the motivation of EA components in terms of stakeholder goals [5].

Stakeholder goals can be added to, deleted from and modified in system goal models depending on the type of change occurring in the business environment. Among these changes, change in the satisfaction level of (soft) goals due to influence relations is the primary reason behind the dynamic aspect of the business environments [6]. In goals related with influence relations, a change in the satisfaction level of a goal affects the satisfaction level of its adjacent goal, which will in turn affect its adjacent goal and so on. Capabilities to reason on these kind of influence relations helps in maintaining continuous satisfaction of stakeholders' interest by supporting the required adaptability of business processes, products, service and their supporting IT systems.

Unfortunately, current object oriented RE methodologies give negligible attention to the management of influence relations among goals. On the other hand, there are few GORE based attempts to reason on goal satisfaction levels (e.g. [7–9]). But these recent attempts are not tailored or verified to be applicable in managing adaptable stakeholder goals in EA design contexts.

Recently however, researchers have started publishing promising approaches that include requirement and goal management in EA frameworks and modeling languages (e.g. [10], [11]). These approaches emphasize capturing the rationale why EA components are available by incorporating intentional concepts like stakeholder, goals and requirements in EA designs. Moreover these approaches go one step further by representing goal influence effects among directly related goals via notations like ++ and --.

Nevertheless, these approaches are not adequate to model and predict the effect of stakeholder goal change on higher level and indirectly related goals of the system. This demands investigating the applicability of current change impact analysis techniques to analyze influence relations of EA goal models. Based on the results of this investigation, it is also necessary to develop an algorithm that can analyze, reason, simulate and visualize influence relation impacts on satisfaction level of goals.

1.1: Project Context: BiZZdesign[1] and EA Modeling

BiZZdesign is a company primarily involved in assisting organizations to be stronger by providing the necessary approaches for designing, improving and managing business processes effectively. The approach employed by the company includes proven and easy to use software tools, best practice models, methods, trainings, business consultancy etc [12].

One of the primary expertise's of BiZZdesign is enterprise architecture (EA) modeling and management. BiZZdesign is also one of the primary contributors and authors of ArchiMate[2], an emerging language for enterprise architecture modeling [13]. Even though originated and widely used in Netherlands, ArchiMate eventually became a standard of The Open Group and it is getting more and more worldwide attention these days [13].

BiZZdesign is not only the adopter and contributor of ArchiMate; it is also the first company to develop a tool for ArchiMate by the name BiZZdesign Architect [13]. BiZZdesign architect is a tool that can be used by the majority of EA methodologies; though it is primarily used for The Open Group Architecture Framework (TOGAF) in BiZZdesign projects. It is also one of the EA modeling tools accredited by The Open Group [14].

BiZZdesign has been using ArchiMate as a modeling language and TOGAF's Architecture development method (ADM) as a methodology in designing EA for its client organizations. As already noted earlier, most EA frameworks used to ignore goal and requirement modeling in EA designs. TOGAF's ADM is by no means different. Though ADM denotes requirement as a central method for developing an enterprise architecture, little or no attention is paid to represent (explicitly) the motivations or rationale behind the components of the EA. Putting it in other words, the why behind the architectures is not adequately addressed in TOGAF's ADM [5].

Consequently, the lack of goal and requirement management was an apparent problem in most of the BiZZdesign EA projects. In order to solve this problem, BiZZdesign has developed an extension of ArchiMate language that enables modeling of RE concepts in EA designs.

The new extension enables modeling of business goals, requirements, principles, stakeholders and other intentional elements. It also provides representation of influence relations among goals and requirements via classic influence notations like "++" and "--". Considering the importance of these intentional elements for EA, The Open Group has incorporated these extensions in ArchiMate 2.0 specification [4].

But this kind of goal influence modeling among directly related goals is not adequate for maintaining the requited adaptability of EA and its sub systems. Instead, EA goal models should support modeling and analysis of influence relations among indirectly related goals. This analysis in turn demands the presence of adequate reasoning techniques on goal influence relations. Consequently, indirect influence reasoning support on goal satisfaction levels is a must to have technique for BiZZdesign to maintain developing adaptable solutions for its client organizations.

[1] www.bizzdesign.com
[2] http://www.archimate.nl/

1.2 Problem Definition

This sub section explains the problem statement in more detail. It also explains the main objective of the research. Four research questions intended to achieve the research objective will also be formulated. Finally, this subsection identifies the scope of this thesis project.

1.2.1: Problem Statement

BiZZdesign and its clients are not the only organizations who realize the importance of goal formalizations and goal change impact analysis. Among others, i* framework for instance uses the notion of ++ and -- to model goal change impacts in requirements, process and Architecture models [15].

A similar approach that emphasizes on goal contribution analysis is the Non Functional Requirements (NFR) framework [8]. NFR tries to formulate various goal achievement levels and contribution types to enable qualitative analysis on goal change impacts. However this kind of qualitative reasoning on goal change analysis may suffer from ambiguous and undetermined levels of goal satisfaction values [16].

Some business goals, especially those termed as soft goals, are difficult to precisely specify their satisfaction levels due to their fuzzy nature. For this kind of goal, qualitative reasoning may seem the best option for describing their satisfaction values. Nevertheless, quantitative reasoning on goal change impacts would have been possible for these fuzzy goals via fuzzy logic reasoning techniques [17].

Goal influence cycles (feedback loops of influence relations) are critical influential factors in shaping the dynamics of system behaviors [6]. However none of the current goal and requirement change impact analysis techniques (e.g. KAOS, i*, NFR) consider effects of feedback loops in their goal contribution relation definitions. An exception to this is the TROPOS methodology which mentions the possibility of cyclic goal influence relations. But it does not explicitly indicate how to determine the long term effect of cyclic goal influences relations.

Consequently, *there is no formalized, complete and automatable reasoning technique for stakeholder goal change impact analysis in EA goal designs*. This lack in automated reasoning is forcing EA developers and users to analyze goal change impacts using their intuition and past experiences. Obviously, these kinds of practice are incomplete, time consuming, and usually error prone.

1.2.2: Research Objective

The lack of automated goal change impact analysis has been identified as the main research problem in the previous section. To address this problem, the process of analyzing both direct and indirect influence relations among goals should be formalized and automated to the maximum possible extent. Doing so will enable EA users and other stakeholders to assess and estimate the impact of changing a stakeholder goal on other goals of the EA designs.

To provide automated reasoning on goal influence relations, existing tools of BiZZdesign (BiZZdesign Architect) should be also extended to handle formal definitions of goal influence relations. These formal definitions can be used as inputs for possible reasoning on the influence relations.

This will lead to the main objective of this research:

Research Objective: Developing of algorithms that enable analysis, reasoning and simulation of direct and indirect influence relations among goals in Enterprise Architecture designs.

1.2.3: Research Questions

In order to achieve the research objective, the following main research question will be answered in this master thesis project.

How can we analyze and simulate the effect of influence relations among goal models in EA designs?

This research question can be decomposed in to four sub research questions outlined below.

RQ1. What are the formalizations of the existing goal-oriented requirements engineering approaches and what kind of reasoning do they allow?

This question involves identification of existing formalization trends in current GORE practices. A detailed literature review on available GORE methods is used to answer this research question. GORE methodologies can have multiple aspects that can be defined formally. Nevertheless the main focus of this research question will be formal definitions of goal satisfaction and goal-to-goal influence relations.

RQ2. Which goal formalization techniques support reasoning on indirect influence relations and change impact analysis for goal models?

Influence relations are not the only types of relations available among goals of EA models. There can also be other relation types like contain, refine etc. This research question helps in identifying formal definitions of influence relations that can be used in analyzing goal contribution and feedback loop effects. The formalization can be used to estimate the impact of changing the satisfaction level of goals.

RQ3. How can we simulate influence relation impacts on goal satisfaction values and how can we visualize the simulation?

BiZZdesign and its client organizations are interested in developing a tool support for indirect influence relations analysis. Hence the technique identified in RQ3, should be formally defined as an algorithm. Moreover a prototype implementation of this algorithm shall be realized as an extension to BiZZdesign Architect.

The prototype will be designed to accept certain goal changes scenarios. It will then use its influence relation reasoning engine to predict the goal satisfaction values. The simulations results will be visualized on the goal models developed in BiZZdesign Architect.

An immediate advantage of this kind of simulation is its supportive role in decision-making activities like alternative resource allocation. A project manager may need to know the best way of using available resources. Simulating all possible change scenarios using indirect influence reasoning tool will enable the manager to invest on the best change scenario that maximizes benefit of the organization.

RQ4. How can we model, simulate and visualize the effect of feedback loops in goal models?

One of the primary causes of business dynamics are feedback loops [6]. In goal feedback loops, a change in a goal satisfaction level will affect the source of the changing goal itself either positively or negatively. By answering this research question, it will be possible to model, analyze and reason on goal-cycle effects which will add substantial understanding to dynamic business environments.

1.2.4: Research Scope

Goal change impact analysis is not limited to changes in the satisfaction levels Goals. Goals can be added, deleted, modified etc. Even goal model modifications are not limited to changes in the satisfaction level of goals. It can be extended upward to analyze how a change in a goal will affect business processes, enterprise architecture designs and ultimately organizational structures. It can also be extended downward to system requirements, architectures, implementations and other technology artifacts. Analysis of all these change scenarios is too broad to cover in this thesis project. This project will focus only the analysis of change impacts in goal satisfaction levels due to goal-to-goal influence relations.

The reader may argue that influence relations are not the only type of goal-to-goal relations. Yes, there are other type of relations like *contains relation* where a goal will contain another goal, *mean end relations* where a number of sub goals are required to realize a higher level goal etc. These kinds of relations can be relevant in designing goal models by decomposing and refining goals and requirements. But they are either of no practical use in dealing with influence relations or they can be reasonably substituted by one of the influence relations stated in table 1.1 below. For instance, "means end relations" and "aggregation relations" can be considered as "AND decompositions" if we are concerned only on influence relations. Table 1.1 shows the types of relations that are relevant for goal indirect influence relations that will be used in this project.

Influence Relation Name	**AND**	**OR**	**Break**	**Hurt**	**Help**	**Make**
Symbol	(G2 ∧G3) ➔ G1	(G2 ∨G3) ➔ G1	G2 ➔- -G1	G2 ➔ - G1	G2 ➔+ G1	G2 ➔++ G1

Table 1.1 Goal relations relevant for indirect influence relations

Furthermore, since this project is undertaken in the context of BiZZdesign and its already developed and matured EA modeling tools, the impact analysis formalization and algorithm development may be geared towards the organizational contexts of BiZZdesign. Finally, as a master project, there is limited time to realize full implementation of the impact analysis algorithm; rather a demonstrative prototype application featuring some of the important parts of the algorithm will be implemented.

1.3: Research Methodology

This master thesis project will be addressed on the basis of Design Science research guidelines [18]. Design Science research enables understanding of a problem domain and realization of its solutions by building application artifacts like algorithms, formal logics, and even informal language descriptions [18].

In a nutshell, a design science research is about solving a certain problem. Hence the top level problem of any design science research is a practical problem [19]. Correspondingly, the main technical problem of this project will be designing an algorithm for indirect influence relation analysis.

The first step to solve this kind of top-level technical problem is strengthening the understanding of the problem itself [20]. This will lead to the knowledge questions (RQ1 and RQ2) to be answered by acquiring the relevant knowledge from a literature review on existing goal formalization and change impact analysis techniques.

The next step will be formulating the required goal formalization and impact analysis technique that can reason on indirect influence relations. This step may adapt the results from the literature review. Selection of appropriate formalization and reasoning technique will be a vital input to the process of answering the practical problem posed by RQ3.

The third step in this project will be developing the algorithm based on the results of the previous step. This step will lead to building a prototype application as an extension for BiZZdesign Architect tool.

The final step is using case studies to validate both the algorithm and the prototype application. This project conducts two case studies: one real industrial case on a client organization of BiZZdesign and a second case study on hypothetical example from The Open Group [21]. The hypothetical example is adapted to incorporate more complex goal cycles to enable simulation and validation of goal cycle effect.

A research following design science methodology is expected to incorporate seven guidelines though the degree of realizing each guideline can vary depending on the problem context. Some of these guidelines that will act as a base for this project include: design as an artifact, problem relevance, design evaluation, research contributions and communication of research [18]. The details of each guideline when applied to this project context, is too early to show here. The important steps to be followed in answering the research questions in this project are shown in Figure 1.1 below.

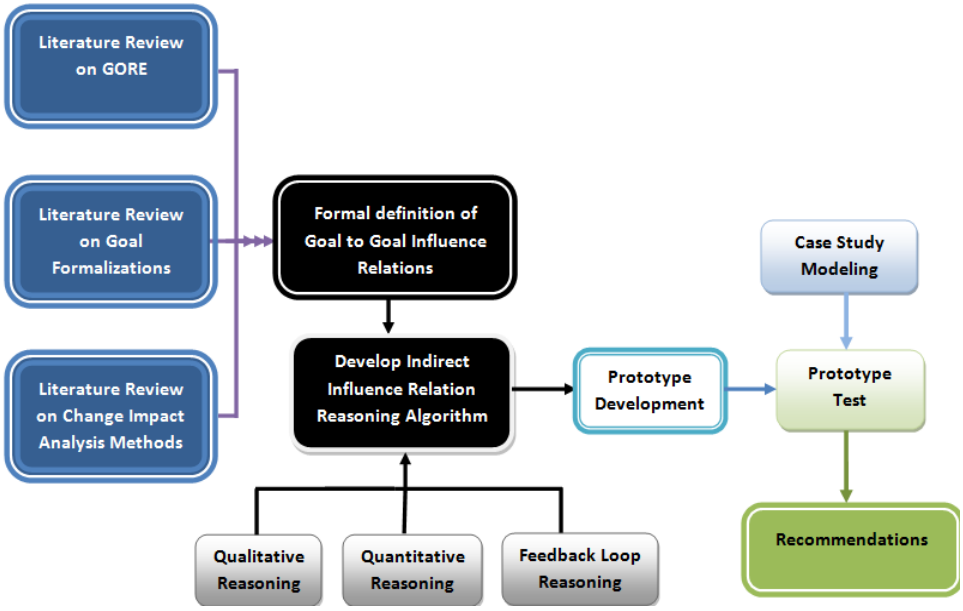

Figure 1.1: Research approach

1.4: Significance of the Project

A number of theoretical and practical contributions are expected from this project. Some of the major contributions are outlined below. The significance of the first three contributions is more of academic, whereas the last two are industrially significant.

The primary significance of this thesis project is its contribution towards semantically reach definitions for influence relations. The semantic is added to the goal relations via formal definitions that will enable reasoning on direct and indirect goal influence relations. An algorithm that utilizes this reasoning to simulate indirect-influence relations will also be developed.

Second, the algorithms developed from this project will support not only qualitative reasoning, but also quantitative reasoning. This will allow goal reasoning at different levels of abstractions for different kinds of users. Higher-level managers and non-technical users can use the qualitative reasoning for more abstracted goal reasoning while RE experts and other technical stakeholders can use the quantitative analysis for more detailed goal analysis. These will enhance the usability of goal models in particular and EA designs in general.

Third, goal change impact analysis is not limited to goal-to-goal relations. Analyzing the impact of changing a certain goal can be extended to higher-level EA components and organizational structures or lower-level software architecture and implementation components. This project can serve as a basis for studying the impacts of goal changes on other components of the EA.

Fourth, literature on goal formalization, goal change impact analysis and related concepts will be compiled and synthesized. The information from the literature study of this project can be a good starting point in further enhancements of Goal Oriented Requirement Engineering (GORE) techniques.

Fifth, prototype application for indirect influence relations on goal models will be realized based on the algorithms with formalized relations. The prototype will be developed as an extension of BiZZdesign Architect. An application based on this prototype will enable BiZZdesign to provide automated reasoning on indirect influence relations for its client organizations. The reasoning will allow BiZZdesign Architect tool to provide better and more accurate analysis on goal change impacts. This will improve the dynamicity of the organization by predicting possible change impacts.

Sixth, the project will provide a tool support for predicting goal change impacts, thus greatly enhancing decision-making process. Stakeholders usually have limited resources that need to be invested in a way that can result the maximum possible benefit for the organization. Simulating effects of possible resource allocation scenarios (correspondingly, simulating goal change impacts) will provide the relevant data for deciding which resource should be allocated to which stakeholder goal.

Finally, an immediate consequence of the last two contributions is the enhancement of BiZZdesign Architect functionality. This will enhance its usability and eventually increase the satisfaction of BiZZdesign architect users.

1.5: Thesis Outline

This report is organized in eleven chapters and three appendixes. The major content of important chapters is discussed below:

- **Chapter 1 : Introduction**

  Chapter one contains the introduction which provides brief background information, problem statement, research questions, scope and related concepts.

- **Chapter 2 : Background of the Study**

  This chapter introduces the basic concepts that are used in the remaining parts of the report. The most important concepts covered are the need for change impact modeling, causal loops, fuzzy logic reasoning and an example case for illustrating goal change impacts.

- **Chapter 3: GORE and Formal Definition of Goal Relations**

  This section we will present the result of the literature study on topics related to goal oriented requirements engineering. The topics of the literature study are the reasons behind adopting GORE as a new technique for RE, the kind of formalization techniques employed by major GORE techniques and the type of reasoning the formalizations allow. Four goal oriented approaches are discussed in detail: KAOS, NFR, i* and TROPOS.

- **Chapter 4: Artifact Design**

  This chapter presents two vital steps in a design science methodology based research: specification of the requirements and architectural design of the intended system. The requirements and constraints of the system are gathered from discussions with BiZZdesign stakeholders and from the literature review presented in the previous chapter.

  The proposed solution structure and its alignment with the existing BiZZdesign tools will also be presented.

- **Chapter 5: Qualitative Reasoning**

  The first reasoning approach, which is based on TROPOS software development methodology, is discussed in detail. Adapted algorithm, prototype application and result of test case study are presented.

- **Chapter 6: Quantitative Reasoning**

  The second reasoning approach, which is based on fuzzy quantitative reasoning engine, is presented in this section. The rules in the fuzzy reasoning engine were adapted from the NFR framework. Just like the previous chapter, adapted algorithm, prototype application and result of test case study is presented.

- **Chapter 7: Comparison of Qualitative and Quantitative Reasoning**

  This chapter compares the results of the two reasoning approaches discussed in chapters six and seven.

- **Chapter 8: Reasoning on Feedback Loops of Goal Models**

  Feedback loops of influence relations in goal models will be explained here. An algorithm and a prototype application for simulating feedback loops is also presented.

- **Chapter 9: Conclusion and Future Work**

  This chapter summarizes the results of the whole project and discusses how these results answered the research questions posed in the beginning. Based on the results, possible recommendations to be carried out as a future research will also be proposed.

2. Background of the study

This chapter introduces the basic concepts to be used in the remaining parts of the report. The most important items to be introduced are the need of analyzing change impacts, causal loops, fuzzy logic reasoning and an example goal model for illustrating upcoming "goal-oriented" sections. This chapter also presents an introduction to some of the formalization techniques employed in GORE activities.

2.1: Why Care about Changes and Their Impacts?

As the old saying goes, change is the only thing that does not change. A peculiar characteristic of the Information Age is the continuous change in our personal, cultural, environmental, political, economical, technical environment. Business processes and their environments are by no means different. Organizations are always under continuous change as a result of policy, strategy, financial, resource, marketing, and other factors. In short, the ability to predict and cope with the inevitable changes is becoming the crucial factor for determining the success of any organization.

Along with the increased dynamicity of business process, the necessity of adaptable systems is also increasing as information systems are migrating from business process supporters to business process enablers [22]. But an adaptable system development is not a one night task. In order to ensure, the required adaptability, every step and artifact of a system development process, starting from the top level goal identification to the realization of the system components should be designed keeping in mind the possibility of change at any time during and after the development process.

But before trying to develop adaptable systems and business processes, anticipating and modeling the possible change scenarios (evolution requirements) is an important prerequisite step. And whenever a change occurs, its influence on other entities of the organization should be identified and modeled. For these and related reasons, the change case, associated scenarios and their impacts should be studied as well.

Change is not something that exists by itself; it is transition of a certain entity from one state into another due to a certain cause usually called an event and it results an outcome that can be referred as an action. Putting it in another way a change in certain entity will always have an effect on some other entity in its environment.

There are a number of ways of representing this cause - effect relationships including casual loops and Ishikawa diagrams. Casual loops are prominent in change impact modeling while Ishikawa diagrams are common in product design, defect and quality modeling. Additionally since BiZZdesign Architect goal and requirement modeling feature resembles casual diagrams; we are going to use causal loops for modeling goals and analyzing change impacts.

2.2: Causal Diagrams

Causal diagrams show the influence of one entity (influencer entity) on a second entity (affected entity). The influencer entity will be connected to the affected entity by using arrows with heads pointing in the direction of the influence. Generally two types of influences are common, positive influence and negative influence.

In a positive influence, the change in the influencer entity is directly proportional to the change in the affected entity. For instance an increase in the strength of encryption algorithm of customers account will increase the security of the account security data while a weaker encryption algorithm results a lower security standard.

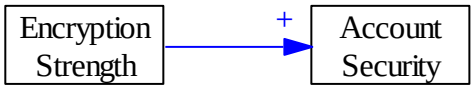

Figure 2.1a: a positive causal effect

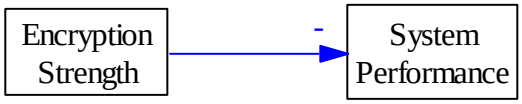

Figure 2.1b: A negative causal effect

On the other hand, in a negative influence relation, a change in the influencer is inversely proportional to a change in the affected entity. Considering figure 2.1b above, sophisticated encryption algorithms might cause delay in accessing accounts thereby decreasing system performance while employing naïve encryption algorithms may result in faster input output operations.

These relations hold if and only if third entities are kept constant or their effects are ignored. For instance, though a sophisticated encryption algorithm is used, response time may remain constant if a faster machine is used. Hence the above equations will hold if every other entity in the system environment except A and B are ignored or kept constant.

It is also possible to represent these influence relations by using mathematical equations. Assume two entities 'A' and 'B' are related by an influence relation where 'A' is the influencer and 'B' is the affected, then if:

$$\frac{\Delta A}{\Delta B} = \frac{A_2 - A_1}{B_2 - B_1} > 0 \ \ the \ \ influence\ type\ is\ positive$$

$$\frac{\Delta A}{\Delta B} = \frac{A_2 - A_1}{B_2 - B_1} < 0 \ \ the \ \ influence\ type\ is\ \ negative$$

As an example let us take the case in figure 2.1b shown above; let A represents the encryption strength and B represents the performance of the system. Assume we increase the encryption strength, then we will have a positive value for $\Delta A = A_2 - A_1$ due to the increase in the value of encryption strengths. Now assuming every other factor (e.g.: processor speed) remains constant, $\Delta B = B_2 - B_1$ will result a negative value since the performance measure of the system decreases as a result of the increase in the encryption strength. From the ratio of ΔA to ΔB, we will have then a negative value indicating a negative influence type.

Just like the effect of change in entity 'A' can have an effect on B, the resulting changes in 'B' can also have an effect on a third entity (entity 'C') or even back on the original entity (entity 'A') which causes the change. These change interactions, also called feedbacks are the sources of dynamics in any complex system. And every dynamics in all kinds of systems is as a result of two types of feedback loops; positive (or self-reinforcing) and negative (or self-correcting) loops [6]. As their names imply positive loops tend to intensify the systems change while the negative loops tend to oppose the change thereby attempting to counteract the change.

Important loops in casual diagrams are highlighted by a loop identifier using "R" for reinforcing or positive loops and "B" for negative or balancing loops. In figure 2.2 left, In order to increase the reliability of a system to be developed, the formal verification efforts should be increased. And an increase on formal verification will enhance the reliability of the system to be developed there by forming a positive(R) loop. But so much formal verification approach may escalate testing costs which can force developers to decrease the formal verification effort resulting in a negative (B) loop formation.

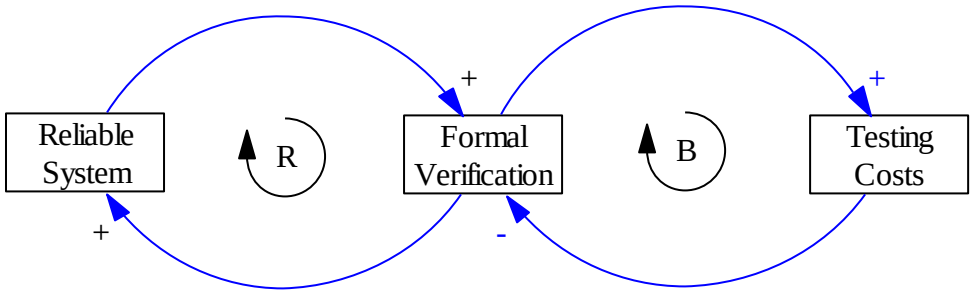

Figure 2.2: Positive and negative feedback loops

2.3: Goal Change Impacts: an Introduction

In previous subsection, dynamicity of business processes, the needs in adaptability and techniques of modeling change impacts have been discussed. But the reason behind dynamicity of business processes and IT systems were not discussed. We will discuss this reason now and it is going to be short and precise:

There can be numerous causes of changes in business environments. But these changes will be relevant to a system under study if and only if they have a possibility of affecting some stakeholder goal. And these are the types of changes that will determine the dynamics of the business environments.

Putting it in other words, though changes occur frequently, only those changes that are "relevant" to the system stakeholders are influential in governing the business dynamics. A point to remember is that "relevant" does not necessarily mean beneficial to stakeholders. Changes can be constructive where users will encourage them to happen all the time and at the same time changes can be destructive where appropriate remedial are necessary to cope with them.

When these "relevant" changes occur, some of the stakeholders' goal will be affected either positively or negatively. The affected goal will then affect goal directly related to them and the newly affected goals will affect goals elated to them and the change impact will continue until the change effect reaches the root goal of the goal tree. Sometimes there can be cyclic goal influence loops like the one shown in figure 2.2. A change in one of the goals of a goal loop diagram will then circulate for longer period of time until all the goals in the system reach common satisfaction level.

Adaptability to these "relevant" changes is then the means to maintain the adaptability of the entire system, i.e. failure to manage changes in stakeholders' goals and requirements poses significant risks to the adaptability of business environments. Consequently, designing of adaptable systems and management of dynamic stakeholder goals are two inseparable processes.

2.3.1: Modeling Goal Change Impacts

The notations used in modeling influence relations can be extended in modeling influences between stakeholder's goals. In order to do so, causal diagrams can be extended with AND/OR decomposition notations and goal conflict representations to model goal change impacts.

To analyze goal change impacts, the influence relations need to have formal definitions that allow reasoning on indirect influence relations among goals. Ultimately, the result of goal change impact analysis will be used as an input in decision making processes and in developing more adaptable systems. Figure 2.3 below shows an example goal model with causal arrows and AND/OR decompositions. The light box represents "OR" decomposition while the dark box represents "AND" decomposition. The formal definitions of goal relations will be covered in detail in the upcoming subsections.

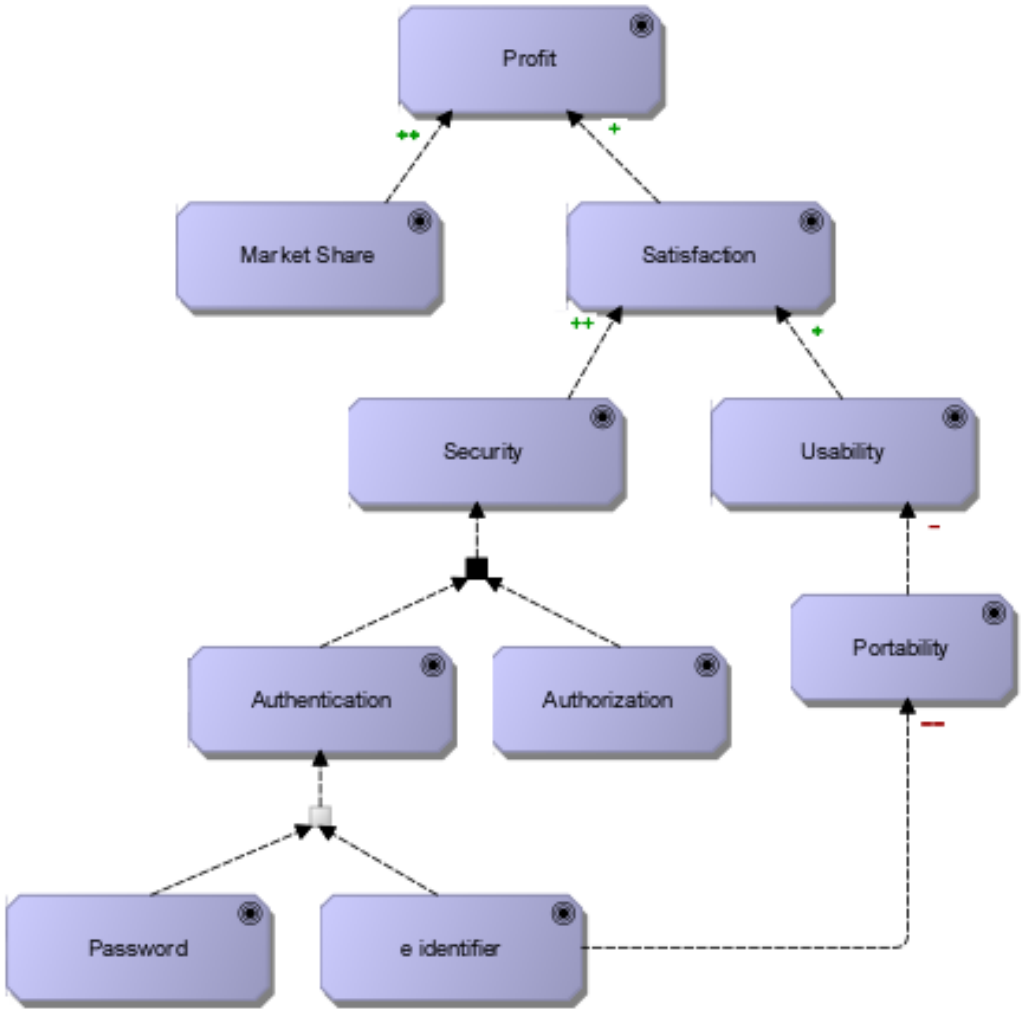

Figure 2.3: Sample Goal model from BiZZdesign Architect.

2.4: Formal Specification - A Quick Recap

Though the names might seem quite new, formal specifications and languages have been around probably as old as computers; And with or without knowing it, we are using formal languages when we write snippets of codes in Java or C++, when we write first order logic predicates etc.

Just like any other language, a formal language is a set of expressions but of course with some tight constraints. A language is formal if and only if it has well defined syntax and semantic grammar. Programming languages like Java or C++ are formal languages while human languages like English and Dutch are not formal language since expressions on these languages can have vague interpretations.

In system analysis activities, formal languages can be used to write formal specifications. These formal expressions will be then the necessary ingredients for clear, precise and automated analysis and reasoning on system properties.

A formal expression is "an expression, in some formal language and at some level of abstraction, of a collection of properties some system should satisfy"[23]. As an example, the textual (and of course the non formal) goal "The elevator shall deliver a passenger to a specific floor" using temporal logic can be defined formally as:

$$\forall p: p.buttonPressed(x) \Rightarrow \Diamond (p.Arrive(x) \wedge OpenDoor(x))$$

Although these kinds of expressions are more abstract and less appealing than their human language form, they provide formal and unambiguous specifications for automated analysis and reasoning processes [23].

Other uses of formal specifications include ability for inferring consequences, generation of concrete scenarios and counter examples, to check and simulate consistency and completeness etc. For the details of this and many more benefits of formal specifications [23] can be refereed.

2.4.1 Formalization and GORE

Though the degree and type of employed formalization may vary, majority of the current GORE techniques employ some sort of formalization in one or more GORE activities like goal elicitation, elaboration, analysis and specification. In fact the literature study conducted by Kavakali et.al identifies that out of the 15 goal oriented approaches identified in a literature study, 14 of them uses one or more formal definitions in at least one of the sub activities of GORE process [24].

To give an example of formalization usage on well known GORE approaches KAOS uses temporal logic for specifying and elaborating goals[7] and a probabilistic based reasoning to reason on partial satisfaction of soft goals [16], i* uses formalized intentional concepts from artificial intelligence [15], [25] in specifying the ability of agents to achieve goals, NFR employs qualitative reasoning for goal contribution analysis [8] and TROPOS uses the i* concepts in first order logic format to reason about requirement analysis [26].

As noted earlier, these GORE techniques employ formalizations for providing more precise meanings to the constructs used in modeling the goals, requirements, agents, various types of relations etc [15], [16], [23].

2.5: Fuzzy Sets: an Introduction

Members of conventional sets like the set of operating systems, the set of Android phones etc. can have only two possible values: Either they are not a member or members of the given set, just like we use 0 and 1 in the digital world. There is no concept of partial membership in conventional crisp sets. For instance a phone is an android based system or not! It does not make sense to say this phone is partially android.

Crisp sets have competitive advantages in computers, since either "yes" or "no" membership values match the digital world 0's and 1's. But there are situations where entities can be a member of a given set partially. Take the set of fast computers in the world, the set of young people in a city or the set of beautiful bitches in South America etc; it is difficult if not impossible to have a consensus on the members of these kinds of sets.

There are other types of sets, called Fuzzy sets that can be applied in these kinds of ambiguous set definitions. Fuzzy sets, proposed by LA Zadeh, use the concept of membership functions to deal with these kinds of vague sets [27], [28]. The membership values will range from 0 to 1, 0 representing no membership value while 1 represents full membership. Any number between 0 and 1 can be used then to specify the degree of membership. It is worthy to mention that conventional sets can be considered as special type of fuzzy sets whose membership values are "0" and "1" only.

2.5.1: Application of Fuzzy Sets

Fuzzy sets are applicable in areas where approximates reasoning is required. Considering the fact that most of mankind reasoning is based on approximation like "if temperature is high then increase fan speed", fuzzy sets can be used in many approximate reasoning techniques [27].

Humans are experts on approximate reasoning because we can take actions based on common sense. Boolean logic based computers cannot use "common sense" reasoning because the exact meaning of "hot" temperature should be defined to them. But the definition of "hot" temperature is not something that can be agreed by everyone. Different places and even different people at the same place may have different definition for "hot" weather.

When dealing with this kind of vague ideas, different perceptions of the current situations should be taken in to consideration before deciding the desired action. Take a temperature controller as an example: if 20°C is the sensor reading, we can assume the controller will perceive this as 80% moderate temperature and 20% cold. We will see shortly, how these percentage values can be determined for each possible input in section 2.5.2.

The decision of a fuzzy logic controller will be then based on both percentage values. i.e. it increases the temperature of the thermostat (because it is 20% cold) while considering the room temperature is 80% normal. These will clap down the amount the temperature of the thermostat will be raised by a significant portion since the room temperature is 80% normal.

These kind of fuzzy logic based reasoning are found to be much more efficient, economical and yield a very smooth transition when applied to motion controllers [29]. An interesting and very successful

example of fuzzy logic controller is Sendai Subway system in Sendai, Japan. Thanks to its fuzzy logic based controller, this subway is one of the smoothest running subway systems in the world [29].

Fuzzy logic based reasoning requires few more steps than the conventional if – else based rule inference systems. The most important steps are fuzzification, Fuzzy rule application, Rule Aggregation and defuzzification processes which are shown in figure 2.4 below. Detailed discussion of each step will follow the figure.

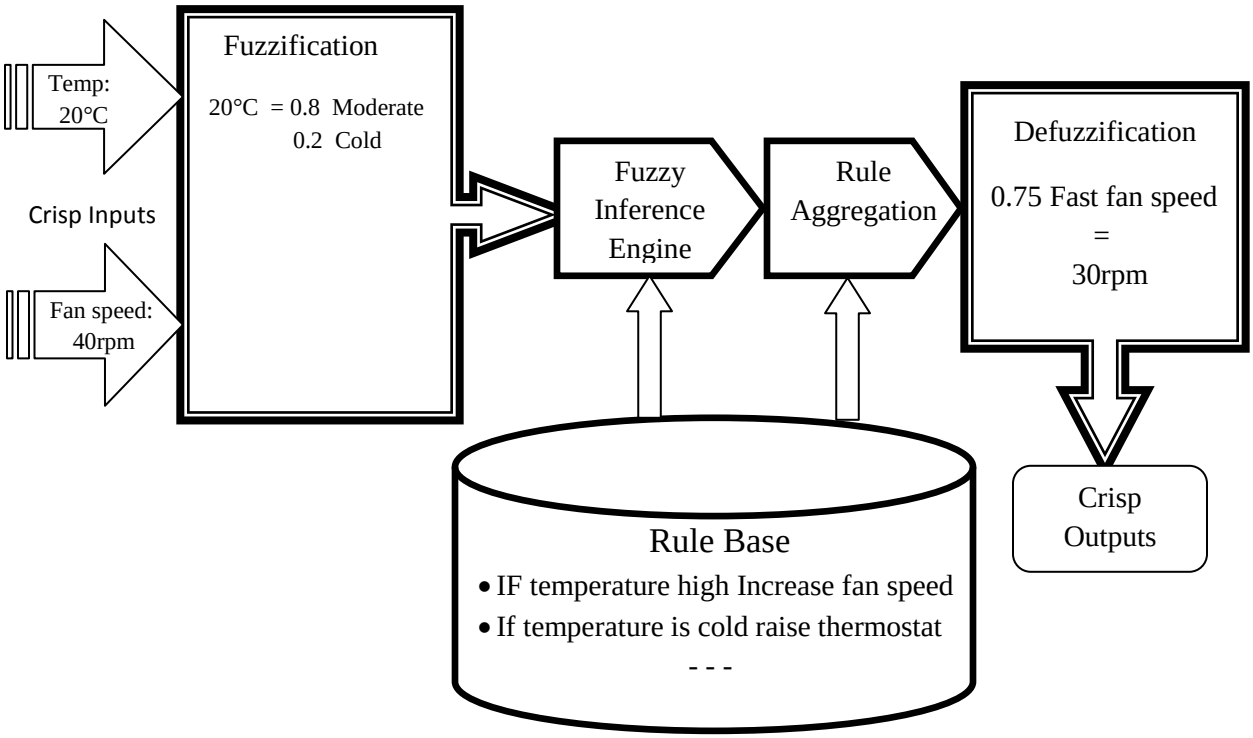

Figure 2.4: Fuzzy Logic based reasoning engine

2.5.2: Membership Functions and Fuzzification

Fuzzification is a process of changing a crisp input to a membership values. As noted earlier some vague concepts may belong to two sets. To what extent these concepts belongs to available fuzzy sets will be determined by membership functions and the process of determining the membership values is the fuzzification process[27], [30]. Assigning a membership value of 20% cold and 80% moderate values for the temperature in the previous section was an example of fuzzification process. To fuzzify a given input value, we need to define membership functions. There are a number of predefined and customizable membership functions like Triangular, Trapezoidal, Gaussian etc [30] as shown in figure 2.5 a, b and c. Looking at the pictures in figure 2.4 a, b and c, it is obvious that the these sets get their names from their respective shapes.

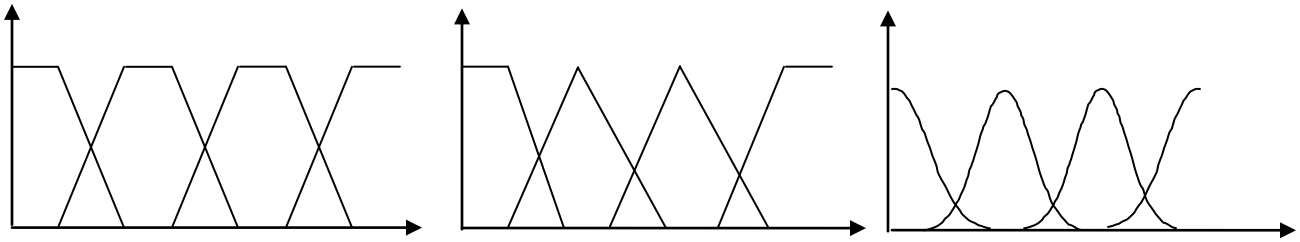

Figure 2.5 Trapezoidal, Triangular and Gaussian fuzzy sets.

The complete discussion of each of these fuzzy sets is out of the scope of this project. Interested reader can refer [30] for more information. But we'll elaborate more trapezoidal fuzzy sets here since it will be used extensively in the quantitative reasoning section (chapter seven) of this report. The reason why we choose trapezoidal fuzzy sets for our approach will be discussed in chapter seven.

A trapezoid, as shown in figure 2.6 has four vertices that can be used to specify and identify the trapezoid.

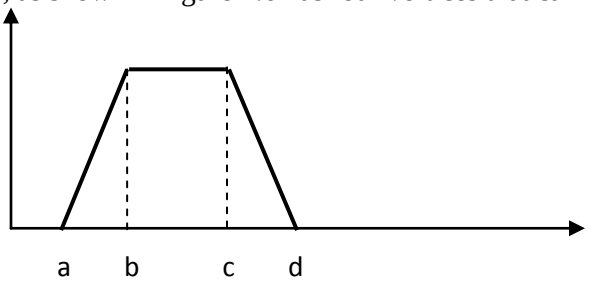

Figure 2.6 A typical trapezoidal function with four vertices a, b, c and d.

From now on, we refer to a trapezoid by its four vertices. Using these four vertices and standard linear equation formula (y = mx + b), the equations of the lines that make the trapezoidal sets can be easily calculated. And from that equation, the membership values can be easily found for any given crisp input.

The following example will clarify the fuzzy set and membership values concept discussed so far.

Let us take the concept of temperature controller. A given temperature can be "Freezing", "Cold", "Moderate", "Warm" or "Hot" which we will take them as fuzzy sets. The question will be then; given a temperature of 12°C, to which fuzzy set will it belong to? It can belong to cold temperature in tropical regions and moderate temperature in Scandinavian counties! If we are using the conventional crisp set concept, we would be forced to choose either cold or moderate temperature but not both!

But if a fuzzy set approach is used, a temperature of 12°C can be a member of two sets (Moderate and Cold) with different degree of membership. Let's use standard linear line equations to determine these membership degree values. Diagrammatical representations of the two possible membership values are shown in Figure 2.7.

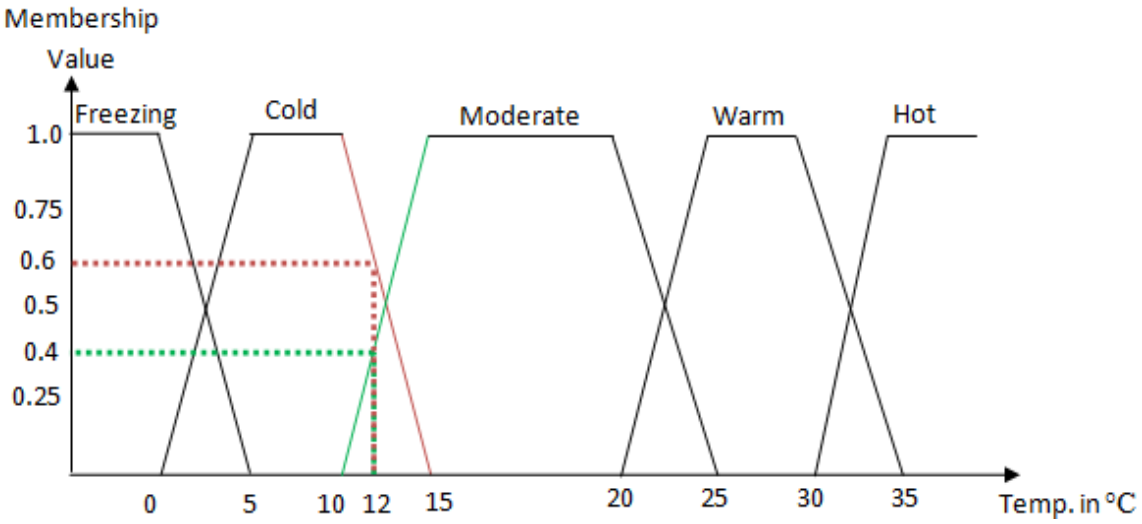

Figure 2.7: A single temperature value belonging to two different fuzzy sets.

The values 0.4 and 0.6 are derived using linear equations as follow. From the ranges specified in figure 2.6, an input of 12 °C belongs to a region where both cold and moderate temperatures are defined.

Using points (10, 0) and (15, 1) to find the linear equation of the diagonal line in Moderate region (Green line), we will have:

m (Slope) = $\frac{Y2-y1}{x2-x1} = \frac{1-0}{15-10} = \frac{1}{5} = 0.2$

Then the equation of the line, using linear equation formula, will be:

$y = mx+b$

$y = 0.02x+ b$

To find b, we can use either of the points. Let's use Point (10, 0).

$y = 0.02x+ b$

$0 = 0.02*10 + b$

$0 = 0.2 + b$

➔ $b = -0.2$

The equation of the line will be then: $y = 0.02x – 0.2$

Using this equation, the membership value of a temperature (with input level is 12) to the fuzzy set "Moderate Temperature" as shown in figure 5.7 can be determined as:

$y_2 = 0.02x – 0.2$

$y_2 = 0.05*12 - 0.2$

$y_2 = 0.6 – 0.2$

$y_2 = 0.4.$

i.e. the temperature whose measure in °C is 12°C, belongs to "Moderate" fuzzy set with a membership value of 0.4.

The second membership value, y2, which is the membership value of a goal to the set "Cold" (the red one), can also be shown to be 0.6 using a similar approach.

2.5.4 Fuzzy Inference Engine and Rule Aggregation:

The fuzzy inference engine applies the rules stored in the rule base on the membership values obtained in the fuzzification step. The rules to be applied are specified in terms of "if-else" statements that will be invoked based on the input membership values. Taking temperature controller as an example the following are candidate rules for the rule base.

i. *If temperature is cold Turn on heater.*
ii. *If temperature is Warm AND fan speed is low fan speed should be medium.*
iii. *If temperature is moderate AND fan speed is average no change to speed of the fan.*

There can be rules that can be formed from combination of two or more rules (e.g. Rule ii and iii). In this kind of situations, the "AND" combination takes the minimum of the individual rules. The minimum result will be used as a representant of the AND combination [30]. This kind of reasoning is also used by Mamdani inference engine system which utilizes the Max –min inference system [31].

Consider figure 2.8, next page there are two preconditions, namely "*If temperature is Warm*" and "*IF fan speed is low*". One of the membership values of these preconditions is 0.6 and 0.75 respectively. According to the AND propagation rule of fuzzy logic, the minimum value of the two will be the aggregated result of both values.

Minimum (0.6, 0.75) = 0.6

If we assume that the three candidate rules are applicable in this case, then Rule ii, will be selected by the fuzzy reasoning engine since it matches the preconditions. The result of applying this rule will belong to the fuzzy set medium according to rule ii and its membership value will be the minimum of 0.6 and 0.75 which will be 0.6.

In real world applications, there is a high probability of two or more rule need to be applied at the same time. In this kind of situations, rule aggregation is employed. There are different kinds of rule aggregation techniques. The commonly used one is taking the average, ordered weighted averaging operations, or the minimum of the individual results [30].

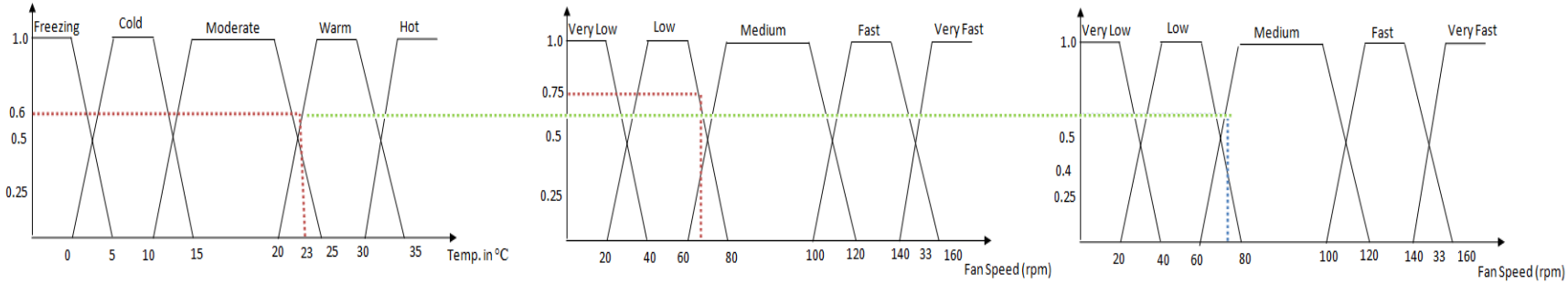

Figure2.8: Rule aggregation in fuzzy inference engine

2.5.5 Fuzzy Reasoning Output - Defuzzification

The output of the fuzzy rule engine we saw earlier is a membership value to a certain fuzzy set. For instance, in the previous example our output was set the fan speed to medium with a membership value of 0.6. But almost all of today's machine s accepts only crisp commands like "set the revolution of a motor to 65 rpm", or "set the temperature to 25 °C". Hence, membership values and fuzzy set outputs we saw in the previous section have limited use in these machines. This is why we need to change these kinds of fuzzy membership values to crisp outputs to be applied in real machines. The process of changing the membership values to crisp outputs is called defuzzification.

There are a number of defuzzification techniques. Some of the most commonly used ones are "Max membership principle", "Centroid method", "Weighted average method" and "Mean max membership" [30]. Of these four defuzzification methods, "Centroid method" is most prevalent and physically appealing technique and "Weighted average method" is the most frequent, efficient and reasonably accurate technique [30].

Since we are going to use "Weighted average method" later in section 7 of this report, we will elaborate it more here. It is formed by weighting each membership function in the output by its respective maximum membership value.

Mathematically this is given as:

Crisp Output = $\frac{\sum ai*Mai}{\sum Mai}$

Where ai is the average value of i^{th} fuzzy set and M_{ai} is the membership value of fuzzy set a_i.

Take the picture in Figure 2.9 as an example:

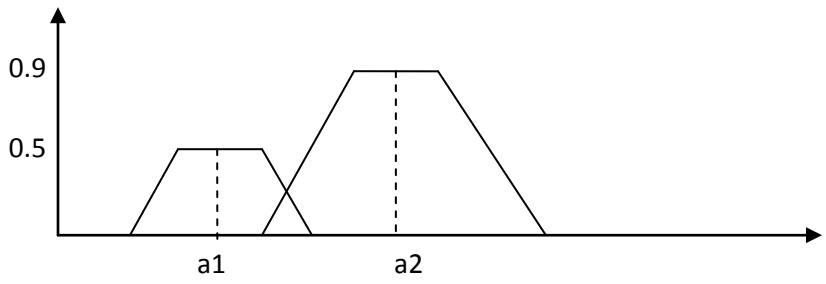

Figure 2.9: Weighted average method to defuzzify two fuzzy sets

Crisp Output = $\frac{a1(0.5)+b2(0.9)}{0.5+0.9}$

2.6: Summary

To maintain adaptability in today's dynamic business environments, analyzing and predicting the impact of various change scenarios is essential. Causal diagrams are one of the tools used in modeling the dynamic behavior of systems. They use arrows labeled with notations like ++ and – to represent positive and negative influences among system components. The dynamic behavior of goals can also be represented using causal notations. More specifically, a change in the satisfaction level of a goal on other goals of a system can be represented using causal diagrams and AND/OR goal refinements.

To automate analysis and reasoning on goal influences, formal definition of the goal-influence relations is required. First order logic, temporal logic and fuzzy logic are few of the formalization techniques used in defining goal to goal influence relations and satisfaction levels.

Majority of (business) goals are soft goals whose satisfaction level is ambiguous to specify precisely. To reason on these satisfaction levels, fuzzy logic based inference engines can be used. A fuzzy inference engine uses approximate reasoning techniques that resemble human behavior. To do so it uses a special type of sets called fuzzy sets.

In conventional crisp sets, an entity is a member of a certain set or not. But in fuzzy set, an entity membership is assigned a value from 0 to 1. A membership value of 0 means it is not a member of the set while 1 means it is a full member of the set. Any value between 0 and 1 will represent partial membership to the given set. The rule bases of the inference engine will then use this membership values to perform the intended approximate reasoning.

3: GORE and Formal Definition of Goal Relations

This chapter presents the result of the related literature study on topics related to goal-oriented requirements engineering. The main focus of the literature study is exploration of current GORE techniques, the kind of formalizations employed by this GORE techniques and the type of reasoning the formalizations allows. But before presenting the result of the literature review, a discussion on why the common object oriented RE approaches are problematic and how the emerging GORE approaches can treat the problems will be presented first.

This chapter also explains four popular goal oriented approaches in detail along with an overview on the type of formalization they support. The four goal oriented approaches to be discussed are: KAOS, NFR, i* and TROPOS. (Almost exhaustive overview of current GORE methods is shown in [24]). In addition to the popular GORE approaches this chapter also covers some of the emerging approaches in requirement and goal change management.

3.1: Why Do We Need a New Approach to RE?

Along with the increase in popularity of object orientation 1980's, object oriented analysis and requirements specifications have been gaining wide acceptance. These approaches were somehow good for some period of time. But their limitations in modeling early phases RE activities make them inadequate for today's highly dynamic and complex systems [32]. Various researchers have published a number of reasons why object oriented RE activities are problematic. Some of the limitations as summarized in [22], [32] include:

- Limited Scope: Only specifying the software alone. Leaving alone the composite system or the environment of the system. And inability to specify nonfunctional requirements.
- Lack of rationale capture: limitations to specify the motive behind the presence of the requirements
- Lack of understandability of the system domain
- Poor Guidance: OORE was entirely focusing on analysis after requirement elicitation. Constructive and incremental methods for building correct model specifications for complex systems in a systematic way doesn't exist to adequate extent.
- Change resistance and lack of traceability to business objectives
- Lack of support for exploration of alternatives: Interchangeability of requirement – software component is not allowed.

These problems are prevalent in traditional object oriented RE approaches because they are concerned with later phases of RE activities [22]. Late RE activities focus on providing a specification document to the system developers ignoring the intentions behind the system development process. i.e., no attention has been given to the "why" questions behind the system development process. Ignoring the why questions is resulting in ignorance of the stakeholder goals behind the system to be developed. A system developed without stakeholders' goal in mind will face difficulties in satisfying the intended objectives and users interests [3].

This will lead us to one conclusion: current RE practices, mostly based on object oriented analysis approaches, are not adequate enough in delivering products that can satisfy stakeholders goals! It is the time to move to another RE methodology that gives a good emphasis on goals of stakeholders.

3.2: Shifting form Object Orientation to Goal Orientation

GORE is an emerging and stakeholder goal driven approach to Requirements Engineering. Different scholars give different but related definitions to goals. Lamsweerde defines a stakeholder goal as "an objective the system under consideration should achieve" [7] while Yu defines a goal as "a condition or state of affairs in the world that the actor would like to achieve" [15]. But these and other goals definitions share one common thing: A goal is a state desired to be achieved by a certain stakeholder which explains the intentions behind developing a system.

Goals can be characterized by goal attributes like name, priority, description etc. They can also be expressed by links with other goals or with other requirement design elements [7]. Furthermore goals can be classified based on various criteria like functional vs. non-functional goal, soft goal vs. hard goal, Achieve vs. maintain vs. optimize goals etc.[32]. Our main focus is on influence relations among goals where soft goals relations are much more important than hard goal relations. Influence relations in hard goal are less important because hard goals satisfaction levels are discrete, either fully satisfied or fully denied.

GORE, in addition to treating traditional RE problems, brings a number of reasons to incorporate goals in to the RE process. The majority of these reasons have been identified in [7] which includes support in identification of requirements, assurance of requirement completeness, avoiding of irrelevant requirements, providing rationale of requirements and management of requirement conflicts.

Moreover, focusing on early phases of RE activities will result in understanding the "why" behind the system requirements, i.e. Having an answer to the "why" questions will not only helps in developing successful systems but also in facilitating the development of cooperation with other systems [22].

The majority of GORE techniques take identification of the stakeholders' goals as the first step in the RE process. Stakeholders are likely to know at least some of the goals behind the system to be developed. In fact if the current business process and its problem have been studied and documented well, goal identification can be a smooth process. But sometimes the reverse can happen. Goals are implicitly known by the stakeholders forcing system developers to perform goal elicitation activities [32].

Classic goal elicitation starts form analysis of the current system followed by refinement and abstraction through "Why" and "How" questions on the already identified goals [32]. Once the necessary goals are identified, goal modeling is performed for providing an input for a better analysis and support for formal reasoning approaches to be carried out in the requirement engineering process [32].

Goals can be specified in a number of ways to support requirements elicitation, elaboration, verification/validation, conflict management, negotiation and evolution. Correspondingly, current GORE approaches can be categorized based on the requirement engineering activity they primarily support. Table 3.1 adapted from [33] shows the current Goal oriented approaches mapped to the common RE activities.

RE Activity	Goal Analysis Contribution	Goal-oriented Approach
Requirement Elicitation	1. Understanding current organizational situation 2. Understanding the need for change	GOMS, **i*,** EKD, TROPOS ISAC, TROPOS, F[3]
Requirement Negotiation	3. Providing the deliberation context of the RE process	SIBYL, REMAP, Reasoning loop
Requirement Specification	4. Relating business goals to functional and non functional system components	KAOS, GBRAM , NFR
Requirement Validation	5. Validating systems specifications against stakeholders goals	GSN, GQM

Table 3.1: Goal analysis approaches mapped to RE activities

The detail description of each of the above goal oriented approaches is out of the scope of this project. But [34] identifies that NFR, i*, KAOS and GBRAM are the main GORE approaches. Hence explanation of these approaches will be presented here except GBRAM. GBRAM was skipped because we observe it has very limited support in analyzing goal contribution relations. In addition to these three approaches, we incorporate TROPOS software development methodology since we find it applicable for goal contribution analysis. For more information on the other approaches [33] can be refereed.

3.4 NFR Framework

NFR Framework, standing for Non-Functional Requirements framework, is a popular framework in goal and requirements modeling. NFR goal analysis starts first by drawing the dependency relations between goals of a system in a tree like graphical notation called software interdependency graphs (SIG) [8], [9]. A SIG model start with a top level system goal and refines this top level goals in to more concrete levels at each refinement level.

In SIG models, business goals like customer satisfaction and IT system goals like security, reliability, performance, usability etc. are more or less abstracted goals and are located in the upper areas of the goal decomposition tree. These goals, by their nature, are difficult (if not impossible) to precisely quantify and measure their achievement levels [35]. This demands the usage of "partial satisfaction" or "range of satisfaction level" concepts in goal refinements and contribution analysis. The primary reason behind the presence of partially satisfied goal concept is the limitation of available resources, inevitability of failures, presence of conflicting goals etc. [16].

The refinements are guided by already defined goal decomposition patterns [24]. In these decompositions goal to goal interdependencies depict the interrelationships among goals of a system showing the refinement of soft goals into offspring soft Goals and the contribution of the offspring goals towards the satisfaction of parent goals. The notations used in SIG are similar to casual loops we saw in chapter two where goal nodes can be interconnected by interdependency links with arrowheads and associated labels representing the degree to which a goal is achieved [8]. Figure 3.1 shows a simple decomposition of "customer satisfaction" goal via software interdependency graphs.

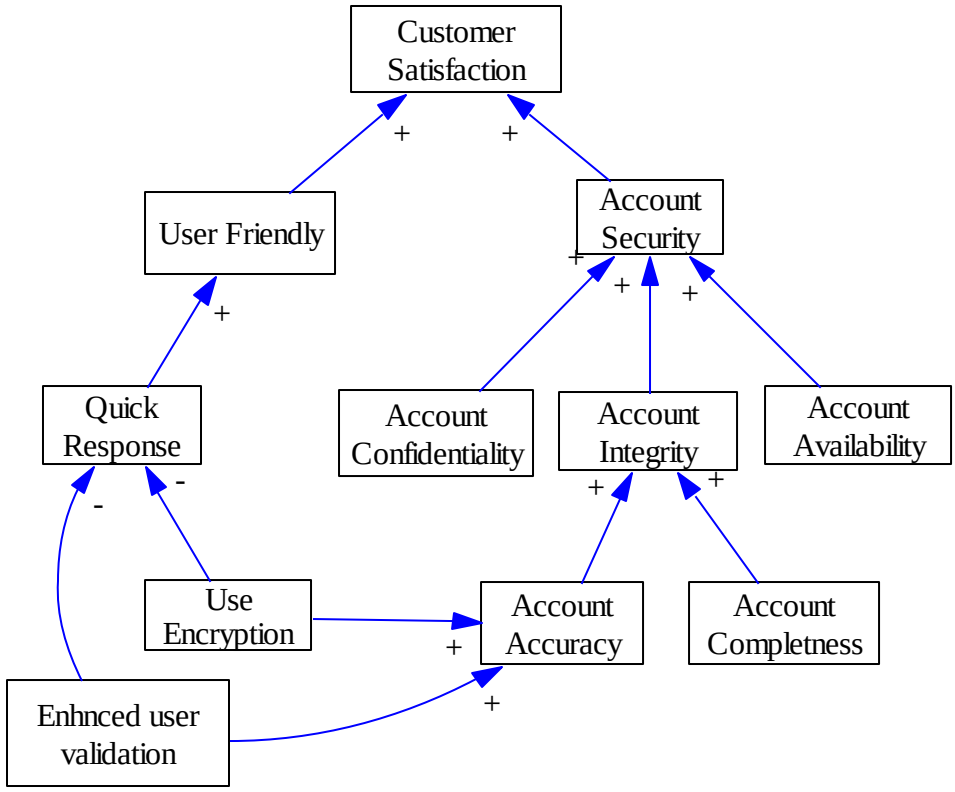

Figure 3.1: Sample Software Interdependency Graph (SIG).

3.4.1 Goal Contribution Types in NFR Framework

Influence between a sub goals and a super goal is not the only type of interdependency that exists between the goals of a system. A goal, in addition to its sub goals, can also be affected both positively and negatively by other goals of the system [8], [15], [16], [26]. However, identifying the sign of the contribution is not enough. The offspring can contribute a little, partial or to a great extent in both positive and negative ways. In fact the NFR framework in [8] identifies seven different types of contribution types to be used in qualitative reasoning: BREAK,HELP, HURT, SOME+, SOME- and Unknown.

In SIG, a system goal can be affected by a number of children goals in any of the seven types of contributions types discussed above. The ultimate effect of these contributions to a goal is expressed in NFR framework in four satisfaction levels: Satisfied, denied, satisfiable and deniable [8]. The first two is used when there is a clear understanding whether the goal is achieved or not. And to express the presence of some level of evidence for goal achievement levels, the notion of "satisfiable" and "deniable" are used.

3.4.2: Formal Definition of Contribution types

NFR framework does not employ any kind of quantitative formalizations on goal contribution types. It simply uses the qualitative reasoning (based on the notion of ++ and --) to reason out the goal contribution effects. But in order to give guidelines for requirement engineers on how to use the frame work, NFR gives formal but textual definitions for contributions. The details of theses definitions are not

relevant here. But we'll show the "OR" and "Makes" relations as an illustration in table 3.2. Interested reader can refer [8]for further details.

OR ({Offspring}) SATISFICE parent	Offspring Makes Parent:
If any of the offspring is satisfied and When the interdependency is self satisfied Then the parent is self satisfiable	If the offspring is satisfied and When the interdependency is self satisfied Then the parent is self satisfiable

Table 3.2: Sample contribution relations from the NFR framework [8].

3.4.3: NFR Influence Decision Procedure

A goal can be AND/OR refined and there can be a number of contribution relations. Identification of these relations alone is not quite enough; we need to determine the extent to which a certain goal is satisfied or denied by its offspring. NFR employs label assignments based on qualitative reasoning for contribution links. This allows determination of the satisfaction levels of soft goals.

The initial values of the labels can be obtained from the decisions made to accept or reject alternatives available usually in the leaf level goals of the SIG. With the help of satisfiable and deniable notations from the previous section, a goal or interdependency graph is labeled as Satisfied(S), Denied (D), Conflicting (C), Undetermined (U), Weak Positive (W+) or Weak Negative (W-)

Using these notations, whenever a goal or interdependency graph is assigned a new label, the decision procedure is activated and propagates labels from offspring to a parent. The decision procedure will have two important steps: individual impact analysis and grouping of interdependence impacts.

The individual impact analysis determines the individual impact of an offspring contribution towards a parent for each contribution type discussed in the previous section while the grouping step will sum up the individual impacts into a bag (not a set) and combine them so that a single representant label is created. These two steps are not simple tasks and have a number of sub steps. An interested reader can refer again [8] for more details.

3.5: KAOS

KAOS, standing for Keep All Objects Satisfied, is a RE methodology that provides a specification language for capturing why, who and when aspects of requirements in addition to the answers to the common what questions addressed by object oriented analysis approaches [36]. KAOS based approach to RE activities encompasses five basic steps: Identifying relevant goals from preliminary documents, goal elaboration by asking "how" and "why" questions, object and agent modeling, operationalization of goals. Moreover conflict and obstacles are handled in parallel with these five steps[32].

The KAOS approach to requirements engineering involves three different levels of modeling: the meta-level, the domain-level, and the instance level [37]. The ***meta-model*** provides domain-independent abstractions in terms of meta-concepts like goals, agents, relationships and meta-relationships such as Refinement between Goals. The ***domain model*** is composed of domain-specific instances of meta-

concepts and meta-relationships and the ***instance model*** refers to specific instances of domain-level concepts [37]. Figure 3.1 shows the basic components of these diagrams.

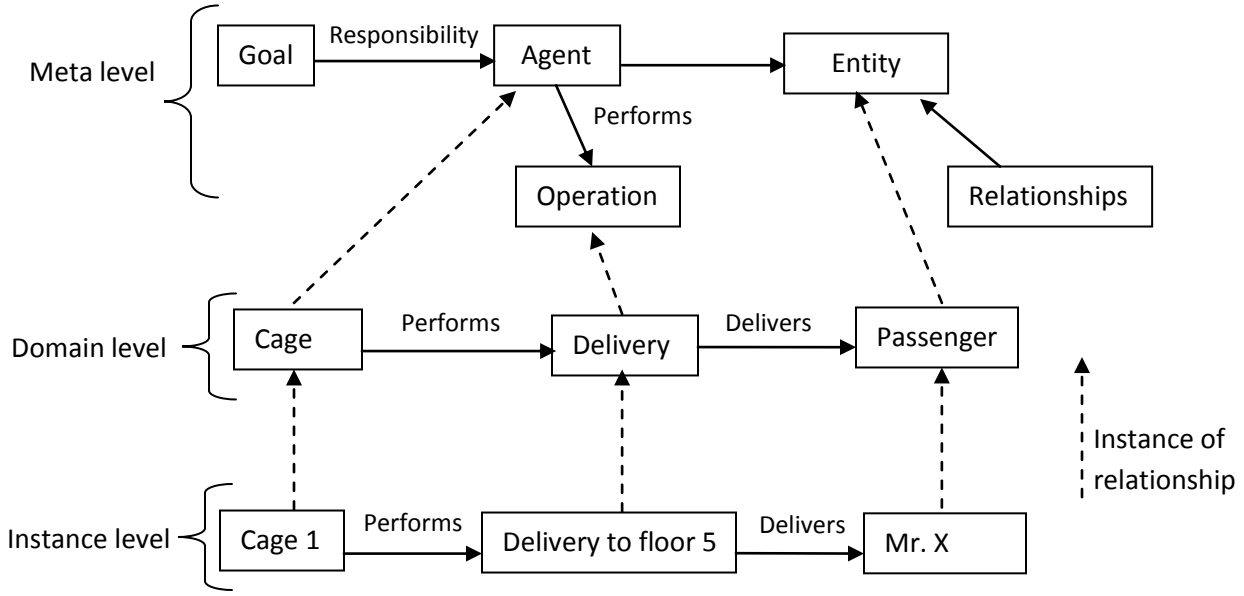

Figure 3.2: Three levels of KAOS models

The central concept in KAOS is the goal concept which can be represented in a natural language and in an optional semiformal real-time temporal logic. As an example, consider the goal "Achieve delivery of a passenger to floor x" for an elevator can be expressed as:

Goal Achieve [Deliver Passenger]
Definition Each passenger should be delivered to the floor of his choice.
FormalDef $\forall p$: p.buttonPressed(x) $\Rightarrow \Diamond$ (p.Arrive(x) $\wedge$ OpenDoor(x))

Where the symbol ◊(x) is a temporal logic property stating x will occur after some time t. there are many of course many more temporal properties like: ♦ (sometime in the past), ■ (always in the past) , □ (always in the future). Interested reader can read about these and other details of formalizations in [38]. To assist requirement engineers in eliciting goal and model them in goal models, KAOS also employs well defined and proved goal decomposition patterns that can help in decomposing goals in to a fine grained goal that can be assigned to an agent [39].

3.5.1: Reasoning about Goal Influences in KAOS

The original KAOS framework has a very limited power in supporting the reasoning of goal contribution relations and in providing support for deciding better alternative resource allocations [32]. The authors of KAOS approach realize this limitation and came up with a probabilistic density function based extension to KAOS. The extension enables specifying partial degree of goal satisfiability and contribution relations [16].

This extension to KAOS models is based on quantitative approaches for goal contribution types. Contrary the subjective criteria employed in the NFR, it is based on objective criteria [16]. An objective assessment

based reasoning uses domain specific criteria like the no. of security breaches per unit time for measuring security or the average time between failures in a given time for measuring reliability.

To specify goal satisfaction level, the aspect of the goal that is under change (quality variable) and the corresponding objective functions should be clearly determined. As an example, the goal "Increase reliability of a system" can use the number of failures, time to repair failures or average time between failures as quality criteria.

Once the quality criteria are determined the next step will be providing the formal specification of the objective functions. Taking mean time between failures as a reliability criterion, the objective function can be defined as:

$\forall f_1, f_2$: failure records $\Rightarrow$ (f2-f1 < 24 hours)

Which states the given system should keep running continuously for a minimum of 24 hours.

Finding the probabilistic density function is the next step. This is not an easy task. For instance, let the delay due to maintenance of a system is given by the sum of two delays as shown below:

delay0 = delay1 + delay2

The probability density function pdf_0 is given by $\int_0^x pdf_{delay1(i)} * pdf_{delay2(x-i)}\, di$

The values of these types of probabilistic mathematical expressions will be then used for estimating the propagation effects. They can also be used for comparing effects of alternative resource allocations [16].

3.6: i*

NFR and KAOS frameworks add one more step to answer the "why" questions about RE activities by incorporating the concept of goals to the RE process. But these approaches do not distinguish early phase of RE activities to late phases of RE activities. These two phases of RE activities have different objectives and presuppositions. Hence it would be appropriate to provide different modeling and reasoning support for the very early phases of RE activities [22].

i*, where its name is referring to distributed intentionality, is a relatively new framework proposed to enable analysis and reasoning on the very early phases of RE activities [15]. i* is an agent-oriented modeling framework that can be used not only for RE, but also for business process reengineering, organizational impact analysis and software process modeling activities [34]. Being agent-oriented approach, the central concept behind i* modeling is the notion of an intentional actor which has motivations, intents and rationales behind its actions [15].

i* frame work has two main components: The Strategic Dependency (SD) model which describes a network of dependency relationships among various actors in an organizational context and Strategic Rationale(SR) model which models the reasons associated with each actor and their dependencies along with the information about how actors achieve their goals and soft goals [22].

3.6.1: Strategic Dependency (SD) Model

The SD model focuses on modeling intentional relationships among organizational actors via a set of nodes and links. The nodes represent an actor and the links representing the dependency relationships. i* identifies four types of dependency relationships between actors, namely: goal, task, resource and soft goal dependencies.

Each of these dependency types can have either of open, committed or critical dependency strengths. In order to deal with the needs of strategic actor the concepts of routines, ability and workability is introduced in i* models. For the sake of illustration, let's see the formal definition of routine:

A routine of an agent is a skeleton plan that provides a rough outlines for a specific actions to be carried out when instantiated. An agent is said to have an ability to achieve a certain goal when it has the necessary routines to accomplish that goal. Combining these two concepts we got [15].

$$A(y, \eta) \equiv \exists u(U_y(u) \wedge purpose(u, \eta))$$

i* employs this and much more formal notations to represent how actors can communicate and depend on each other to achieve their goals.

3.6.2: The Strategic Rationale (SR) Model

While the Strategic Dependency - SD model maintains a level of abstraction by modeling only the external relationships among actors, the SR model takes one step further this abstraction in order to allow a deeper understanding and reasoning about strategic actors' intentions [15]. Incorporating intentional details will enable modeling how the interaction of actors results achievements of a goal. It also helps to reason out to what extent the goal is achieved [40].

These actor specific intentional details are shown using internal intentional relationship graphical models. An SR graphical model will have goals, tasks, resources and soft goals as nodes and two main classes of links: means end links and task decomposition links. Means end links indicates the relationship between an end and a means of attaining it. A task decomposition links shows a link between a task node and its component nodes that with a way of representing how these goals can be met [22].

i * also uses + and – contribution types to identify different levels of contributions and to reason on selection of alternatives [22]. But Regardless of contribution types, satisfiability of a goal in i* is determined based on the ability of actors to achieve their goals. And the ability to achieve a goal is based on the analysis of the actor behavior as modeled in the strategic rationale model [24].

3.6.3 i* for Requirement Engineering and Goal Reasoning

Earlier we discussed that i* is a frame work intended to be used in the early phases of requirements elicitation process where the primary concerns is understanding intentionality of stakeholders. In i* based RE methodologies, the requirements engineer role should be limited to guidance of stakeholders by proposing a candidate solutions to their problems [22].

To identify the intentions of stakeholders, SD and SR models of i* can be used as a tool for understanding the organizational environment and exploring alternate system proposals. They can also be used in the analysis of the impact of alternatives system arrangements on organizational participants[15].

Relations among goals and other intentional elements in i* are modeled using NFR like influence notations like ++ and - -, means end relations, aggregations relations etc [15]. These relations have well formulated textual definitions though no formal definition is given that can help in reasoning on goal influence relations.

3.7: TROPOS

TROPOS is an i* based, agent oriented and requirements driven software development methodology [25]. Though it is an i* based methodology , it is not only concerned with early phases of system development activities; rather it deals with early requirements, late requirements, architectural design, detailed design and interface with agent programming platforms [41].

TROPOS based goal models provides two approaches to reason on goal contribution types: Qualitative and Quantitative reasoning approaches. The qualitative approaches are similar to the NFR framework except they are expressed in more mathematical way. For instance, assuming Goal G1 is AND refined in to two G2 and G3, If G2 and is fully satisfied and G3 is partially satisfied NFR tells us that G1 will be eventually partially satisfied. This reasoning, as formalized in the TROPOS approach, can be specified as:

$$(G_2, G_3)\ \overrightarrow{and}\ G_1 : (FS(G_2) \wedge PS(G_3)) \rightarrow PS(G_1)$$

It is not difficult to infer from the above expression that AND refinement propagates the minimum satisfaction of goals G2 and G3. This and other axioms listed in [26] can be used as the skeletons of the propagation algorithm TROPOS uses to reason on goal models.

The qualitative analysis of TROPOS allows “full”, “partial” and “none” level for both satisfiability and deniability levels of goals. But there can be of course a satisfiability level between full and partial levels or there can be different kinds of partial deniability levels. Putting it in other words, it is desirable to have more fine grained level of satisfiability rather than having only three levels of satisfiability for goal satisfaction analysis.

And this is exactly why TROPOS also includes quantitative analysis techniques. The quantitative approach is capable of providing more concrete and detailed analysis on goal contribution relations. This technique is based on assigning a value between 0 and 1for goal satisfaction and deniability values [26].

For validating the qualitative and quantitative approaches, TROPOS authors also compare the qualitative and quantitative approaches on industrial test cases. Though both lead to similar results, though the quantitative approach allows drawing of more precise conclusions on contribution types [41].

3.8 Other Approaches for Managing Goals and Requirements Evolutions

Evolving requirements and impact analysis is not a new concept in system development activities. A number of change impact analysis techniques have been and are being published in literature. Majority of

the popular goal change impact analysis techniques already covered in the previous section. This section we will focus more on recent trends in impact analysis techniques.

3.8.1: Change Impact Analysis Based on Formalization of Trace Relations

As the name suggests this approach employs formalization of different kinds of relations between requirements to manage evolutionary requirements. This technique uses a metamodel depicting requirement and their relation types. The metamodel is based on common requirement to requirement relation types extracted from a literature study [42].

The relations are also formally defined using first order logic based expressions. Some of these relations includes requires, refines, conflicts and contains. For each relation type adding, deleting and modification of a requirement and a relation scenarios are also considered [42].

This approach practically covers all types of requirement change scenarios. But the relations formalizations are not adequate enough to analyze impacts of changes in goal satisfaction values which are of course a vital part of this study.

Though this approach is limited in analyzing soft goal satisfaction levels, it revealed us an important change scenario we didn't observe in the previous well known approaches: the possibility of changes not only in the requirements/goals of a system but also in the relation types themselves.

3.8.2 DepRVSim: Requirement Volatility Simulation Considering Dependency Relationship

DepRVSim is a recently proposed discrete-event based simulation approach for managing evolution requirements. Compared to the system dynamics based approaches like TROPOS, the authors of DepRVSim argue that discrete-event simulation allows more detailed descriptions of activity, resource and work products and is more suitable for modeling processes [43].

A DepRVSim approach for managing requirement changes has four major components as shown in figure 3.3 below. The change event routines handles all the events generated by the event generator and updates both the requirements and software project plans accordingly.

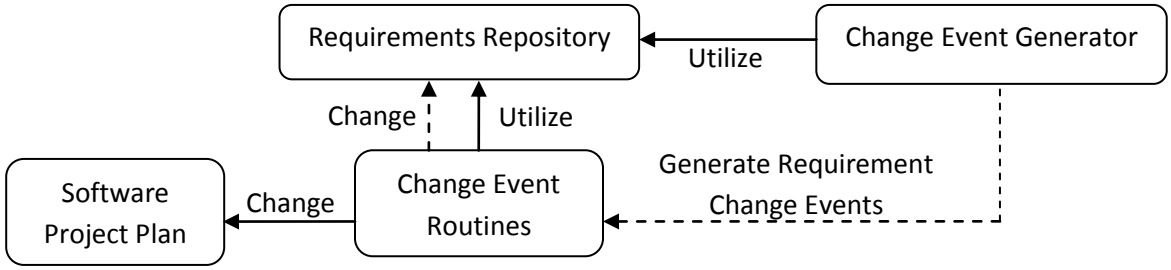

Figure 3.3: DepRVSim structure [43]

DepRVSim identifies and handles Requires, Explains, similar to, Conflicts and Influences dependency types among requirements. It also uses the terms Major, Moderate and Minor influence types for identifying different levels of influence dependencies.

Though DepRVSim is a good approach for handling the effect of evolving requirements on architectural components and other requirements, the employed contribution reasoning technique is not neither formalized nor strong enough to analyze the indirect influence relations among stakeholder goals.

3.9: Summary

Though popular for quite some time, object oriented system analysis methodology has some drawbacks. Some of these drawbacks include tendency to resist changes, limited scope and lack of rationale to capture the specified requirements. To alleviate this problem, goal oriented approaches to RE were introduced. Popular goal based approaches includes NFR framework, i*, KAOS and TROPOS.

The NFR framework is based on the concept of non functional requirements as soft goals. It uses software interdependency graphs to refine non functional requirements. The refined (soft) goal graph will be used to analyze the influence relations between goals. NFR framework influence reasoning is entirely based on qualitative notations like fully positive contribution (++), partial negative contribution (-) etc.

KAOS is a popular GORE technique that can be used to specify why, who and when aspects of requirements. The formal definitions of these requirement features are specified using temporal logic. Beside temporal logic, KAOS uses probabilistic density functions to reason on partial goal satisfaction values.

i* is a relatively new framework proposed to enable analysis and reasoning on the very early phases of RE activities. It uses strategic dependency model to specify intentional relationships among organizational actors. It also uses strategic rationale model to allow a deeper understanding and reasoning about strategic actors' intentions behind their goals and requirements. Various attributes of these models are specified using predicate logic and the contribution relations among the model components are specified using classic influence notations like ++, --.

TROPOS is an i* based, agent oriented software development methodology. Being requirements driven methodology, it can be used in early and late phases of RE activities. Additionally, it is also applicable for architectural design and detailed design phases of system development process. It also employs a qualitative reasoning technique for goal influence analysis that can be used in the early phases of the RE process.

There are also other less popular techniques like change impact analysis based on formalization of trace relations and requirement volatility based on dependency relationships which can be used to analyze goal and requirement change impacts.

The first research question of this thesis project is "*What are the formalizations of the existing goal-oriented requirements engineering approaches and what kind of reasoning do they allow?*" By presenting the available formalization techniques in current GORE approaches, this chapter addresses the first half of this research question. It also partially answers the second part of the research question by specifying the kind of influence reasoning available in each method. The answer to this research question will be finalized in chapter four by providing comparison on existing goal influence reasoning techniques.

4. Artifact Design

This chapter presents two vital steps in a design science methodology based research: specification of the system requirements and architectural design of the system under development. The requirements and constraints of the system are gathered from discussions with BiZZdesign stakeholders and from the literature review presented in the previous chapter.

One of the constraints of the desired indirect influence reasoning system is it should be developed as an extension to BiZZdesign Architect. This makes the solution structure to utilize some of the components from BiZZdesign Architect. The alignment of BiZZdesign Architect and the desired reasoning techniques will be discussed in this chapter.

The reasoning techniques encompasses two approaches for goal satisfaction specification; qualitative and quantitative. Qualitative approach uses textual goal satisfaction specifications while quantitative approaches use numeric goal satisfaction values. To select an appropriate technique for these approaches, a comparative analysis on existing GORE techniques is presented in this chapter. The comparative study reveals that TROPOS software development methodology and Fuzzy logic reasoning engines are suitable for qualitative and quantitative reasoning respectively.

4.1: Project Requirements

This section specifies the project requirements from the desired artifact perspective. The constraints imposed on the system by BiZZdesign are also discussed. The primary users of the extension are expected to be EA users, requirement engineers and decision makers.

4.1.1: Functional Requirements:

i. Use ArchiMate modeling notations.
   To enhance the usability of the system, the system shall use the standard ArchiMate goal and requirement modeling notations. These notations are specified as the motivation extension of ArchiMate 2.0.
ii. Work with BiZZdesign Architect.
   The system shall be developed as an extension of BiZZdesign Architect. This allows the system to use goal and relation attributes already designed in BiZZdesign Architect. Primarily the system shall read and update the goal satisfaction values and influence relation strengths of goal models in EA designs.
iii. Accept goal and influence relation values.
   The system shall allow users to specify goal satisfaction level change scenarios. This can be done by adding external attributes to existing goal models of ArchiMate.
iv. Accept different kinds of goal satisfaction specifications.
   The system shall allow both textual and numeric goal satisfaction values. It shall also allow textual and numeric values to influence relation strengths.
v. predict the effect of goal changes on *directly* related goals
   Based on the specified inputs, the system shall predict the effect of changing a goal satisfaction value on *directly* related goals.

vi. predict the effect of these changes on *indirectly* related goals
Based on the specified inputs, the system shall predict the effect of changing a goal satisfaction value on *indirectly* related goals.

vii. Detect conflicting goal contributions
The system shall detect conflicting goal influences on a single goal. This happens when one goal receives both positive and negative influences from two or more goals.

viii. Determine the effect of cyclic goal influence relations
The system shall predict the effect of both positive and negative loops in goal influence models.

ix. Detect diverging and converging feedback loops
The system shall detect if the satisfaction value of a goal converges to a certain value or not when the goal is a member of a certain feedback loop.

x. Work with Microsoft Office Excel
The system shall export goal satisfaction data to Microsoft Office Excel. The excel data can be used for further analysis.

4.1.2: Quality Requirements:

i. Usability:
The system should be easy to use especially for existing BiZZdesign Architect users.

ii. Adaptability:
The system shall be easily modifiable to support adaptation to organization and project contexts.

iii. Efficiency:
The system shall be reasonably efficient. The system should mot require more resources than an average task in BiZZdesign architect EA design processes.

4.2: Major GORE techniques and Indirect Influence Relations

4.2.1: NFR and Indirect Influence Relations

Considering the year of its publication (1999), the NFR framework can be considered a pioneer approach to non functional requirement analysis. It uses the concept of non functional requirements as soft goals [8]. Due to the fuzzy satisfaction levels of major business goals, the NFR framework can also be extended to goal contribution analysis.

The NFR framework-based goal analysis has a number of advantages. First there are guidelines that can facilitate eliciting and decomposing of goals. Second it provides a wide range well-defined and yet subjective goal achievement levels and contribution types for indirect influence analysis. Further, based on the results from the previous step, it helps in choosing the better alternatives among available goal change options.

Indirect influence relations analysis is quite possible via NFR framework. But the reasoning techniques employed in the NFR are hierarchical influence relations and AND/OR decompositions. This is not a problem in itself but it complicates the task of analyzing and simulating feedback loop effects.

Furthermore, the NFR based qualitative reasoning approaches tend to be too vague for deep and accurate understanding of goal models. Decision procedures for multiple goals contributions also tend to result in undetermined satisfaction levels for higher level goals [16], [17].

4.2.2: KAOS and Indirect Influence Relations

KAOS is more applicable in generating complete and concrete requirements specifications and in handling possible conflicts and obstacles among goals and requirements. Though we find it difficult to reason on indirect influence relations based on the original KAOS goal formalizations, the relatively recent extension of KAOS by KAOS authors themselves makes it at least theoretically possible to reason on indirect influence relations [16].

But handling influence relation of most real life business goal models using KAOS can be a difficult process at least for two reasons. First formulating the objective criteria requires deep analysis of current organizational structure and domain specific knowledge. Second, finding the probabilistic density functions required by KAOS approach is a cumbersome process especially when there are interdependent quality variables [16]. We suggest using KAOS approach for analyzing mission critical influence relations.

4.2.3: i* and Indirect Influence Relations

We have seen i* uses the notion of ++ and -- in goal contribution relations that were adopted from the NFR framework. And compared to NFR, i* excels in supporting very early phases of RE activities, but it adds nothing extra to the contribution types we already saw in the NFR framework.

The difficulties in the early phases of RE activities that forces i* to adopt NFR style qualitative reasoning is the high degree of incompleteness in the early phase of RE. Consequently, we can conclude that both i* and NFR are the same with respect to the objective of this research: they both use qualitative reasoning on goal-to-goal contribution relations. It should be noted that, the NFR framework presents well documented and adequately explained qualitative reasoning on goal to goal contribution relation reasoning.

4.2.4: TROPOS and Indirect Influence Relations

TROPOS is based on i* which enables it to use i*'s early RE activities modeling features that we saw in the previous sections (which are similar to NFR). Hence indirect influence relation is possible in TROPOS. But TROPOS employs both qualitative and reasonably understandable quantitative techniques for indirect influence relation analysis, making it more convenient approach than the ones we saw earlier.

Even a more interesting feature we observe in example goal models of TROPOS is the presence of goal cycles (feedback loops). In addition to the common AND - OR goal decomposition types, TROPOS allows modeling of contribution link chains that makes loops in goal diagrams.

But the very nature of (soft) goals vague satisfaction level is not adequately addressed in TROPOS qualitative reasoning. It only uses three levels of satisfiability (None, Partial or Full evidence) for reasoning on goal satisfaction levels. However, as we will see shortly, a give goal can be perceived both satisfied and partially satisfied when viewed from different perspectives.

The quantitative reasoning of TROPOS provides discrete values of goals since it uses any numeric value between 0 and 1 to represent the satisfaction level. This allows various satisfaction levels instead of three levels we saw in the qualitative approach. Nevertheless, goal-to-goal relations in the quantitative approach have limited reasoning power since TROPOS uses textual descriptions for goal relations

4.3: Selecting Desired Approach

To satisfy theses requirements the goal models shall, appropriate goal reasoning techniques shall be selected. This section will present the comparative analysis we perform on goal oriented approaches specified in chapter three and previous section.

4.3.1: Candidates

The candidates selected for our approaches are:

- KAOS
- NFR framework
- NFR based fuzzy reasoning
- DepRVSim
- Change impact analysis based on Formalization of Trace Relations
- TROPOS

These approaches are obtained from the detailed literature study we presented in chapter 3. Without repeating the details, this section provides only the comparative analysis of the six approaches.

4.3.2: Selection Criteria and Procedure

A number of selection criteria have been considered to choose the relevant change impact analysis technique. Common selection criteria like usability, algorithm complexity, effectiveness and efficiency as well as more specific criteria like applicability to goal analysis are considered.

Each candidate approach is assigned a value of “Low”, “Medium” and “High” for each selection criteria. These values are not adapted from an existing comparative study or from the papers that propose the approaches themselves. Rather, we use the information from the literature study, comments from monthly meetings with BiZZdesign stakeholders and our assessment on the applicability of these approaches in BiZZdesign context to determine the values reported in table 4.2.

For instance KAOS goal reasoning approach needs relatively complex integral functions which are difficult to be used in BiZZdesign Architect scripting language. Hence we label the usability of KAOS in goal influence reasoning “Low” as shown below.

Our intention is to provide a generic solution for different types of stakeholders. This demands high level text based and abstracted reasoning approach in one hand and a more detailed, numeric based goal satisfaction reasoning technique on the other hand. The former can be perceived as qualitative reasoning approach while the later is a quantitative reasoning on goal satisfaction levels.

Qualitative reasoning uses textual goal satisfaction levels like a goal is "partially satisfied", "fully denied" etc. The majority of top level managers and non-technical users are likely to prefer qualitative reasoning approaches because of its simplicity to understand. Sometimes even technical people may use qualitative reasoning for abstracted analysis of goal change impacts. Requirement engineers, strategic planners, policy makers and similar stakeholders may need more detailed level of goal analysis. Quantitative approaches, which use numbered satisfaction levels like "a goal is 85% satisfied", suit more these kind of users.

Approach / Criteria	KAOS	NFR	NFR based Fuzzy reasoning	DepRVSim	Formal Trace Relation	TROPOS
Usability	Low	High	High	Medium	Medium	High
Effectiveness	High	Medium	High	Low	Low	High
Efficiency	Medium	High	High	High	Medium	High
Algorithm Complexity	High	Low	Medium	Low	Medium	Low[3]

Table 4.2: Summary of the comparison criteria values about current GORE methodologies.

Criteria like usability, effectiveness and efficiency are subjective and vague by their nature. A more precise meaning of these criteria with respect to this thesis is given in table 4.3.

Criteria	Definition in this Project
Usability	This refers to usability of the proposed approach with respect to EA, BiZZdesign and BiZZdesign Architect contexts. More specifically the usability of the proposed approach with respect to ArchiMate modeling language and BiZZdesign Architect's EA design environments. An approach that is easy to work with BiZZdesign Architect is assigned "High" usability while an approach that is difficult to integrate it to BiZZdesign Architect is assigned a "Low" usability.
Effectiveness	Effectiveness determines the extent to which a method achieves its objectives [44]. In this analysis, if a method is not capable of reasoning on influence relations adequately, it is termed to have low effectiveness else it will be assigned medium and high values based on the extent to which indirect influence reasoning results appears to be satisfactory.
Efficiency	Efficiency represents the effort required to apply a method [44]. No strict measuring unit is used to determine efficiency of each approach. Rather we compare the available approaches with each other and assign values from low medium and high. The values assigned are not estimated directly since we didn't implement a prototype application for each approach; rather, these values are assigned based on our perception about each method's algorithm and formalization technique.

Table 4.2: Definition of comparison criteria

[3] TROPOS has a top-down approach which is relatively more complex. But this top-down approach is out of the scope of this project and is not considered in this comparison.

To accommodate all types of users, the kind of reasoning approach (qualitative vs. quantitative) is also used as additional selection criteria. This comparison is summarized in table 4.4.

Approach	KAOS	NFR	NFR based Fuzzy reasoning	DepRVSim	Formal Trace Relation	TROPOS
Qualitative vs. Quantitative	Quantitative	Qualitative	Quantitative	Qualitative	Qualitative	Qualitative and Quantitative

Table 4.4: Quantitative vs. qualitative comparison of existing GORE approaches.

4.3.3: Selected Approaches

The results from the previous section indicate that each method has its own strengths and weaknesses. Due to time and other constraints, it is not possible to design algorithm and prototype application for each approach for further investigation.

As can be noted from tables 4.2, "*TROPOS"* and "*NFR Based fuzzy reasoning"* approaches seem more applicable for a number of reasons. First, both are highly usable and applicable for goal influence change impact analysis, second they are highly effective and third both are rated "high" in efficiency criteria. Moreover they have acceptable level of algorithm complexity; while TROPOS has low algorithm complexity, the fuzzy reasoning approach has medium level of complexity.

Our choice was also supported by discussions with stakeholders of BiZZdesign. The discussions revealed that there is demand for qualitative and quantitative reasoning approaches since they can be used at different levels of abstraction. The main idea is that TROPOS qualitative reasoning can be used in high level goal analysis and NFR based fuzzy logic analysis is used for more detailed reasoning on goal influence relations.

4.4: Architectural Design

As an extension to BiZZdesign architect, the system is dependent on BiZZdesign Architect for inputs and outputs. The main component of the system the Reasoning engine which accepts change scenarios and update the goal models accordingly.

Two types of events can occur in goal contribution analysis of goal models; change in goal satisfaction levels and change in contribution relations. The change in goal satisfaction levels occurs when the achievement levels of a certain goal increase or decrease. A change in contribution relations events occur when the links joining any two goals of the system are affected. The result of a change in contribution relation can be considered as a new goal model and we didn't explicitly consider it here.

These changes are inputted to goal models design in BiZZdesign Architect. The change event generator will read the changes and creates a goal change event to the reasoning engine. The reasoning engine will then determine which goal in the goal model should be modified based on its rule base. The reasoning engines will also use the goals and contribution list repositories to determine the required events and necessary actions.

An abstracted structure of the system is shown in figure 4.1. The components in the shaded region are parts of the new extension. The remaining components are already available in BiZZdesign Architect.

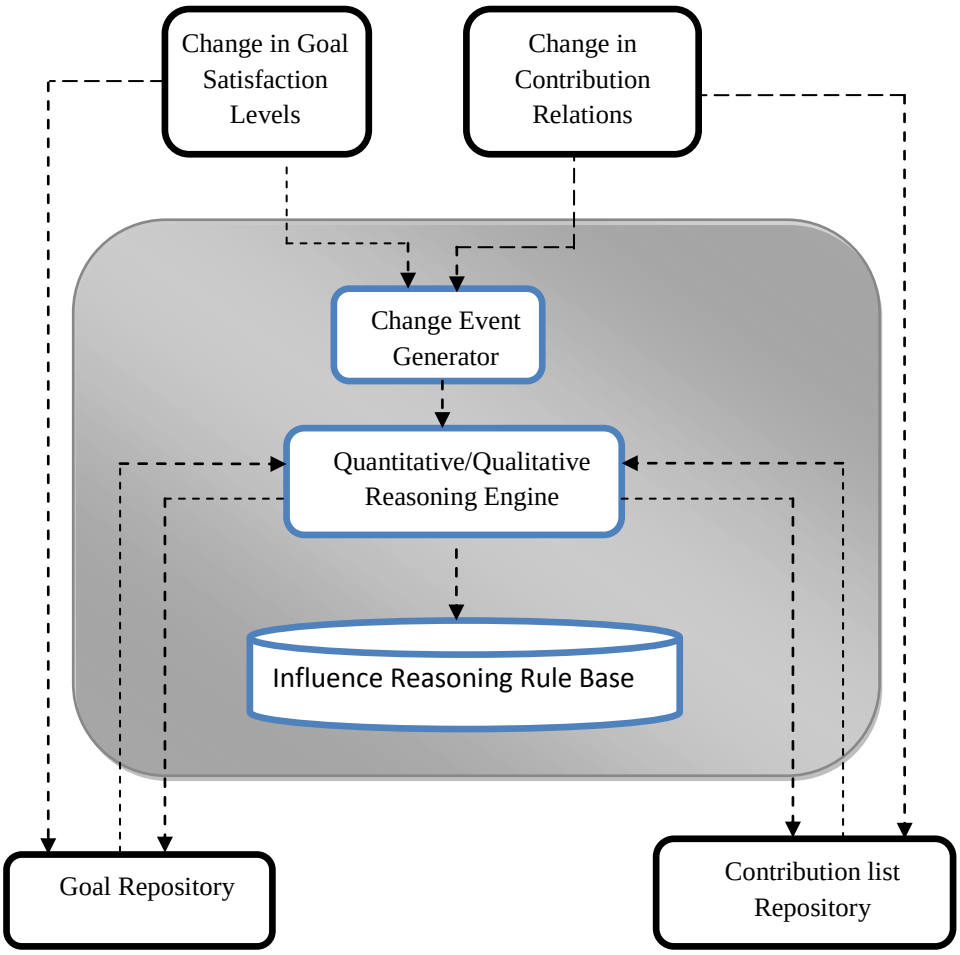

Figure 4.1: Proposed solution structure

4.5: Summary

To leverage existing EA modeling environments of BiZZdesign, the desired goal influence analyzing system has to be developed as an extension of BiZZdesign Architect. The core requirement of the new extension is the capability to provide reasoning on satisfaction level of goals. Other sub requirements like ability to detect goal conflicts and interfacing with excel are also expected from the system.

Two types of specification techniques are found to be applicable for describing goal satisfaction values: textual (qualitative) and numeric (quantitative) specifications. Literature study reveals that there are ranges of existing goal formalization approaches that can provide reasoning on these qualitative and quantitative specifications. To identify the most suitable approach for goal analysis in EA goal models, a comparative analysis is performed. The analysis identifies TROPOS software development methodology and Fuzzy logic extension of the NFR framework as a good candidate for qualitative and quantitative reasoning techniques respectively.

By explaining the kind of reasoning techniques available via current GORE methodologies; this chapter completes the first research question that was partly answered in chapter three.

This chapter also answers the second research question, *which goal formalization techniques support reasoning on indirect influence relations and change impact analysis for goal models*? As specified in the previous paragraph, TROPOS and Fuzzy reasoning engine are the goal formalization techniques that are found to be applicable for goal reasoning.

5: Qualitative Reasoning: TROPOS Based Goal Reasoning

This chapter presents the qualitative reasoning methodology selected for reasoning on goal influence relations. The approach is based on TROPOS software development methodology. A quick overview of TROPOS software development methodology will be presented first followed by the application of TROPOS for RE activities, goal modeling and goal reasoning.

This chapter also presents an adapted algorithm and the prototype application of the TROPOS based goal influence reasoning methodology. To demonstrate the applicability of the approach, the results of two test cases will be presented. Finally the advantages and disadvantages of using TROPOS based qualitative reasoning will be discussed.

5.1: TROPOS Software Development Methodology

TROPOS is a relatively new and requirement –driven software development methodology [25]. The majority of the concepts used in TROPOS like actors, roles, tasks, goals and requirements are adapted from i* conceptual modeling framework [15].

Though its predecessor, i*, is mainly concerned on early phases of requirement engineering activities, TROPOS is extended to cover the following four phases of software development process[41].

1. **Early Requirements:** This phase is concerned with questions like what does the current organization looks like? what are its actors? what are the goal and roles of actors in organizations?
2. **Late Requirements:** Description of the system to be in its operational environment.
3. **Architectural Design:** Defining abstracted global view of the system to be in terms of communications between subsystems.
4. **Detailed Design:** Refining components of step 3 to incorporate inputs, outputs, controller etc.

The main focus of this paper is reasoning on goals, which are the main sources of early requirements. Hence we will focus only on the first phase of TROPOS software development methodology activities.

5.1.1: TROPOS for Early Requirements

As we have noted in chapter 3 of this report, i* based early requirement engineering activities tries to capture and analyze the intention of stakeholders. Intentional elements (goals, requirements and related concepts) in i*can be expressed in terms of Strategic Dependency (SD) model and Strategic Rationale (SR) model. Since TROPOS is based on i*, it is capable of using these i* concepts to reason on goal satisfaction values.

A Strategic Dependency (SD) model is a graph which models how actors depend on each other to achieve their goals. In these actor interactions, an actor (a Depender) will depend upon another actor (Dependee) to get a resource commonly called Dependum [41].

The Strategic Rationale(SR) model is a graph that adds further information to the SD model by providing a deeper understanding and reasoning about strategic actors' intentions [15]. Incorporating intentional details enables to model the interaction of actors that results achievements of goals and to reason out to

what extent the goal is achieved [40]. SR models primarily uses means-end and decomposition links and other influence notations like ++ and -- to provide intentional details and alternative configurations [41].

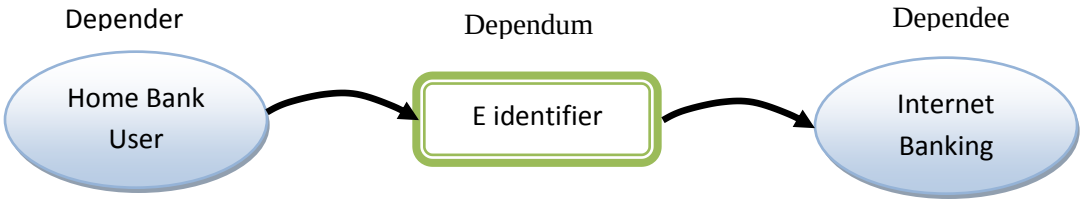

Figure 5.1: A sample SD model

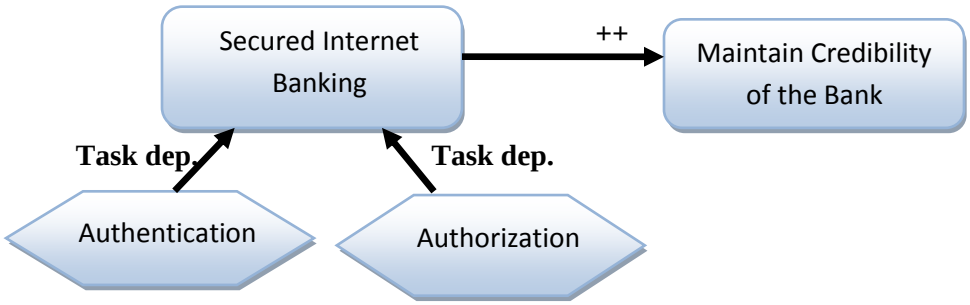

Figure 5.2: A sample SR model

In figure 5.1 , a home bank user (Depender entity) will depend on E identifier (Dependum) to access internet banking services. In figure 5.2, the goal "Secured Internet Banking" depends on "Authentication" and "Authorization" tasks. Secured internet banking has also positive influence on the goal "Maintain Credibility of the bank".

Using this kind of relations, it is possible to have qualitative indirect influence relation in TROPOS. Additionally, by using values from 0 to 1 for goal satisfaction levels and relation strengths, TROPOS also supports quantitative reasoning on indirect influence relations [26].

5.2: TROPOS for Requirements Engineering and Goal Reasoning

TROPOS uses three levels of predefined satisfaction levels for satisfaction values: Full, partial and none evidence of satisfiability. The satisfaction levels are assigned to two special attributes. These two attributes, named satisfiability and deniability, represent the degree of satisfiability and deniability levels of each goal [26]. This kind of separation is useful in situations where a goal can be considered satisfied and denied at the same time when perceived from different views. It is also helpful in handling conflicting goal contributions when there are both positive and negative influences on a single goal.

Table 5.2 shows the satisfiability and deniability values in TROPOS. The influence relations that can also determine the satisfaction level of goals are AND, OR, --, - , + and ++ [25].

	Satisfiability Evidence			Deniability Evidence		
Value	Full	Partial	None	Full	Partial	None
Symbol	FS	PS	NS	FD	PD	ND

Table 5.2: Three levels of satisfiability of goals in TROPOS.

In addition to separation of satisfiability and deniability TROPOS employs asymmetrical contribution relations. Asymmetrical relation is a relation where either satisfiability or deniability, but not both, is propagated. Though these kinds of relation types can be helpful in dealing with complex scenarios, their usefulness with higher level qualitative goal reasoning is less likely. In fact, the discussion with the stakeholders of BiZZdesign reveals that this might reduce the usability of the technique.

Moreover, we realized that discarding these asymmetrical relations will also help in creating common comparison criteria of the qualitative and quantitative reasoning later in this report. Hence we chose to adopt the symmetrical propagation rules of TROPOS in this paper as shown in table 5.3. *"Sat"* and "*Den*" represent *Satisfiability* and *Deniability* respectively of a goal in a goal model while the notation "*P*" represent *Partial level of satisfiability of deniability.*

	(G2 ∧G3) ➔ G1	**(G2 ∨G3) ➔ G1**	**G2 ➔- -G1**	**G2 ➔ - G1**	**G2 ➔+ G1**	**G2 ➔++ G1**
Sat(G1)	Min $\begin{Bmatrix}\mathbf{Sat(G2)},\\ \mathbf{Sat(G3)}\end{Bmatrix}$	Max $\begin{Bmatrix}\mathbf{Sat(G2)}\\ \mathbf{Sat(G3)}\end{Bmatrix}$	Den(G2)	Min $\begin{Bmatrix}\mathbf{Den(G2)},\\ \mathbf{P}\end{Bmatrix}$	Min $\begin{Bmatrix}\mathbf{Sat(G2)}\\ \mathbf{P}\end{Bmatrix}$	Sat(G2)
Den(G1)	Max $\begin{Bmatrix}\mathbf{Den(G2)},\\ \mathbf{Den(G3)}\end{Bmatrix}$	Min $\begin{Bmatrix}\mathbf{Den(G2)}\\ \mathbf{Den(G3)}\end{Bmatrix}$	Sat(G2)	Min $\begin{Bmatrix}\mathbf{Sat(G2)},\\ \mathbf{P}\end{Bmatrix}$	Min $\begin{Bmatrix}\mathbf{Den(G2)}\\ \mathbf{P}\end{Bmatrix}$	Den(G2)

Table 5.3: Adapted influence relation definitions in goal models.

5.2.1: Adapted TROPOS Methodology for Goal Reasoning

The TROPOS goal satisfaction reasoning rules discussed in the previous subsection was found to be less usable after a discussion with the intended stakeholders of BiZZdesign. There were two main reasons for this. First, separating satisfiability and deniability of a certain goal was suspected to be ambiguous and confusing for end users. The ambiguity can be severe if the end users are non technical users of the system.

Second, having three levels of satisfiability was found to be less usable for reasoning with influence relations. If we use only None, Partial and Full satisfiability for goal satisfaction levels, it will be too vague to differentiate goals with different levels of partial satisfaction levels.

To alleviate the first problem, we opt to merge the satisfiability and deniability values into one qualitative reasoning satisfiability variable. And to treat the second problem, we introduce four more possible values

of satisfiability: Much Evidence of Satisfiability/Deniability and Little Evidence of Satisfiability/Deniability.

Given an influencing goal G1 influencing a second goal G2 via influence relation r as shown in figure 5.3, the possible values of the adapted TROPOS based goal satisfaction levels are shown in table 5.4 below.

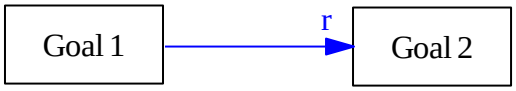

Figure 5.3: A simple goal influence diagram with two goals and one influence relation r.

Evidence for Satisfiability of G1	**Satisfiability of G2, when the influence relation r with G1 is**				
	--	-	+	++	C[4]
Full Evidence Den FD	FS	**Min** {Sat(**G2**), **PS**}	**Min** {Sat(**G2**), PD}	FD	FS
Much Evidence Den MD	MS	**Min** {Sat(**G2**), **PS**}	**Min** {**Sat**(**G2**), PD}	MD	MS
Partial Evidence Den PD	PS	PS	PD	PD	PS
Little Evidence Den LD	LS	LS	LD	LD	LS
No Evidence of Den ND	NS	NS	ND	ND	NS
No Evidence Sat NS	ND	ND	NS	NS	ND
Little Evidence Sat LS	LD	LD	LS	LS	LD
Partial Evidence Sat PS	PD	PD	PS	PS	PD
Much Evidence Sat MS	MD	**Min** {Sat(**G2**), PD}	**Min** {Sat(**G2**), **PS**}	MS	MD
Full Evidence Sat FS	FD	**Min** {**Sat**(**G2**), PD}	**Min** {Sat(**G2**), **PS**}	FS	FD

Table 5.4: Extended TROPOS based Goal Influence reasoning rules.

[4] C represents conflicting relation. It is used to represent a relation where satisfying of an influencing goal prohibits satisfiability of another goal (Makes it Denied). This is practically applicable in hard goal to hard goal relations.
But the main focus of this project, influence relations, is predominantly useful in soft goals (Because there is no influence relations among hard goals since they are either Satisfied or Not Satisfied).From the influence relation perspective, this is the same as the having a break (--) relation. So we didn't explicitly show it in the implementation.

In addition to the values shown in the first column of table 5.4, a goal can assume different levels of "On conflict" state when it receives the following conflicting contributions:

- Little satisfaction and Little Deniability
- Partial Satisfaction and Partial Deniability
- Much Evidence of Satisfiability and Much Evidence of Deniability
- Fully Satisfied and Fully Denied

A point we would like to stress here is that we are not claiming our new rule sets will be complete for all goal satisfaction levels. After all, it is possible to have infinite levels of goal satisfaction levels due to the vague nature of soft goals satisfaction levels. What we aimed is to provide easily understandable and at the same time reasonably complete qualitative reasoning tool.

5.2.2: TROPOS Qualitative Reasoning Algorithm

Being intended to be used in the early phases of requirements engineering activities, TROPOS authors propose algorithms for both qualitative and quantitative goal analysis techniques [26]. We adopt the qualitative TROPOS algorithm with some modifications.

The most important modification was, in the original TROPOS algorithm, once a goal reaches Full satisfiability levels; it was not clear how a negatively affecting influence will reduce the satisfaction level of a fully satisfied goal. We modify the algorithm to allow reduction of goal satisfaction levels via negatively contributing goals. The modification allows using the average values of available goal influences to specify goal satisfaction values.

Additionally, we also update the algorithm to work with the new qualitative reasoning rules shown in table 5.4. Listing 5.1 shows the adapted and an abstracted TROPOS based algorithm for goal analysis.

```
forall "MotivationGoal" goal in model {
        if ( !isLeaf(goal) ) {
                satList = List();
                resultantSat = undefined;
                forall "InfluenceRelation" r in goal.relationsTo() {
                    obj = relation.relatedTo(goal);
                     if (obj is " MotivationGoal ") {
                          satisfaction = applyRulesSat(goal, r);// Rules in table 5.4
                          satList.add(satisfaction);
                       }

                      else if ( obj is "AndJunction" ) { // assumption: no nesting of junctions
                           andSatList = List();
                           forall "InfluenceRelation" r in obj.relationsTo() {
                             g = r.relatedTo(obj);
                             andSatList.add(g.attrValue("QualitativeSatisfaction"));
                           }
```

```
                    newSat = minList(andSatList);
                    newSat = toQualitative(newSat);
                }

            else if ( obj is "OrJunction" ) { // assumption: no nesting of junctions
                    orSatList = List();
                    forall "InfluenceRelation" r in obj.relationsTo() {
                        g = r.relatedTo(obj);
                        orSatList.add(g.attrValue("QualitativeSatisfaction"));
                    }
                newSat = maxList(orSatList);
                newSat = toQualitative(newSat);
        }
        return newSat;

        }

          if (CheckConflict(satList) == false) {
            resultantSat = averageList (satList);
                goal.setAttrValue("QualitativeSatisfaction", resultantSat);
                goal.setAttrValue("Conflict", false);
              } else {
                goal.setAttrValue("Conflict", true);
              }
        }
}


function averageList(l){
  if ( l.empty() ) {
        return undefined;
        }
        Else     {
        m = goalVal(l[1]);
        i = 2;
        while ( i <= l.size() ) {
           m = avg(m,  goalVal(l[i]));
                i = i + 1;
        }
        return m;
        }
}
```

```
function CheckConflict(satList) {
  if (satList.contains("Fully Satisfied") && satList.contains ("Fully Denied")) {
    return StrongConflict;
  }
else if (satList.contains("Partially Satisfied") && satList.contains ("Partially Denied")) {
    return PartialConflict;
  }
   ……..

else if (satList.contains("Little Satisfied") && satList.contains ("Little Denied")) {
    return LittleConflict;
  }

  return false;
}
```

Listing 5.1: Abstracted TROPOS Algorithm

5.2.3: Pros and Cons of TROPOS based goal analysis

Being based on i*, a well known modeling framework, TROPOS based goal analysis enables modeling and description of various intentional element in RE activities like goals, requirements, stakeholders etc. The presence of both qualitative and quantitative reasoning approaches makes TROPOS applicable for different abstractions of goal analysis.

But TROPOS also comes with its own limitations. First, the qualitative reasoning has only three levels of satisfaction and deniability levels. This makes it impossible differentiate two goals with different partial satisfaction levels. We alleviate this problem by introducing more qualitative levels of goal satisfactions like “Little Denied” and “Much Evidence of satisfaction”. But it would not be wrong if someone says there can be two goals whose satisfaction level is “mostly Satisfied” with different degree. Our qualitative reasoning approach will not differentiate these kind goal satisfaction values.

The modified version has also its own limitation since it returns only the average of all input contributions. For instance if there are a number of small positive contributions then these lines will return small contribution whereas in reality the small contributions can be summed up to form bigger contribution.

Moreover, our adopted approach doesn’t take into consideration of goal to be influenced satisfaction level. It only depends only on the affecting goal satisfaction level and contribution relation strength.

5.3 Testing Qualitative Reasoning on Industrial Case Study

For validating the algorithms and prototype applications two case studies were used. The first one, ArchiSurance insurance company, is based on an example from The Open Group[5]. It is also a standard example set by The Open Group for certified TOGAF Enterprise Architecture academies [21]. The second test case is based on a drinking water company in Netherlands. The company was under some structural change due to budget reductions which makes it good candidate for investigative stakeholder change impacts.

The water company case study would have been much more practical and complete if we apply our reasoning tool on the company before it implements the business process reengineering. In that way we could have the opportunity to compare our tool predictions to what has been observed by the company in practice.

But this kind of empirical studies would take probably a year or longer (longer than this project time span). And even if we had time, the company already undergoes the business process reengineering and we have no full data that can be used to see which of their goals' satisfaction levels have been altered to enable satisfaction of the business process reengineering process objective.

Nevertheless, we argue that the ArchiSurance and water company case studies are adequate enough to show our approach can be applied in practical goal analysis of real industrial cases. Additionally, the results of case studies were vital in comparing and contrasting the two approaches. The details of this comparison will be presented in chapter 8.

5.3.1: Qualitative Reasoning Case Study on Goal Models of ArchiSurance

The ArchiSurance case study is a fictitious example published by The Open Group to be used in ArchiMate trainings in the context of The Open Group Architecture Framework (TOGAF) [21]. Three companies involved in home, auto and legal insurances are merged to form ArchiSurance. The primary drivers behind the merge of the three companies were twofold; first the companies want to stay competitive despite new lower cost insurance companies are emerging. Second each company needs big investment on IT infrastructure and common IT system was a viable option.

BiZZdesign has developed an EA model for ArchiSurance using BiZZdesign Architect. One of the features of BiZZdesign architect is the auto generation of specific view of EA components. Using BiZZdesign architect, we generate a view for the motivation aspect (see chapter one) of the ArchiSurance EA design that show the drivers, goals, stakeholders principles, requirements etc.

Since the primary objective of this thesis is the analysis of indirect influence relation, we strip off all the motivation aspect components except the goal models and we end up in a goal model shown in figure 5.4.

But the goal model shown in figure 5.4 is a little simple to be used in this project primarily for three reasons. First the goal model doesn't have enough number of goal contribution links to show goal change impact propagation effects. Second it has only positive goal influence types and third there is no dynamic

[5] The Open Group is a global consortium that tries to enable the achievement of business objectives through IT standards. http://www.opengroup.org/.

loop among goal to goal influences. Hence we extend the goal model in figure 5.4 to a reasonably complex goal model as shown in figure 5.5.

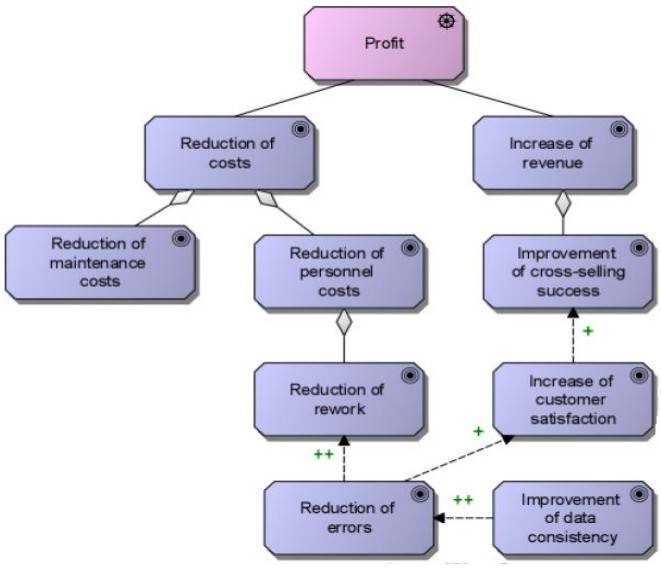

Figure 5.4: ArchiSurance goal models

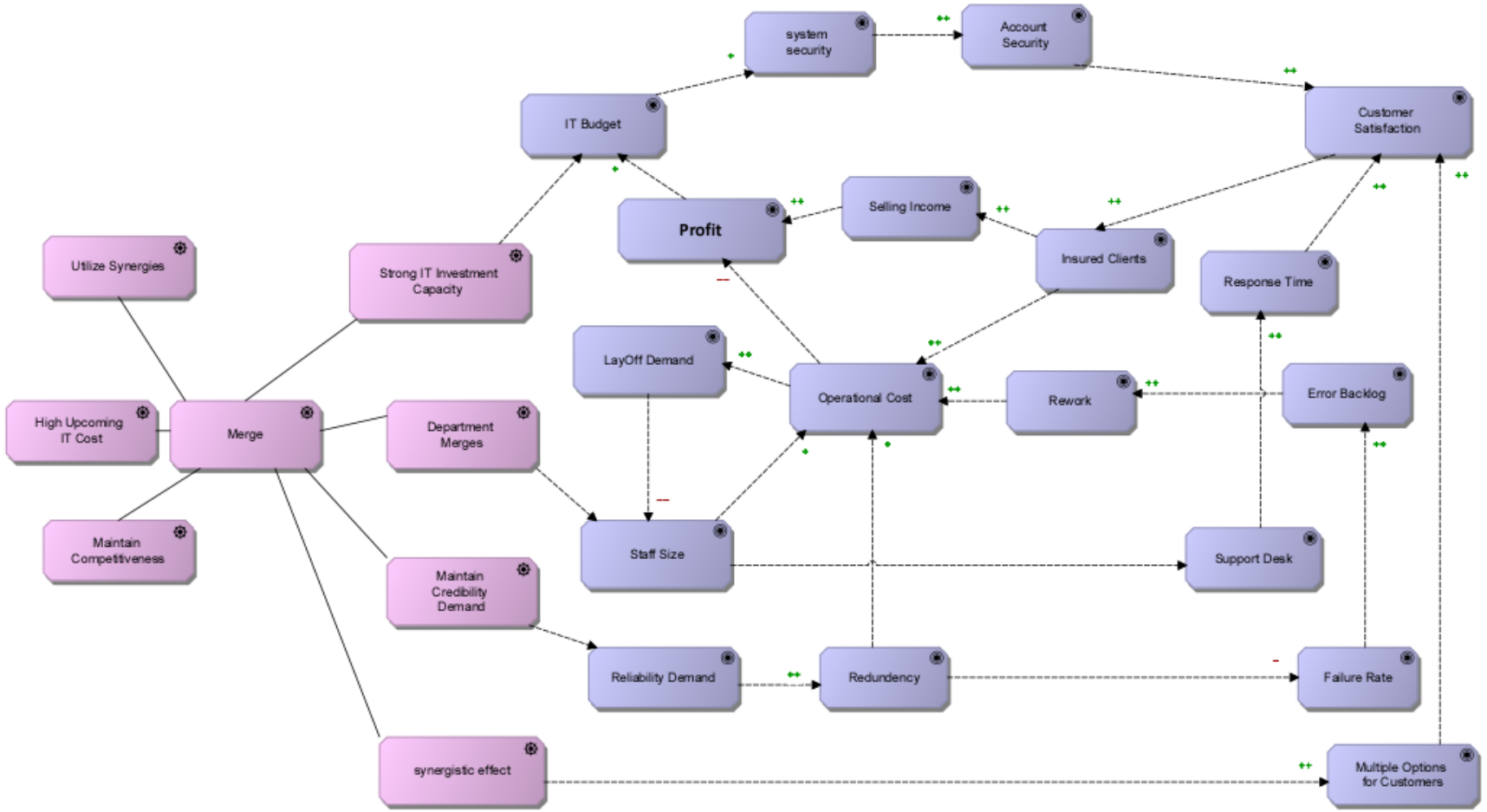

Figure 5.5: Extended ArchiSurance Goal model

As ArchiSurance was a fictions company, we couldn't set "real" values as inputs for our tool. Rather, we select inputs that match the scenario. For instance, the goal IT budget was set to Fully Satisfied because the combined IT budgets of the three companies is assumed to be satisfactory for the new company. The initial satisfaction levels of all other goal were set to be partially satisfied because they were not affected directly by the merge.

Leaf Goal	Initial Value	Symbol
It Budget	Full Evidence of Satisfiability	FS
Staff Size	Much Evidence of Satisfiability	MS
Reliability Demand	Partial Evidence of Satisfiability	PS
Multiple Options for Customers	Much Evidence of Satisfiability	MS

Table 5.5: Initial goal assignments for satisfaction levels

Taking the values of table 5.5 as inputs, the result of the output of our tool is shown in table 5.6 below. The maximum possible goal traversing to the top "Profit" goal has six steps. Table 5.6 shows the output of the result at the end of the 6th traverse. The details of what will happen in the cyclic phases will be discussed in detail in section 9.

Goal Name	Satisfaction Level
Profit	NotSat
Insured Clients	NotSat
Common IT Infrastructure	PartiallySat
Operational Cost	NotDen
Rework	PartiallyDen
Failure Rate	PartiallyDen
Selling Income	NotSat
Support Desk	NotDen
Response Time	NotDen
Reliability Demand	PartiallySat
Error Backlog	PartiallyDen
Customer Satisfaction	NotSat
Account Security	NotSat
system security	NotSat
IT Budget	NotSat
Multiple Options for Customers	MostlySat
Layoff Demand	NotDen
Staff Size	NotDen
Redundancy	PartiallySat

Table 5.6 Goal Analysis Result for ArchiSurance case study

5.3.2: Qualitative Reasoning Case Study on Goal Models of a Water Company

This is the second case study we conduct to verify the applicability of our qualitative reasoning tool on industrial contexts. The study is based on the data collected from one of BiZZdesign client organization involved in drinking water production in Netherlands. The organization was under some structural changes due to shrinking budget resulting in changes of various stakeholder goals making it a good case study for testing our approach.

The primary data about goals of the organization was collected by BiZZdesign and is presented in detail in [11]. There are about ninety goals identified from relevant documents and interviews. Due to privacy issues, space limitations and complexity of the entire goal graph, we will use only a small portion of the organization goal graph. Figure 5.6 shows decomposition of "customer satisfaction" goal, which is of course one of the many goals of the company's Board of Directors.

The primary contributors of "Customer satisfaction" goal are "Correct and fast invoicing", "Water price below national average" and "High quality customer perception". Each of these goals is further decomposed in to a number of more concrete goals. There are also AND/OR decompositions like the "OR" decomposition of "collaborative buying" in to "Procurement Agency" and "Buy with other water companies" and the "AND" decomposition of "Excellent water Quality" in to "Odorless water", "Safe water" and "Clean Water" Goals.

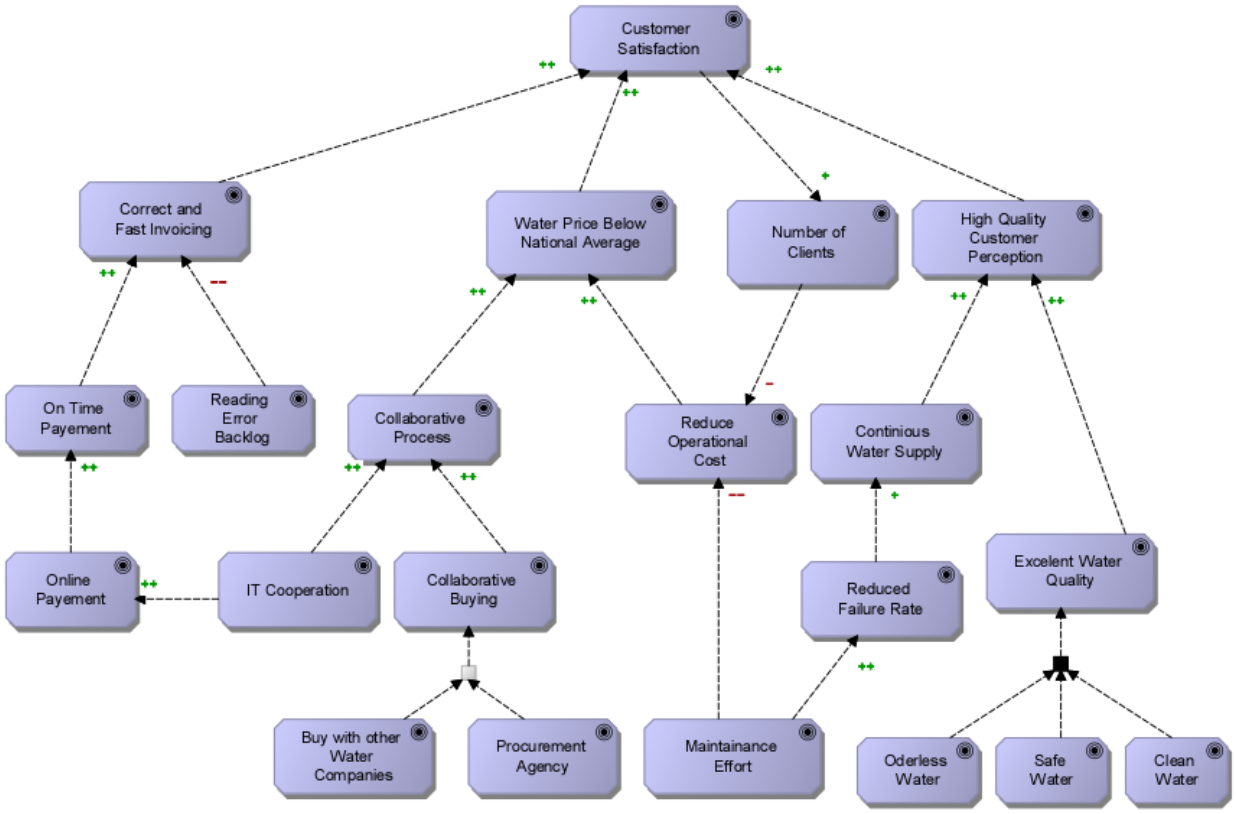

Figure 5.6: A portion of goal models applied in goal analysis

Initial inputs were provided for the leaf goals of the goal model to trigger the impacts of the inputs on other goals of the system. The input goals were selected because they were located down on the goal decomposition hierarchy which makes them easier to estimate their satisfaction value. But the concrete values assigned to the satisfaction level of these goals was not a total random; rather we use the data collected by BiZZdesign to estimate the initial assignment values. We also try to select leaf goals whose satisfaction levels will cover majority of the goal satisfaction levels shown in table 5.6.

We use leaf goals as initial inputs because in correctly decomposed goal models, realizing upper level goals is possible if and only if the leaf goals are fulfilled. Since complete decomposition of goals maybe subjective and sometimes difficult to achieve, our approach can also work even if the input goals were not leaf goals. The initial satisfaction levels of all other goal were assumed to be partially satisfied.

Leaf Goal	Qualitative Satisfaction Level	Symbol
Buy with other water companies	Partial Evidence of Deniability	PD
Procurement Agency	Much Evidence of Satisfiability	MS
IT collaboration	Partial Evidence of Satisfiability	PS
Maintenance Effort	Much Evidence of Deniability	MD
Reading Error Backlog	Little Evidence of Satisfiability	LS
Odorless Water	Much Evidence of Satisfiability	MS
Safe Water	Full Evidence of Satisfiability	FS
Clean Water	Full Evidence of Satisfiability	FS

Table 5.6: Input values for leaf goals satisfiability and deniability values.

The output of the TROPOS based reasoning tool is shown in table 5.7. Goals in which weak conflict (Partial Satisfaction and partial Deniability) influence is received are marked with * and Goals which receive strong conflicts (Fully satisfied and fully Denied) are marked with **. In both cases, we use the minimum of the contributing values to continue propagating change impacts.

The results shown in table 5.8 are after traversing four steps in the goal model shown in figure 5.6. The four traversal steps are chosen to incorporate the four levels of goal models (from the bottom line of goals to the top goal) shown in fig. 5.6.

Goal Name	Satisfaction Level	Symbol
Excellent Water Quality	MostlySat	MS
Collaborative Buying	MostlySat	MS
Customer Satisfaction	NotSat	NS
Correct and Fast Invoicing	NotSat	NS
Water Price Below National Average	LittleSat	LS
Number of Clients	NotSat	NS
High Quality Customer Perception	NotSat	NS
Reduce Operational Cost	LittleSat	LS
On Time Payment	PartiallySat	PS
Online Payment	PartiallySat	PS
IT Cooperation	PartiallySat	PS
Collaborative Process	PartiallySat	PS
Maintenance Effort	MostlyDen	MD
Continuous Water Supply	PartiallyDen	PD
Reduced Failure Rate	MostlyDen	MD
Odourless Water	MostlySat	MS
Safe Water	FullySat	FS
Clean Water	FullySat	FS
Procurement Agency	MostlySat	MS
Buy with other Water Companies	PartiallyDen	PD
Reading Error Backlog	LittleSat	LS

Table 5.8: The result of the TROPOS based goal analysis on satisfaction levels.

5.4: Summary

Qualitative reasoning techniques use predefined textual specifications like a goal is partially satisfied or fully denied to reason on goal satisfaction levels. The influence relations are also specified in semi textual notations like ++, -- and AND/OR decompositions.

The qualitative reasoning technique selected for goal influence reasoning is TROPOS software development methodology. It is chosen because of its high usability and effectiveness in analyzing indirect influence relations among EA goal models.

The adapted qualitative reasoning engine has ten levels of goal satisfaction values. These levels, arranged from lowest to highest, are full evidence of deniability, much evidence of deniability, partial evidence of deniability, little evidence of deniability, no evidence of deniability, no evidence of satisfiability, little evidence of satisfiability, partial evidence of satisfiability, much evidence of satisfiability and full evidence of satisfiability.

The influence relations that will affect these satisfiability values are - - (full negative), - (partial negative), + (partial positive), ++ (Full positive) and conflicting contributions.

A reasoning algorithm based on these goal satisfaction and contribution values is developed. The two test cases conducted on this algorithm shows that, it is indeed possible to use TROPOS based reasoning engine to analyze indirect influence relations in EA goal models. The test cases also reveal that the qualitative reasoning technique is capable of detecting conflicting goal contributions.

The third research question of this master project is *"How can we simulate influence relation impacts on goal satisfaction values and how can we visualize the simulation?"* This chapter provides one answer to this research question by presenting a qualitative reasoning engine to analyze goal change impacts. It also presents the reasoning engine algorithm and a tool support to visualize the change impact effects. Two test cases are also used to demonstrate the applicability of the qualitative reasoning engine for analyzing goal influence relations.

6: Quantitative Reasoning: NFR Based Fuzzy Logic Reasoning

This chapter presents the second approaches selected for reasoning on goal influence relations. The approach is based on NFR framework goal satisfaction propagation rules. It also employs fuzzy reasoning techniques to quantitatively reason on goal influence relations.

This chapter also presents the adapted definitions of goal to goal influence relations of the NFR framework. This discussion will be followed by introducing of the applicability of fuzzy sets and fuzzy reasoning techniques to goal influence reasoning. The fuzzy reasoning engine uses a range of numeric values from -100 to 100 to specify goal satisfaction values and influence relation strengths.

To demonstrate the validity of the quantitative approach, two test cases are applied. The first test case is based on ArchiSurance goal model while the second one is based on water company goal model.

6.1 NFR framework and Goal Reasoning

NFR framework, which is a pioneer in managing non functional requirements, was selected as a candidate approach analyze goal relations due to the high resemblance of non functional requirements and majority of business (soft) goals [8], [9], [45].

NFR framework is used in modeling and operationalization of stakeholder goals. It does so by enabling understanding and prediction of possible interdependencies among various goals of a system. The main tool used by the NFR framework for modeling and reasoning about goals of a system is the software interdependency Graphs (SIG) [8]. Figure 6.1 shows a typical software interdependency graph.

In SIG goal decomposition is represented by interconnected interdependency links with arrowheads and associated labels representing the degree to which a goal is achieved. Goals can contribute positively or negatively to the satisfaction level of other goals. Positive contributions are represented with an arrow and a plus sign while negative contributions are denoted by an arrow and a minus sign [9].

There are also cases where all the sub goals of a given goal are required to satisfy the super goal. This kind of relation is termed as AND type contribution while if only one of the sub goals is enough for satisfying the super goal, the relation is referred as an OR type contribution. The seven contributions, “AND” refinements and “OR” refinement types available in the NFR framework are summarized in table 6.1.

Based on the contribution types defined in table 6.1, system developers can decide among the following types of satisfaction levels: Satisfied, denied, satisfiable and deniable. The first two is used when there is a clear understanding whether the goal is achieved or not. But since a goal can receive both positive and negative types of contributions from offspring’s, a clear measure of satisfaction extent may not be possible. In that case “satisfiable” and “deniable” indicators can be used.

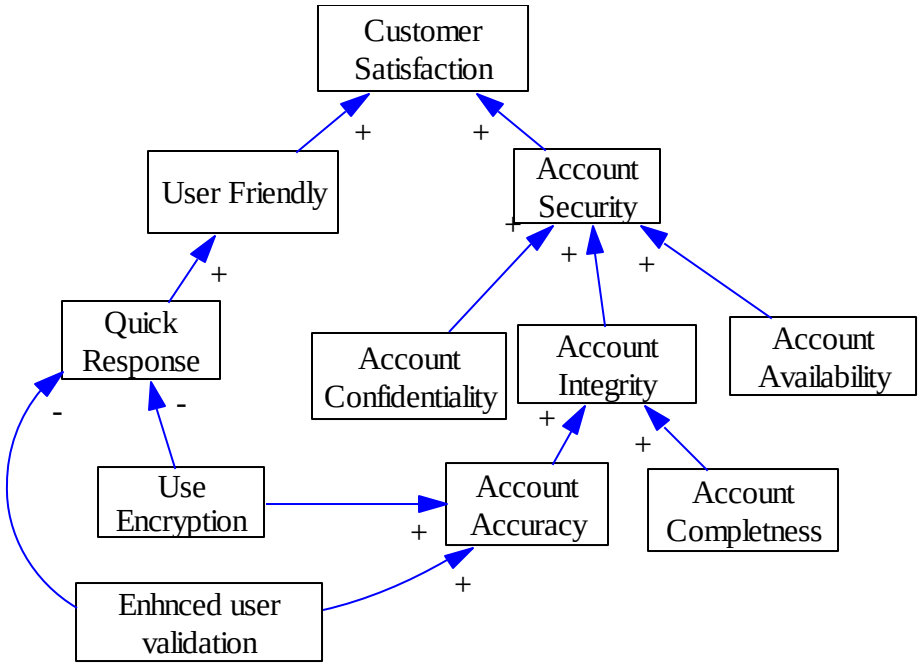

Figure 6.1: A typical Software Interdependency Graph (SIG).

Contribution Types	**Contribution Symbol**	**Description**
AND	∧	If all the offspring are all satisfied so is the parent goal.
OR	∧	If any of the offspring are satisfied the so is the parent goal.
MAKE	↑ ++	A single offspring analogy of AND. If the single offspring goal is satisfied so is the parent goal.
BREAK	↑ --	If the single offspring is satisfied the parent goal will be denied.
HELP	↑ +	Offspring provides partial positive support.
HURT	↑ -	Offspring provides partial negative support.
SOME+	↑ Some+	Any type of positive contribution: Either HELP or MAKE
SOME-	↑ Some -	Any type of negative contribution: Either BREAK or HURT
Unknown	?	Makes some contribution of unknown sign and unknown extent.

Table 6.1: Goals contribution types and symbols of "AND" and "OR" contribution combinations [8].

Satisfaction Level	Description
Satisfied	Achievement of goal is satisfactory
Denied	Achievement of goal is unsatisfactory
Satisfiable	Achievement of goal is potentially satisfactory
Deniable	Achievement of goal is potentially unsatisfactory

Table 6.2: Goal satisfaction level descriptions [9].

6.1.2 Formal Definition of Contribution types

i. AND: Let offspring$_1$ AND offspring$_2$ AND ….AND offspring$_n$ be called as Offspring, AND({Offspring}) SATISFICE parent can be defined as
If all the off springs are satisfied and
When the interdependency itself is satisfied then
The parent is satisfied

ii. OR ({Offspring}) SATISFICE parent
If any of the offspring is satisfied and
When the interdependency is self satisfied
Then the parent is self satisfiable

iii. Offspring Makes Parent:
If the offspring is satisfied and
When the interdependency is self satisfied
Then the parent is self satisfiable

iv. Offspring Breaks Parent
If the offspring is satisfied and
When the interdependency is self satisfied
Then the parent is self deniable

v. Offspring Hurts Parent:
If offspring is denied and
When the interdependency is self satisfied
Then the parent is self satisfiable

vi. Offspring HELPS Parent:
If offspring is denied and
When the interdependency is self satisfied
Then the parent is deniable

vii. Offspring SOME+ Parent ≡ Offspring HELPS Parent or Offspring MAKES Parent

viii. Offspring SOME- Parent ≡ Offspring HURTS Parent or Offspring breaks Parent

ix. Offspring UNKNOWN Parent ≡ Offspring SOME- parent or Offspring Equals Parent or Offspring SOME+ parent

6.1.3: The NFR framework decision Process

Till now the goal decompositions, contribution types and their notation is discussed. But the most important concept remains: How to determine the extent to which a certain goal is satisfied or denied by its offspring's? The approach suggested in [8] to determine the extent of goal satisfaction is through the usage of label assignments.

The initial values of the labels can be obtained from the decisions made to accept or reject alternatives available usually in the leaf level goals of the SIG [8], [9]. Once the initial values are set; it is possible to evaluate change effects on upper level of goal until the effect on the final goal is reached.

With the help of satisfiable and deniable notations of previous section, a goal in an interdependency graph is labeled as:

- Satisfied (S): If it is satisfiable and not deniable
- Denied (D): If it is deniable not satisfiable
- Conflicting (C): If it is Both Satisfiable and deniable
- Undetermined (U): If it is neither satisfiable nor deniable
- Weak Positive (W+): Inconclusive positive support for a parent
- Weak Negative (W-): Representing inconclusive negative support for a parent

Using these notations, whenever a goal or interdependency graph is assigned a new label, the decision procedure is activated and propagates labels from offspring to a parent. The decision procedure will have two important steps: individual impact computation and aggregation of the individual impacts.

i. First Step:

This step determines the individual impact of an offspring contribution towards a parent for each contribution relation.

The AND contribution is determined from the minimum of the offspring contribution while the OR contribution is determined by the maximum of the offspring's' contributions. The order of the contributions can be decided based on the relation: $D \leq U \approx C \leq S$.

MAKE propagates "S" and "D" from offspring to parent

BREAK inverts the sign of an offspring "S" label into "D" for the parent

HELP keeps the sign but weakly propagate the changes.

HURT inverts the sign and weakly propagates the changes.

SOME+ labels the parent with a weak label (W+ or W-) while keeping the sign.

SOME- labels the parent with a weak label (W+ or W-) while inverting the sign.

Unknown and undetermined always propagates U.

Conflicting offspring propagates "C" unless the contribution type is unknown.

Individual Impact of offspring with label:	Upon Parent label, given off-spring parent contribution types							
	Break	SOME-	HURT	?	HELP	Some+	Make	=
D	W+	W+	W+	U	W-	W-	D	D
C	C	C	C	U	C	C	C	C
U	U	U	U	U	U	U	U	U
S	D	W-	W-	U	W+	W+	S	S

Table 6.3: Contribution impacts reference table

ii. Second Step:

During the first stage, all the contributions to a given parent are collected, what remains will be putting these contributions into a bag (not a set) and combine them so that a single representant label is created. The combination of contribution is performed in incremental steps. First W+ and W- labels are combined to form one or more "D", "C", "U", "S". For instance two W+ labels can be combined into one "S".

The resulting labels will be then combined into a single one by choosing the minimal label of the bag, with a label ordering:

"C" ≤ "U" ≤ D ≤ "S".

As an Example consider one satisfied offspring with denied contribution to its parent and another denied offspring with satisfied contribution to the parent, according to table 6.3, both offspring will contribute Denied contribution to the parent and the parent will be denied.

Of course it is not always easy to make contribution decisions, especially in the case where two or more offspring have conflicting contributions to the parent. In such cases, the developer can use the expertise about NFRs, the domain and development as well as other knowledge in order to resolve conflicts. More on conflict resolution can be found in [8], [9].

6.2 Fuzzy Logic based Inference systems

Earlier in chapter three of this report, we have noted that due to the very nature of soft (goals), accurate satisfaction level measurement is not possible. Take the goal "Maintain availability of a system" as an example, for a mission critical system anything less than 100% availability is considered as failure while to an online based game, 95% availability can be considered as acceptable level of availability.

In this kind of vague ideas, fuzzy logic can be applied. Fuzzy logic, based on fuzzy sets proposed by Zadeh [27], [30], uses a concept of membership function to determine the membership value of a certain input to a set. Quite contrary to conventional crisp sets, a given value can belong to two fuzzy sets in fuzzy logic. We already present more detailed discussion on fuzzy logic in section 2.5; we will focus only on the application of fuzzy reasoning on goal models.

Due to the fuzzy nature of goal satisfactions, goal satisfaction values can be assigned in to two different sets. As an example, consider a goal that is perceived both as satisfied and partially satisfied from different perspectives as shown in figure 6.2. Fuzzy logic has got its applications in requirements engineering especially in reasoning with non functional requirements. In fact, Maurício Serrano et.al in [17] shows a fuzzy logic extension of NFR framework for a dynamic management of non functional requirements management. But the approach in [17] doesn't consider the fuzzy nature of goal contribution types. This limitation is prevalent in areas where relation between two goals is fuzzy just like the satisfaction level of goals themselves.

Consider the relation between, security measures and system performance. Under normal circumstances establishing tight security (increasing the security measure) will affect the performance of the system (resulting lower response time). Hence it is logical to expect the relation between Security measure goal and System Performance to be negative contribution types.

But the security measures can be made tighter in a way which leaves the system performance intact. For instance the new security measure may be tighter but may use much more efficient algorithms or the new system had a faster machine. In this kind of situation the contribution relation between the two goals will be then "Undetermined" or "no effect".

Figure 6.3, shows how these kind of vague goal contribution relations are represented using fuzzy logic sets and membership functions just like the goal satisfaction levels of figure 6.2.

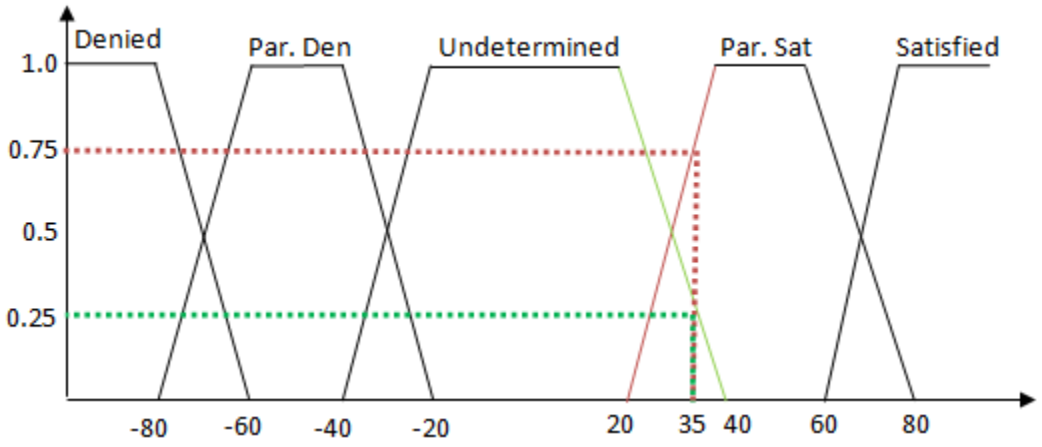

Figure 6.2: A goal, whose satisfaction level belonging to two different fuzzy sets.

The fuzzy sets used in figure 6.2 and 6.3 are trapezoidal fuzzy sets. Obviously, these set gets their name from their shapes. There are also other fuzzy sets, whose membership functions can be termed as triangular, Gaussian etc. (see chapter 2.5 for more details).

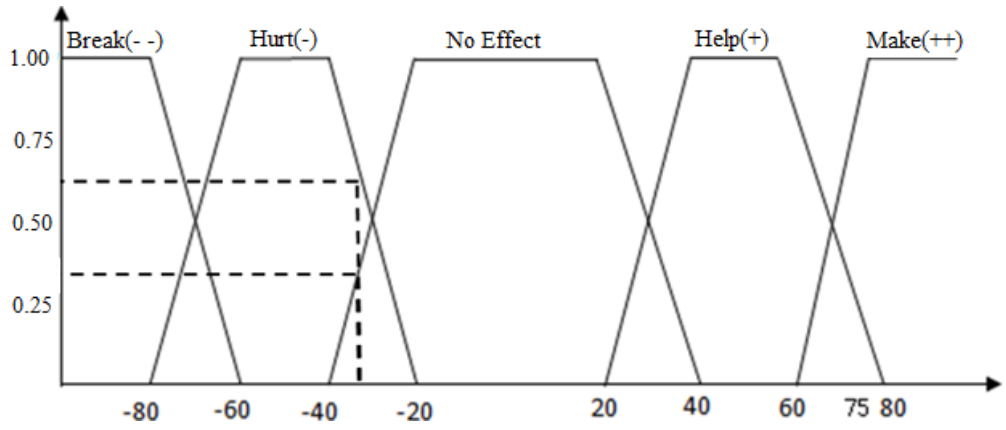

Figure 6.3: A contribution relation, whose satisfaction level belonging to two different fuzzy sets.

We select trapezoidal approaches because there are certain goals where a certain range of values can be considered as the same. For instance, two goals achieved 97% and 99% respectively can be both considered as fully achieved. Trapezoidal fuzzy membership functions are suitable for this kind of fuzzy sets. The usability of fuzzy trapezoidal fuzzy sets is also found applicable and used for extending the NFR framework in [17].

6.2.1: Fuzzification of goal and contribution relation values

Both i* and NFR states a goal can have a satisfaction value ranging from -100 to 100 [9], [15]. As we noted earlier (Figure 6.3), contribution relations are also subject to vague values. Hence we decide to adapt the concept of vague goal satisfaction levels to goal contribution relations also by assigning values from -100 to 100 to goal contributions relations.

A trapezoid, as shown in figure 5.5 has four vertices that can be used to specify the trapezoid.

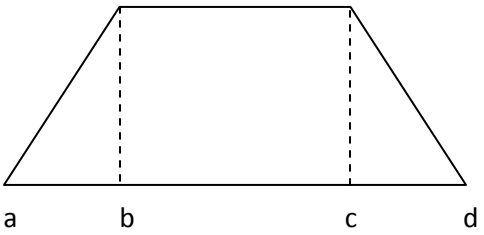

Figure 6.4: A typical trapezoid specified using four vertices.

From now on, we refer to a trapezoid by its four vertices. Using these four vertices the goal satisfaction levels and goal contribution relations can be assigned to different trapezoidal fuzzy sets as shown in tables 6.4 and 6.5 shown below.

Goal Satisfaction Level	Value on scale from -100 to 100
Fully Denied only	-100 to -80
Fully Denied and Partially Denied	-80 to -60
Partially Denied only	-60 to -40
Partially Denied and Undetermined	-40 to -20
Undetermined only	-20 to 20
Undetermined and Partially Satisfied	20 to 40
Partially Satisfied only	40 to 60
Partially Satisfied and Fully Satisfied	60 to 80
Fully Satisfied only	80 to 100

Table 6.4: Goal satisfaction levels assigned to various trapezoidal fuzzy sets.

Relations Contribution Level	Value on scale from -100 to 100
Break (- -) only	-100 to -80
Break (- -) and Hurt (-)	-80 to -60
Hurt (-) only	-60 to -40
Hurt (-) and No effect	-40 to -20
No Effect	-20 to 20
No Effect and Help (+)	20 to 40
Help (+) only	40 to 60
Help (+) and Make (++)	60 to 80
Make (++) only	80 to 100

Table 6.5: Contribution types assigned to various trapezoidal fuzzy sets.

In addition to the contribution types discussed in table 6.5, Goal can be decomposed based on AND/OR decomposition guidelines. AND decomposition propagates the minimum value of the contributing goals satisfaction level while OR propagates the maximum value of the contributing goals satisfaction level.

The choice of the boundary values in tables 6.4 and 6.5 is not a random choice. We took the values used in [17] as inspiration and make the sets more relaxed to allow accommodation of various level of goal satisfaction levels. BiZZdesign stakeholders also don't complain on these selections but insist that these values should not be fixed. As we reported earlier, these boundary values are never meant to be final; rather they can be adjusted depending on the interest of users and the domain under study.

Once goal satisfaction and contributions relation trapezoidal sets are specified, Goal satisfaction and contribution strength crisp values (like 75 and -50) should be changed to membership values. The membership values are obtained by following three steps.

i. Find the equation of each line in the figure 6.4 and 6.5.
ii. Identify to which line the specified crisp input belongs to
iii. Find the membership value from the equation of the line identified in step ii using the crisp input as an x value.

A point we would like to stress again is that a given crisp value, like a goal whose satisfaction level is 75, can be assigned to a maximum of two membership functions as shown in figure 6.5. In this case, the three steps mentioned above should be performed again to find the second membership value.

As an example, let' see how the fuzzification of a goal whose crisp value is 75 can be calculated.

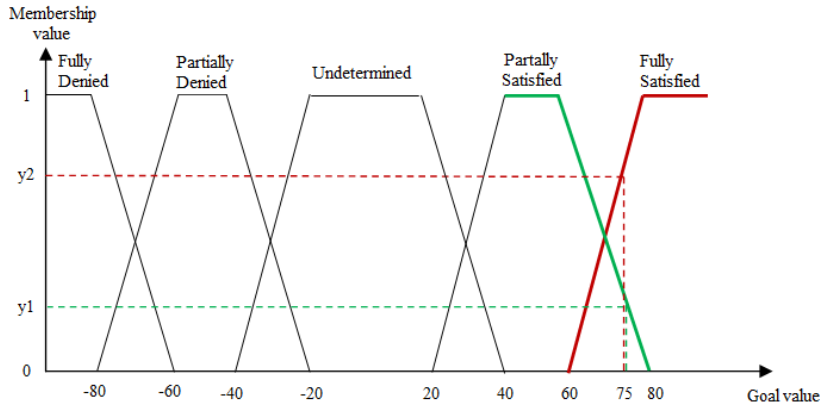

Figure 6.5: Identifying the membership functions of a crisp input.

From the ranges specified in table 6.4, an input of 75 belongs to a region where both partial satisfaction and Full satisfaction membership values are possible.

Using points (60, 0) and (80,1) to find the linear equation of the diagonal line in Fully Satisfied region (Red line), we will have :

$$\text{m (Slope)} = \frac{Y2-y1}{x2-x1} = \frac{1-0}{80-60} = \frac{1}{20} = 0.05$$

Then the equation of the line, using linear equation formula, will be:

y = mx+b
y = 0.05x+ b

To find b, we can use either of the points. Let's use Point (60, 0).

y = 0.05x+ b
0 = 0.05*60 + b
0 = 3 + b
➔ b = -3

The equation of the line will be then: y = 0.05x - 3

Using this equation, the membership value of a goal (with satisfaction level is 75) to the fuzzy set "Fully satisfied" as shown in figure 5.7 can be determined as:

y_2 = 0.05x - 3
y_2 = 0.05*75 - 3
y_2 = 3.75 – 3
y_2 = 0.75.
i.e. the goal whose satisfaction level is 75, belongs to "Fully Satisfied" fuzzy set with a membership value of 0.75.

The value of y_1, which is the membership value of a goal to the set partially satisfied (the green one) can also be determined to be 0.25 using a similar approach.

6.2.2: Adapting NFR rules for Fuzzy reasoning tools

From figures 6.5, we remember that a given goal satisfaction level can belong up to two different fuzzy sets. Since we have five fuzzy sets for goal satisfaction levels, we will have a maximum of 2*5 = 10 possible cases that a single crisp input value will be assigned to the fuzzy membership functions.

Similarly, from figure 5.4, a given influencing relation can also belong to two different fuzzy sets. Since there are five fuzzy sets for satisfaction levels, we will also have 2*5 = 10 maximum possible cases for influence relation types.

The NFR framework reasoning rules works by combining the goal satisfaction levels (10 possible cases) and contribution relations (another 10 possible cases). Hence if we adopt the NFR style fuzzy reasoning rules we will have:

10 * 10 = 100 possible rule combinations in NFR based fuzzy reasoning engine.

Diagrammatically, this can be depicted as shown in figure 6.6.

The rule combinations in figure 6.6 don't show the outcome of applying these rules. The following table shows the outcomes of the first five rules of figure 6.6. All of the fuzzy reasoning rules and their outcomes can be found in Appendix A in detailed Algorithm explanation section. Remember that both GoalA and RelA each can have two membership values (value 1 and 2) that can belong to any of the 5 different sets specified in tables 6.4 and 6.5.

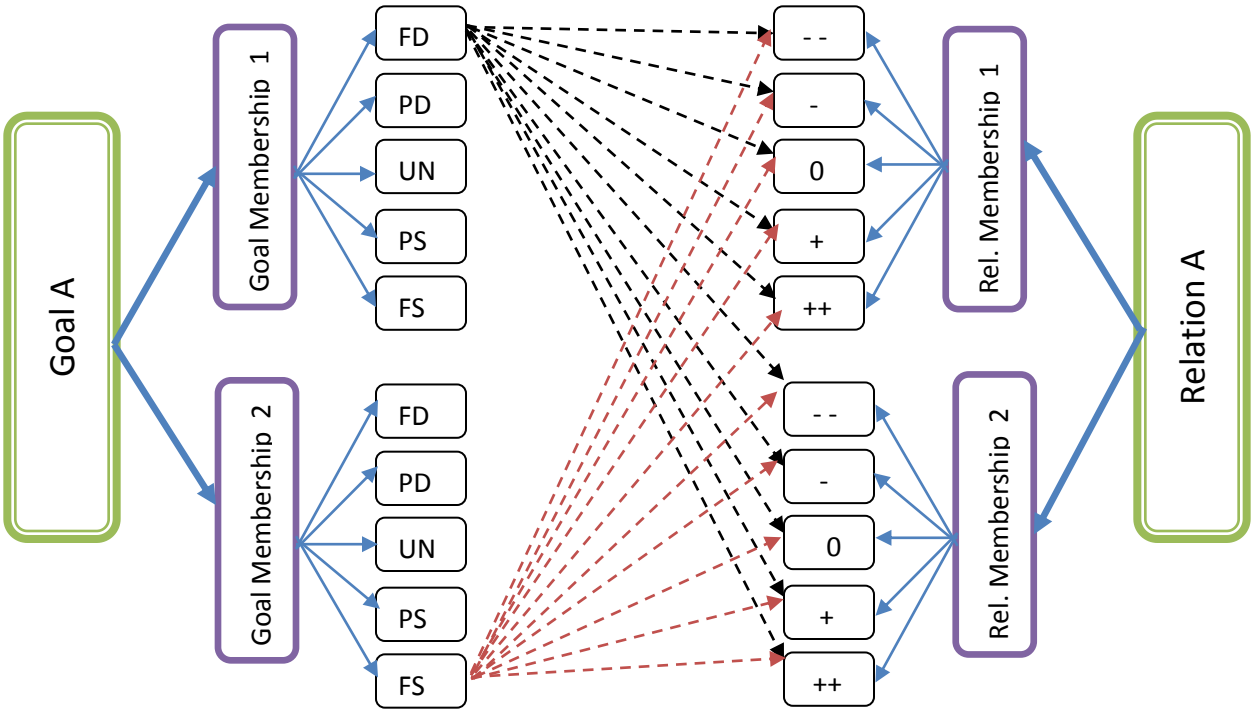

Figure 6.6: Twenty of the hundred possible rule combinations in NFR based Fuzzy logic reasoning.

RelaA M.Ship 1 / GoalA M.Ship 1	**BREAK(- -)**	**HURT(-)**	**NEUTRAL(0)**	**HELP(+)**	**MAKE(++)**
Fully Denied (FD)	Fully Satisfied	Partially Satisfied	Neutral	Partially Denied	Fully Denied
Partially Denied (PD)	Partially Satisfied	Partially Satisfied	Neutral	Partially Denied	Partially Denied
Neutral (Un)	Neutral	Neutral	Neutral	Neutral	Neutral
Partially Satisfied (PS)	Partially Denied	Partially Denied	Neutral	Partially Satisfied	Partially Satisfied
Fully Satisfied (FS)	Fully Denied	Partially Denied	Neutral	Partially Satisfied	Fully Satisfied

Table 6.6: Sample rules for goal satisfaction levels.

6.2.3: Fuzzy Rule application

This is the step where selected applicable fuzzy rules will be applied to the membership values obtained in step two. The fuzzy values to be applied are selected from the hundred fuzzy rules shown in figure 6.6. Example of such rules looks like:

- If Influencing goal is Full Satisfied **AND** Relation is Break Then Effect is Fully Denied
- If Influencing Goal is Partially Denied **AND** Relations is Help then Effect is Partially Denied.
- If influencing goal is Fully Denied **AND** Relation is Hurt then Effect is Partially Satisfied.
- If a goal is AND decomposed to n sub goals then Effect is the minimum of the satisfaction of the sub goals.
- If a goal is OR decomposed to n sub goals then Effect is the maximum of the satisfaction of the sub goals
- etc…..

Note that the majority of the fuzzy rules used in our algorithm are AND combination of goal satisfaction levels and contribution relation strengths (e.g. the first three fuzzy rules above). The basic rule of combining this kind of AND combination fuzzy rules is taking the minimum of the goal satisfaction and the relation membership values [30].

For instance for the rule:

*If influencing goal is Fully Denied **AND** Relation is Hurt then Effect is Partially Satisfied*

If the Fully Denied membership value is 0.7 and the Hurt relation membership value is 0.4, then the effect will be propagating partially satisfied effect whose membership value is the minimum of 0.6 and 0.4 which is 0.4.

6.2.4: Fuzzy Rule Aggregation

So far we have covered two vital steps in fuzzy logic reasoning engines. The fuzzification of crisp inputs to membership values and application of fuzzy rules on this membership values. We will see now how we can aggregate different rule results in to a single result. This is crucial when there are more than one goal influencing a goal understudy and we need to know the aggregate effect of the individual combinations.

As we have seen in section 2 of this report, there are a number of techniques to aggregate multiple fuzzy rules. For the sake of lower complexity and acceptable level of accuracy, we will use the average of available impacts to aggregate multiple fuzzy rules.

6.2.5: Defuzzification of Fuzzy Rule Results

Enterprise Architecture users and other business stakeholders are interested neither in fuzzy logic approach or the membership values specified in section 6.2.3. What they really want is a crisp numbered result in the range of -100 to 100. The process of changing fuzzy rule results like 0.75 and 0.6 to concrete goal satisfaction results is termed as Defuzzification.

As we have noted in section 2, we have selected "Weighted average defuzzification technique" for the sake of simplicity, efficiency and reasonable level of accuracy. But this weighted average technique is only applicable if the individual fuzzy sets have symmetrical shape. The trapezoidal membership functions we see in the previous subsections are symmetrical most of the time. But at the extreme ends, (Fully satisfied and fully denied values), there are membership sets whose shapes are asymmetrical.

A possible way to handle this situation is to use a modified version of weighted average technique called center of sums method. The center of sums method is similar to the weighted average method except in the center of sums method the weights are the areas of the respective membership functions whereas in the weighted average method the weights are individual membership values [30]. Fig. 6.7 shows an example of center of sum methods.

$$\text{Crisp Output} = \frac{c1*m1+c2*m2}{m1+m2}$$

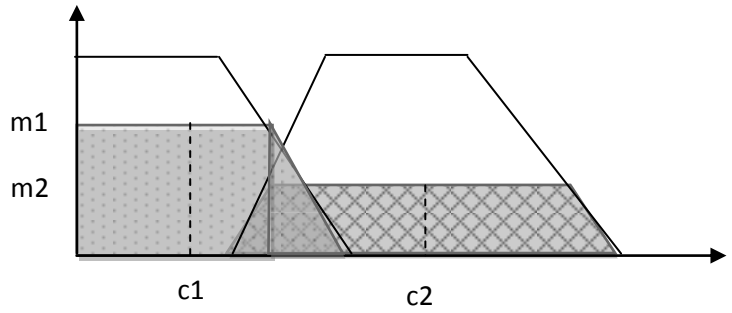

Figure 6.7: Weighted average method to defuzzify two fuzzy sets

Note that the values c1 and c2 are the centroid of the two shaded regions in figure 6.7. Their values can be computed by finding the x value of a centroid point of any trapezoid. This can be done by using the formula [46]:

$$\text{Centroid } Cx = \frac{2ac+a^2+cb+ab+b^2}{3(a+b)}$$

Where the values of a, b and c are shown in figure 6.8

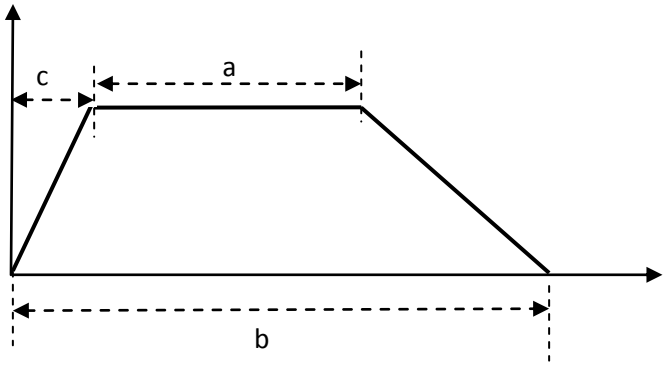

Figure 6.8: Finding the Centroid of a trapezoid

6.2.6 NFR based fuzzy reasoning Algorithm

The full Algorithm of fuzzy logic based reasoning tool for goal analysis is too vast to be covered here and we only present a vey abstracted view of the algorithm here. But the reader may expect individual steps in the algorithm will encompass some of the steps fuzzy logic reasoning steps already discussed.

For each goal determine the satisfaction level membership value

For all goal g in Goal graph G {

```
if ( !isLeaf(goal) ) {
    satList = List();

    forall "InfluenceRelation" relation in goal.relationsTo() {
      sourceObject = goal.relatedTo(relation);
       if  (sourceObject is "MotivationGoal"){
         gInput = sourceObject.attrValue("satisfactionLevel");
         rInput = relation.attrValue("infDetailedType");
         goalValue = GoalBelongsTo(gInput);// determine the goal satisfaction membership value
        relationValue = RelationbelongsTo(rInput); );// determine the relation strength membership value
         finalMembershipValue = applyFuzzyRules(goalValue,relationValue);
         crispResult = DeffuzzifyWeightedAverage(finalMembershipValue); // change to crisp output
       satList.add(crispResult);
      }
    }
     finalValue = AverageList(satList);
     goal.setAttrValue("satisfactionLevel", finalValue);
  }

  if ( !isLeaf(goal) ) {
    andList = List();

    forall "InfluenceRelation" relation in goal.relationsTo() {
      sourceObject = goal.relatedTo(relation);
      // check here if source object is AND Junction
      if  (sourceObject is "AndJunction"){
       forall "InfluenceRelation" r in sourceObject.relationsTo() {
          g = r.relatedTo(sourceObject);
          andList.add(g.attrValue("satisfactionLevel"));
          }
           representant = min(andList); // AND propagates the minimum value
          goal.setAttrValue("satisfactionLevel", representant);
      }
    }
  }
```

```
if ( !isLeaf(goal) ) {
  orList = List();
  forall "InfluenceRelation" relation in goal.relationsTo() {
    sourceObject = goal.relatedTo(relation);
    if  (sourceObject is "OrJunction"){
     forall "InfluenceRelation" r in sourceObject.relationsTo() {
        g = r.relatedTo(sourceObject);
        orList.add(g.attrValue("satisfactionLevel"));
     }
       representant = max(orList);// OR propagates the maximum value
      goal.setAttrValue("satisfactionLevel", representant);
    }
  }
}
```

Listing 6.1: Abstracted fuzzy reasoning algorithm

6.2.7 Benefits of NFR based fuzzy logic analysis

The main advantage of using NFR based fuzzy logic analysis is the ability to reason on crisp inputs. Though the TROPOS's has a qualitative reasoning technique, the vague nature of goal satisfaction levels and contribution types is overlooked in TROPOS approach. Employing NFR based fuzzy reasoning engine will ease such problems.

6.2.8: Limitations of NFR based fuzzy logic analysis

The main limitation of the NFR based fuzzy logic approach is that it highly relies on the user inputs which some of the users find it difficult to respond to. i.e. users may find it difficult to say the satisfaction level of a goal is 87 instead of 88.

The other less severe limitation is the moderate complexity of the algorithm especially in the fuzzification and defuzzification process. Unless a more detailed advanced algorithm for these steps is carefully designed based on the membership functions, inconsistent results can occur. For instance we use Sum weighted average method of defuzzification instead of integration approach which makes the algorithm easier but will make the outcome less accurate.

6.3: Testing the Quantitative Reasoning on Industrial Case Study

Just like we did for the qualitative reasoning, we will present the result of two industrial case studies in this subsection. As a reminder, one of the test cases is a fictitious insurance company from The Open Group while the second one is a drinking water company in the Netherlands.

Only the goal models of the companies will be shown here since both case studies are adequately explained in section six of this report.

6.3.1: Quantitative Reasoning Case Study on Goal Models of ArchiSurance

The inputs selected for the qualitative reasoning are again assumed values that approximate the given scenario. The next chapter of the report will compare the results of the approaches based on real cases. To make this comparison fair, the inputs were also selected to match the qualitative inputs used in table 6.5 of chapter 6.

Leaf Goal	Initial Value
It Budget	85
Staff Size	75
Reliability Demand	50
Multiple Options for Customers	75

Table 6.7 Sample inputs for the ArchiSurance goal model

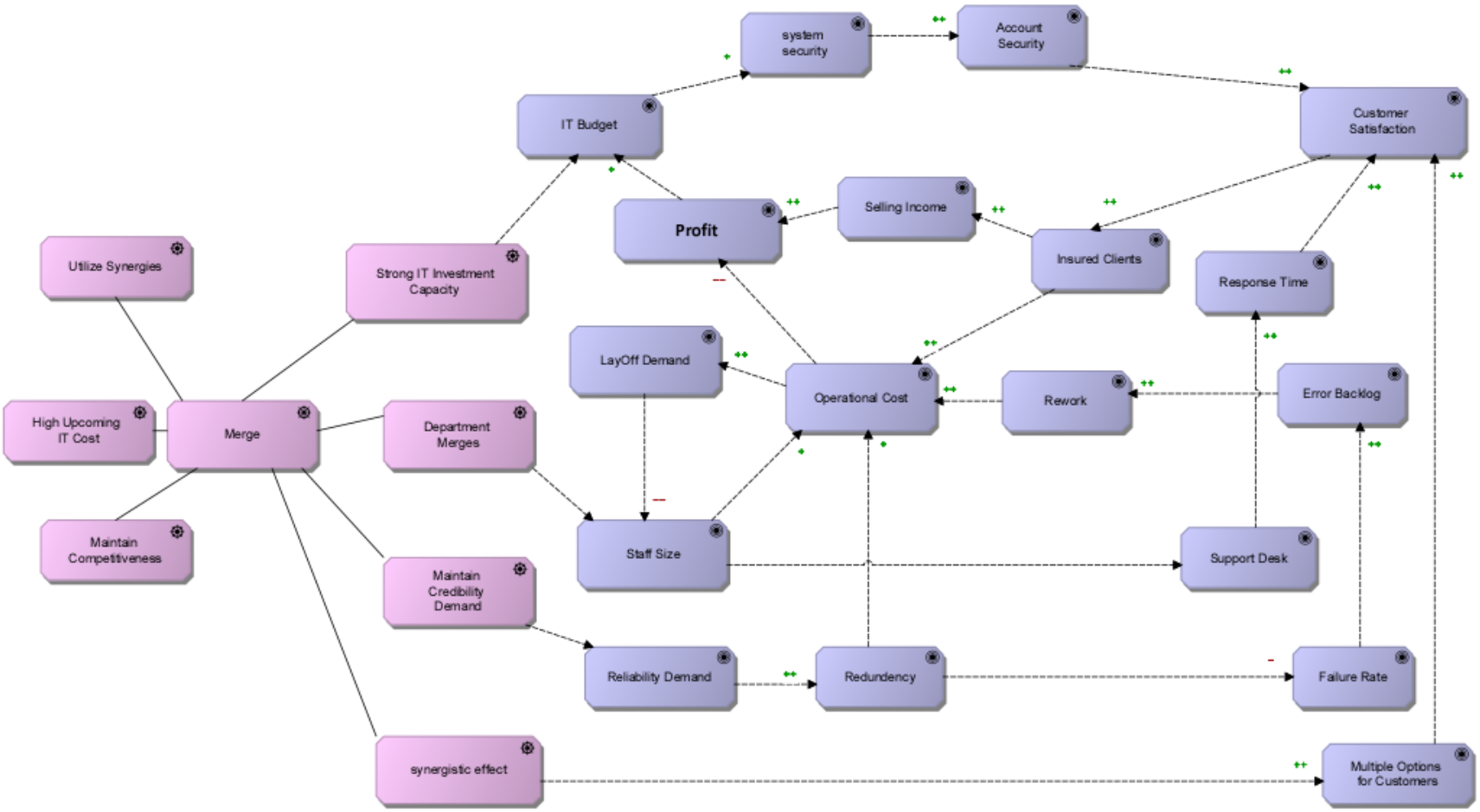

Figure 6.9 ArchiSurance goal model.

Again to make the comparison with the qualitative reasoning approach, our tool was set to traverse seven goal influence relations. The result of the qualitative reasoning tool is shown in table 6.8 below.

Goal Name	Satisfaction Level
Profit	0
Insured Clients	8.557607008
Common IT Infrastructure	0
Operational Cost	0
Rework	-50
Failure Rate	-50
Selling Income	0
Support Desk	0
Response Time	0
Reliability Demand	50
Error Backlog	-50
Customer Satisfaction	23.79781421
Account Security	0
system security	0
IT Budget	0
Multiple Options for Customers	75
Layoff Demand	0
Staff Size	0
Redundancy	50

Table 6.8: ArchiSurance quantitative goal model results

These results are reasonably consistent with the results of the qualitative approach presented in table 5.6. To what extent the two results are consistent and the major difference observed will be discussed in chapter 8.

6.3.2: Quantitative Reasoning Case Study on Goal Models of a Water Company

This is the second case study we conduct on the water company whose goal model is shown in figure 6.10. The first one was a qualitative reasoning case study explained in chapter six. The values assigned as inputs for this case study are also set to be similar to the ones used in the qualitative reasoning. This helps in making the comparison of the qualitative and quantitative approaches fair.

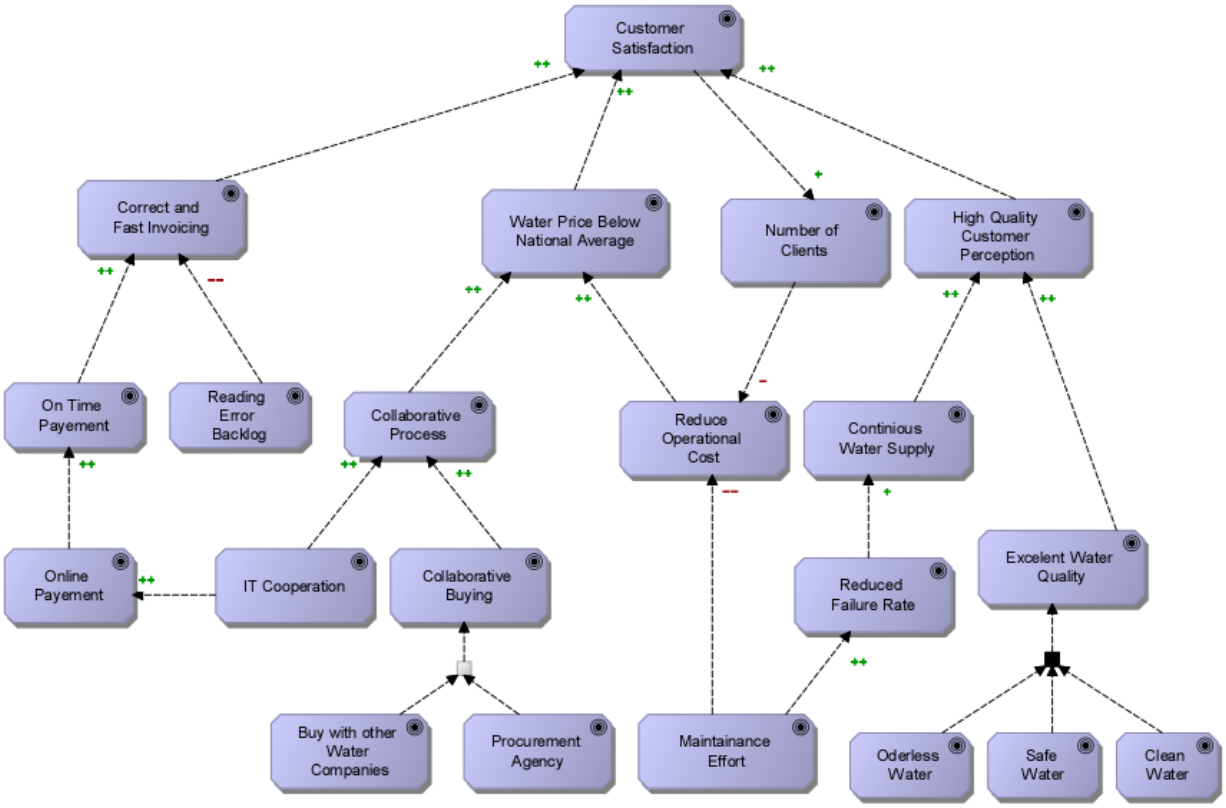

Figure 6.10: Goal model of the water company

Leaf Goal	Satisfaction Level
Buy with other water companies	-50
Procurement Agency	75
IT Cooperation	50
Maintenance Effort	-75
Reading Error Backlog	10
Odorless Water	75
Safe Water	85
Clean Water	85

Table 6.9: Input values for leaf goals satisfaction levels

Goal Name	Satisfaction Level
Excellent Water Quality	75
Collaborative Buying	75
Customer Satisfaction	7.673200176
Correct and Fast Invoicing	25
Water Price Below National Average	42.53767619
Number of Clients	0
High Quality Customer Perception	10.69672131
Reduce Operational Cost	35.69672131
On Time Payment	50
Online Payment	50
IT Cooperation	50
Collaborative Process	60.69672131
Maintenance Effort	-75
Continuous Water Supply	-50
Reduced Failure Rate	-71.39344262
Odorless Water	75
Safe Water	85
Clean Water	85
Procurement Agency	75
Buy with other Water Companies	-50
Reading Error Backlog	10

Table 6.10: The result of the fuzzy logic analysis on goal satisfaction levels.

Just like the ArchiSurance case study, these results are reasonably consistent with the results of the qualitative approach presented in table 5.8. The next chapter will elaborate more to what extent two results are consistent.

6.4: Summary

Quantitative reasoning techniques use numeric specifications like a goal is 90% Satisfied or 50% denied to reason on goal satisfaction levels. The influence relations are also specified in numeric formats like 70% positive contribution relation or 30% negative contribution relation.

Fuzzy reasoning is selected to be used for quantitative reasoning on goal satisfaction values because of its applicability in fuzzy concepts like satisfaction levels of soft goals. The rule base of the quantitative reasoning engine uses adapted definitions of contribution rules from the NFR framework.

The fuzzy reasoning engine assigns membership values for the initial goal satisfaction and relation strength values. It then uses these membership values as inputs to the inference engine to determine the resultant goal satisfaction values. The results of the inference engine are also membership values in the range 0 to 1. Finally, these values will be changed to goal satisfaction values on the range -100 to 100.

A reasoning algorithm and a prototype application of this fuzzy inference engine are developed. The two test cases conducted on this prototype application shows that it is indeed possible to use fuzzy reasoning engine to analyze indirect influence relations in EA goal models. The test cases also reveal that quantitative reasoning techniques are capable of providing discrete and detailed goal satisfaction values.

The third research question of this master project is *"How can we simulate influence relation impacts on goal satisfaction values and how can we visualize the simulation?"* This chapter provides a second answer to this research question by presenting a quantitative reasoning engine algorithm and a tool support to analyze goal change impacts.

7. Comparison of Qualitative and Quantitative approaches

This chapter presents the comparison of qualitative and quantitative approaches. The comparison is based on the ArchiSurance and water company test case results presented in the chapter five and six. Based on the results of the comparison, merits and demerits of each method will also be presented.

Due to their difference in describing goal satisfaction levels (TROPOS uses textual descriptions and Fuzzy logic uses numeric values), it is difficult to have a common comparison criteria. Yet, it is still logical to assume high positive values of fuzzy approach (like 85) will represent availability of Full (F) evidence for deniability of a goal. Similarly, a partially (P) satisfied goal in qualitative reasoning approach is a good analogy for a goal whose satisfaction level is approximately 50 in the quantitative approach. Table 7.1 shows the complete mapping of textual goal values to their numeric counterparts.

Qualitative Goal values	**Corresponding Quantitative goal values**
-100 to -80	Fully Denied
-80 to -60	Mostly Denied
-60 to -40	Partially Denied
-40 to -20	Little Denied
-20 to 0	Not Denied
0 to 20	Not Satisfied
20 to 40	Little Satisfied
40 to 60	Partially Satisfied
60 to 80	Mostly Satisfied
80 to 100	Fully Satisfied

Table 7.1: Goal value mapping between qualitative and quantitative specifications.

7.1: Comparing results of Water Company Case Study

Using the mapping in table 7.1, the inputs of the qualitative and quantitative studies of Water Company was is set to be equivalent. Table 7.2 shows the inputs used for analyzing water company goal satisfaction values.

Leaf Goal	Qualitative Satisfaction Level	Quantitative Satisfaction Level
Buy with other water companies	Partial Evidence of Deniability	-50
Procurement Agency	Much Evidence of Satisfiability	75
IT collaboration	Partial Evidence of Satisfiability	50
Maintenance Effort	Much Evidence of Deniability	-75
Reading Error Backlog	Little Evidence of Satisfiability	10
Odourless Water	Much Evidence of Satisfiability	75
Safe Water	Full Evidence of Satisfiability	85
Clean Water	Full Evidence of Satisfiability	85

Table 7.2: Inputs for qualitative and quantitative reasoning approaches of water company case study.

Using these logically equivalent inputs, the results of the two approaches were compared. As can be seen in table 7.3, the results obtained from the two methodologies are fairly consistent. For instance the goal "Collaborative Process" is predicted as partially satisfied in the qualitative approach and an equivalent value of 60.7 in the quantitative approach.

Goal Name	Qualitative Satisfaction Level	Quantitative Satisfaction Level
Excellent Water Quality	Mostly Satisfied	75
Collaborative Buying	Mostly Satisfied	75
Customer Satisfaction	Not Satisfied	7.673200176
Correct and Fast Invoicing	Not Satisfied	25
Water Price Below National Average	Little Satisfied	42.53767619
Number of Clients	Not Satisfied	0
High Quality Customer Perception	Not Satisfied	10.69672131
Reduce Operational Cost	Little Satisfied	35.69672131
On Time Payment	Partially Satisfied	50
Online Payment	Partially Satisfied	50
IT Cooperation	Partially Satisfied	50

Collaborative Process	Partially Satisfied	60.69672131
Maintenance Effort	Mostly Denied	-75
Continuous Water Supply	Partially Denied	-50
Reduced Failure Rate	Mostly Denied	-71.39344262
Odourless Water	Mostly Satisfied	75
Safe Water	Fully Satisfied	85
Clean Water	Fully Satisfied	85
Procurement Agency	Mostly Satisfied	75
Buy with other Water Companies	Partially Denied	-50
Reading Error Backlog	Little Satisfied	10

Table 7.3: Predicted satisfaction level of goal models for the water company case study.

A notable difference is the presence explicit of goal conflict detection in TROPOS based approach. Fuzzy logic based reasoning engine rule aggregation feature is useful in combining two or more influence effects but the combined result can be misleading when there are conflicting goal influences.

Take the goal "High Quality Customer Perception", it receives negative influence from continuous water supply and a positive influence from excellent water quality. But a satisfaction level of 10 can also be an effect of a goal which satisfied to small extent. This can make fuzzy logic based reasoning approach vague and less useful in decision making process.

The TROPOS approach can identify little, partial and full conflicts. As an example, the goal "Correct and Fast Invoicing" receives little conflicting contributions due to the conflicting goal influences from "Reading Error Backlog" and "On Time Payment" goals.

7.2: Comparing Results of ArchiSurance Case Study

The procedures used in comparing the results of ArchiSurance case study were the same as that of the water company discussed above. Again only the inputs (table 7.4) and outputs (table 7.5) of the case study will be presented here to avoid repetition. Relatively new inconsistencies between these approaches, which are observed in this case study, will also be presented at the end.

Leaf Goal	Initial Value	Initial Value
It Budget	Full Evidence of Satisfiability	85
Staff Size	Much Evidence of Satisfiability	75
Reliability Demand	Partial Evidence of Satisfiability	50
Multiple Options for Customers	Much Evidence of Satisfiability	75

Table 7.4: Inputs for qualitative and quantitative reasoning approaches for ArchiSurance cases study

Goal Name	Qualitative Satisfaction Level	Quantitative Satisfaction Level
Profit	Not Satisfied	0
Common IT Infrastructure	Partially Satisfied	0
Operational Cost	Not Denied	0
Insured Clients	Not Satisfied	8.557607008
Rework	Partially Denied	-50
Failure Rate	Partially Denied	-50
Selling Income	Not Satisfied	0
Support Desk	Not Denied	0
Response Time	Not Denied	0
Reliability Demand	Partially Satisfied	50
Error Backlog	Partially Denied	-50
Customer Satisfaction	Not Satisfied	23.79781421
Account Security	Not Satisfied	0
system security	Not Satisfied	0
IT Budget	Not Satisfied	0
Multiple Options for Customers	Mostly Satisfied	75
LayOff Demand	Not Denied	0
StaffSize	Not Denied	0
Redundancy	Partially Satisfied	50

Table 7.5: Predicted satisfaction level of goal models for the ArchiSurance case study.

The outputs of these two approaches are also reasonably consistent. But the degree of consistency is a little bit lower than the result observed in water company case study comparison (table 7.3). As an example, the first three goals in the table 7.4 are predicted to have zero satisfaction level in the quantitative approach while they are termed as “Not Sat”, “Partially Sat” and “Not Den” in the qualitative approach.

Table 7.5 also shows that zero levels of satisfaction is (i.e. a goal is not satisfied and not denied) occurs frequently than the previous case study. This can be a result of the aggregation effect of multiple contribution types. These strengths the conclusion we took from the previous case study: fuzzy logic based reasoning can lead to ambiguous goal conflict satisfaction levels due to aggregation effects of fuzzy reasoning engines.

The presence of a number of zero levels of satisfaction for goals is also consistent with the claims of KAOS authors’ opinion about current goal satisfaction reasoning approaches. In [16], Letier and Lamsweerde claim that majority of goal satisfaction reasoning approaches tend to result undetermined level of goal satisfaction levels.

7.3: The verdict: Which approach is better?

The results showed in the previous subsections shows that both qualitative and quantitative approaches are indeed applicable in analyzing indirect influence relations. Both approaches have their own good and bad sides. The degree to which these applications are applicable can also depend on the targeted user.

Qualitative reasoning is easy to use and understand even by none technical users due to the textual descriptions of goal satisfaction levels. Besides, the conflict detection feature of TROPOS based qualitative approach makes it more usable in decision making activities.

Quantitative reasoning techniques have also their advantages. One of these advantages is their discrete expressive power of goal satisfaction levels in numeric format. Consider the goals "Customer Satisfaction" and "Correct and Fast invoicing" in table 7.3. Both are predicted to be "Not Satisfied" levels of satisfiability in the qualitative approach. But the degree to which they are satisfied is different as can be seen from the fuzzy logic approach (7.6 and 25). This will make quantitative reasoning techniques more useful in detailed goal analysis.

The presence of limitations on both sides will make it difficult to choose one of them as a winner approach. But from the discussions presented above it is reasonably logical to conclude:

- Both approaches can be used for direct and indirect goal influence reasoning.
- For high level goal analysis on goal models, TROPOS based approach can be a better choice. This is especially advantageous for nontechnical people who prefer natural language expression for reasoning on goal models.
- TROPOS based qualitative reasoning is also better in detecting/handling conflicting goal satisfactions. This makes it more applicable in goal models where conflicting stakeholder interests are frequent.
- For a deep investigation of goals and requirements, NFR based fuzzy reasoning approach will be a good choice. This is mainly due to the possibility of using extended range of goal satisfaction values (-100 to 100).
- Requirement engineers, policy makers and other technical people can benefit from the detailed analysis of quantitative approaches provided that they are capable of assigning concrete values to input goals of the goal model.

7.4 Summary

Two test cases are applied on the qualitative and quantitative reasoning techniques. The test case results show that both approaches are indeed applicable in analyzing indirect influence relations. Besides, both approaches provide reasonably consistent results.

The availability of two kinds of reasoning techniques poses a question, which approach should users chose to reason on a certain goal model?

From the results of the two case studies conducted on goal models, the qualitative reasoning technique is applicable for high level goal analysis. It is easily understandable because it uses textual specifications of goal satisfaction values. This makes it highly usable for non-technical people.

The quantitative approach is more useful in detailed and discrete level of goal analysis. It can be useful for policy makers, requirement engineers and other technical stakeholders. The users of the system however may face difficulties in assigning concrete satisfaction values for the input goals.

8. Goal Feedback Loops and Simulations

The qualitative and quantitative reasoning techniques discussed in section six and seven are based on direct (acyclic) influence relations and AND/OR decompositions. The behavior of feedback loops resulting from goal influences were not discussed. But from chapter two of this report, we remember that feedback loops are the primary sources of system dynamics that play a crucial role in determining the dynamics of EA goal models [6].

Goal influence feedback loops occur when a series of influence relations form cyclic influence paths in goal models. There are two types of feedback loops: positive and negative loops. If a goal satisfaction level is increased in a positive loop, the loop effect tends to increase the satisfaction level of goals. If the loop was a negative one, an increase in satisfaction level of a given goal will eventually decrease the satisfaction level of the goal.

This chapter will present an algorithm developed to reason on these feedback loops of goal influence relations. The algorithm first finds all the goals whose satisfaction levels are changed, compute the change percentage and propagate the change effect to the influenced goals.

The satisfaction values of goals in feedback loops can keep increasing, keep decreasing, converge to certain or can keep oscillating between two or more satisfaction values. The feedback loop algorithm presented in this chapter will also be able to detect whether a goal in a feedback loop will have a converging or diverging satisfaction level.

8.1: Goal Cycle Simulation Algorithm

From the two goal reasoning approaches presented in section six and seven, the quantitative approach is more suitable for goal feedback simulation since it provides a range of discrete goal satisfaction values that range from -100 to 100.

Actually it is practically impossible to simulate goal cycle effects in qualitative approach since there are only three levels of satisfaction between a “not satisfied” goal and “a fully satisfied” goal. These values are almost useless to feedback loop reasoning when compared to the infinite possibilities in the numbers between -100 and 100.

We will use a simple feedback loop shown in figure 8.1, to illustrate our approach to analyze the dynamic of goal cycles. On the scale from -100 to 100, the goal “Formal Verification effort” was assigned an initial value +60 and has a positive influence on “Testing Cost” whose initial value was +40. The testing cost in turn has a partial negative influence on the goal “Formal Verification Effort”.

The First question to be answered here is what will be the new satisfaction value of “Testing Cost” if we increase the satisfaction level of the “Formal Verification Effort” from +60 to + 85? A second question to complete the loop will be then what effect will changing the “Testing Cost” will have back on the “Formal Verification Effort”?

We will answer these questions step by step.

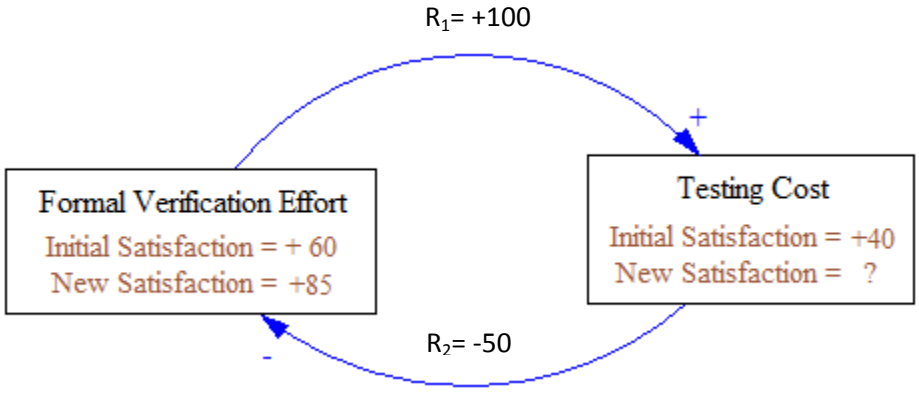

Figure 8.1: Simple goal feedback loop for illustrating change impact analysis algorithm.

1. Accept the initial goal values:
   In our case + 60 for "Formal Verification Effort" and + 40 for "Testing Cost".

2. Accept the relation strength values.
   For the sake of Simplicity let's assume R1 has a value of +100 and R2 has a value of -50.

3. Assign the new satisfaction level to the goal under change.
   In our case assume the "Formal Verification Effort" is increased from +60 to +85.

4. Compute the change as the percentage of the new satisfaction value.
   We are only interested in computing the effect of the change: the effect of changing the value of "Formal Verification Effort" from + 60 to + 70 on the satisfaction level of "Testing Cost". To compute the effect of the change, we need to find by how much is the influencing goal is changed.
   This can be calculated using the following equation:

$$Change\ \% = \frac{New\ Value - Old\ Value}{Old\ Value}$$

   e.g: Goal "Formal Verification Effort" is now set to have +70 Satisfaction level
   Now the change percentage can be calculated as:

$$\frac{85-60}{60} = 0.417$$

5. Compute to what extent the relation R1, will propagate the given change in satisfaction level.
   Relations strengths in the quantitative approach are also assigned a value from -100 to 100. -100 will propagate the inverse of the change (same magnitude different sign) while 100 will propagate

the change as it is. The other values between -100 and 100 will propagate some percentage of the change depending on the strength and sign of the influence relation.

e.g. R1 has a value of +100. This makes it possible to propagate the changed value as it is.
i.e. 0.417*100 = 41.7 Change value is propagated to the "Testing Cost" goal.

6. Adjust the new satisfaction level of "Testing Cost" based on the percentage in step 5.
New Sat Value = Existing value + Existing Value * Percentage in step 6.

NewSatValue (Testing Cost) = 40 + 40*41.7%
= 40 + 16.68
= 56.68

7. Once the new value is set to "Testing Cost", the initial and the new satisfaction level of the influencing goal will be set to the same new satisfaction values. This is necessary because the new value propagates its effect and it is not "new" anymore.

Figure 8.2: shows the effect of the steps we did so far for our simple example.

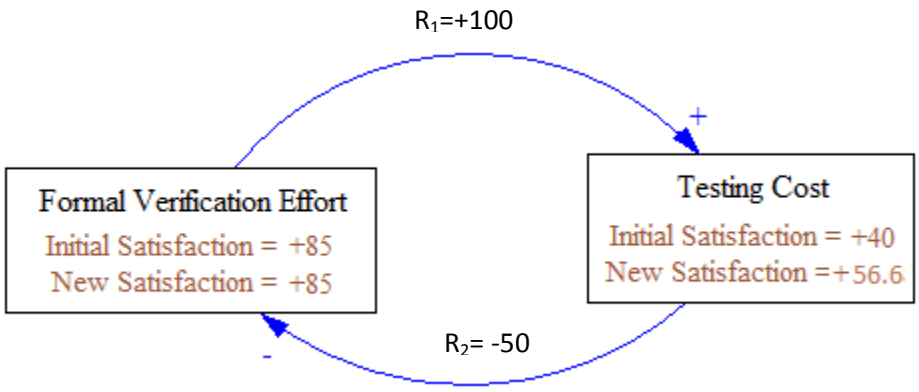

Figure 8.2: Effect of changing "Formal Verification Effort" on "Testing Cost" .

Note that once the "Testing Cost" is affected, the initial satisfaction of "Formal Verification Effort" should be set to the same value as the new satisfaction value since the effect of the new value is already propagated.

8. At this stage, "Testing Cost" satisfaction level is changed and it is time to calculate the effect of this change on "Formal verification Effort".

Using similar procedure as shown from step 1 to 7, we compute the new value of "Formal verification Effort" as follows:

$$Change\ in\ \text{testing Cost} = \frac{54.6-40}{40} = 0.415$$

This time the value of R2 is -50, which will propagate half to the change influence and invert the sign of the influence.

The propagated effect will be then 0.415 (-50) = -20.75

This is the percentage value that will affect the "Formal Verification Effort" goal:
i.e. New Satisfaction value of = Existing value + Existing Value * -20.75%
New Satisfaction value of "Formal Verification Effort" = 85 + -0.2075*85
New Satisfaction value of "Formal Verification Effort" = 85 – 17.6375
New Satisfaction value of "Formal Verification Effort" = 67.36

At this stage the goal cycle diagram will have the values shown in figure 8.3

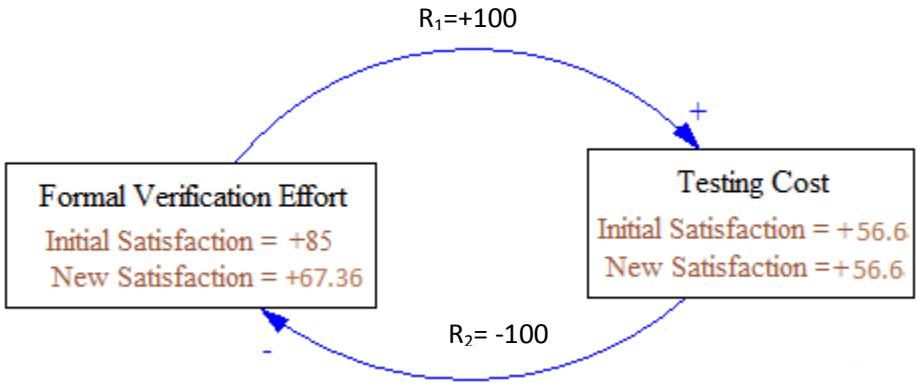

Figure 8.3: Effect of changing "Testing Cost" on "Formal Verification Effort" .

Note that once the "Formal Verification Effort" is affected, the initial satisfaction of Testing Cost" should be set to the same value as the new satisfaction value since the effect of the new value is already propagated.

9. The "Formal Verification Effort" will then start affecting "Testing Cost" due to the new change in its satisfaction value and the influence cycle continues.

10. The loop will continue until the users stop it (if it becomes diverging loop) or when both goals reach identical satisfaction levels (if the loop is converging loop). A goal can also exhibit oscillating (not converging, not diverging) goal satisfaction values.

Note that in these steps the effect of two or more goals influencing another goal in the goal cycle is not considered. In this kind of situations, the average of the individual contributions can be taken. To entertain the condition where different goals may have different priorities, a simple average or weighted average technique can be employed.

8.2: Goal Cycles Simulation Algorithm

An algorithm that reasons on cyclic goal influences needs the initial and new satisfaction levels as well as the influence relation strengths. Listing 8.1 below shows an algorithm to reason on influence relations. The algorithm is outlined based on the steps discussed in the previous sub section.

```
forall "MotivationGoal" g in model {
  fuzzyList = List();
  goalFuzzyEffect = 0;
  influencingObject = undefined;
  junction  = undefined;
   ANDcurrSatList = List();
          ANDoldSatList = List();
          ORcurrSatList = List();
          ORoldSatList = List();

   forall  "InfluenceRelation" r in g.relationsTo() {
    influencingObject = r.relatedTo(g);
     if (influencingObject is "MotivationGoal"){
       goal = influencingObject;
       if(goal.attrValue("satisfactionLevel")!=goal.attrValue("oldSatisfactionLevel")){
          changePercentage=(goal.attrValue("satisfactionLevel")-goal.attrValue("oldSatisfactionLevel"))
            changePercentage =        changePercentage/goal.attrValue("oldSatisfactionLevel");
          relationPropagatingEffect =  r.attrValue("infDetailedType");
          propagatingEffect = relationPropagatingEffect*changePercentage;
          finalEffect = g.attrValue("satisfactionLevel") +(g.attrValue("satisfactionLevel")*(propagatingEffect/100));
          fuzzyList.add(finalEffect);
         // output g, fuzzyList;
       }
     }
     else if(influencingObject is "AndJunction"){
       junction  = influencingObject;
       forall "InfluenceRelation" reln in junction.relationsTo() {
                          junctiongoal  = reln.relatedTo(junction);
                ANDcurrSatList.add(junctiongoal.attrValue("satisfactionLevel"));
                      ANDoldSatList.add(junctiongoal.attrValue("oldSatisfactionLevel"));
                   }
                 }

     else if(influencingObject is "OrJunction"){
      junction  = influencingObject;
       forall "InfluenceRelation" reln in junction.relationsTo() {
                          junctiongoal  = reln.relatedTo(junction);
                ORcurrSatList.add(junctiongoal.attrValue("satisfactionLevel"));
```

```
                    ORoldSatList.add(junctiongoal.attrValue("oldSatisfactionLevel"));
                }
    }
}

    //Find if there is any change in the AND decomposition
 if (ANDcurrSatList.empty() == false){
    if(minList(ANDcurrSatList)!= minList(ANDoldSatList)){
                     fuzzyList.add(min(minList(ANDcurrSatList), minList(ANDoldSatList)));
                    }
 }

 if (ORcurrSatList.empty() == false){
   if(maxList(ORcurrSatList)!= maxList(ORoldSatList)){
           fuzzyList.add(max(maxList(ORcurrSatList), maxList(ORoldSatList)));
           }
 }

 if(fuzzyList.empty() == false)   {
    ultimateResult = AverageList  (fuzzyList);
    g.setAttrValue("newSatisfactionLevel", ultimateResult);
  }
 }

 forall "MotivationGoal" goal in model {
 // setting the new value as an old value before the next round begins
  if (goal.attrValue("newSatisfactionLevel")==goal.attrValue("oldSatisfactionLevel")){
       goal.setAttrValue("oldSatisfactionLevel", goal.attrValue("satisfactionLevel"));
       goal.setAttrValue("newSatisfactionLevel", goal.attrValue("oldSatisfactionLevel"));
   }

   //Setting the effect of the previous previous step on to the satisfaction level of new goals
  else if (goal.attrValue("newSatisfactionLevel")!=goal.attrValue("satisfactionLevel")){
       goal.setAttrValue("satisfactionLevel", goal.attrValue("newSatisfactionLevel"));
       goal.setAttrValue("newSatisfactionLevel", goal.attrValue("oldSatisfactionLevel"));
  }
 }
```

Listing 8.1: an algorithm for reasoning on feedback loop of goal models

8.3: Test Case on Goal Cycle Simulations

Two case studied have been used throughout this report to validate the applicability of the indirect influence reasoning approaches. As shown in figure 6.1 and 7.1, the water company case study has only one cyclic influence path while the ArchiSurance case study has four goal cycles. This makes the ArchiSurance case study a better choice for validating the goal cycle simulation. To make goal cycle simulation case study easily understandable, we slightly change the ArchiSurance case study to the one shown in Figure 8.4.

In chapter one, decision support on resource allocation problem is stated as one of the benefits of indirect influence reasoning. The modified ArchiSurance case study will also be used to demonstrate the application of indirect influence reasoning in supporting decision making process. For this application we will use the scenario listed in section 8.3.1.

8.3.1: Sample Goal Change Scenario for ArchiSurance Case Study

ArchiSurance Company is established with a purpose of utilizing synergies and to maintain competiveness. Assume the newly appointed General Manager has a budget of €1,000,000 for the first quarter of the company. She has to decide how to spend the money in a way that generates the maximum possible benefit for the company (e.g. Score the highest possible profit for the company).

From figure 8.4, there are three possible ways of spending this budget:

i. Increase the IT budget and implement state of the art IT system,
ii. Maintain its employees (probably recruit more)
iii. Reengineer the business process of the old companies and establish new and multiple insurance options.

To simplify the case study, two assumptions will be taken.

First, the concept of delay in achieving the result is skipped here. Delays are usually, time, project and company specific which makes them relatively difficult to calculate. More on delays can be found in Appendix A.

Second, we will assume €1,000,000 is adequate enough to have either of a strong IT budget, to keep the current staff size (avoid layoff) or to reengineer the business processes (create multiple insurance options for customers). i.e. €1,000,000 is not enough to achieve more than one task.

Using the feedback loop reasoning approach discussed in the previous section, the consequences of the three cases are simulated. To make the comparison fair, the initial satisfaction values of all goals are set to be +50(partially satisfied).

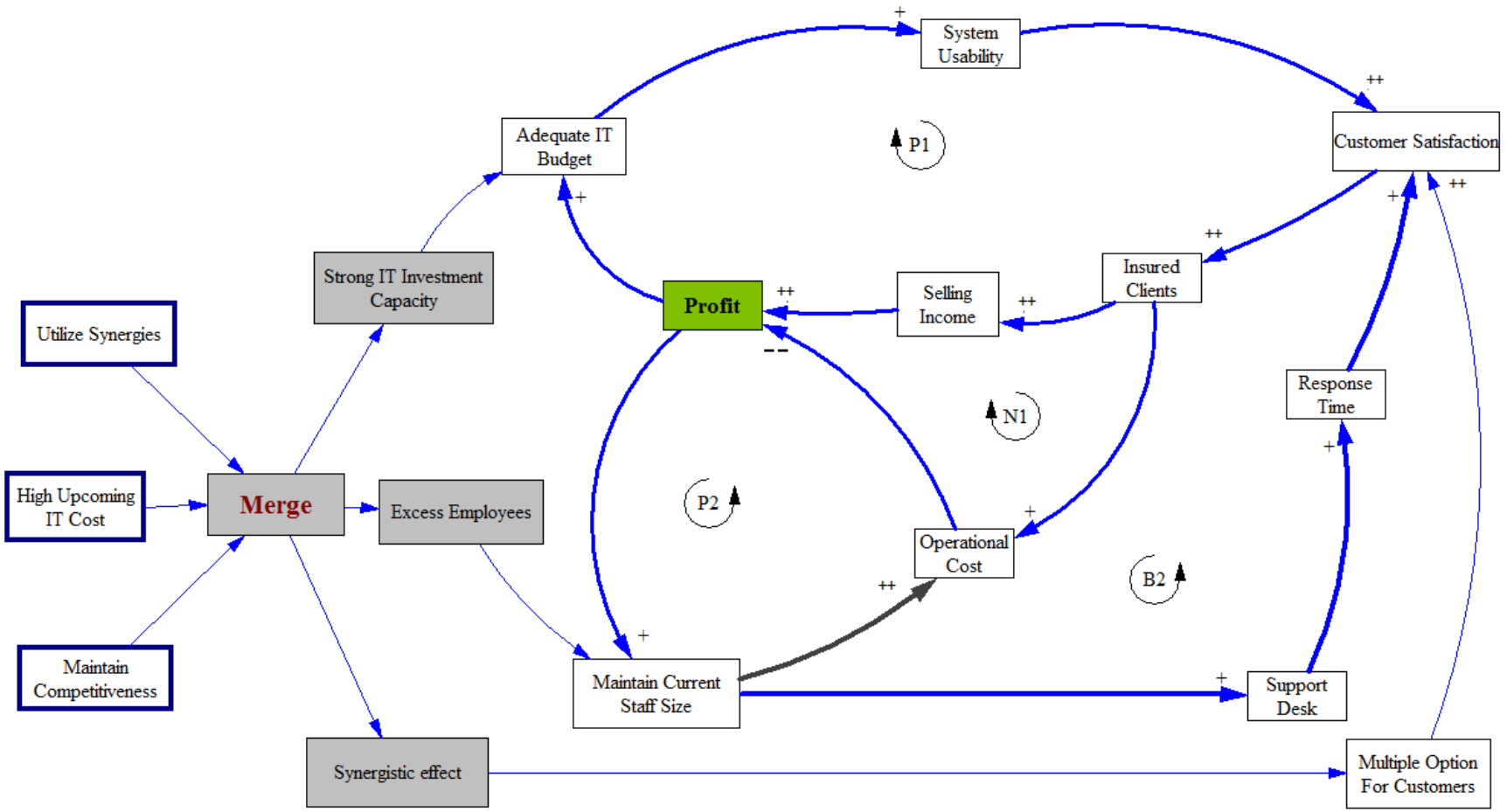

Figure 8.4: Simplified ArchiSurance goal Model

i. Budget Allocated for Enhancing the IT system of the Company

If the manager decides to invest on the IT system of the company, the IT budget will be fully satisfied. From the fuzzy sets used in chapter seven, a goal is fully satisfied if its satisfaction value is between 80 and 100. Let's take the satisfaction level of the goal to be 85. Table 8.1 shows the result of increasing the satisfaction value of "IT Budget" to 85 on the goals involved in the first positive loop (P1).

Goal Name	Satisfaction Level
Adequate IT Budget	85
System Usability	83.25
Customer Satisfaction	83.25
Insured Clients	79.925
Profit	78.42875

Table 8.1: Effect of increasing IT budget on the profit of the company.

The "Adequate IT Budget" goal is involved in two loops. One positive loop (P1 in figure 8.1) and one negative loop (N1 in figure 8.1). Loop P1 repeats itself every five transition and Loop N1 repeats itself every six transitions. Table 8.2 below shows the goal satisfaction values in the first 16 goal influence path traversals.

Since P1 is a positive loop, it tends to continuously increase the IT budget of the system. On the other hand N1 is negative and will try to decrease the IT Budget of the goal. From section 8.2, we remember that when there are multiple contributions affecting a single goal, the resultant is the average of the two contributions.

From Table 8.2, increasing the IT budget ultimately result an increases in the selling Income of the company. At the same time, increasing the IT Budget also results an increase in the operational cost of the company. But the raise in selling income is much more than the increase in the operational cost. Hence the combined effect on the net profit of the company is continuously increasing reaching a value of 79.2 from the initial value of 50.0 at the end of the 16^{th} iteration.

If the simulation keeps iterating, the company profit will keep increasing. Consequently, the IT budget will not converge in to a certain value. Rather; it continues to grow along with the increase in net profit of the company.

Goal Name	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Adequate IT Budget	85	85	85.0	85.0	85.0	85.0	96.1	96.1	96.1	100.2	100.2	100.2	103.8	117.6	117.6	119.2	126.8
System Usability	50	83.3	83.3	83.3	83.3	83.3	83.3	93.5	93.5	93.5	97.3	97.3	97.3	100.7	117.3	117.3	118.9
Customer Satisfaction	50	50	83.3	83.3	83.3	83.3	83.3	83.3	93.5	92.5	92.5	96.3	95.9	95.9	99.2	109.3	107.1
Multiple Option for Customers	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50
Support Desk	50	50	50	50	50	50	50	47.8	47.8	47.8	47.1	47.1	47.1	46.6	44.0	44.0	43.8
Response Time	50	50	50	50	50	50	50	50	48.9	48.9	48.9	48.6	48.6	48.6	48.3	46.7	46.7
Maintain Current Staff Size	50	50	50	50	50	50	45.7	45.7	45.7	44.4	44.4	44.4	43.3	39.4	39.4	39.1	37.4
Profit	**50**	**50**	**50**	**50**	**50**	**58.7**	**58.7**	**58.7**	**62.0**	**62.0**	**62.0**	**65.0**	**73.1**	**73.1**	**74.4**	**79.2**	**79.2**
Insured Clients	50	50	50	79.9	79.9	79.9	79.9	79.9	79.9	88.8	97.7	97.7	101.3	104.7	104.7	107.9	121.4
Operational cost	**50**	**50**	**50**	**50.0**	**65.0**	**65.0**	**65.0**	**60.3**	**60.3**	**60.3**	**61.3**	**68.1**	**68.1**	**68.1**	**66.2**	**66.2**	**66.5**
Selling Income	**50**	**50**	**50**	**50.0**	**79.3**	**79.3**	**79.3**	**79.3**	**79.3**	**79.3**	**88.0**	**107.2**	**107.2**	**111.0**	**118.7**	**118.7**	**122.3**

Table 8.2: The result of investign on the IT budget of the ArchiSurance Company.

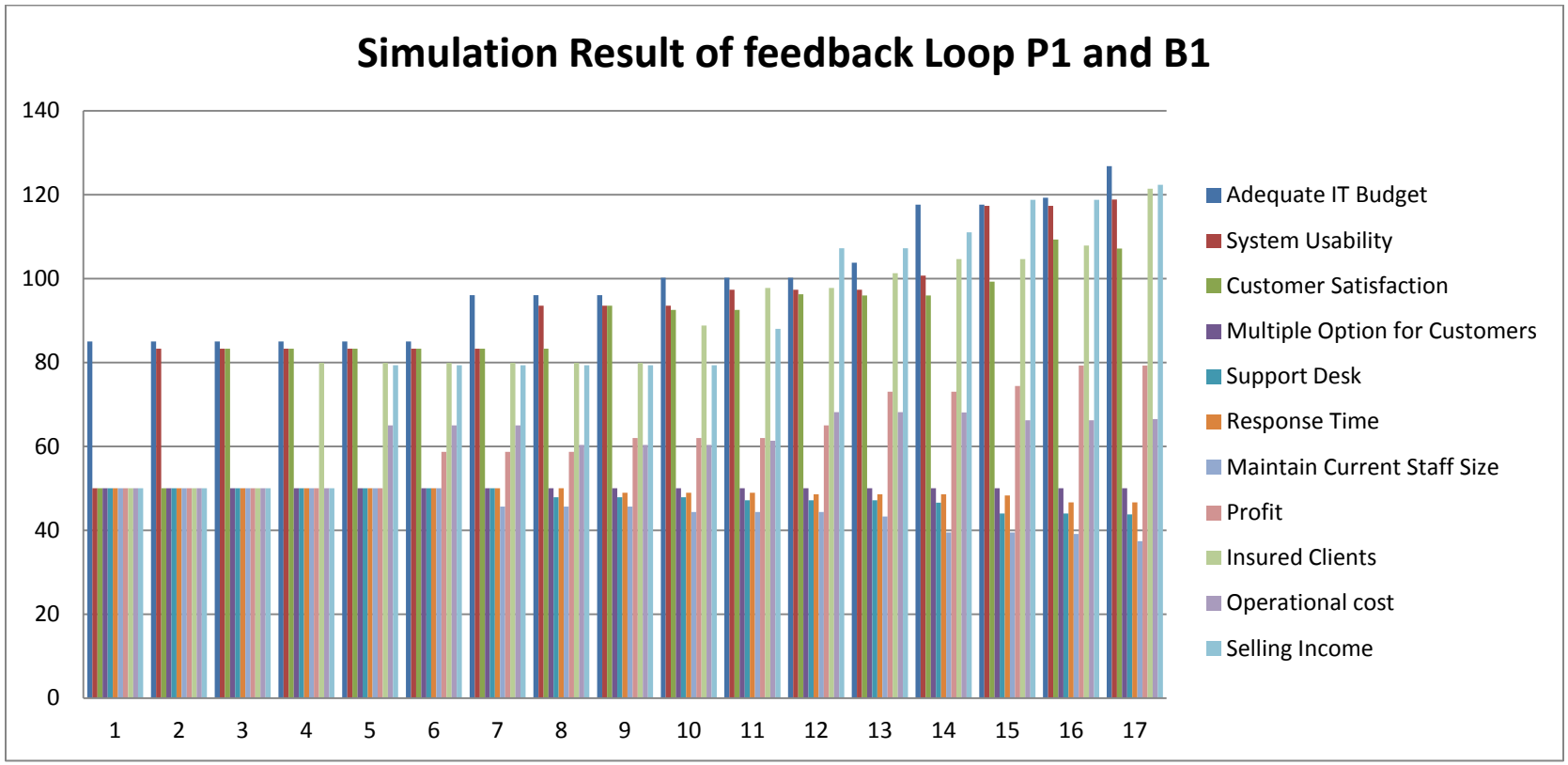

Figure 8.5: Graphical representation of investing on IT budget effects.

ii. Budget Allocated to Maintain the Current Staff Size /Recruit More Employees

If the manager decides to invest on spending the budget on human resources, she can keep all the current employees and probably recruit more employees. This will make the goal "Maintain Current Staff" fully satisfied. From the fuzzy sets used in chapter seven again, a goal is fully satisfied if its satisfaction value is between 80 and 100. Let's take the satisfaction level of the goal to be 85. Table 8.3 shows the result of increasing the satisfaction value of "Maintain Current Staff" to 85 on the Profit and directly related goals.

Goal Name	Satisfaction Level
Adequate IT Budget	29.0770663
System Usability	33.641
Customer Satisfaction	37.63456516
Multiple Option for Customers	50
Support Desk	75.249
Response Time	62.12225
Maintain Current Staff Size	112.3869745
Profit	21.69407143
Insured Clients	38.05495588
Operational cost	97.19923853
Selling Income	53.85875

Table 8.3: Effect of changing the "Maintain current staff size" on other goals of the system.

As can be shown in table 8.3 Spending money on maintaining employees will eventually decrease the company's profit. And more interestingly successive iterations show that investing on HR will further decrease the company's profit. The project manager can keep hiring more and more employees but she is going to bankrupt the company in the near future. This successive decrease in the company's profit is shown in Table 8.4 and Figure 8.6.

From Table 8.2 and 8.6, the goal which hasn't showed a change in satisfaction value is "Multiple Option for Customers". This is because there is no influence relation coming in to the goal. The next case will trigger a change in this goal as a result of investing the allocated budget on business process reengineering.

Goal Name	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Adequate IT Budget	50	50	50	32.8	32.8	32.8	29.1	24.8	24.8	24.8	16.2	16.2	15.6	12.9	8.9	4.9
System Usability	50	50	50	50.0	50.0	33.6	33.6	30.0	23.1	23.1	21.0	21.0	14.0	13.6	11.0	6.3
Customer Satisfaction	50	50	50	54.4	54.4	36.6	37.6	37.6	33.6	33.6	29.7	29.7	21.8	22.9	22.1	19.7
Multiple Option for Customers	50	50	50	50.0	50.0	50	50	50	50	50	50	50	50	50	50	50
Support Desk	50	67.5	67.5	67.5	67.5	75.2	75.2	78.1	85.2	85.2	87.9	87.9	99.3	100.5	107.7	128.1
Response Time	50	50	58.8	58.8	58.8	62.1	62.1	62.1	63.3	63.3	67.5	67.5	74.2	74.2	74.7	77.8
Maintain Current Staff Size	85	85	85.0	104.5	104.5	104.5	112.4	123.5	123.5	123.5	155.7	155.7	159.5	177.9	214.8	278.2
Profit	50	50	27.0	27.0	27.0	23.0	21.7	21.7	18.9	18.9	13.6	13.6	10.5	8.0	5.6	3.8
Insured Clients	50	50	50	50.0	50.0	53.9	38.1	27.5	27.5	27.5	19.4	19.4	11.7	7.3	4.7	3.0
Operational cost	50	78.7	78.7	78.7	78.7	97.2	97.2	93.0	88.6	88.6	88.7	88.7	72.1	62.4	54.6	51.8
Selling Income	50	50	50	50.0	50.0	53.9	53.9	38.3	19.9	19.9	18.0	18.0	7.4	3.3	0.9	0.2

Table 8.4: the first 16 iteration results of allocating the budget fully to HR department.

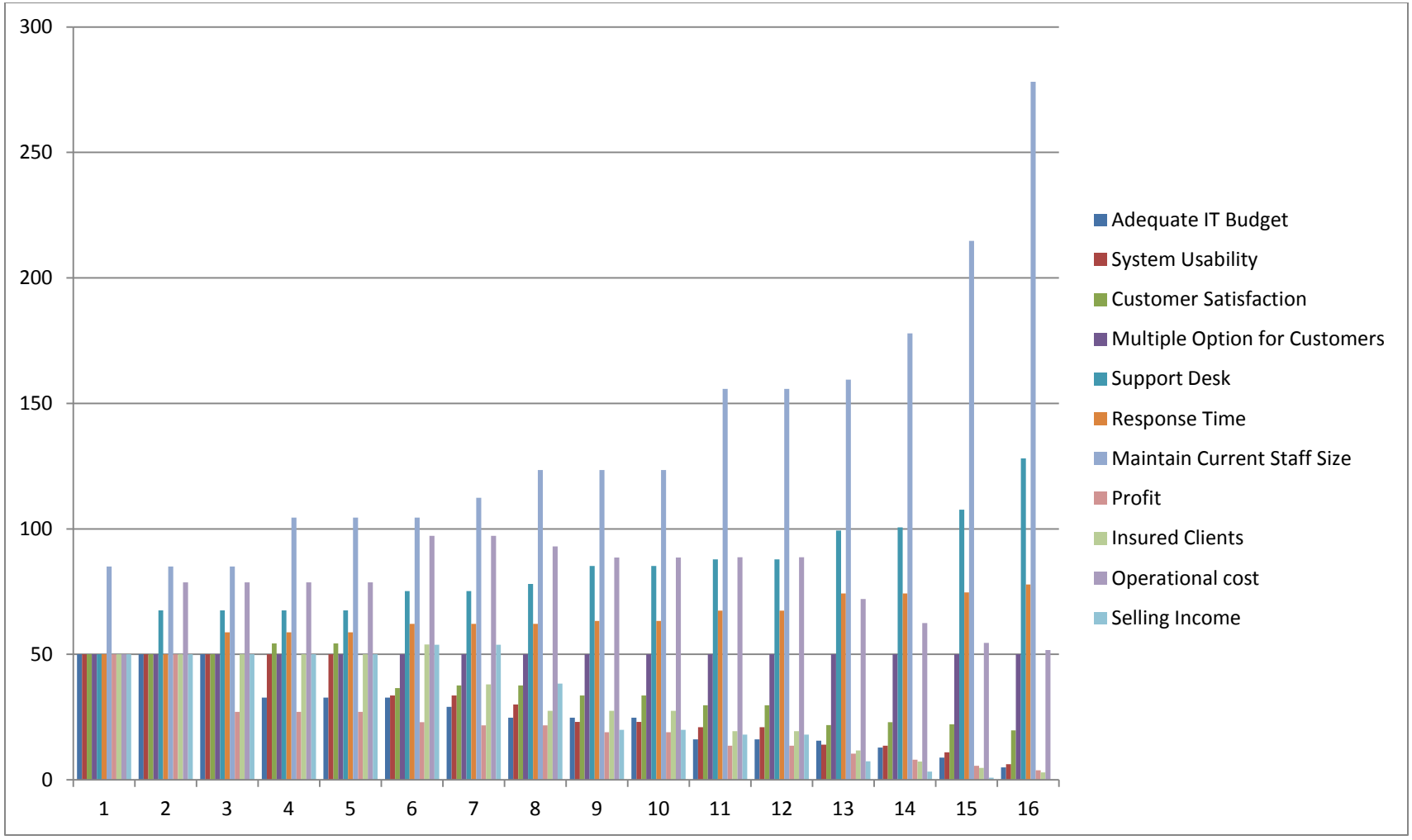

Figure 8.6: Graphical representation of investing on “HR” effects on other goals of the system.

iii. Budget Allocated to Utilize the Synergetic Effects/Business Process Reengineering

If the manager decides to invest on spending the budget on reengineering the business process, she can to create multiple insurance options for customers by combining/ adapting existing insurance policies. This will make the goal "Multiple Options for Customers" fully satisfied. From the fuzzy sets used in chapter seven again, a goal is fully satisfied if its satisfaction value is between 80 and 100. Let's take the satisfaction level of the goal to be 85. Table 8.4 shows the result of increasing the satisfaction value of "Multiple Options for Customers" to 85 on the Profit of the company and other related goals.

Goal Name	Satisfaction Level
Adequate IT Budget	50
System Usability	50
Customer Satisfaction	85
Multiple Option for Customers	85
Support Desk	50
Response Time	50
Maintain Current Staff Size	50
Profit	59.135
Insured Clients	81.5
Operational cost	65.75
Selling Income	80.87

Table 8.4: Effect of creating more insurance options for customers

From Table 8.5 and Figure 8.7, creating more insurance options keeps increasing the profit of the company. At the end of 16^{th} influence path traversal, the company's profit is predicted to be 81.1 which is higher than its 50.0 initial value.

Goal Name	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Adequate IT Budget	50	50	50	50	50	56.9	56.9	56.9	59.4	59.4	59.4	61.7	70.4	70.4	71.4	76.2
System Usability	50	50	50	50	50	50	56.5	56.5	56.5	58.9	58.9	58.9	61.0	71.8	71.8	72.7
Customer Satisfaction	50	85	85	85	85	85	85.0	96.1	95.0	95.0	99.0	98.7	98.7	102.2	113.2	110.8
Multiple Option for Customers	85	85	85	85	85	85	85.0	85	85	85	85	85	85	85	85	85
Support Desk	50	50	50	50	50	50	47.7	47.7	47.7	47.0	47.0	47.0	46.4	43.7	43.7	43.5
Response Time	50	50	50	50	50	50	50.0	48.9	48.9	48.9	48.5	48.5	48.5	48.2	46.5	46.5
Maintain Current Staff Size	50	50	50	50	50	45.4	45.4	45.4	44.1	44.1	44.1	43.0	38.9	38.9	38.5	36.8
Profit	50	50	50	50	59.1	59.1	59.1	62.7	62.7	62.7	65.9	74.5	74.5	75.9	81.1	81.1
Insured Clients	50	50	81.5	81.5	81.5	81.5	81.5	81.5	91.0	100.7	100.7	104.5	108.2	108.2	111.7	126.5
Operational cost	50	50	50	65.8	65.8	65.8	60.8	60.8	60.8	61.9	69.1	69.1	69.1	67.0	67.0	67.3
Selling Income	50	50	50	80.9	80.9	80.9	80.9	80.9	80.9	90.2	110.9	110.9	115.1	123.5	123.5	127.5

Table 8.6: The first 16 iteration effect as a result of creating more insurance options for customers

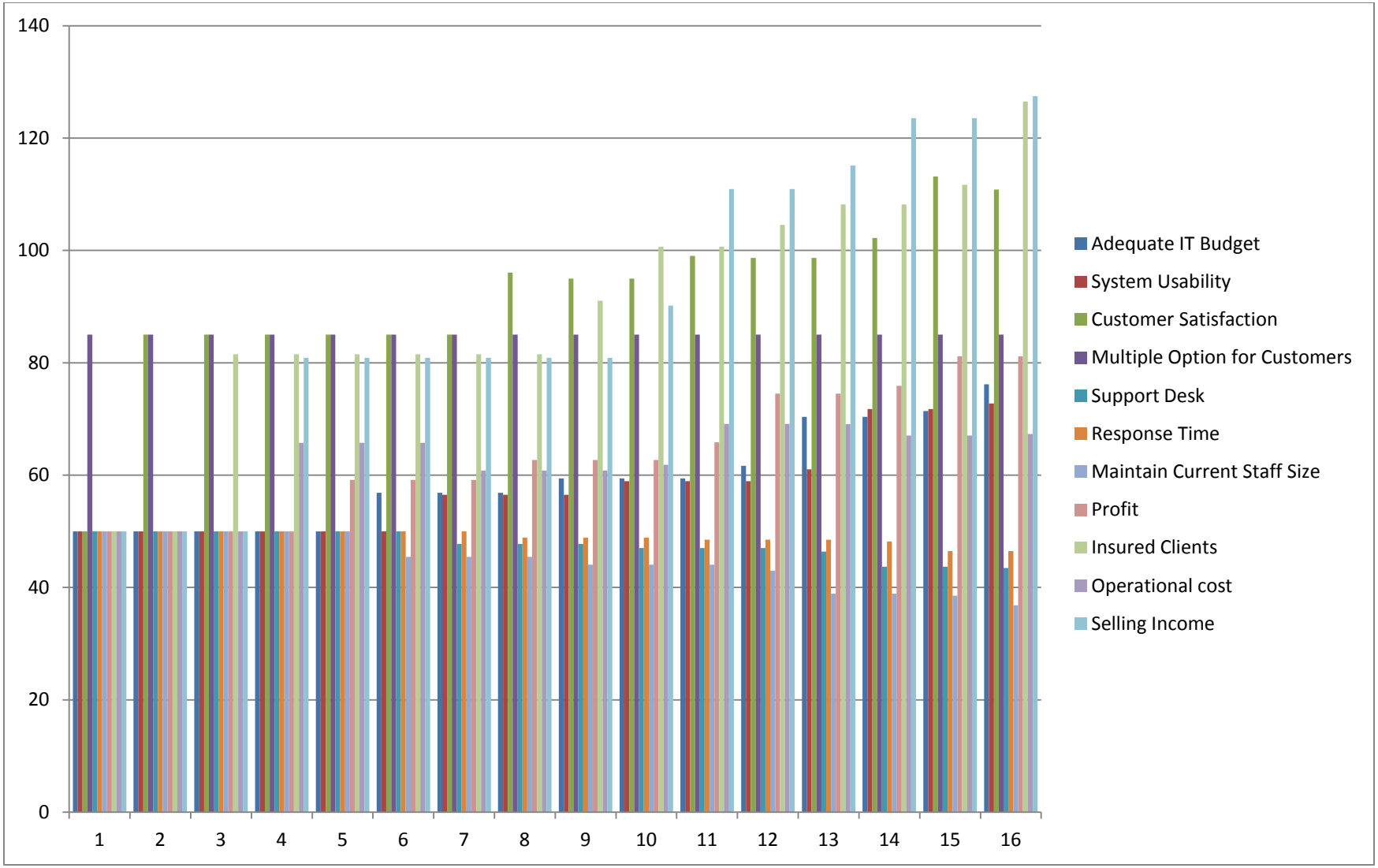

Figure 8.7: Graphical representation effects as a result of creating more insurance options.

8.3.2: Decision support via Indirect Influence Reasoning

The company's manager decision making process is relatively easy now because she can see the results of table 8.2, 8.4 and 8.6. Table 8.7 summarizes the ultimate effect of each alternative decision scenario on the profit of the company.

Decision	Invest on HR	Invest on IT	Invest on BPR (Create more Insurance options)
Effect on Profit of the company	3.8	79.2	81.1

Table 8.7: Effect of each candidate decision on the profit of the company.

From table 8.7, it is apparent that investing on HR is not a wise decision for ArchiSurance manager. Rather she should invest either on creating more types of insurance options or on enhancing the IT system of the company. Choosing from the last two options can be still difficult since they result approximately the same level of profit for the company. The manager can use the priorities of each goal and other factors that were not considered explicitly in the goal model to make her decision.

8.4: Summary

Feedback loops are set of contribution relations that form a cyclic path of influence relations. They are one of the primary causes of dynamic system behaviors. Two types of feedback loops exist: positive and negative feedback loops.

Positive feedback loops intensify a change in the satisfaction level of goals in a feedback loop, i.e. if a goal satisfaction value is increased in a positive loop, its satisfaction value keeps growing or vice versa.

Negative feedback loops resist any kind of change in the loop by counteracting any change in the satisfaction level o the goals in the loop. i.e. if a goal satisfaction value is raised in a negative loop, The loop will eventually force the satisfaction value to decrease or vice versa.

To compute the effect of goal influence feedback loops, a propagation algorithm that computes the change percentage in the satisfaction level of the influencing goal is used. The algorithm then updates the influenced goal based on the change percentage and its initial satisfaction value.

A test case study is used to show the applicability of the feedback loop algorithm on goal satisfaction values. Beside demonstrating the applicability of the algorithm in feedback loop analysis, the test case also shows one important application of this master thesis project: Decision support in alternative resource allocation problem. This is done by simulating every possible change scenario to predict and compare the effect of the change scenarios. Decision makers can use this predicted values to make a better decision.

The fourth research question of this master project is *"How can we model, simulate and visualize the effect of feedback loops in goal models?"*This chapter provides an answer to this research question by presenting a quantitative reasoning engine to simulate and analyze feedback loops of influence relations. It also presents the reasoning engine algorithm and a tool support to visualize the feedback loop effects.

9. Conclusions and Recommendations

This chapter summarizes the whole project and specifies the relevant findings. It also presents the significance of the study and its limitations. Finally, possible recommendations to be carried out as a future research will also be proposed.

9.1: Summary

This MSC thesis report investigates the role of techniques designed for analyzing indirect influence goal-to-goal relations. The goal relations are defined in the context of the motivation extension for ArchiMate EA modeling language. Two different semantics to these relations were defined to enable qualitative and quantitative reasoning on goal influence relations. The two semantics were selected based on comparative analysis performed on existing goal formalization techniques.

The qualitative reasoning is based on TROPOS software development methodology while the quantitative reasoning is based on NFR framework based fuzzy reasoning engine. A reasoning technique to analyze goal influence relations that form cyclic influence paths (feedback loops) is also realized.

For all reasoning techniques, ArchiMate modeling language tailored algorithms are developed. Prototype applications for qualitative and quantitative reasoning as well as feedback loop simulations were also realized as an extension of already existing EA modeling tool from BiZZdesign (BiZZdesign Architect).

To demonstrate the applicability of the reasoning approaches, two test case studies have been conducted on one real and another fictitious company goal models. The comparative analysis performed on both approaches reveals that both approaches are applicable for indirect influence reasoning and they generate reasonably consistent results.

9.2: Relevant Findings

The main objective of this project is developing of algorithms that enable analysis, reasoning and simulation of influence relations among goals in Enterprise Architecture designs. By analyzing indirect influence relations in goal models, it is possible to enhance understanding of the effect of changing the satisfaction level of goals. This understanding can be a vital input for decision making processes like alternative resource allocation problems.

Moreover, by developing two kinds of reasoning techniques on goal influence relations, this study finds out that:

i. Both qualitative and quantitative reasoning approaches are applicable in goal influence reasoning.
ii. The results obtained from the two reasoning approaches are reasonably consistent.
iii. Each approach has its own merits and demerits
iv. TROPOS based qualitative reasoning approach is a good choice for high level goal analysis. This is especially advantageous for nontechnical people who prefer natural language expression for reasoning on goal models.
v. The qualitative approach is also found to be better in handling goal conflicts due to small number of goal satisfaction value types.

vi. The qualitative approach can be ambiguous due to lack of discrete values in goal satisfaction specifications.

vii. For a deep investigation of goals and requirements demanded by technical people, NFR based fuzzy reasoning approach will be a good choice. This is mainly due to the possibility of using extended range of goal satisfaction values (-100 to 100).

viii. The quantitative reasoning technique can lead to undetermined (zero level) goal satisfaction values. It can also be less useful if there is a difficulty in assigning crisp numeric values for initial goal satisfaction inputs due to lack of domain experts.

9.3: Relevance of the Study

Most of the theoretical and practical implications of this study are outlined in chapter one of the report. This subsection will revisit those and other implications of this project from the perspective of the results obtained.

9.3.1: Theoretical Significance

i. The thesis provides well defined semantics for goal to goal influence relation. This is probably the most important theoretical contribution of this project. The semantics are defined based on existing trends in GORE practices. These makes the reasoning approaches easy to understand and apply on practice.

ii. The thesis provides a comparison of existing approaches for applicability in practical case studies. This has both theoretical and practical significance. From the theoretical side, two reasoning techniques for goal satisfactions are compared and contrasted to identify the merits and demerits of each approach.
From the practical side, the results of our case study can help requirement engineers who want to use goal analysis can use to decide which approach is better for their goal analysis. The consequent reasoning adds another dimension for requirement traceability especially in early phases of RE.

iii. The paper presents feedback loops for goal influence relation. GORE approaches already attract significant attention by the RE community. But to the author's best knowledge, there is no goal reasoning algorithm that explicitly considers goal influence feedback loops. The goal feedback loop algorithm presented in this report can be used as a starting point for investigating the goal influence dynamics.

9.3.2: Practical Significance

i. Decision support in analyzing dynamic business environments: Considering the inevitable dynamic of today's' business environments, tool support for managing dynamic stakeholder goals will enable requirement engineers, enterprise architects and other stakeholders ability to build adaptable systems.

ii. Extended functionality for ArchiMate language: ArchiMate is emerging as a standard in modeling EA designs. Incorporating indirect influence relation reasoning support will enhance the usability of the language.

iii. Similarly: realizing indirect influence reasoning functionality in BiZZdesign will enhance the functionality of BiZZdesign Architect in situation like decision making processes and alternative resource allocation problems.

iv. Finally, an immediate consequence of the last two is the enhancement of the usability of BiZZdesign Architect which will eventually enhance the satisfaction of BiZZdesign Architect users.

9.4: Limitations of the Study

Despite its apparent significances, this study is also subjected to a number of important limitations to be considered. The first limitation is the lack of top down goal influence reasoning support. Top down goal reasoning involves setting a satisfaction level of a (higher level) and search for goal satisfaction assignments that can result the desired goal satisfaction level. Both the qualitative and quantitative approaches are designed to work with bottom-up approaches. Discussions with the stakeholders of BiZZdesign revealed that the proposed goal reasoning approaches might have been more usable if top-down goal analysis features were added.

The second limitation of this project is related to the consideration of symmetrical goal influence relations only. There are also goal influence relations where only positive or only negative influences are propagated. These kinds of influence relations are termed as asymmetric relations and might have been useful for detailed goal analysis if incorporated in this study.

A third limitation of this project is the lack of domain experts' opinion about the case study results. If relevant domain experts (e.g. the water company enterprise architects) had verified the results, the validity of the project would have been enhanced.

The effect of goal model sizes on the applicability of the reasoning techniques could have been also investigated in more detail. The size of the goal model can depend both on the number of goals and the number of influence relations. The slight difference in the consistencies of qualitative and quantitative approaches in the two case studies could have been a good starting point in investigating goal model size effects on the proposed approaches.

9.5: Recommendations for Future Work

The results of this project can be extended in many ways. An immediately related potential future work is combining the advantages of the two approaches to provide more accurate analysis on goal satisfaction levels. An example for this combination can be incorporating the conflict detection feature of TROPOS and NFR framework to the fuzzy reasoning engine.

To improve the conflict management feature of the proposed solution, Multi-Objective Programming could also be employed. Multi-objective programming allows optimization of conflicting goal satisfaction values that are subject to zero or more constraints [47]. Incorporating this feature in the influence reasoning engines will enhance the practical usability of the system.

It is also possible to incorporate asymmetrical goal relations of TROPOS in goal reasoning techniques. The outputs of the reasoning tool with asymmetrical relations can be compared against the results shown in this report to estimate the significance of asymmetrical relations in indirect influence analysis.

Another possible extension of this research is testing the applicability of fuzzy logic as an extension to TROPOS based approach. This will change the qualitative reasoning to quantitative reasoning. The new quantitative reasoning can be compared against existing quantitative reasoning approach of TROPOS or the fuzzy logic based quantitative reasoning approach described in this report.

To enhance the validity of the proposed approaches, a full scale empirical study can be performed on the satisfaction level of goals in an organization going through changes in its goals. The observed satisfaction levels and the predicted values can be compared to assess the validity of the proposed approaches.

10. References

[1] S. Lehnert, "A taxonomy for software change impact analysis," in Proceedings of the 12th International Workshop on Principles of Software Evolution and the 7th annual ERCIM Workshop on Software Evolution, New York, NY, USA, 2011, pp. 41–50.

[2] S. A. Bohner, "Impact analysis in the software change process: a year 2000 perspective," in , International Conference on Software Maintenance 1996, Proceedings, 1996, pp. 42 –51.

[3] A. Josey, TOGAF Version 9 Enterprise Edition: An Introduction (White Paper). The Open Group, 2009.

[4] V. H. Publishing, Archimate 2.0 Specification. Van Haren Publishing, 2012.

[5] W. Engelsman, D. Quartel, H. Jonkers, and M. van Sinderen, "Extending enterprise architecture modelling with business goals and requirements," Enterp. Inf. Syst., vol. 5, no. 1, pp. 9–36, Feb. 2011.

[6] J. Sterman and J. D. Sterman, Business Dynamics: Systems Thinking and Modeling for a Complex World with CD-ROM. McGraw-Hill/Irwin, 2000.

[7] A. van Lamsweerde, "Goal-oriented requirements engineering: a guided tour," in Fifth IEEE International Symposium on Requirements Engineering, 2001. Proceedings, 2001, pp. 249–262.

[8] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, Non-Functional Requirements in Software Engineering, 1st ed. Springer, 1999.

[9] J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using nonfunctional requirements: a process-oriented approach," Software Engineering, IEEE Transactions on, vol. 18, no. 6, pp. 483 – 497, Jun. 1992.

[10] Wilco Engelsman, Henk Jonkers, and Dick Quartel, "Supporting Requirements Management in TOGAF and ArchiMate," The Open Group, Feb. 2010.

[11] W. Engelsman and R. Wieringa, "Goal-Oriented Requirements Engineering and Enterprise Architecture: Two Case Studies and Some Lessons Learned," in Requirements Engineering: Foundation for Software Quality, vol. 7195, B. Regnell and D. Damian, Eds. Springer Berlin / Heidelberg, 2012, pp. 306–320.

[12] BiZZdesign B.V., "BiZZdesign Home Page," Building Strong Organizations. [Online]. Available: http://www.bizzdesign.com/. [Accessed: 03-Mar-2012].

[13] Remeco Bloom, "Introduction: ArchiMate - Recorded video," Bizzdesign. .

[14] "BIZZdesign Holding registers Tool Support Product." [Online]. Available: http://www.opengroup.org/homepage-items/c662.html. [Accessed: 06-Aug-2012].

[15] E. S.-K. Yu, "Modelling strategic relationships for process reengineering," University of Toronto, Toronto, Ont., Canada, Canada, 1996.

[16] E. Letier and A. van Lamsweerde, "Reasoning about partial goal satisfaction for requirements and design engineering," in Proceedings of the 12th ACM SIGSOFT twelfth international symposium on Foundations of software engineering, New York, NY, USA, 2004, pp. 53–62.

[17] M. Serrano, M. Serrano, and J. C. S. do P. Leite, "Dealing with softgoals at runtime: A fuzzy logic approach," 2011, pp. 23–31.

[18] A. Hevner, S. March, J. Park, and S. Ram, "Design Science in Information Systems Research," Management Information Systems Quarterly, vol. 28, no. 1, Dec. 2008.

[19] R. Wieringa, "Design science as nested problem solving," in Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology, New York, NY, USA, 2009, pp. 8:1–8:12.

[20] R. Wieringa, "Design science methodology: principles and practice," in Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2, New York, NY, USA, 2010, pp. 493–494.

[21] Henk Jonkers, Iver Band, and Dick Quartel, "ArchiSurance Case Study." The Open Group, Jan-2012.

[22] E. S. . Yu, “Towards modelling and reasoning support for early-phase requirements engineering,” in , Proceedings of the Third IEEE International Symposium on Requirements Engineering, 1997, 1997, pp. 226–235.
[23] A. van Lamsweerde, “Formal specification: a roadmap,” in Proceedings of the Conference on The Future of Software Engineering, New York, NY, USA, 2000, pp. 147–159.
[24] E. Kavakli, “Goal oriented requirements engineering: a unifying framework,” REQUIREMENTS ENGINEERING JOURNAL, SPRINGER-VERLAG LONDON, vol. 6, pp. 237–251, 2002.
[25] P. Giorgini, J. Mylopoulos, and R. Sebastiani, “Goal-oriented requirements analysis and reasoning in the Tropos methodology,” Eng. Appl. Artif. Intell., vol. 18, no. 2, pp. 159–171, Mar. 2005.
[26] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, “Reasoning with Goal Models,” in Proceedings of the 21st International Conference on Conceptual Modeling, London, UK, UK, 2002, pp. 167–181.
[27] L. A. Zadeh, “Fuzzy sets,” Information and Control, vol. 8, no. 3, pp. 338–353, Jun. 1965.
[28] L. A. Zadeh, “Is there a need for fuzzy logic?,” in Fuzzy Information Processing Society, 2008. NAFIPS 2008. Annual Meeting of the North American, 2008, pp. 1 –3.
[29] Bart Kosko and Satoru Isaka, “Fuzzy Logic,” Scientific American, vol. vol. 269, pp. 76–81, Jul. 1993.
[30] T. J. Ross, Fuzzy Logic with Engineering Applications, 2nd ed. Wiley, 2004.
[31] E. H. Mamdani and S. Assilian, “An experiment in linguistic synthesis with a fuzzy logic controller,” International Journal of Man-Machine Studies, vol. 7, no. 1, pp. 1–13, Jan. 1975.
[32] Van Lamsweerde and E. Letier, “From object orientation to goal orientation: A paradigm shift for requirements engineering,” Radical Innovations of Software and Systems Engineering in the Future, vol. 2941, no. I, pp. 153–166, 2004.
[33] E. Kavakli and P. Loucopoulos, “Goal Modeling in Requirements Engineering: Analysis and Critique,” OF CURRENT METHODS, INFORMATION MODELING METHODS AND METHODOLOGIES, pp. 102–124, 2004.
[34] A. Lapouchnian, “Goal-Oriented Requirements Engineering : An Overview of the Current Research,” Requirements Engineering, vol. 8, no. 3, p. 32, 2005.
[35] Bo Wei, Zhi Jin, and Lin Liu, “A Formalism for Extending the NFR Framework to Support the Composition of the Goal Trees,” in Software Engineering Conference (APSEC), 2010 17th Asia Pacific, 2010, pp. 23–32.
[36] R. Darimont, E. Delor, P. Massonet, and A. van Lamsweerde, “GRAIL/KAOS: An Environment for Goal-Driven Requirements Engineering,” in , Proceedings of the 1997 (19th) International Conference on Software Engineering, 1997, 1997, pp. 612–613.
[37] E. Letier, “Reasoning about Agents in Goal-Oriented Requirements Engineering,” 2001.
[38] A. Dardenne, A. van Lamsweerde, and S. Fickas, “Goal-directed requirements acquisition,” Science of Computer Programming, vol. 20, no. 1–2, pp. 3–50, Apr. 1993.
[39] R. Darimont and A. van Lamsweerde, “Formal refinement patterns for goal-driven requirements elaboration,” SIGSOFT Softw. Eng. Notes, vol. 21, no. 6, pp. 179–190, Oct. 1996.
[40] E. S. K. Yu and J. Mylopoulos, “Using Goals, Rules, and Methods to Support Reasoning in Business Process Reengineering,” IN BUSINESS PROCESS REENGINEERING”, INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS IN ACCOUNTING, FINANCE AND MANAGEMENT, vol. 5, pp. 1–13, 1994.
[41] J. Mylopoulos, J. Castro, and M. Kolp, “Tropos: A Framework for Requirements-Driven Software Development,” in INFORMATION SYSTEMS ENGINEERING: STATE OF THE ART AND RESEARCH THEMES, 2000, pp. 261–273.
[42] A. Göknil, I. Kurtev, and K. G. van den Berg, “Change Impact Analysis based on Formalization of Trace Relations for Requirements,” pp. 59–75, Jun. 2008.
[43] J. Wang, J. Li, Q. Wang, H. Zhang, and H. Wang, “A Simulation Approach for Impact Analysis of Requirement Volatility Considering Dependency Change,” in Requirements Engineering: Foundation

for Software Quality, vol. 7195, B. Regnell and D. Damian, Eds. Springer Berlin / Heidelberg, 2012, pp. 59–76.
[44] S. España, N. Condori-Fernandez, A. González, and Ó. Pastor, “An empirical comparative evaluation of requirements engineering methods,” Journal of the Brazilian Computer Society, vol. 16, no. 1, pp. 3–19, 2010.
[45] L. Chung and J. C. Prado Leite, “Conceptual Modeling: Foundations and Applications,” A. T. Borgida, V. K. Chaudhri, P. Giorgini, and E. S. Yu, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 363–379.
[46]“Centroid, Area, Moments of Inertia, Polar Moments of Inertia, & Radius of Gyration of a General Trapezoid.” [Online]. Available: http://www.efunda.com/math/areas/Trapezoid.cfm. [Accessed: 01-Aug-2012].
[47] P. Korhonen and G. Macdonald, “Multiple Objective Programming Support,” in In: C.A. Floudas, P.M. Pardalos (eds.) Encyclopedia of Optimization, Vol. III (Kluwer, Dordrecht 2001) 566–574, 1998.
[48] J. P. M. Silva and K. A. Sakallah, “GRASP—a new search algorithm for satisfiability,” in Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design, Washington, DC, USA, 1996, pp. 220–227.
[49] R. Zabih and D. Mcallester, “A Rearrangement Search Strategy for Determining Propositional Satisfiability,” in in Proceedings of the National Conference on Artificial Intelligence, 1988, pp. 155–160.
[50] R. Sebastiani, P. Giorgini, and J. Mylopoulos, “Simple and Minimum-Cost Satisfiability for Goal Models,” 2004, pp. 20–35.

Appendix A: Top-down Goal Influence Reasoning

The reasoning techniques we saw in chapter six and seven work in bottom up reasoning fashion. A bottom up reasoning technique involves assigning certain values to (leaf) goals by stakeholders and then estimate the effect on upper level goals of the goal model. This kind of reasoning is practical and applicable when we have a change in the satisfaction level of few of the goals in the goal model and we need to determine their effect on other goals. Predicting the effect of merging two departments, analyzing the consequence of decreasing a budget for a certain department, forecasting the result of increasing production rate etc. are some of the applications of bottom up reasoning techniques.

Let's take a different kind of goal analysis scenario now; imagine a software project manager who likes to develop "Excellent Rated" software. She can do this by choosing most applicable software development methodology, she can also hire good software engineers, good testers etc. But hiring all this "good" employees may escalate the project cost. Selecting the most appropriate software development methodology can also be time taking which is undesirable. But the project manager then has to make decision who to hire, which software development methodology to use etc. in a way that result the best possible "Quality Software" under time and budget constraints.

Top down goal influence reasoning is applicable on these kinds of scenarios. It is a kind of goal reasoning that involves setting of a certain value to a certain (top level) goal and then exploring possible ways of achieving this goal by altering its sub goals under given constraints. The value assigned to the top level goal is to be select by stakeholders owning the goal possibly in consultation with requirement engineers.

If possible conflicts and constraints are ignored; the top down goal analysis is similar to the bottom up analysis where every possible combination of leaf goal assignments are checked until an optimum level of satisfaction is achieved for the top level goal. This process can take quite some time especially in large organizations, where a goal model can have thousands of goals. A top down goal analysis using propositional satisfiability technique can be used in this kind of problems [25].

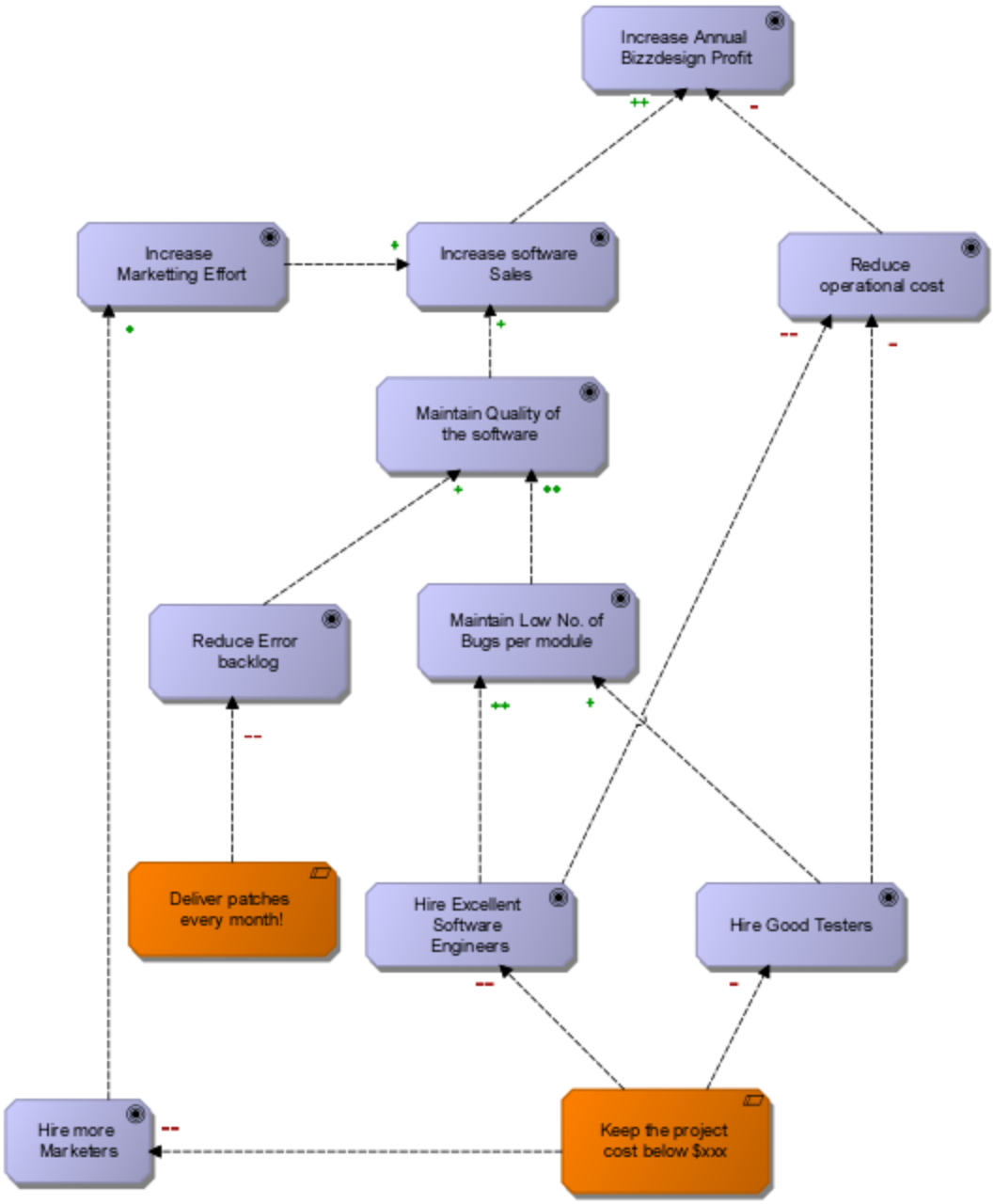

Figure A.1: Ensuring software quality (Top level goal) can be achieved to different extent by altering sub goals and depending on certain constraints (time and budget in this case).

A.2: Simple and Minimum Cost Satisfiability Goal for model

Propositional satisfiability (SA T) is the problem of determining whether a Boolean formula P admits at least one satisfying truth assignment "a" to its variables [25]. In worst case, SAT problems are believed to have an exponential complexity though there are efforts to provide more efficient algorithms [48].

One of the well known techniques for solving SAT problems is using a backtracking search algorithm where each node in the search tree elects an assignment and shorten subsequent search by iteratively applying the unit clause and the pure literal rules [49].

Using this backtracking algorithm, Sebastiani et al.[50] Proposes a technique for determining the lowest cost node assignment selection that satisfies a certain Boolean value for a given (top level) node. This technique named, Simple and Minimum Cost Satisfaibility for goal Models, uses a Conjunctive Normal Form (CNF) of the graph that represents the entire nodes in the graph, the desired output, the backward reasoning algorithm and optionally available constraints.

Mathematically, this can be written as:

$$P = P_{graph} \wedge P_{output} \wedge P_{backward} [\wedge P_{optional}] \quad \text{A.1}$$

If we use the qualitative reasoning goal satisfaction values (chapter 6) in equation A.1, the Boolean variables of the formula will be the variables in table A.1 shown below.

Boolean Variable	**Description**
FS(g)	Goal g is Fully Satisfied or not
MS(g)	Goal g is mostly Satisfied or not
PS(g)	Goal g is Partially Satisfied or not
LS(g)	Goal g is Little Satisfied or not
NS(g)	Goal g is not Satisfied or not
ND(g)	Goal g is not Denied or not
LD(g)	Goal g is Little Denied or not
PD(g)	Goal g is partially Denied or not
MD(g)	Goal g is Mostly Denied or not
FD(g)	Goal g is Fully Denied or not

Table A.1: Boolean variables for qualitative top down reasoning.

Using these variables, equation A.1 can be formulated to the desired goal model as well as the desired output and constraints. The equation can be then checked using backward tracking algorithm to find the Boolean value assignments to these variables that can result the desired output (if there exists any possible solution).

We will not cover the details of each of the formulas in equation A.1 since it is out of the scope of this project. Interested reader can refer [25], [48], [50] for more details.

Appendixes B: Delays and Influence Propagations

Both the qualitative and quantitative reasoning approaches we have seen in chapter six and seven use adapted forms of causal diagrams to model the influence relations between goals of a system. Inspired by the Software Interdependency Graphs (SIGs), the adaptation we choose enables us to incorporate AND/OR decompositions to causal loop models.

Causal loops are best suited for representing interdependence and feedback loops. But there are goal influences relations where a change in the satisfaction level of a goal will take some time to affect the influenced goal. The influence relation between market demand and production rate shown in figure 9.4 will illustrate this.

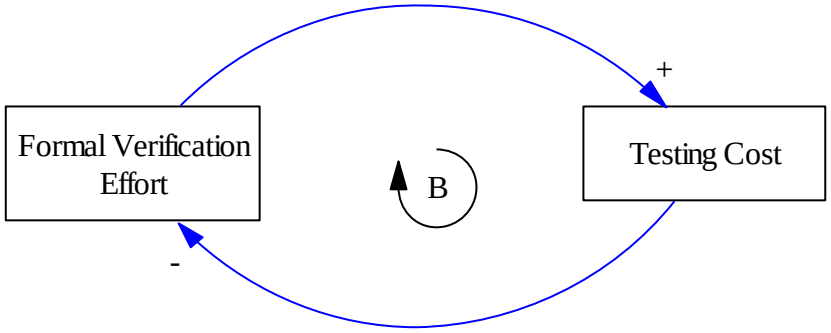

Figure B.1: A negative feedback loop between formal verification effort and associated testing costs.

An increase in verification and testing procedures of a software module will definitely escalate the testing costs. Ultimately the testing cost will reach a certain limit and the project manager will decided to limit the budget allocated for testing. This kind of influence propagations will not happen overnight.

Project manager need to be sure if the software to be developed is bug free to the required level and has to verify this with the intended stakeholders before he/she decided to cut the budget of testing team. Unfortunately, casual loops like the one in figure 9.4 cannot delays in propagating influence effects.

Delays occur when an output of a certain process lags behind an input [6]. Considering the time required in information processing, decision making, manufacturing etc, we can reasonably assume delays occur in every practical system.

Two types of delays can occur in goal change impact analysis and other systems [6]. The first one is, "***Material Delay"*** where time spent when physical entities are processed. E.g: time to produce a new car. The second one is ***"Information Delay"*** Involves delays associated with beliefs, expectations, forecasts and projects. All of these causes delays because they are based on information available to the decision maker at that time and gathering and analyzing information will take a while.

B.1: Delays in Goal Change Impacts

In goal change impact analysis delays can be caused by numerous factors like time difference between change occurrence and the time the change is felt by stakeholders, time spent for gathering information

about the change, time to make decision what to do about the change involving stakeholders, the time to implement the decisions etc.

From the two types of delays we saw earlier, we argue Information delays are prevalent in goal analysis because goal analysis doesn't involve usually production of entities. Rather it involves analyzing of information about entities in the environment and predicts upcoming effects based on the gathered information. Moreover goal satisfaction levels are not physical entities; rather they are conceptual entities that are a result of stakeholder decision making process based on available information.

In information delay, what is perceived as a delay is stakeholders' belief about the entities in the environment. When new information is discovered about an entity in the business environment, stakeholders will analyze it and change their belief about it. The simplest and most widely used models for belief adjustment and forecasting is the technique of exponential smoothing or adaptive expectations [6]. This technique involves gradual adjusting of stakeholders' goal satisfaction prediction to the reported value of the goal satisfaction level. More on this and other delays in system dynamics can be found in [6].

B.2: Estimating Duration of Delay

There are two possible ways of estimating delays in dynamic processes [6]. If it is a well known change, statistical prediction can also be employed to measure the expected delay based on lessons learned from the previous delays.

But if the delay is due to a new kind of situation or if there is no data about previous experiences (This is usually the case), we must estimate the parameters to determine delays from direct inspection of the delay process. Experience with delays in similar systems or personal judgments can be used here. But judgmental estimates of delays can be quite unreliable and usually underestimate their duration [6].

B.3: Possible ways of modeling Delay in BiZZdesign Architect

BiZZdesign extensible feature via custom configuration files can be used to assign delay variables for the influence relations expected to manifest a delay. Goals that receive influences from relations with delay can also be assigned a variable to possess the estimated delay.

The estimated delays can be calculated by one of the methods discussed above. Regardless of the method chosen, delay estimations process need an active involvement of the relevant stakeholders and subject area experts.

Due to numerous factors affecting delays, delays are usually underestimated. To avoid this, a process can be decomposed in to multiple steps. The delay of the smaller processes can be calculated more accurately. Finally the individual delays in these steps can be decomposed to represent the delay of the main process [6] .

Appendix C: Sample Code for Qualitative Reasoning Tool

The sample codes given here are written in BiZZdesign architect scripting language. This scripting language is used to manipulate EA design components developed in BiZZdesign Architect.

C1: Qualitative Rule Application Process

```
function applyRulesSat(goal, relation) {
	obj = relation.relatedTo(goal);
	newSat = undefined;
	infList = undefined;
	infList = List();
	if ( obj is "MotivationGoal" ) {
		strength = relation.attrValue("infType");
		influencingGoal = obj;
		if (influencingGoal.attrValue("QualitativeSatisfaction") == Enum( "FullyDen","SatisfiedType")){

			if ( strength == Enum("strongneg", "InfluenceType") ) {
				newSat = Enum( "FullySat","SatisfiedType");
			}
			else if ( strength == Enum("weakneg", "InfluenceType") ) {
				newSat =  Enum("PartiallySat", "SatisfiedType");
			}
			else if ( strength == Enum("weakpos", "InfluenceType") ) {
				newSat = Enum("PartiallyDen", "SatisfiedType");
			}
			else if ( strength == Enum("strongpos", "InfluenceType") ) {
				newSat = Enum( "FullyDen","SatisfiedType");
			}
		}

	else if ( obj is "AndJunction" ) { // assumption: no nesting of junctions
		andSatList = List();
		forall "InfluenceRelation" r in obj.relationsTo() {
			g = r.relatedTo(obj);
			andSatList.add(g.attrValue("QualitativeSatisfaction"));
		}
		newSat = minList(andSatList);
		newSat = toQualitative(newSat);
	}
	else if ( obj is "OrJunction" ) { // assumption: no nesting of junctions
		orSatList = List();
		forall "InfluenceRelation" r in obj.relationsTo() {
			g = r.relatedTo(obj);
			orSatList.add(g.attrValue("QualitativeSatisfaction"));
		}
```

```
                    newSat = maxList(orSatList);
                    newSat = toQualitative(newSat);
            }
            return newSat;
            //resultantSat = CheckConflict(infList);
    //return  resultantSat;
}
```

Appendix D: Sample Code for Interfacing with Excel.

```
excel = ExternalObject("Excel.Application");
workbooks = excel.Workbooks;
//workbooks.Open("D:\\Shared\\Projects\\ArchiVal\\Archisurance\\Bedell.xls");
//excel.ActiveWorkbook.Sheets(3).Activate();
newWorkbook = workbooks.Add();
newSheet = newWorkbook.Sheets().Add();

excel.Visible = true;

// Prepare Table

startRow = 1;

goalCol = 1;
satisfactionCol = 2;

excel.Cells(startRow, goalCol).Value = "Goal Name";
excel.Cells(startRow, satisfactionCol).Value = "Satisfaction Level";

newSheet.Rows(1).Font.Bold = true;
newSheet.Columns(2).HorizontalAlignment = xlRight;
newSheet.Columns(3).HorizontalAlignment = xlRight;
newSheet.Columns(1).ColumnWidth = 30;
newSheet.Columns(2).ColumnWidth = 30;

i=1;
forall "MotivationGoal" goal in model {
   excel.Cells(startRow+i, goalCol).Value = goal;
            excel.Cells(startRow+i, satisfactionCol).Value = goal.attrValue("QualitativeSatisfaction");
            i = i + 1;

}

newChart = newSheet.ChartObjects().Add(10, 10, 360, 260).Chart;
//newChart.Name = "Goal Satisfaction levels";
newChart.Type = xlColumnClustered;
newChart.HasTitle = true;
newChart.ChartTitle.Text = "Goal Satisfaction levels";
newChart.ChartTitle.Font.Size = 16;

newChart.SetSourceData (newSheet.Range("A2:B22"));
//workbooks.Close();
//excel.Quit();
```

Appendix E: Sample Code for Quantitative Reasoning Tool.

E1: Fuzzification Process

```
function GoalBelongsTo(gInput){
   belongToSet1= undefined;
   belongToSet2= undefined;
   setValue1 = 0.00;
   setValue2 = 0.00;

 if (gInput <= -80.0){
   belongToSet1 = Pair(-2, 1);
   belongToSet2 = Pair(-1, 0);
 }
  else if (-80 < gInput && gInput< -60 ){
    setValue1 = (-60-gInput)/(-60--80);
    setValue2 = (gInput--80)/(-60--80);
    belongToSet1 = Pair(-2, setValue1);
    belongToSet2 = Pair(-1, setValue2);
  }
  else if (-60 <= gInput && gInput<= -40){
   belongToSet1 = Pair(-1, 1);
   belongToSet2 = Pair(0, 0);
  }
  else if (-40 < gInput && gInput<-20){
    setValue1 = (-20-gInput)/(-20--40);
    setValue2 = (gInput--40)/(-20--40);
    belongToSet1 = Pair(-1, setValue1);
    belongToSet2 = Pair(0, setValue2);
  }
  else if (-20 <= gInput && gInput<=20){
   belongToSet1 = Pair(0, 1);
   belongToSet2 = Pair(1, 0);
  }
  else if (20 < gInput && gInput<40){
    setValue1 = (40-gInput)/(40-20);
    setValue2 = (gInput-20)/(40-20);
    belongToSet1 = Pair(0, setValue1);
    belongToSet2 = Pair(1, setValue2);
  }
  else if (40 <= gInput && gInput<=60){
    belongToSet1 = Pair(1, 1);
    belongToSet2 = Pair(2, 0);
  }
  else if (60 < gInput && gInput<80){
```

```
  setValue1 = (80-gInput)/(80-60);
  setValue2 = (gInput-60)/(80-60);
  belongToSet1 = Pair(1, setValue1);
  belongToSet2 = Pair(2, setValue2);
 }
 else {
  belongToSet1 = Pair(1, 0);
  belongToSet2 = Pair(2, 1);
 }
 goalResult = Index(1,belongToSet1, 2, belongToSet2);
 return goalResult;
// return belongToSet1;
}
```

E2: Fuzzy Rule Application Sample Process

```
function applyFuzzyRules(goalValue, relationValue){

  SatValue1 = Pair(0,0);
  SatValue2 = Pair(0,0);
  SatValue3 = Pair(0,0);
  SatValue4 = Pair(0,0);

  resultantSet = undefined;


                goalSatValue1 = goalValue.valueFor(1);
                goalSatValue2 = goalValue.valueFor(2);

                relSatValue1 = relationValue.valueFor(1);
                relSatValue2 = relationValue.valueFor(2);

                        newSatPlace = undefined;
                        newSatValue = undefined;

            if ( goalSatValue1.first == -2 && relSatValue1.first== -2) {
                        newSatPlace = 2;
                        newSatValue = min(goalSatValue1.second, relSatValue1.second);
                }
                else if ( goalSatValue1.first == -2 && relSatValue1.first== -1) {
                        newSatPlace = 1;
                        newSatValue = min(goalSatValue1.second, relSatValue1.second);
                }
                else if ( goalSatValue1.first == -2 && relSatValue1.first== 0) {
```

```
                newSatPlace = 0;
                newSatValue = min(goalSatValue1.second, relSatValue1.second);
        }
        else if ( goalSatValue1.first == -2 && relSatValue1.first== 1) {
                newSatPlace = -1;
                newSatValue = min(goalSatValue1.second, relSatValue1.second);
        }
        else if ( goalSatValue1.first == -2 && relSatValue1.first== 2) {
                newSatPlace = -2;
                newSatValue = min(goalSatValue1.second, relSatValue1.second);
        }

        …….

        else if ( goalSatValue2.first == 2 && relSatValue2.first== -2) {
                newSatPlace = -2;
                newSatValue = min(goalSatValue2.second, relSatValue2.second);
        }
        else if ( goalSatValue2.first == 2 && relSatValue2.first== -1) {
                newSatPlace = -1;
                newSatValue = min(goalSatValue2.second, relSatValue2.second);
        }
        else if ( goalSatValue2.first == 2 && relSatValue2.first== 0) {
                newSatPlace = 0;
                newSatValue = min(goalSatValue2.second, relSatValue2.second);
        }
        else if ( goalSatValue2.first == 2 && relSatValue2.first== 1) {
                newSatPlace = 1;
                newSatValue = min(goalSatValue2.second, relSatValue2.second);
        }
        else if ( goalSatValue2.first == 2 && relSatValue2.first== 2){
                newSatPlace = 2;
                newSatValue = min(goalSatValue2.second, relSatValue2.second);
        }

        // Assigning the third option
    SatValue4 = Pair(newSatPlace,newSatValue);

 resultantSet = Index(1,SatValue1, 2,SatValue2, 3,SatValue3, 4,SatValue4);
 return resultantSet;
}
```

E3: Finding Centroid and Defuzzification Process

```
function centroid(pairValue){
   // This function uses the formula centroid = (2ac+a*a+cb+ab+b*b)/(3*(a+b)) c is the skew, a is upper base and b
is the lower base.
   // Ref: http://www.efunda.com/math/areas/Trapezoid.cfm
   area = undefined;
   skew = undefined;
   position = pairValue.first;
   membership = pairValue.second;
   if (position == -2){
    topBase = upperBase(position, membership);
    skew = skewValue(position, membership);
    area = (membership*0.5)*(topBase+40);
    tempCentroid = ((2*topBase*skew)+(topBase*topBase)+(skew*40)+(40*topBase)+(40*40))/(3*(topBase+40));
    tempCentroid = tempCentroid+(-100);
   }

   else if (position == -1){
    topBase = upperBase(position, membership);
    skew = skewValue(position, membership);
    area = (membership*0.5)*(topBase+60);
    tempCentroid = ((2*topBase*skew)+(topBase*topBase)+(skew*60)+(60*topBase)+(60*60))/(3*(topBase+60));
    tempCentroid = tempCentroid+(-80);
   }

   else if (position == 0){
    topBase = upperBase(position, membership);
    skew = skewValue(position, membership);
    area = (membership*0.5)*(topBase+80);
    tempCentroid = ((2*topBase*skew)+(topBase*topBase)+(skew*80)+(80*topBase)+(80*80))/(3*(topBase+80));
    tempCentroid = tempCentroid+(-40);
   }

   else if (position == 1){
    topBase = upperBase(position, membership);
    skew = skewValue(position, membership);
    area = (membership*0.5)*(topBase+60);
    tempCentroid = ((2*topBase*skew)+(topBase*topBase)+(skew*60)+(60*topBase)+(60*60))/(3*(topBase+60));
    tempCentroid = tempCentroid+(20);

   }

   else if (position ==2){
    topBase = upperBase(position, membership);
    skew = skewValue(position, membership);
```

```
   area = (membership*0.5)*(topBase+40);
   tempCentroid = ((2*topBase*skew)+(topBase*topBase)+(skew*40)+(40*topBase)+(40*40))/(3*(topBase+40));
   tempCentroid = tempCentroid+(60);
  }

 return Pair(area,tempCentroid);
}


function DeffuzzifyWeightedAverage(memberShipValue){


   tempCentroid1 = memberShipValue.valueFor(1);
   tempCentroid2 = memberShipValue.valueFor(2);
   tempCentroid3 = memberShipValue.valueFor(3);
   tempCentroid4 = memberShipValue.valueFor(4);

   centroidPair1 = centroid(tempCentroid1);
   centroidPair2 = centroid(tempCentroid2);
   centroidPair3 = centroid(tempCentroid3);
   centroidPair4 = centroid(tempCentroid4);
   /*
   output "centroid1",centroidPair1;
   output "centroid2",centroidPair2;
   output "centroid3",centroidPair3;
   output "centroid4",centroidPair4;
   */


           area1 = centroidPair1.first;
           area2 = centroidPair2.first;
           area3 = centroidPair3.first;
           area4 = centroidPair4.first;

           centroid1 = centroidPair1.second;
           centroid2 = centroidPair2.second;
           centroid3 = centroidPair3.second;
           centroid4 = centroidPair4.second;

   crispOutput =
((centroid1*area1)+(centroid2*area2)+(centroid3*area3)+(centroid4*area4))/(area1+area2+area3+area4);
   return crispOutput;

}
```