

TimedGRL: Specifying Goal Models Over Time

Aprajita

Master of Engineering

Department of Electrical and Computer Engineering
McGill University, Montreal

June 2017

A thesis submitted to McGill University in partial fulfillment of the requirements of
the degree of Master of Engineering

© Aprajita 2017

ABSTRACT

This thesis introduces TimedGRL, an extension of the current GRL (Goal-oriented Requirement Language) standard, which enables the modeling and analysis of a comprehensive set of changes to a goal model over time. Goal-oriented analysis, in general, deals with finding a solution to the modeled problem, which tries to accomplish desired system qualities while taking care of stakeholder needs simultaneously. It can be used to influence design decisions as it allows performing trade-off analysis among alternative solutions by determining which proposed solution strikes a more appropriate balance of stakeholder objectives and desired qualities of a system. Some common goal-oriented languages such as i*, GRL, the NFR Framework, and KAOS offer standard language definitions to support varying forms of such analysis. However, in most of these cases, the analysis is focused on only one snapshot in time. To capture evolving systems, several goal models need to be created that represent different stages of that system. The current language definitions do not allow these models to be connected and make it difficult to analyze them simultaneously. Moreover, it requires extra effort to maintain and consistently update these models that actually represent the same system and have many repeated elements.

TimedGRL proposes capturing the goal model and the changes to the goal model in a single model to facilitate system evolution. The metamodel for TimedGRL is presented, and an analysis algorithm for TimedGRL models is defined and implemented in jUCMNav, the most comprehensive GRL modeling tool to date. A hypothetical but realistic example from the sustainability domain is used to illustrate the introduced concepts. Moreover, a detailed case study has been performed to show TimedGRL's functionality and usefulness.

ABRÉGÉ

Cette thèse présente TimedGRL, une extension de la norme GRL (Goal-oriented Requirement Language) qui permet de modéliser et d'analyser un ensemble complet de modifications à un modèle orienté but au fil du temps.

L'analyse orientée but, en général, consiste à trouver une solution au problème modélisé, qui tente de satisfaire les qualités souhaitées du système tout en considérant simultanément les besoins des parties prenantes. Cette analyse peut être utilisée pour influencer les décisions de conception car elle permet d'analyser les compromis entre solutions alternatives en déterminant la solution proposée qui offrira l'équilibre le plus approprié entre objectifs des parties prenantes et qualités souhaitées d'un système. Certains langages orientés but populaires tels que i^* , GRL, NFR Framework et KAOS offrent des concepts standards pour supporter des représentations diverses de ce type d'analyse. Cependant, dans la plupart de ces cas, l'analyse se concentre sur un seul moment instantané. Pour capturer les systèmes en évolution, plusieurs modèles orientés but doivent donc être créés pour représenter différents moments de ce système dans le temps. Les langages actuels ne permettent pas de connecter ces modèles, ce qui rend difficile leur analyse simultanée. De plus, il faut des efforts supplémentaires pour maintenir et constamment mettre à jour ces modèles qui représentent réellement le même système et qui ont de nombreux éléments répétés.

TimedGRL propose de représenter un modèle orienté but et les modifications apportées à ce modèle dans un modèle unique afin de faciliter l'évolution du système. Le métamodèle pour TimedGRL est présenté et un algorithme d'analyse pour les modèles TimedGRL est défini et

implémenté dans jUCMNav, l'outil de modélisation GRL le plus complet à ce jour. Un exemple hypothétique mais réaliste relié à la durabilité est utilisé pour illustrer les concepts introduits. De plus, une étude de cas détaillée a été effectuée pour démontrer la fonctionnalité et l'utilité de TimedGRL.

ACKNOWLEDGMENTS

I would like to thank all the people who contributed in some way to the work described in this thesis. Foremost, I would like to express my sincere gratitude to my supervisor Prof. Gunter Mussbacher, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way. He introduced me to the general research area of requirements modeling and made sure I was always on right track. Without his encouragement and guidance, this thesis would not have been completed. I could not have imagined having a better supervisor and mentor for my Masters study.

My sincere thanks go to Prof. Jörg Kienzle and members from Software Engineering lab for providing valuable feedback during my thesis. Their inputs always helped in taking the best decision and hence, steered my work into the right direction.

I am grateful for the funding source that allowed me to pursue my graduate school studies: Fonds de recherche du Québec - Nature et technologies (FRQNT).

I thank all my friends here at McGill University, who always challenged me to do better. They made my stay here in Montreal memorable and fun.

Finally, I must express my very profound gratitude to my parents and to my brother for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

TABLE OF CONTENTS

| | |
|--|------|
| ABSTRACT..... | ii |
| ABRÉGÉ..... | iii |
| ACKNOWLEDGMENTS | v |
| TABLE OF CONTENTS..... | vi |
| LIST OF TABLES..... | viii |
| LIST OF FIGURES | ix |
| CHAPTER 1: Introduction | 1 |
| 1.1 Motivation..... | 1 |
| 1.2 Contributions and Methodology | 2 |
| 1.3 Thesis Organization | 3 |
| CHAPTER 2: Background Information..... | 5 |
| 2.1 Goal-oriented Requirements Language | 5 |
| 2.1.1 Basic GRL Notation..... | 5 |
| 2.2 Goal Model Evaluation | 7 |
| 2.3 Summary | 9 |
| CHAPTER 3: TimedGRL Concepts..... | 10 |
| 3.1 Abstract Metamodel..... | 10 |
| 3.2 Illustrating Example..... | 14 |
| 3.3 Changes Supported in TimedGRL..... | 17 |
| 3.3.1 Deactivation Changes | 18 |
| 3.3.2 Numeric Changes | 19 |
| 3.3.3 Enumeration Changes | 24 |
| 3.4 Model Changes not supported by TimedGRL | 25 |
| 3.5 Discussion | 26 |
| 3.6 Summary | 28 |
| CHAPTER 4: TimedGRL Evaluation | 29 |
| 4.1 Algorithm Overview | 29 |
| 4.2 Included Dynamic Contexts..... | 37 |
| 4.3 Evaluation Trends over a Time Interval | 39 |
| 4.4 Summary | 48 |
| CHAPTER 5: Validation using Case Study | 49 |
| 5.1 The Case Study: Audit Report on Ontario Power Generation’s HR | 49 |
| 5.1.1 System Description | 51 |
| 5.2 Identification and Addition of Changes to Model Elements | 69 |
| 5.2.1 Expected Changes as of 10 th December, 2013..... | 72 |
| 5.2.2 Changes after Status Update as of 30 th June, 2014..... | 78 |
| 5.2.3 Changes after Follow-up Review as of 31 st July, 2015 | 82 |
| 5.3 Evaluation of Goal Model at Timepoints..... | 84 |
| 5.3.1 Evaluations on 10 th December, 2013 | 85 |
| 5.3.2 Evaluations on 30 th June, 2014 | 86 |

| | |
|--|-----|
| 5.3.3 Evaluations on 31 st July, 2015 | 89 |
| 5.3.4 Evaluations on 31 st December, 2015 | 91 |
| 5.3.5 Evaluations on 31 st December, 2017 | 92 |
| 5.3.6 Discussion | 93 |
| 5.4 Overall Evaluation of Goal Model over a Timepoint Group | 94 |
| 5.4.1 Overall System Behavior for “DynamicContext (Expectation)” | 96 |
| 5.4.2 Overall System Behavior for “DynamicContext (2014_Update)” | 100 |
| 5.4.3 Overall System Behavior for “DynamicContext (Final)” | 103 |
| 5.4.4 Discussion | 105 |
| 5.5 Summary | 107 |
| CHAPTER 6: Related Work | 108 |
| CHAPTER 7: Conclusions and Future Work | 115 |
| REFERENCES | 117 |

LIST OF TABLES

| | |
|--|----|
| Table 1 ELEMENTS, AFFECTED PROPERTIES, AND CHANGES | 18 |
|--|----|

LIST OF FIGURES

| | |
|--|----|
| Figure 1: Excerpt of Goal Model for Energy Efficiency Goals of a City | 7 |
| Figure 2: An example to demonstrate the Quantitative Evaluation Algorithm | 8 |
| Figure 3: Extension to GRL Metamodel for TimedGRL..... | 11 |
| Figure 4: Goal Model for Energy Efficiency Goals of a City..... | 14 |
| Figure 5: Evaluation on 1 st January, 2016 | 21 |
| Figure 6: Evaluation on 1 st January, 2026 | 22 |
| Figure 7: Evaluation on 1 st January, 2031 | 22 |
| Figure 8: Model Changes not supported by TimedGRL | 25 |
| Figure 9: Combining Numeric Changes in Included Dynamic Contexts | 38 |
| Figure 10: Evaluation on 1 st January, 2026 (with Actor Importance values defined) | 42 |
| Figure 11: Heatmap depicting Overall Evaluation from 15 th December, 2025, to 15 th January, 2026..... | 43 |
| Figure 12: Variation in Evaluation and Importance of the goal "Encourage Usage of Renewable Energy" | 44 |
| Figure 13: Variation in Evaluation by Decomposition Links of "Encourage Usage of Renewable Energy" | 46 |
| Figure 14: Variation in Evaluation and Importance of the resource "Research facility" | 47 |
| Figure 15: Variation in Evaluation and Importance of the actor "Renewable Nuclear Fusion Research Group" | 48 |
| Figure 16: Goal Model for Recommendation 1 | 55 |
| Figure 17: Goal Model for Recommendation 2 | 58 |
| Figure 18: Goal Model for Recommendation 3 | 61 |
| Figure 19: Goal Model for Recommendation 4 | 63 |
| Figure 20: Goal Model for Recommendations 5 and 6..... | 65 |
| Figure 21: Comprehensive Goal Model for OPG Audit Report | 69 |
| Figure 22: Initial evaluation (EvaluationStrategy6) of goal model of Recommendation 1 | 73 |
| Figure 23: Initial evaluation (EvaluationStrategy6) of the comprehensive goal model | 73 |
| Figure 24: (a) Specification of Dynamic Contexts and (b) Specification of Timepoints | 75 |
| Figure 25: Addition of a Linear Change through "Manage Change" dialog box | 76 |
| Figure 26: Addition of a Deactivation Change through "Manage Change" dialog box | 77 |
| Figure 27: Changes visualized in the table for the resource "Hiring panel" for the selected Dynamic Context | 77 |
| Figure 28: Changes in the Dynamic Context "DynamicContext (2014_Update)" | 82 |
| Figure 29: Evaluation of Recommendation 1 Model on 10 th December, 2013 | 85 |
| Figure 30: Evaluation of Comprehensive Model on 10 th December, 2013 | 86 |
| Figure 31: Evaluation of Recommendation 1 Model on 30 th June, 2014 (DynamicContext (Expectation)) | 86 |
| Figure 32: Evaluation of Comprehensive Model on 30 th June, 2014 (DynamicContext (Expectation)) | 87 |

| | |
|--|-----|
| Figure 33: Evaluation of Recommendation 1 Model on 30 th June, 2014 (DynamicContext (2014_Update)) | 87 |
| Figure 34: Evaluation of Comprehensive Model on 30 th June, 2014 (DynamicContext (2014_Update)) | 88 |
| Figure 35: Evaluation of Comprehensive Model on 30 th June, 2014 (DynamicContext (Final)) | 88 |
| Figure 36: Evaluation of Recommendation 1 Model on 31 st July, 2015 (DynamicContext (Expectation)) | 89 |
| Figure 37: Evaluation of Comprehensive Model on 31 st July, 2015 (DynamicContext (Expectation)) | 90 |
| Figure 38: Evaluation of Recommendation 1 Model on 31 st July, 2015 (DynamicContext (2014_Update)) | 90 |
| Figure 39: Evaluation of Comprehensive Model on 31 st July, 2015 (DynamicContext (2014_Update)) | 91 |
| Figure 40: Evaluation of Comprehensive Model on 31 st July, 2015 (DynamicContext (Final)) .. | 91 |
| Figure 41: Evaluation of Recommendation 1 Model on 31 st December, 2015 | 92 |
| Figure 42: Evaluation of Comprehensive Model on 31 st December, 2015 | 92 |
| Figure 43: Updating Granularity in jUCMNav Preferences | 96 |
| Figure 44: Heatmap for DynamicContext (Expectation) | 97 |
| Figure 45: Variation in Evaluation and Importance for DynamicContext (Expectation) (for mentioned goal) | 98 |
| Figure 46: Variation in Evaluation by Contribution Links for DynamicContext (Expectation) (for mentioned goal) | 99 |
| Figure 47: Behavior of the system for DynamicContext (Expectation) | 99 |
| Figure 48: Heatmap for DynamicContext (2014_Update) | 100 |
| Figure 49: Variation in Evaluation and Importance for DynamicContext (2014_Update) (for mentioned goal) | 101 |
| Figure 50: Variation in Evaluation by Contribution Links for DynamicContext (2014_Update) (for mentioned goal) | 102 |
| Figure 51: Behavior of the system for DynamicContext (2014_Update) | 102 |
| Figure 52: Heatmap for DynamicContext (Final) | 103 |
| Figure 53: Variation in Evaluation and Importance for DynamicContext (Final) (for mentioned goal) | 104 |
| Figure 54: Variation in Evaluation by Contribution Links for DynamicContext (Final) (for mentioned goal) | 105 |
| Figure 55: Behavior of the system for DynamicContext (Final) | 105 |

CHAPTER 1: Introduction

Goal modeling plays a significant role in Requirements Engineering (RE) [17]. It is used to capture and visualize different steps of the RE process. It encourages study of requirements early in the development phase, rather than later. A goal model of a system represents its qualities, requirements, stakeholder objectives, and relationships among those elements. Thus, in short, it is used to define the purpose of the system it represents, along with the ways to achieve that purpose. Construction of a goal model requires thorough understanding of the system and all its stakeholders. Popular languages used to model goals and their relationships include i* [16], the Goal-oriented Requirement Language (GRL) [11], the NFR framework [4], and KAOS [15].

Goal model analysis, also known as evaluation, deals with finding a solution to the modeled problem, which tries to accomplish desired system qualities while taking care of stakeholder needs simultaneously. It can be used to influence design decisions by providing quantitative or qualitative analysis of different alternatives. It can also be used as a tool to judge the status of the system and, to identify the tasks needed to achieve the goals of the system.

1.1 Motivation

Goal model analyses are usually focused on a single snapshot in time, i.e., they possess static behavior. During analysis, the complete model is evaluated without considering any changes to this model over time. However, systems do evolve over time. As the time advances, the objectives, requirements, stakeholders, and relationships among those elements change. Sometimes it is possible to predict these changes as the patterns they follow can be determined. A perfect example for this can be any system belonging to the sustainability domain, which by

its very nature requires the problem to be analyzed over a longer period of time. Other examples are systems where the attitude of the users towards the system may change over time or where disruptive technology may lead to new applicable solutions.

It is still possible to visualize the evaluation of such a system at different points in time using current analyses. One option is to make copies of the goal model to which the changes over time are applied, and the other option is to update the same goal model again and again with the new elements, according to the requirements. Considering the first option, for each model corresponding to a point in time, the modeler needs to define all the associated elements and their relationships. Thus, all these models are going to be similar, except for the elements and attributes that have changed. This is space-consuming, time-consuming, and error-prone as the modeler needs to maintain multiple goal models. The second option to build and update the same goal model may be preferable to the first one as it is less space-consuming. However, this is still error-prone and time-consuming as the modeler has to be vigilant about the changes occurring in each and every element, and update the model accordingly before every evaluation. These approaches make it difficult to maintain a consistent model.

1.2 Contributions and Methodology

A better approach to the problem stated in the previous section uses a single goal model which can visualize the evolution of the system by explicitly defining the expected changes for the required model elements. The Timed Goal-oriented Requirements Language (TimedGRL) [18] introduces this approach to goal modeling, by incorporating the concept of changes over time into the metamodel of the current GRL standard. Any system, where the system's progress over time needs to be tracked and analyzed, could benefit from the proposed

extensions to goal modeling, which is demonstrated on GRL. However, the proposed approach is generally applicable to all goal modeling languages, especially those with a propagation-based evaluation mechanism.

A hypothetical but realistic example from sustainability domain is used to illustrate the introduced concepts. In addition, an algorithm is presented that analyzes the goal model over time based on the proposed TimedGRL metamodel. This algorithm has been implemented in the open-source jUCMNav tool [12], an Eclipse-based editor for URN models. Furthermore, a detailed case study has been performed using the implemented tool to show its complete functionality and usefulness.

1.3 Thesis Organization

The remainder of this thesis document is organized as follows:

Chapter 2 gives background information on GRL and goal model analysis techniques.

Chapter 3 describes the TimedGRL concepts introduced into the GRL metamodel. It also provides an example scenario, which is used to provide detailed description of all the introduced concepts.

Chapter 4 defines an algorithm for the evaluation of TimedGRL, which extends existing goal model analysis techniques. This chapter also describes various visualization techniques for TimedGRL evaluation.

Chapter 5 describes a case study performed to demonstrate successful implementation of TimedGRL in jUCMNav tool as well as its usefulness in real world scenarios.

Chapter 6 gives a brief overview of related work.

Chapter 7 concludes the thesis and presents future work.

CHAPTER 2: Background Information

This chapter discusses important background information needed before introducing the Timed Goal-oriented Requirements Language (TimedGRL).






2.1 Goal-oriented Requirements Language




Goal modeling has been acknowledged as a substantial part of Requirements Engineering [25]. It allows the modelers to carve out system requirements in a way that is comprehensible and can communicate the motivations behind those requirements. It also helps in ensuring the correctness of the system that is being built. Several goal modeling frameworks exist such as *i** [16], the Goal-oriented Requirement Language (GRL) [11], the NFR framework [4], KAOS [15], and Tropos [26].

The Goal-oriented Requirement Language (GRL) is one of the two parts of the User Requirements Notation [1], which is published as a standard by the International Telecommunication Union in the Z.15x series [11]. It is a goal modeling standard based on *i** [16] and the NFR Framework [4]. GRL allows the modelers to visualize and communicate common concepts of goal modeling notations such as stakeholders and their objectives, systems and their desired qualities, and solutions that impact these objectives and qualities. Its support for both qualitative and quantitative attributes and integration with a scenario notation makes it stand out among well-known goal modeling languages.

2.1.1 Basic GRL Notation

A GRL goal model is a connected graph that shows the high-level business goals and qualities of interest to a stakeholder and the solutions considered to accomplish these high-level

elements. A stakeholder of a system or the system itself is represented as an actor (, e.g., “Ministry of National Research” in Figure 1). Various intentional elements (softgoal, goal, task, resource) are available to capture the mentioned concerns and they can either belong to an actor or be independent. Objectives and qualities are modeled with softgoals and goals. A softgoal () is used to represent objectives that are somewhat obscure in nature and have no definite measure of satisfaction, whereas a goal (, e.g., “Support Research in Renewable Energy”) is used when the objective is clear and quantifiable. The solutions required to realize the objectives of a systems are represented as tasks (, e.g., “Build an Advanced Research Facility”). Softgoals, goals, and tasks may sometimes need resources (, e.g., “Research Facility”) in order to be achieved or completed.

Links are used to connect the various elements in a GRL goal model. Decomposition links (, e.g., between “Support Research in Renewable Energy” and “Ease Immigration Rules for Skilled Researchers”) allow an element to be decomposed into sub-elements. GRL supports AND, XOR, and IOR decomposition types. A dependency link (, e.g., between “Discover Renewable Controlled Nuclear Fusion Reaction” and “Support Research in Renewable Energy”) is used to visualize reliance of one element on another element, typically across actor boundaries. Desired influences of one element on another are represented using a contribution link (, e.g., between “Build an Advanced Research Facility” and “Research Facility”). Contributions are indicated qualitatively with labels (+ or –) or quantitatively as an integer value between –100 and 100. It is also possible to specify a quantitative contribution relatively to other contributions of the same element; however, the contribution values must be normalized in such a case [5][6] to allow collective comparison of all the elements in the goal model.

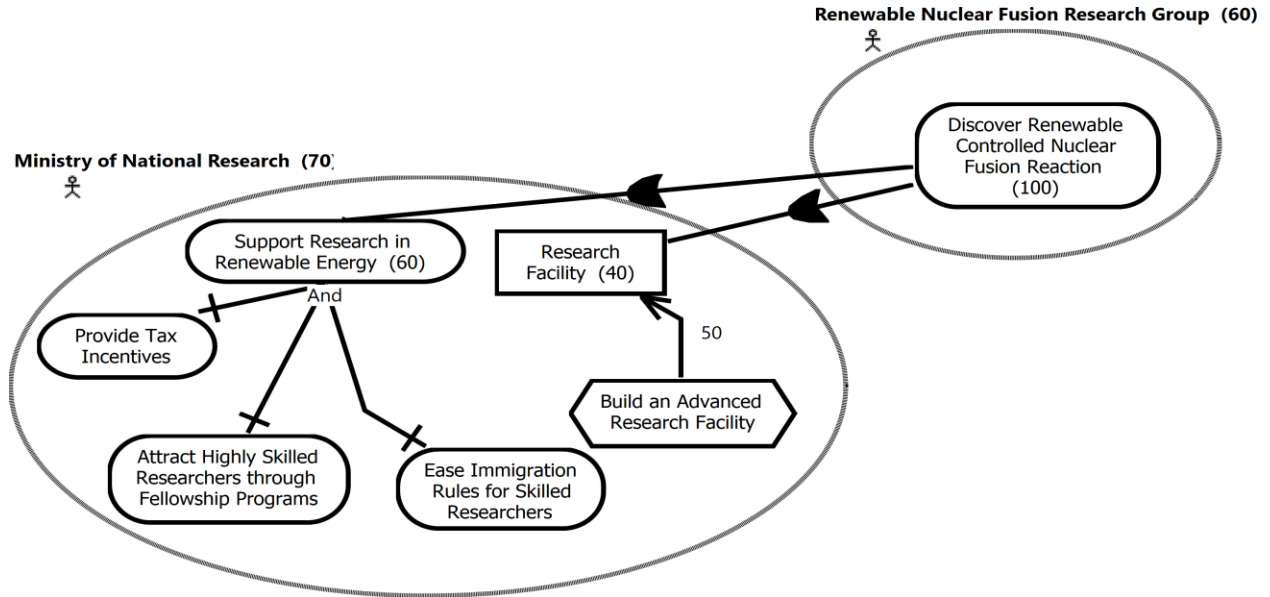


Figure 1: Excerpt of Goal Model for Energy Efficiency Goals of a City

2.2 Goal Model Evaluation

GRL offers propagation-based evaluation mechanisms [2], built on the NFR Framework [4], in order to support the analysis of stakeholder objectives and system qualities. An evaluation mechanism [2] allows the users to reason about high-level goals, system qualities, and desired solutions. The evaluation shows the impact of proposed candidate solutions to find the most appropriate trade-offs of often conflicting stakeholder goals.

A GRL evaluation strategy describes a possible solution by assigning initial qualitative or quantitative satisfaction values to a set of intentional elements in the GRL model, typically leaf nodes (i.e., tasks) (e.g., 20(*)) for “Increase Efficiency of Solar Power” in Figure 5; (*) indicates that this value is an initial value; in addition, an element with an initial value also has a dashed outline). These initial satisfaction values are then propagated to higher-level goal model elements (e.g., 57 for “Encourage usage of Renewable Energy”) by the evaluation mechanism. The satisfaction values are also color-coded from the minimum Red (-100) to Yellow (0) and the

maximum Green (100). It is also possible to define an importance attribute (quantitative or qualitative) for intentional elements inside actors, which is considered while evaluating the actors of the goal model. The importance is shown between parentheses in intentional elements: (H)igh, (M)edium, (L)ow, or None for qualitative evaluations, and an integer between 0 and 100 for quantitative evaluations (e.g., 60 for the goal “Support Research in Renewable Energy” in Figure 4).

The GRL standard provides non-normative examples of evaluation algorithms [2], all of which have been implemented in the jUCMNav tool [12]. Three evaluation algorithms applicable to GRL – a quantitative, a qualitative, and a hybrid approach – are described in more detail in [2].

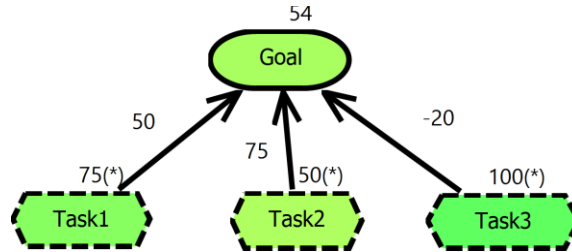


Figure 2: An example to demonstrate the Quantitative Evaluation Algorithm

Figure 2 demonstrates how the leaf nodes’ satisfaction values are propagated across contribution links to calculate the satisfaction value of the parent “Goal”, when the values are specified quantitatively (note that the sum is limited to the [-100,100] range):

$$\sum_{contributing\ elements} \frac{(element\ satisfaction\ value * contribution\ value)}{100}$$

$$= [(75 * 50) / 100] + [(50 * 75) / 100] + [(100 * -20) / 100] = 37 + 37 - 20 = 54 \text{ (Rounding up)}$$

GRL offers limited support for the analysis of goal models over time. Strategies may be used to define different initial satisfaction values for different time periods. However, time periods are not explicitly defined and changes to the goal model are restricted to contribution values with the help of contribution contexts [27]. In addition, the jUCMNav tool supports the visualization of trends given a set of strategies [3].

2.3 Summary

This chapter discusses the essential background information needed before diving into the concepts of TimedGRL. The basic GRL notation is briefly described along with some examples for illustration. GRL is going to be used throughout this thesis. Moreover, the existing evaluation algorithms are discussed. These concepts provide the perfect platform to introduce the TimedGRL metamodel in the next chapter.

CHAPTER 3: TimedGRL Concepts

This chapter first introduces the extensions to the GRL metamodel for TimedGRL. It, then, illustrates the need for the introduced metamodel elements with the help of an example.

3.1 Abstract Metamodel

In TimedGRL, it is possible to define changes to general goal model elements such as intentional elements, actors, and links. A *Change* is the behavior expected in an *element* over time. See Figure 3 for the abstract metamodel of TimedGRL, which introduces this concept, along with some other new concepts as discussed later on in this section, into the existing metamodel of GRL. A change can further be divided into numeric, enumeration, and deactivation changes:

- A *NumericChange* is used to define any behavior that can be expressed mathematically with respect to time. It can be further divided into constant, linear, quadratic, and formula changes, which will be discussed in detail in Section 3.3.2.
- An *EnumerationChange* specifies the switch in values over the course of time for any attribute with an enumeration, which will be discussed further in Section 3.3.3.
- A *DeactivationChange* controls the presence or absence of an element in the goal model over the course of time, which will be discussed in detail in Section 3.3.1.

Numeric and enumeration changes are grouped together as *PropertyChange*, as they both are applied to an attribute (*affectedProperty*) of an element. On the other hand, *DeactivationChanges* are applied to the elements themselves to define their actual existence

timeline in the model. Each change has a *start* date and an *end* date, which specify the time period when that particular change is applicable to the element.

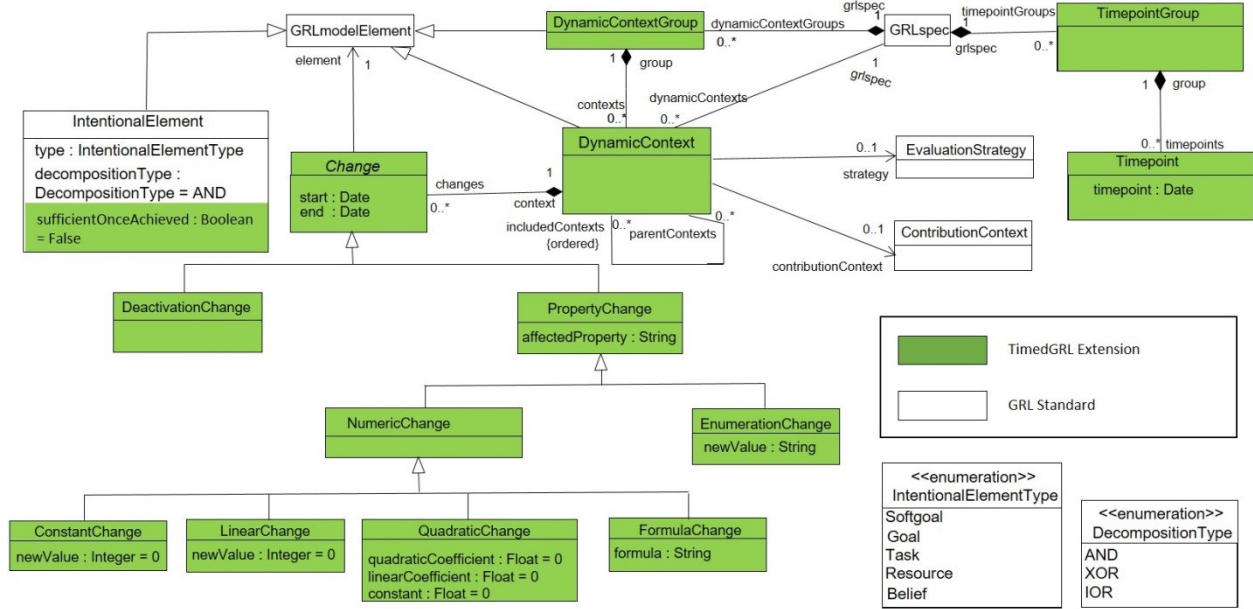


Figure 3: Extension to GRL Metamodel for TimedGRL

A new Boolean variable *sufficientOnceAchieved* is added to *IntentionalElement* to flag any element that stays fully satisfied once it has reached this level of satisfaction even when the contributions impacting the element are deactivated. This is particularly useful for resources which may be created by a task. The task contributes to the resource, but once the resource exists and can be used, the task is not needed anymore and can be deactivated. Nevertheless, the resource should remain fully satisfied.

The required concepts to state the changes in the elements of a goal model have already been discussed. However, that is not enough. We also need to specify the changes that act simultaneously to examine the comprehensive effect of these changes on the model. For this purpose, a *DynamicContext* is defined within which such changes can be grouped. This also gives the option of having multiple groups to facilitate the exploration of alternatives and trade-

offs. The existing GRL evaluation mechanism receives as input a set of initial satisfaction values (*EvaluationStrategy*) and possibly a set of contribution changes to override contribution levels of contribution links in the goal model (*ContributionContext*). Along with any changes, a dynamic context may thus include at the most one *EvaluationStrategy* and at most one *ContributionContext* to more comprehensively specify the situation to be evaluated. It is also possible to include other dynamic contexts in a dynamic context to override default values. These *includedContexts* give the modeler power to structure dynamic contexts more modularly, flexibly consider multiple situations, and get their evaluations at different timepoints by just including the context containing the desired changes. For example, a modeler may wish to explore a series of situations that only vary slightly by creating a base context. This base context is then included in all other contexts that override the base as needed. This can be used by the modeler to either choose the best performing option, or compare various options. A similar kind of inclusion has already been successfully used for evaluation strategies and contribution contexts in the GRL standard [11].

TimedGRL also needs to specify timepoints or timepoint groups, at which the model needs to be evaluated. For that purpose, the concepts of *Timepoint* and *TimepointGroup* are introduced. A *Timepoint* is a specific date, while a *TimepointGroup* is a set of dates of interest. Thus, TimedGRL is able to run the evaluation at a specific timepoint to get the status of the model corresponding to the selected dynamic context on that particular date. Moreover, it is possible to visualize the overall status of a model over a period of time corresponding to the selected dynamic context in the form of a heat map and charts by running the evaluation for a

TimepointGroup. This thesis talks about TimedGRL and its role in the quantitative analysis of a goal model over time and hence builds on the existing quantitative evaluation mechanism [2].

In addition, a valid TimedGRL model must obey the following constraints on the metamodel:

- The start of a change always comes before the end of a change:

OCL: context *Change* inv: *start.isBefore(end)*

- All changes of a particular attribute of an element in a dynamic context do not overlap:

OCL: context *DynamicContext* inv:

self.changes \rightarrow select(*c* : *Change* | *c.ocllsType(PropertyChange)*) \rightarrow forAll(*c1*, *c2* : *PropertyChange* | ((*c1.element* = *c2.element*) and (*c1.affectedProperty* = *c2.affectedProperty*)) implies (not *c1.end.isAfter(c2.start)* or not *c2.end.isAfter(c1.start)*))

- An included context's evaluation strategy and contribution context do not play any role in the evaluation of the goal model corresponding to its parent dynamic context. Only the changes in included contexts are considered by the parent dynamic context.

The last constraint simplifies the inclusion of one dynamic context into another, because it is not needed to define how several evaluation strategies and contribution contexts could possibly be combined.

3.2 Illustrating Example

This part illustrates the usefulness of the concepts introduced in Section 3.1 with the help of a hypothetical but realistic example from the sustainability domain. Let us consider a city that wants to encourage its usage of Renewable Energy by maximizing its overall Renewable Energy Efficiency while minimizing cost of Renewable Energy and limiting usage of Non-Renewable Energy. To get a more complete picture of this problem, the city needs to analyze its renewable energy strategy over the next 25 years. Figure 4 describes the problem with a goal model that focuses on the “Maximize Energy Efficiency of Renewable Resources” goal. This model serves the purpose of explaining different types of changes and how they affect the overall evaluation over time.

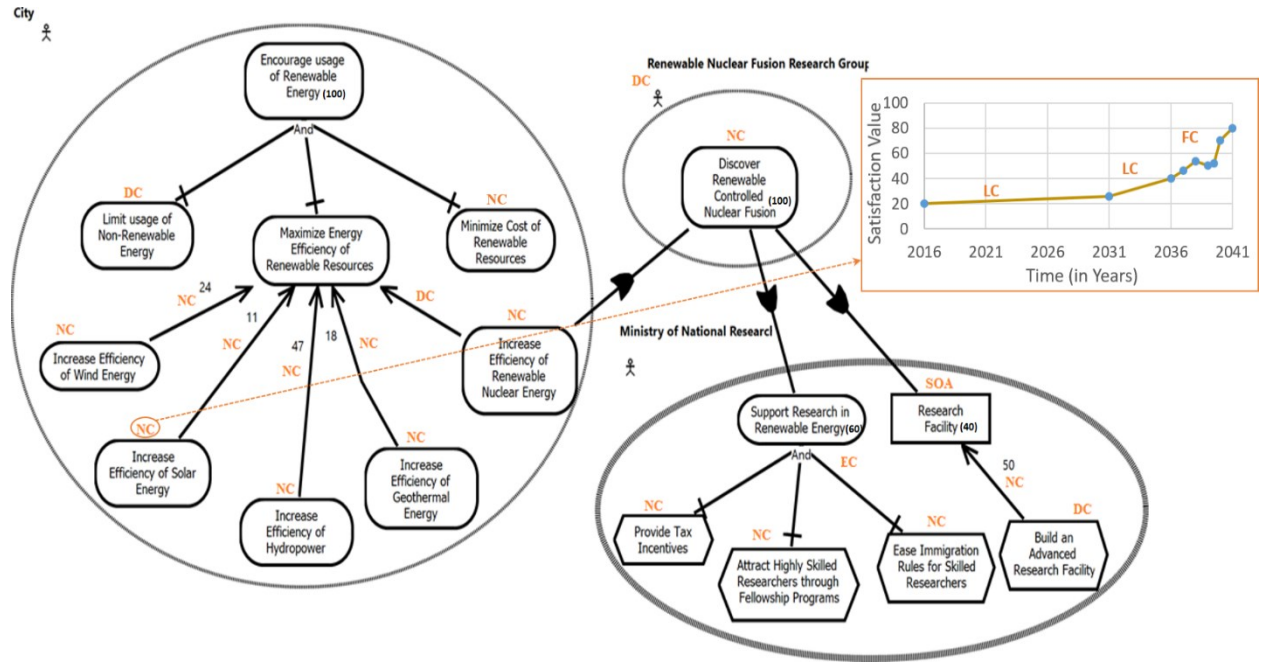


Figure 4: Goal Model for Energy Efficiency Goals of a City

All elements in the model expecting change over time are marked with DC for *DeactivationChange* (i.e., the element is not present in the model throughout the defined change time period), NC for *NumericChange*, or EC for *EnumerationChange*. The element, for which

the *sufficientOnceAchieved* flag is true, is marked with SOA. In the actual implementation, the visualization of change in an element is different than what's portrayed in the example here. Instead of marking DC, NC, or, EC, a single icon for change is used to denote all three kinds of changes.

The main goal “Encourage usage of Renewable Energy” is decomposed into “Maximize Energy Efficiency of Renewable Resources”, “Minimize Cost of Renewable Resources”, and “Limit usage of Non-Renewable Energy”. In the beginning, however, a lot of research still needs to be done in the field of Renewable Resources. Therefore, the city holds off on limiting the usage of Non-Renewable Energy until sufficient renewable resources are available. Note the DC label on the “Limit usage of Non-Renewable Energy” goal. Eventually however, all three sub-goals are equally important for the city to increase its dependency level on Renewable Energy. The city’s Renewable Energy Resources currently (in 2016) include Wind and Solar Energy, Hydro Power, and Geothermal Power. These all have corresponding individual goals, whose satisfaction values show the maximum efficiency they can achieve at this point in time (see Figure 5). They all contribute to the goal “Maximize Energy Efficiency of Renewable Resources”. These contribution values describe the share each resource has relatively to all other energy forms. Thus, 24% of overall renewable energy is obtained from wind energy, 11% from solar energy, 47% from hydro power, and 18% from geothermal power. The efficiencies of renewable resources (i.e., the satisfaction values) change through the course of time as the resources are improved continuously. Representative of all changes in the model, the details of the changes to the attribute of one element – the satisfaction value of “Increase Efficiency of Solar Energy” – are shown in the graph in the top right corner of Figure 4. The relative shares of

the resources (i.e., the contribution values) also change depending on their projected availabilities and increase in efficiencies. The city may define these changes by using any one of the following: estimates based on prior experience, predictions based on research, expert opinions, or simply explorative “what if” scenarios.

The city is also aware of a group of scientists that will begin work on a new highly efficient renewable resource using nuclear fusion reactors in two years’ time. These scientists are represented by the actor “Renewable Nuclear Fusion Research Group”. The main goal of this group is to discover renewable controlled nuclear fusion power with 90% efficiency in the next 15 years. Once available, the city wants to include this new resource in its energy portfolio from 2031 onwards.

The third actor in the goal model, “Ministry of National Research”, provides support to the goal “Discover Renewable Controlled Nuclear Fusion” now (in 2016) in anticipation of the founding of the research group by building an Advanced Research Facility and supporting the research overall via the tasks “Provide Tax Incentives”, “Attract Highly Skilled Researchers through Fellowship Programs”, and “Ease Immigration Rules for Skilled Researchers”. At the beginning, it is necessary to do all of these three tasks to get research into nuclear fusion off the ground (hence, an AND decomposition is required). However, as time progresses and some breakthroughs have been made, it is expected that it is not necessary anymore to do all of these, but at least one task still needs to be done to further support the research (hence, an OR decomposition is required and this change to the decomposition type enumeration is indicated by EC). Moreover, the importance of the main components of this actor, i.e., the goal “Support Research in Renewable Energy” and the resource “Research Facility” is shown in Figure 4 to

specify the role each of these components play in the overall satisfaction of the actor (60 and 40, respectively). As both the other actors have a sole main goal, the quantitative importance of each main goal is marked as 100.

The following sections first describe which kinds of changes are supported in TimedGRL and then describe the various changes in more detail. The timepoints at which the model is evaluated are the dates 1st January, 2016 (i.e., the date when model was built), 1st January, 2026, and 1st January, 2031. The corresponding evaluations are shown in Figure 5, Figure 6, and Figure 7, respectively.

3.3 Changes Supported in TimedGRL

A *Change* is used to define the expected evolution of an individual element or its attribute over time to enable reasoning over time periods. To define a change, the modeler first needs to define its time period by defining a *start* date and an *end* date. It is possible to add more than one change to an attribute of an element, if the time periods do not overlap. This allows the modeler to add wide varieties of behaviors for an unrestricted time period, making the goal model flexible and adaptive to evolutionary trends. If no change is defined for an element in a time period, then the default value from the standard specification of the goal model is used.

All types of change cannot be added to every goal model element and its attributes. E.g., a numeric change (NC) cannot be added to the decomposition type of an intentional element, while an enumeration change (EC) cannot be added to a contribution link. A deactivation change (DC) may only be applied to an element but not an attribute. Table 1 defines all changes that can

currently be associated with the relevant metamodel elements and their attributes (affected properties).

Table 1 ELEMENTS, AFFECTED PROPERTIES, AND CHANGES

| Element | Affected Properties | NC | EC | DC |
|----------------------------|----------------------------------|-----|-----|-----|
| Intentional Element | Intentional Element | No | No | Yes |
| | Importance | Yes | No | No |
| | Decomposition Type | No | Yes | No |
| | Evaluation | Yes | No | No |
| Actor | Actor | No | No | Yes |
| | Importance | Yes | No | No |
| Contribution Link | Contribution Link | No | No | Yes |
| | Quantitative Contribution | Yes | No | No |
| Dependency Link | Dependency Link | No | No | Yes |
| Decomposition Link | Decomposition Link | No | No | Yes |

3.3.1 Deactivation Changes

A *DeactivationChange* is used to define the lifetime of a goal model element. By default, any defined element exists in the model. Adding this change to a model element means that the element is invisible in the model for the entire defined time period and hence does not participate in the evaluation.

There are four elements marked with DC in the goal model. Since the research group is being established in two years' time, a deactivation change is defined for the actor "Renewable Nuclear Fusion Research Group" from 1st January, 2016, until 1st January, 2018. Note that a deactivation change propagates to contained and connected elements. Therefore, the goal within this actor as well as the dependency links of the goal are also deactivated for the same time period (see Figure 5). Deactivated elements are greyed out in the evaluation.

A deactivation change is also defined for the contribution link from the goal “Increase Efficiency of Renewable Nuclear Energy” to the goal “Maximize Efficiency of Renewable Resources” from 1st January, 2016, until 31st December, 2030, because the city expects to make use of this energy form only after that time.

Similarly, the goal “Limit Usage of Non-Renewable Energy” is deactivated until 1st January, 2028, at which point the efficiency and cost of Renewable Energy are expected to be so advanced and affordable that it would be possible to allow limiting of Non-Renewable Energy. Hence, these model elements are only activated in Figure 7.

The task “Build an Advanced Research Facility” is needed initially to construct the research facility, but is not performed anymore once the facility exists. Therefore, it is deactivated from 2nd January, 2026, on, which is when the building is expected to be completed. This is accurate from the ministry’s point of view, because the task is not performed anymore, but the task feeds the resource “Research Facility”, which needs to exist even after the task’s deactivation. For this purpose, the flag “*sufficientOnceAchieved*” (SOA) is raised for the resource, indicating that its satisfaction level remains at the maximum indefinitely once the maximum has been reached at some point in time and regardless of any changes to incoming contributions after that point in time (see Figure 6 and Figure 7).

3.3.2 Numeric Changes

A *NumericChange* can be defined for any element or its attribute that has a numerical value. This change uses various types of formulae with time as variable to define the behavior. This allows the model to calculate the exact numerical value by substituting the required time.

The time variable here is the duration, in days, between the start date of that change and the *timepoint* at which the formula needs to be calculated.

A *ConstantChange* means that the numerical value remains the same throughout the time period. For this change, the expected constant value (*newValue*) needs to be entered. In the given example, a constant change to 86% is defined for the satisfaction value (*affectedProperty*) of “Increase Efficiency of Hydropower” for the time period from 1st January, 2025 (*start*) to 1st January, 2041 (*end*). Hydropower’s efficiency is 85% at the beginning in 2016, which is already rather high. Thus, any increase in its efficiency is expected to be slow-paced, which is captured by the constant change. Therefore, Hydropower’s efficiency is 85 on 1st January, 2016 (Figure 5), and 86 on 1st January, 2026 (Figure 6) and 1st January, 2031 (Figure 7). Similarly, researchers predict “Minimize Cost of Renewable Resources” is going to reach to a halt, once its satisfaction value (*affectedProperty*) reaches 90 (*newValue*), which is expected to be on 1st January, 2026 (= *start*; *end* = 1st Jan, 2041) (i.e., cost is 80 on 1st January, 2016 (Figure 5), and 90 on 1st January, 2026 (Figure 6) and 1st January, 2031 (Figure 7)).

A *LinearChange* means that the numerical value increases/decreases linearly within the defined time period. For this change, the *newValue* expected at the end date needs to be entered. In the given example, a linear change is defined for the quantitative value (*affectedProperty*) of the contribution link from “Hydropower” to “Maximize Efficiency”. Here, researchers are assuming that the share of Hydropower in overall usage of Renewable Energy Resources is going to decrease from the current 47% on 1st January, 2016 (*start*) to 41% (*newValue*) by 1st January, 2028 (*end*), as the efficiencies of other available Renewable Resources improve and their usage increases to avoid heavy dependence on a single resource. The equivalent linear

equation is “ $y = 47 - 0.001368t$ ”, which translates to the contribution link being 42 on 1st January, 2026 (i.e., t = duration between 1st January, 2016, and 1st January, 2026 in days = 3653) (Figure 6).

Other attributes which are also exhibiting linear change in the actor “City” are the satisfaction values of “Wind Energy”, “Solar Energy”, and “Geothermal Power”, and the contribution values of “Wind Energy”, “Solar Energy”, and “Geothermal Power” to the “Maximize Efficiency” goal.

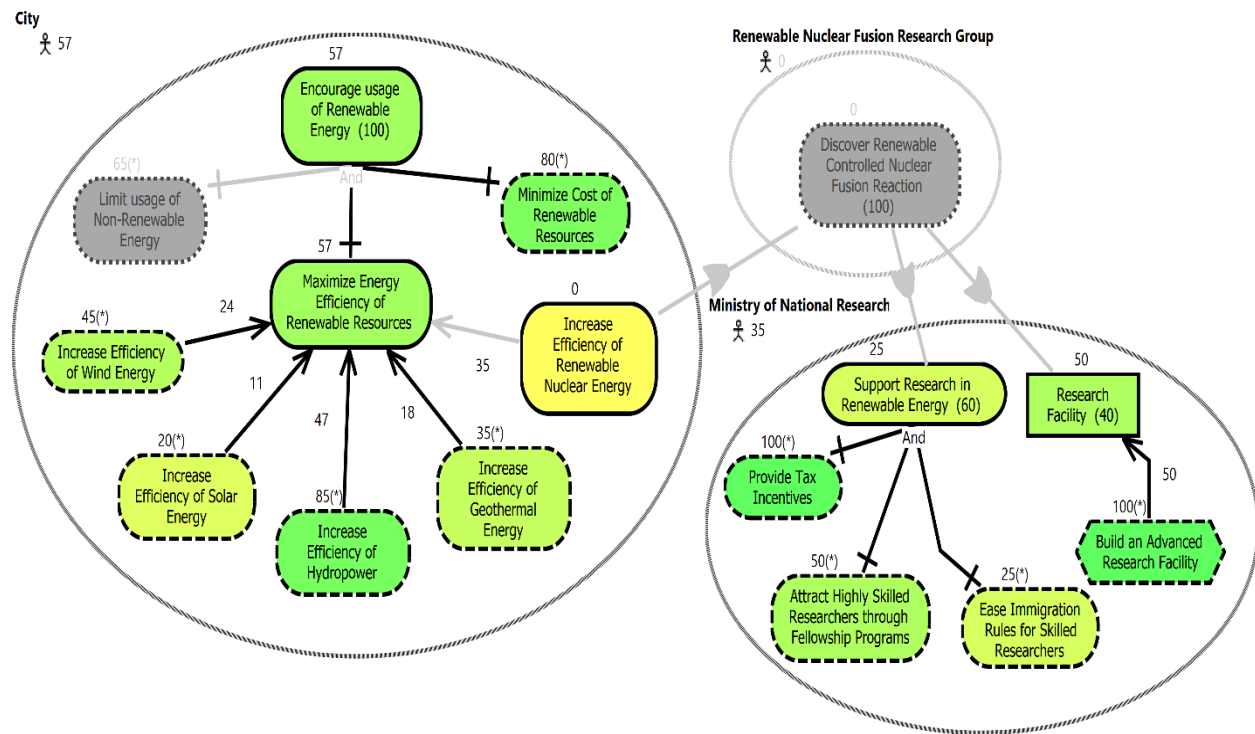
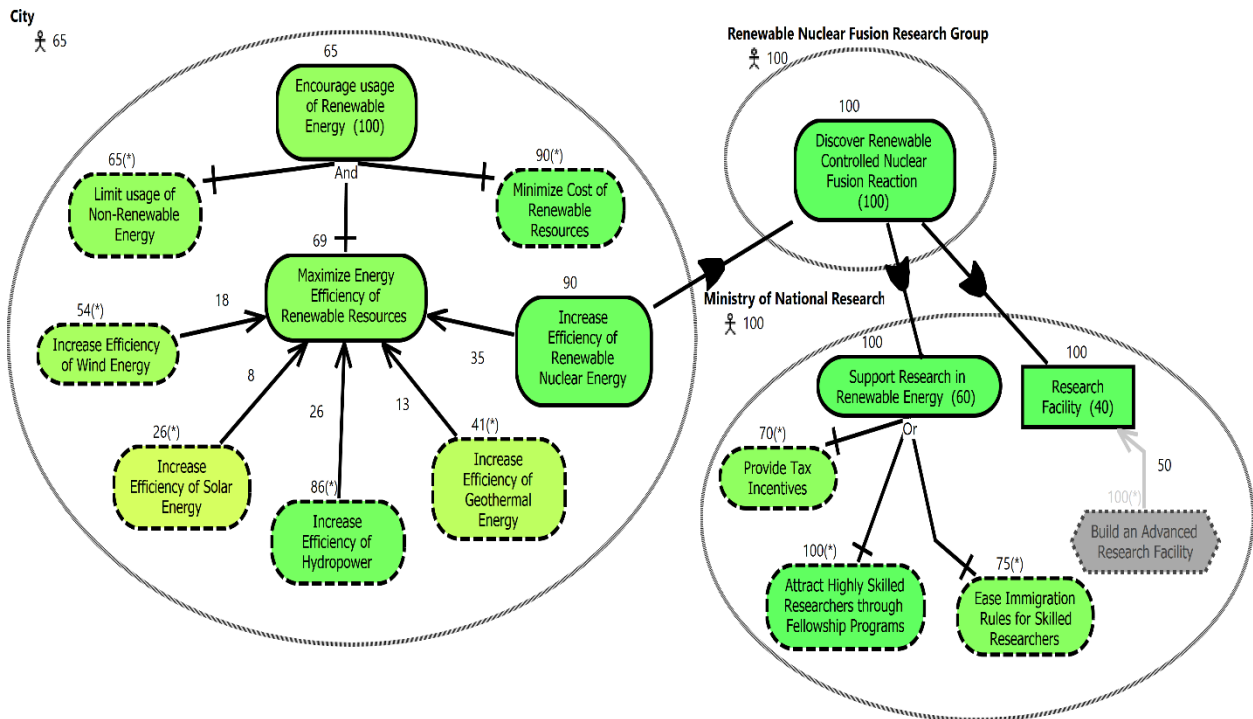
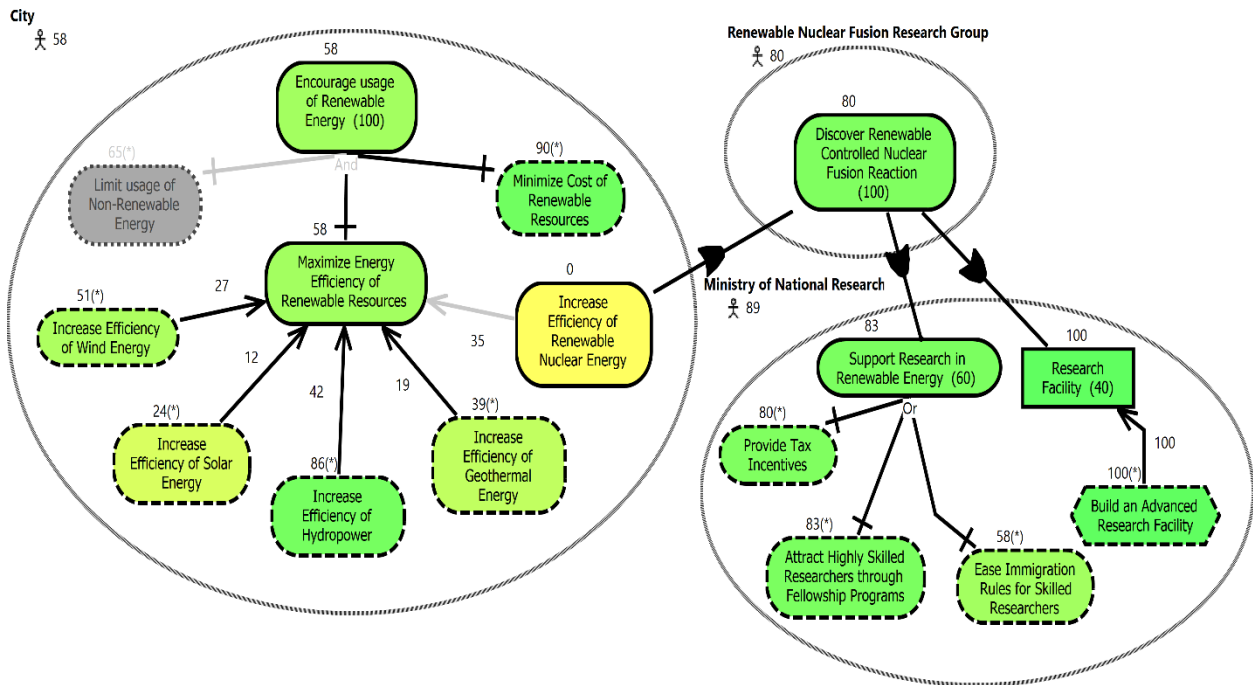


Figure 5: Evaluation on 1st January, 2016



In the actor “Ministry of National Research”, the contribution link from “Build an Advanced Research Facility” task to “Research Facility” resource also has a linear change attached to its quantitative contribution value (*affectedProperty*). This models the fact that it takes some time to build a research facility. During the whole time the facility is built, the task “Build an Advanced Research Facility” is set to 100 as the ministry is continually performing this task. However, the contribution of the task to the resource is initially low (e.g., 50 in Figure 5) and only gradually grows to 100 (Figure 6), at which point the research facility has been completed (i.e., the resource also reaches the maximal satisfaction level of 100 and stays there because of the SOA flag).

A *QuadraticChange* means that the numerical value follows a quadratic behavior. *quadraticCoefficient*, *linearCoefficient*, and *constant* need to be entered to describe this change. In the given example, a quadratic change is defined for the satisfaction value (*affectedProperty*) of the goal “Discover Renewable Controlled Nuclear Fusion”. Here, it is assumed that the research results are limited at first, but occur much more frequent after initial success has been achieved. The change is therefore defined from 1st January, 2016 (*start*) to 1st January, 2026 (*end*). A *quadraticCoefficient* of 0.00001, a *linearCoefficient* of -0.01462 , and a *constant* = 0, result in the formula “ $y = (0.00001 * (t^2)) - (0.01462 * t)$ ”, i.e., the value on 1st January, 2026 (i.e., $t = 3653$) is 80 (Figure 6). This is then combined with a more moderate linear growth from 2026 to reach 100 in 2031. Note that the specific formula in this example is entirely fictitious to illustrate a quadratic change.

A *FormulaChange* means that the numerical value follows a behavior that can be described using a custom mathematical formula in time. For this change, the actual *formula*

needs to be entered. In the given example, a formula change is defined from 1st January, 2016 (*start*) to 1st January, 2031 (*end*) for the satisfaction value (*affectedProperty*) of “Provide Tax Incentives” with the formula “ $y = 100 - (0.005475 * t)$ ”. Thus, the value on 1st January, 2026 ($t = 3653$) is 80 (Figure 6) and on 1st January, 2031 ($t = 5479$) is 70 (Figure 7). Note that each linear change can be expressed as a *FormulaChange*, if the modeler finds it easier to determine the formula instead of figuring out the desired new value at the end of the time period. Other attributes which are also exhibiting formula change are the satisfaction values of the other two sub-tasks of “Support Research in Renewable Energy” in the “Ministry of National Research” actor.

3.3.3 Enumeration Changes

An *EnumerationChange* allows the modeler to define different enumeration values for different time periods for any enumeration attribute. Currently, this change is only needed for the decomposition type of an intentional element. The expected *newValue* of the attribute in the defined time period needs to be entered. In the given example, the decomposition type of the goal “Support Research in Renewable Energy” exhibits an enumeration change. As discussed already, during the initial period, the decomposition type remains “AND” as it is necessary to do all of the three tasks to get the research into nuclear fusion off the ground. As time progresses and some breakthroughs have been made, it is not necessary anymore to do all of these but at least one still needs to be done to keep the research going. Therefore, the decomposition type (*affectedProperty*) is changed to “IOR” from 1st January, 2025 (*start*) to 1st January, 2031 (*end*) (see Figure 6 and Figure 7).

3.4 Model Changes not supported by TimedGRL

To use TimedGRL at its highest capability, every model element that is ever expected to exist in the Goal model should be added in the base model. This is sometimes referred to as the 150% approach. TimedGRL, then, deactivates elements at specified timepoints, according to the applicable changes. There is no provision of addition or, replacement of elements in the metamodel during evaluation. Hence, after analyzing potential change requirements, we discovered a few situations that could not be modeled directly using TimedGRL. This section discusses those situations and provides modeling alternatives for them.

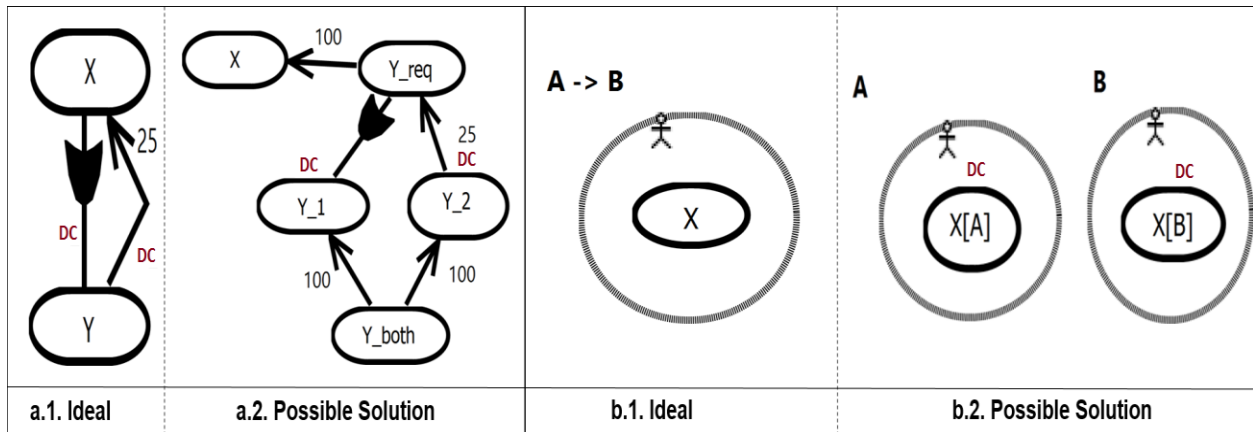


Figure 8: Model Changes not supported by TimedGRL

Currently GRL does not support addition of two different kinds of links between the same pair of intentional elements in a goal model. However, it is probable that the relationship between two elements changes over time. For example, in Figure 8(a), Goal Y contributes to Goal X for some time period, but later on X becomes dependent on Y. Figure 8(a.1) shows the ideal modeling visualization to communicate this situation, where either one of the links can be deactivated depending on the corresponding change and timepoint. However, as discussed, GRL does not allow it. The workaround in this case is to model both the links with different intentional elements and, control them using DeactivationChange, as shown in Figure 8(a.2).

This workaround can also solve situations dealing with link reversal between two elements in a goal model.

Another situation, that cannot be modeled directly, is changing the actor of an intentional element in a goal model. For example, in Figure 8(b), Goal X's parent actor needs to be changed from Actor A to Actor B after a certain time period. Figure 8(b.1) shows the ideal model visualization for this situation, which when evaluated at a timepoint should only show the relevant actor in the model. However, it is not possible to do so in GRL currently and, hence, a workaround is needed. Figure 8(b.2) presents a possible solution, in which two intentional elements X[A] in Actor A and X[B] in Actor B are used, however, they both represent the same concept X. Now, DeactivationChanges can be added to these elements to show only the relevant actor - intentional element pair. It can be argued that the same intentional element could have been used in both the actors, as more than one reference of an intentional element are allowed in the jUCMNav tool. However, deactivation is applied to the definition of an element, and hence, all the references are either deactivated or not. Therefore, two distinct elements such as X[A] and X[B] are needed.

3.5 Discussion

The proposed metamodel and supported changes discussed in Section 3.1 and Section 3.3 allow a comprehensive set of changes over time to be made to a goal model. However, there are still some restrictions because all model elements should be defined in the base model, while making sure of validity of that model. In Section 3.4, a few such situations were discussed along with their possible modeling alternatives. Another restriction is the fact that TimedGRL only

supports formula with time as a variable, but a modeler could theoretically want to express changes in the model based on, e.g., calculated satisfaction values or other model data.

As discussed in Section 3.3, a deactivation change defines the time period for which an element is inactive or absent in the model. The reasons behind TimedGRL only supporting deactivation changes instead of activation or combined activation and deactivation changes are:

- (i) the active periods of an element can be automatically deduced from deactivation changes and,
- (ii) an element in the model is going to be for sure active in some time period, while it is not necessarily inactive in any time period. Therefore, by default, it is assumed that an element is always present in the model, unless a deactivation change is defined.

One could argue that activation or deactivation changes are not needed at all, because the satisfaction value of an element could simply be set to zero if the element is not active. However, there is a fundamental difference between (a) an element being inactive and (b) having a satisfaction value of zero. In the latter case, the element exists in the model and participates in the evaluation. This is important, because TimedGRL may use relative contribution values (e.g., the percentage figures which describe the share each resource has relative to all other energy forms in the example). Relative contribution values have to be normalized to ensure that the best/worst possible solution results in a satisfaction value of 100/0 for the target of the contribution links [5][6]. The contribution link of an element with a satisfaction value of zero is still taken into account during the normalization process (e.g., the target cannot achieve 100

because it does not receive the positive contribution from the element with a satisfaction value of zero). However, if the positively contributing element is inactive, then the contribution link also does not exist in the model and it is possible for the target to reach 100, because the link does not need to be taken into account during the normalization process.

3.6 Summary

This chapter introduces the fundamental concepts of TimedGRL. The TimedGRL metamodel, which is an extension of the GRL metamodel, is described in detail explaining all new elements. Moreover, a hypothetical but realistic example from the sustainability domain is presented to illustrate the newly introduced concepts. Each of the possible change types is explained with the help of the mentioned example. TimedGRL tries to cover as many of the different modeling situations as possible using the introduced changes (DeactivationChanges, NumericChanges, and EnumerationChanges). However, there are a few situations which cannot be modeled directly using TimedGRL. This chapter discusses those situations and provides modeling alternatives for them. Finally, a brief discussion is provided to analyze the sufficiency and restrictions of TimedGRL.

The next chapter defines an algorithm for the evaluation of TimedGRL, which extends existing goal model analysis techniques, and uses the same example as this chapter for demonstration. Moreover, it introduces various visualization techniques used to illustrate TimedGRL evaluations.

CHAPTER 4: TimedGRL Evaluation

This chapter first discusses how the existing evaluation algorithm has been modified to support TimedGRL, and then explains how included dynamic contexts are handled during evaluation. Finally, the overall evaluation of the system in a time period as defined by a *TimepointGroup* is explained.

4.1 Algorithm Overview

GRL currently supports three types of analyses: quantitative evaluation, qualitative evaluation, and hybrid evaluation [2]. The algorithm for TimedGRL builds on the quantitative evaluation algorithm, however the fundamental principles of the proposed approach may also be adapted for other evaluation algorithms. The existing evaluation algorithm works on a model, whose elements have been assigned with contribution values and initial satisfaction values, which are then propagated to higher-level goal model elements. However in TimedGRL, changes such as *NumericChange* may have been defined for these initial values. Moreover, some elements themselves may change behavior with time, which can be expressed by adding changes such as *DeactivationChange* and *EnumerationChange*. This means that at each timepoint, the model is going to be different according to the changes defined for that particular day and the existing algorithm needs that updated model for analysis. Hence before, handing over the goal model for evaluation, TimedGRL has to preprocess it to provide a valid input to the existing algorithm, i.e., an input that corresponds to a regular GRL model with all changes resolved for a particular timepoint. The inputs provided to the evaluation of TimedGRL itself are a *DynamicContext* (including relevant *Changes*, an *EvaluationStrategy*, and, if required, a *ContributionContext*), and a *Timepoint*. As discussed in Chapter 2, an *EvaluationStrategy*

describes a possible solution by assigning initial qualitative or quantitative satisfaction values to a set of intentional elements in the GRL model, typically leaf nodes (i.e., tasks), and a Contribution Context is a set of contribution values to override contribution links.

Before starting with the preprocessing stage, we first have to choose how are we going to store the original/default values in the model, which are going to be overridden by the preprocessing stage. If the model is directly overridden, then it will not be possible to restore the default values in the model once the evaluation at that particular timepoint has been performed. There are three alternatives to choose from:

1. Instead of updating the original model, create a copy of that model and perform preprocessing on that copy. This would make sure that we always have the original values safe with us. For every evaluation at a timepoint, a copy of the model would be created and after evaluation, it would be destroyed. However, the main issue with this option is that before every preprocessing, a deep copy of the model would have to be performed and depending upon the number of elements in that model, it could be time consuming.
2. Create a new file for each evaluation at a particular timepoint, and perform preprocessing on the model in that file. This would also keep the original values in the model intact. However, this option could be both space consuming and time consuming, depending upon the size of the model.
3. Use the original model for preprocessing stage, however store the default values in corresponding element's metadata before overriding it. Once the analysis has been

performed, the original values could be restored from each element's metadata. We select this option as it is less time and space consuming than the first two. Also, this option is easier to implement.

With the choice to store original values made, the next step is to define the actions to be performed during the preprocessing stage:

- EvaluationStrategy and Contribution Context are extracted from the selected DynamicContext to obtain the initial satisfaction values and contribution values for each element.
- Each change of the selected DynamicContext, for which the selected timepoint lies between its start date and end date, is chosen for further processing and its corresponding element is recorded (see Listing 2 and collectChanges() in Listing 1).
- Changes for each group of elements are collected and the changes are applied in the following precedence order of element groups: Actors > Intentional Elements > Links (see the three for loops in Listing 1). Before applying the changes, the original values are stored in the corresponding element's metadata.
- According to the OCL constraints discussed in Section 3.1, for each element in an element group, there can be at the most one change per affectedProperty, for the given timepoint (except the start date of a change can be same as the end date of another change, however in such cases the change with the same start date is given

preference and the other one is ignored). The respective change is applied to each element in a group (see Listing 3).

If an element has a `DeactivationChange` defined for the selected timepoint, then that element is deactivated (see `deactivateElement()` in Listing 3). Moreover, the children elements of the deactivated elements are deactivated (`findAndDeactivateChildrenElements()` in Listing 3). All the elements that are either contained in an element, or are connected to that element are known as its children elements. Hence, an actor can have intentional elements and links, all of which are contained within the actor, as children, and an intentional element can have links connected to it as its children. For example in Figure 5, the actor “Renewable Nuclear Fusion Research Group” is deactivated (indicated by greying out) for the selected timepoint, which has led to deactivation of its children elements - the goal “Discover Renewable Controlled Nuclear Fusion”, and the three dependency links. The precedence order of the element groups assures that elements with a deactivated parent are ignored by the `TimedGRL` evaluation algorithm and all the inactive elements do not participate in the evaluation process, which is visualized by greying them out.

Except `LinearChanges`, all other `NumericChanges` and `EnumerationChanges` can be treated the same way. The reason behind that is for linear change, the modeler only provides the required new value at the end date, and not the linear equation itself. Rather, it is the job of the algorithm to figure out the linear formula first, before calculating the value at the selected timepoint. Determining the linear formula depends on the value at the start date of the change, which is not defined by the linear change itself, but rather the model (i.e., all the changes applied

to the element up until this point). Therefore, if any of the active elements has a LinearChange defined for the given timepoint, then the required value for the start is determined recursively.

In other words, the same preprocessing algorithm is called with a different timepoint by `calculateInitialValue()` in Listing 3. This different timepoint is actually the start date of the change in concern. This method determines if a previous consecutive change (if the end date of a change is equal to the start date of the change in concern) exists. If yes, then the start value is calculated according to that change. Otherwise, the default/original value of that property is used as the start value. Once the start value is known, the linear formula is calculated internally (`calculateLinearFormula()`), the exact value at the needed timepoint is determined, and the model is updated with this value (`updateValue(formula, timepoint)`). This method also internally handles the storage of default value in the metadata of the corresponding element.

Listing 1: TimedGRL Preprocessing Algorithm

```
Algorithm TimedGRLPreprocessing
Inputs currentDynContext:DynamicContext,
        timepoint:Timepoint
Outputs updatedGRLmodel:GRLspec

updatedGRLmodel = currentDynContext.grlspec
changes:List<Change> = ∅
affected:List<String> = ∅
changes = collectChanges(currentDynContext, timepoint, affected)
actorChanges:List<Change> = ∅
intEltsChanges:List<Change> = ∅
linkChanges:List<Change> = ∅
for each change:Change in changes {
    if (change.element isA Actor)
        actorChanges.add(change)
    else if (change.element isA IntentionalElement)
        intEltsChanges.add(change)
```

```

        else
            linkChanges.add(change)
    }
    //First apply changes to Actors
    for each change:Change in actorChanges {
        applyChange(updatedGRLmodel, change, timepoint)
    }
    //Apply changes to Intentional Elements
    for each change:Change in intEltsChanges {
        applyChange(updatedGRLmodel, change, timepoint)
    }
    //Apply changes to Links
    for each change:Change in linkChanges {
        applyChange(updatedGRLmodel, change, timepoint)
    }
    return updatedGRLmodel

```

Listing 2: Collect Changes Algorithm

```

Algorithm collectChanges
Inputs currentDynContext:DynamicContext,
        timepoint:Timepoint,
        affected:List<String>
Outputs changes:List<Change>
changes:List<Change> = ∅
for each change:Change in currentDynContext.changes {
    if (change.start.isBeforeOrEqual(timepoint) &&
        timepoint.isBefore(change.end)) {
        affectedProp:String = change.element.toString()
        if (change isA DeactivationChange) {
            affectedProp += '.deactivate'
        } else if (change isA PropertyChange) {
            affectedProp += change.affectedProperty
        }
        if (affectedProp not in affected) {
            changes.add(change)
            affected.add(affectedProp)
        }
    }
}

```

```

}
// handle included contexts
for each context:DynamicContext in currentDynContext.includedContexts {
    changes.addAll(collectChanges(context, timepoint, affected))
}
return changes

```

Listing 3: Apply Change Algorithm

```

Algorithm GRLspec.applyChange
Inputs updatedGRLmodel:GRLspec,
        change:Change,
        timepoint:Timepoint
if (change isA DeactivationChange) {
    findAndDeactivateChildrenElements(change.element)
    deactivateElement(change.element)
} else if (change isA PropertyChange) {
    if (change isA LinearChange) {
        // determine start value recursively
        start = calculateInitialValue(change)
        formula = calculateLinearFormula(start, change)
        updateValue(formula, timepoint)
    } else {
        updateValue(timepoint, change)
    }
}

```

For changes that are not LinearChange, the information needed to determine the values at the selected timepoint is specified by those changes themselves. For QuadraticChange, at first the formula is figured out using the attributes of that change. Then, the value of timepoint is substituted in that formula to obtain the exact value at the needed timepoint. For FormulaChange, modeler has directly provided the formula. Hence, in this case, just the value of timepoint is substituted to get the correct value. For ConstantChange and EnumerationChange, the newValue is used directly. All these values are updated in the model (see updateValue(timepoint, change) in Listing 3), after storage of default values in the metadata of the corresponding element.

In all the above cases, the timepoint is actually not substituted directly as we know that its unit is date and we need to convert it in some proper format to be able to use that in the formula. There are two options to choose from:

1. Use the number of days from 1st January, 1970, to convert it into proper format, as this is a standard for many Date libraries to calculate amount of time. However, calculating a formula keeping this particular unit of time in mind would be complex as the number of days could be really high if a recent date is considered while, it would be hard to use dates before 1st January, 1970.
2. Use number of days relative to the start date of the change in concern, i.e., start date is converted to “0” day while, any other date until the end date of that change will be converted to the difference between that date and the start date. For example, start date: 1st January, 2017; end date: 31st March, 2017; timepoint: 2nd March, 2017 $\rightarrow t = 60$ days (number of days since 1st January, 2017).

We choose the second option to use the relative number of days as it would be easier for the modeler to calculate a formula keeping this option in mind. Moreover, a formula should remain same for different intervals of time, and using relative values enable that.

At this point, the model has all the updated values and it acts like a regular GRL goal model that could have been manually created for the given timepoint. This model along with the specified `EvaluationStrategy` and `ContributionContext` (if present) is fed to the existing quantitative GRL evaluation algorithm to get the evaluation of the model at that particular time.

Once the evaluation has been performed, the original values are restored in the goal model by accessing those values from the metadata of each element.

Thus, using a single timed goal model, the evaluation of the model at an individual timepoint or a group of timepoints may be determined to explore the model's behavior over time.

4.2 Included Dynamic Contexts

A `DynamicContext`, along with `Changes`, an `EvaluationStrategy`, and a `ContributionContext`, may also include other dynamic contexts. A parent context overrides an included context in all intervals except when there is no change defined for the parent and the included context has a change attached to it for the same time period. Generally, in case a `DynamicContext` has no change defined for an element (or, its attributes) for the timepoint in concern and there is no dynamic context included, then it takes the default value (defined as part of the regular goal model). However, if there is another `DynamicContext` included, which has a change attached for that element in the time period of interest, then the included context overrides the default value of the parent. However, if the included context also has no change defined, then the default value is used again.

For example, Figure 9 (top) shows the changes of two dynamic contexts, which represent two expert opinions on how the efficiency of solar energy is going to change over the next 25 years. The first opinion (solid line) predicts a linear increase (LC) over the next 15 years, while the second opinion (dashed line) forecasts a linear increase (LC) from 2016 to 2036, which is followed by a more erratic 5 years expressed with a custom formula (FC). With the help of included contexts, a modeler may now choose how to combine these two expert opinions. Figure

9 (bottom) shows the case where the first opinion is given priority over the second opinion by including the dynamic context of the second in the one of the first, which leads to the shown combined behavior. Thus, the change in the parentContext takes precedence from time 2016 to 2031 and is only overridden afterwards by the change in the includedContext. However, the modeler could have also given priority to the second opinion by making the corresponding dynamic context the parent instead of the included context.

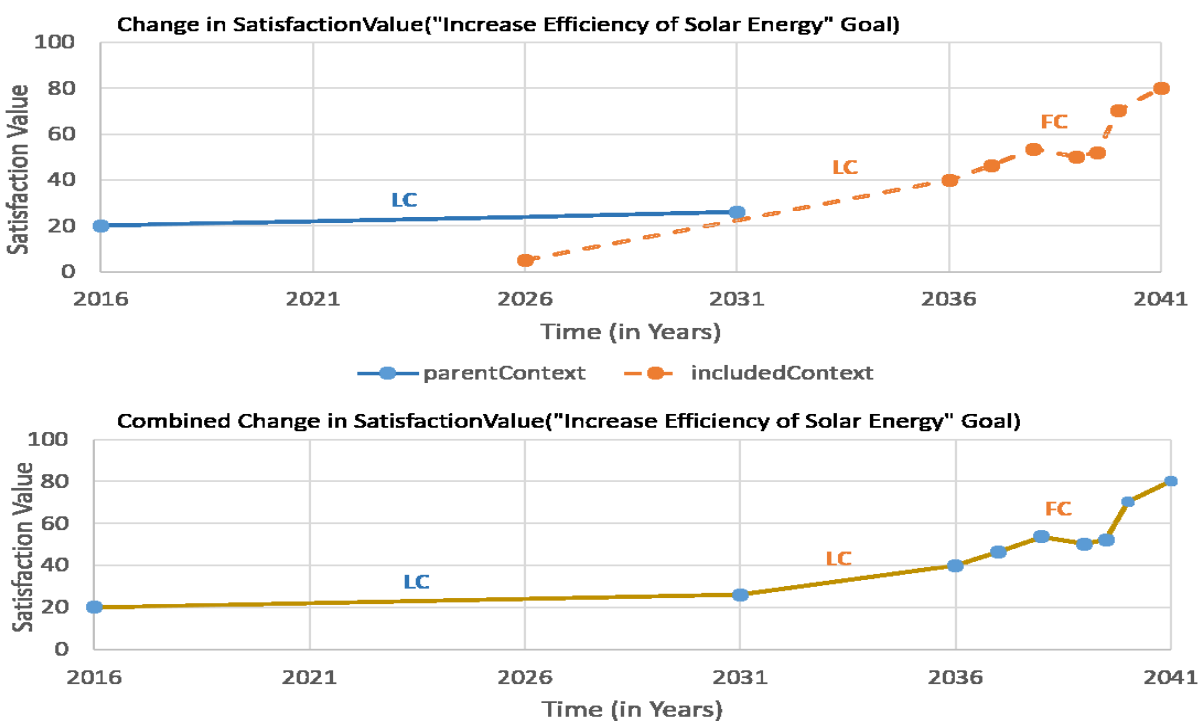


Figure 9: Combining Numeric Changes in Included Dynamic Contexts

It is also possible to include more than one context in a parent context. In that case, the order, in which the contexts are ranked, matters. Conceptually, the bottom-ranked included context is combined first with the included context ranked just above it, i.e., the higher-ranked included context always acts as the parent of the lower-ranked one. This combination process goes on in bottom-up direction, until only the main parent context and one included context

(constructed by combination of all included contexts) are left. At this point, the included context is combined with the parent as discussed earlier.

TimedGRL optimizes this combination process with a top-down approach that only considers the next-ranked included context for an element (or affectedProperty) with default values. Therefore, an additional step is added during the change collection phase (Listing 2) of the evaluation algorithm discussed in the Section 4.1. After selecting all the satisfying changes for the defined timepoint in a dynamic context, elements with the default value for this time period are examined further (see recursive call in collectChanges() in Listing 2). If an element with the default value exists, the changes to the element defined in the included context(s) are used. This continues until there are no more elements with default values or no further included contexts exist.

4.3 Evaluation Trends over a Time Interval

We discussed the TimedGRL algorithm in Section 4.1 and how the existing quantitative evaluation has been tweaked to support it. The inputs needed for the algorithm to work are a DynamicContext (consisting of changes, an EvaluationStrategy, maybe a ContributionContext, and maybe included DynamicContexts) and a Timepoint, at which the model needs to be evaluated. The resultant evaluation corresponds to a single timepoint and is visualized as shown in Figure 5, Figure 6, and Figure 7. Now, in this section, we will discuss how to use the same algorithm for evaluation over a time interval defined using a TimepointGroup as well as visualization of such an evaluation.

Similar to the evaluation at a single timepoint, evaluation over a group requires a `DynamicContext` and a `TimepointGroup` as inputs. A `TimepointGroup` consists of a number of timepoints. Hence, to determine a time interval from a group, the timepoints with the minimum and the maximum dates in that group are considered as the two end points of the interval. For example, selecting a timepoint group having timepoints *1st January, 2017*; *15th March, 2017*; *31st May, 2017*, along with a `DynamicContext` will give an evaluation of the system from 1st January, 2017, to 31st May, 2017.

Once the interval is chosen, the next step is to define an evaluation frequency, i.e., the rate at which we want to perform the analysis in that particular time interval. The default value of evaluation frequency is 1 day, however it can be increased to any desired number. However, that number should be less than the total interval length.

Next, the goal model is analyzed using the `TimedGRL` algorithm, at multiple timepoints starting from the leftmost date of the interval, increasing by the defined frequency, until the rightmost date of the interval is reached. Thus, at the end, the overall evaluation over a timepoint group returns sets of satisfaction values for all the intentional elements and the actors corresponding to each date at which the model was analyzed.

The satisfaction values for intentional elements and actors are already defined in GRL as discussed in Chapter 2. However, the concept of a comprehensive system satisfaction value does not exist in the current GRL description. In `TimedGRL`, this concept is introduced in order to communicate the status of the system itself, once satisfaction values of all its active children actors are known. Along with the satisfaction values, the importance values of the active actors

also participate in the calculation of comprehensive system satisfaction value and any actor with zero importance is ignored. Similar to actor quantitative evaluation, TimedGRL computes system's satisfaction value (also known as system quantitative evaluation) using the following formula:

$$\text{System Evaluation} = \frac{\sum_{actors}(\text{actor evaluation} * \text{actor importance})}{\sum_{actors}(\text{actor importance})}$$

Figure 10 is an updated version of Figure 6, where importance has been assigned to each actor in the system (as shown in the bracket after each actor's name) before evaluating at the timepoint with date 1st January, 2026. The city is the most important actor of this system with an importance value of 100, followed by the Ministry of National Research with an importance value of 70 and Renewable Nuclear Fusion Research Group with an importance value of 60. The main aim of the system to encourage usage of renewable energy in the city makes this actor the most prominent in this system. For this example, the quantitative value of system evaluation is:

$$((100*58) + (60*80) + (70*89)) / (100 + 60 + 70) = 73$$

While evaluating at a single timepoint, the system evaluation is not visualized in the model.

The system evaluation is computed for all the dates at which the model is analyzed in the selected timepoint group. The next step is to visualize this information in a way that can communicate the variation in the system over time. For this purpose, we choose a combination of heatmap along with individual charts corresponding to the satisfaction values.

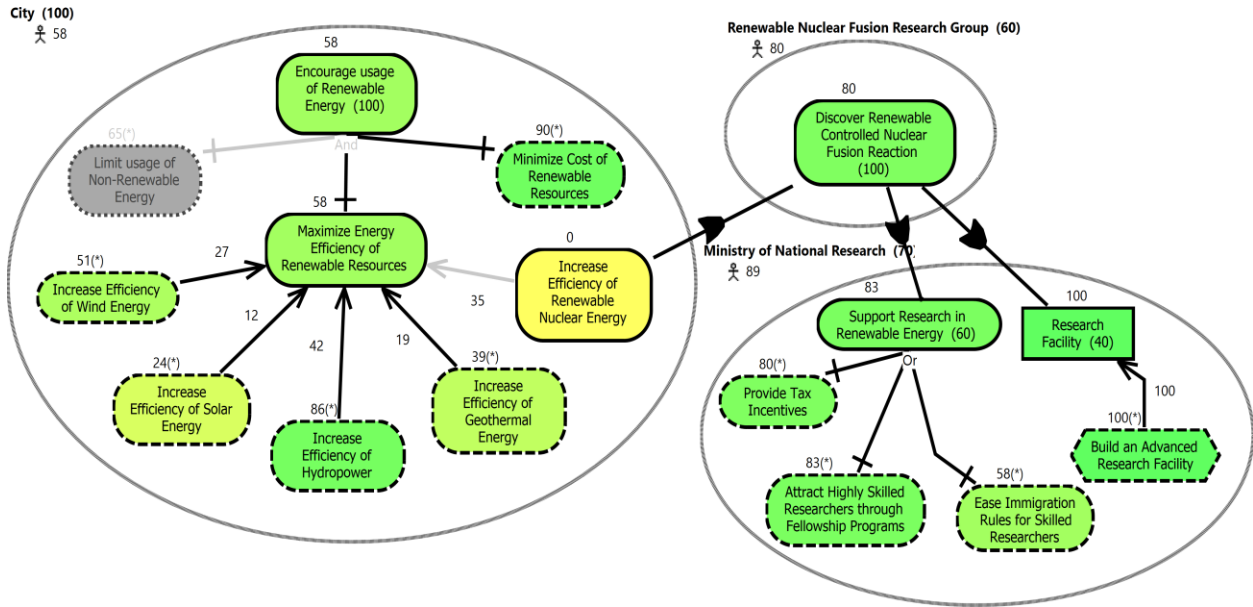


Figure 10: Evaluation on 1st January, 2026 (with Actor Importance values defined)

The heatmap uses the same color-coded scheme as the satisfaction value, i.e., Red (-100) to Yellow (0) and Green (100). This scheme has already been depicted in the figures of goal model evaluations at a single timepoint (Figure 5, Figure 6, Figure 7, and Figure 10). Using the same color-coded scheme ensures consistency making it easier for the user to understand and compare. The only difference in the scheme is addition of grey color to show deactivated elements.

Figure 11 shows a heatmap for the goal model of energy efficiency goals of the city as depicted in Figure 4 over a period of 1 month from 15th December, 2025, to 15th January, 2026. The timepoint group, that is selected as an input for this overall evaluation, consists of the following timepoints: 15th December, 2025; 1st January, 2026; and 15th January, 2026. The dates shown at the top are chosen as per the timepoints defined in that particular timepoint group, as showing all the dates in the interval will make the heatmap too crowded. The evaluation frequency used is the default, i.e., 1 day. We are running the evaluation for only one month just

for explanatory and visualization purpose. This evaluation can be run over any amount of time as needed. The same set of changes as defined in Section 3.2 and Section 3.3 is being used here as well.

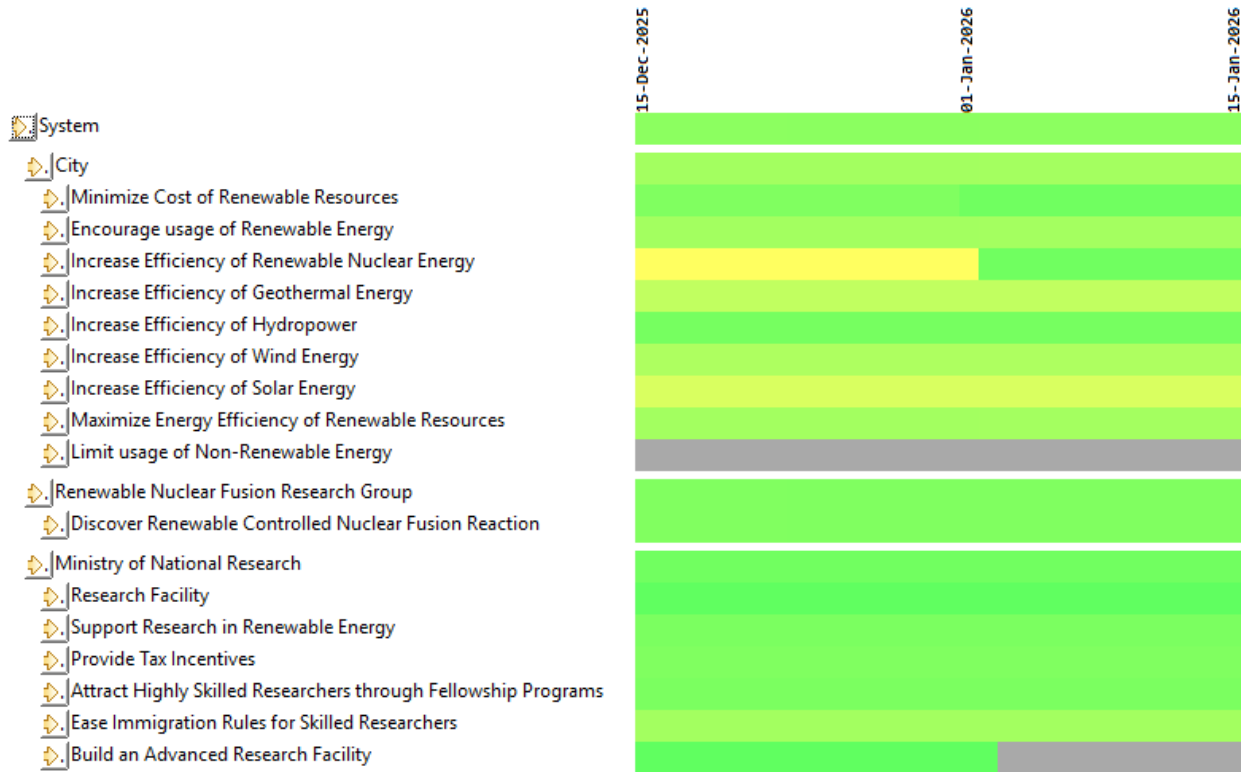


Figure 11: Heatmap depicting Overall Evaluation from 15th December, 2025, to 15th January, 2026

In the heatmap (see Figure 11), all the intentional elements and actors are listed as per their hierarchies (parent-child relationship). All the intentional elements, that do not have a parent actor, are considered to be contained in an undefined actor. The topmost level is the system, which acts as the global parent for all the actors. The color-coded bar next to each element shows the variation in that element's satisfaction value. As discussed in Section 3.3, there is a DeactivationChange defined for the goal "Limit Usage of Non-Renewable Energy" until 2028, and hence the evaluation bar corresponding to that goal is greyed out for the whole interval. On the other hand, the task "Build an Advanced Research Facility" is deactivated from

2nd January, 2026, after completion of the Research Facility's construction and the same is depicted in the heatmap as well. The goal "Increase Efficiency of Renewable Nuclear Energy" has a satisfaction value of zero (as displayed in yellow in the heatmap) before 1st January, 2026, as this goal is waiting for the complete construction of the Research facility. Hence, after that day the goal is satisfied positively as depicted in green in the heatmap. In a similar manner, variations in satisfaction values of all the elements are depicted in this heatmap along with the variations in the system evaluation. This visualization clearly depicts how the status of the system is changing over time, hence giving a comprehensive view.

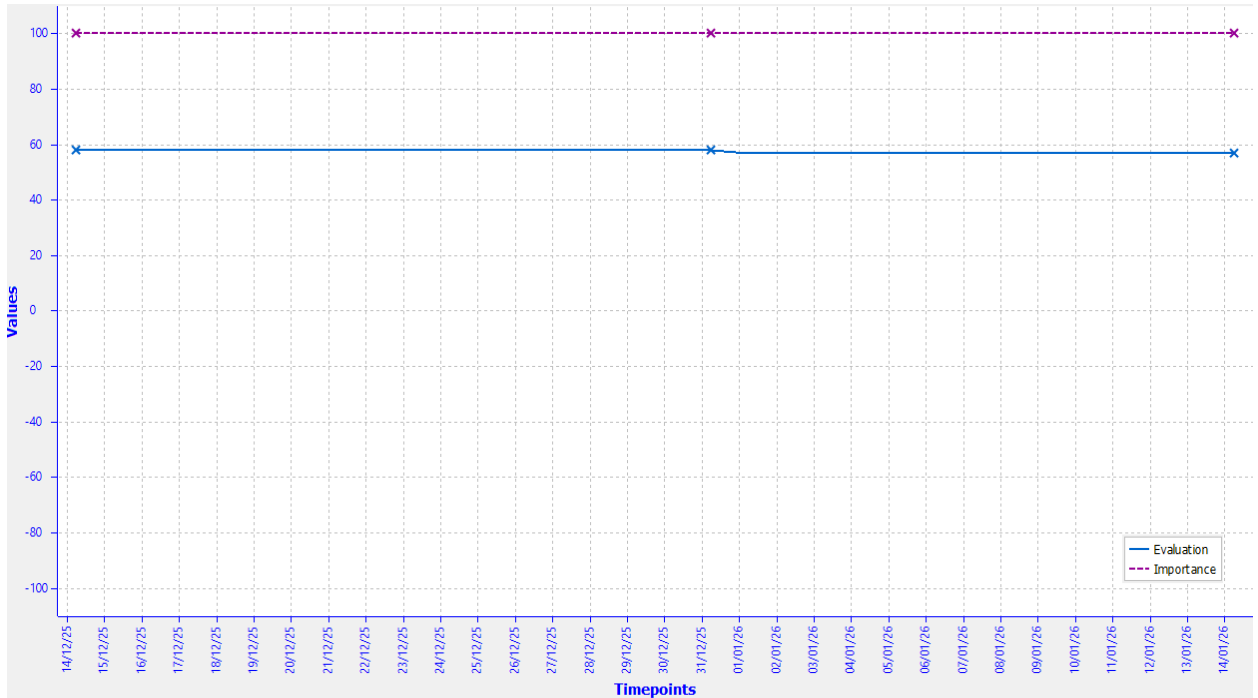


Figure 12: Variation in Evaluation and Importance of the goal "Encourage Usage of Renewable Energy"

However, the heatmap only displays the variation using color and hence, does not express the overall change in depth. Due to this reason, TimedGRL also provides the option to visualize the variation of satisfaction values using line charts, which have the capacity to convey more information. TimedGRL supports visualization of variation in satisfaction value of each

individual intentional element, each actor, and the overall system. Using charts, it is also possible to visualize the importance of an element or demonstrate the participation of different kinds of links like contribution, dependency, and decomposition links in the computation of an intentional element's satisfaction value.

Figure 12 depicts the chart for the variation in satisfaction value and importance of the goal “Encourage Usage of Renewable Energy” over the same timepoint group as the heatmap in Figure 11, i.e., from 15th December, 2025, to 15th January, 2026. The highlighted dates (using crosses) on the lines are the dates corresponding to each timepoint of the selected timepoint group. The importance of the goal for its parent actor does not change, however the satisfaction value decreases a bit after 1st January, 2026. To investigate this further, we look at the chart shown in Figure 13, which shows the effects of individual decomposition links on this satisfaction value. The actual variation in evaluation of the goal from Figure 12 is also represented on this chart using a pink dashed line. As described in Figure 4, there are three decomposition links. However, one of them is deactivated for the concerned time period as the goal “Limit Usage of Non-Renewable Energy” is inactive as shown in the heatmap. That decomposition link's participation is indicated by the three grey crosses on the zero line of the x-axis. The decomposition link corresponding to the goal “Minimize Cost of Renewable Resources”, as indicated using a black solid line, offers a satisfaction value of 80 at first, which increases to 90 on 1st January, 2026. The other decomposition link corresponding to the goal “Maximize Energy Efficiency of Renewable Resources”, as indicated by a blue solid line hidden behind the dashed pink line, offers a satisfaction value of approximately 58 at first, which decreases a bit after 1st January, 2026. As the corresponding decomposition type is “And”, it

means that the actual evaluation will always choose the minimum value at all the dates, which in this case is the satisfaction value offered by the decomposition link represented in blue. If the decomposition type were “Or”, the pink dashed line would have followed the solid black line instead.

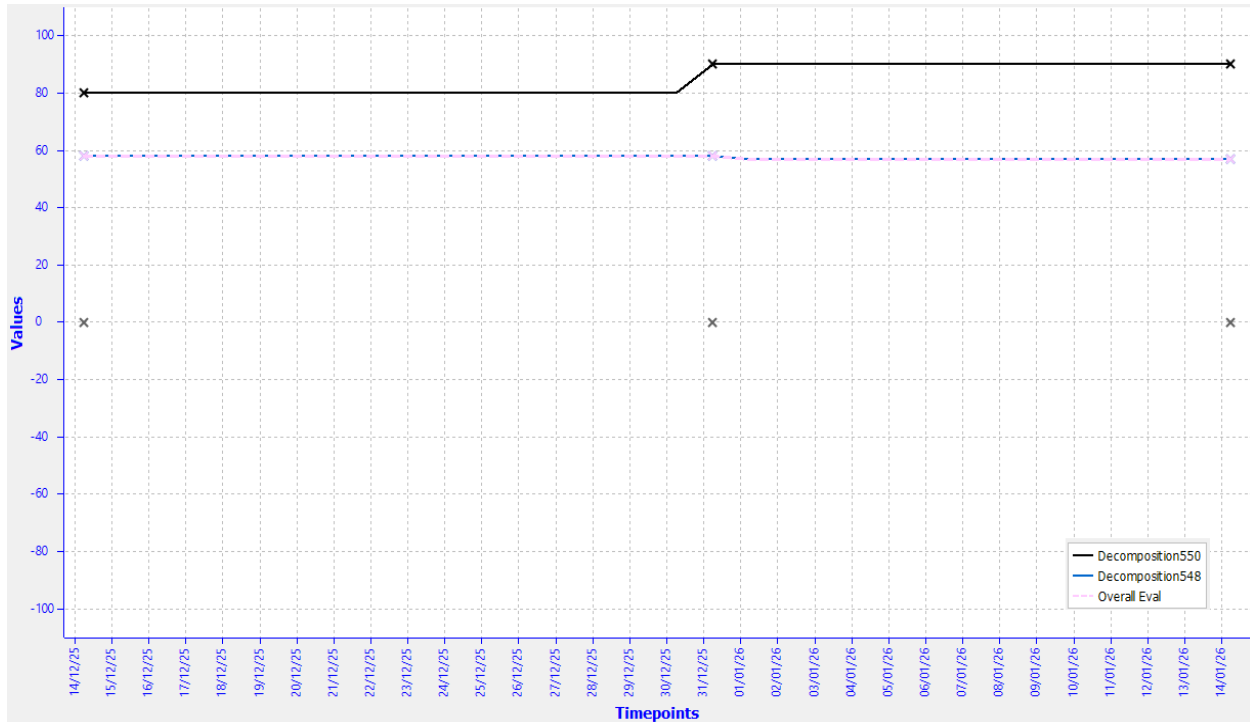


Figure 13: Variation in Evaluation by Decomposition Links of “Encourage Usage of Renewable Energy”

As the previous example was depicted on a small interval of one month, the variation was not that noticeable. To visualize a larger variation, for our next example, we select a timepoint group which contains the following timepoints: 1st January, 2016; 1st January, 2020; 1st January, 2026, hence making the time interval of this overall evaluation as 10 years. The evaluation frequency is still 1 day, which results in a large heatmap. However, the charts are still of the same size and hence, show a clearer and more detailed picture. Figure 14 shows the chart depicting the evaluation and importance of the Resource “Research Facility”. The importance of this resource remains unchanged throughout the period. However, its evaluation shows a linear

increase from 50 to 100, until it becomes saturated (here satisfaction value of 100 indicates completion of construction).

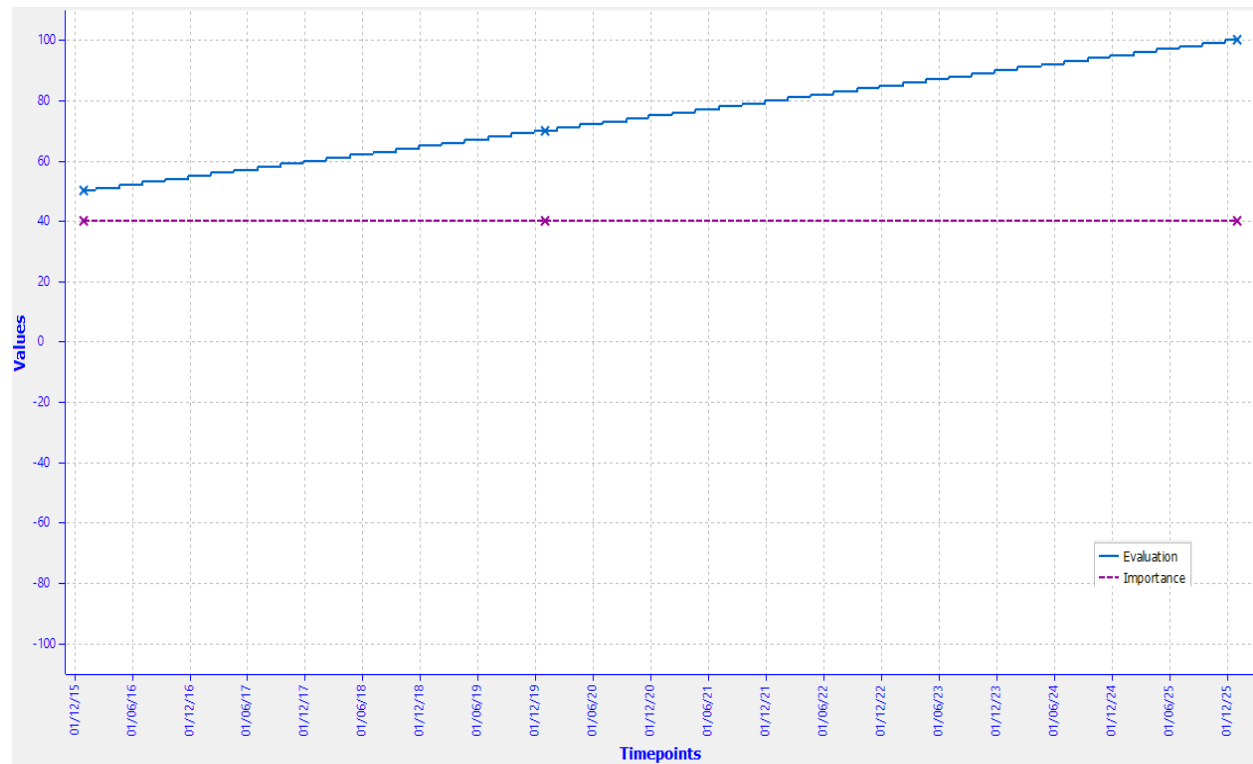


Figure 14: Variation in Evaluation and Importance of the resource "Research facility"

Figure 15 shows the variation in importance and evaluation of the actor “Renewable Nuclear Fusion Research Group” in the same time interval as the previous example (1st January, 2016, to 1st January, 2026), which is deactivated at first. After its activation in 2018, its evaluation shows a variation that is quadratic in nature because of its single child goal “Discover Renewable Controlled Nuclear Fusion”, which has a quadratic change defined for the concerned time period as discussed in Section 3.3.

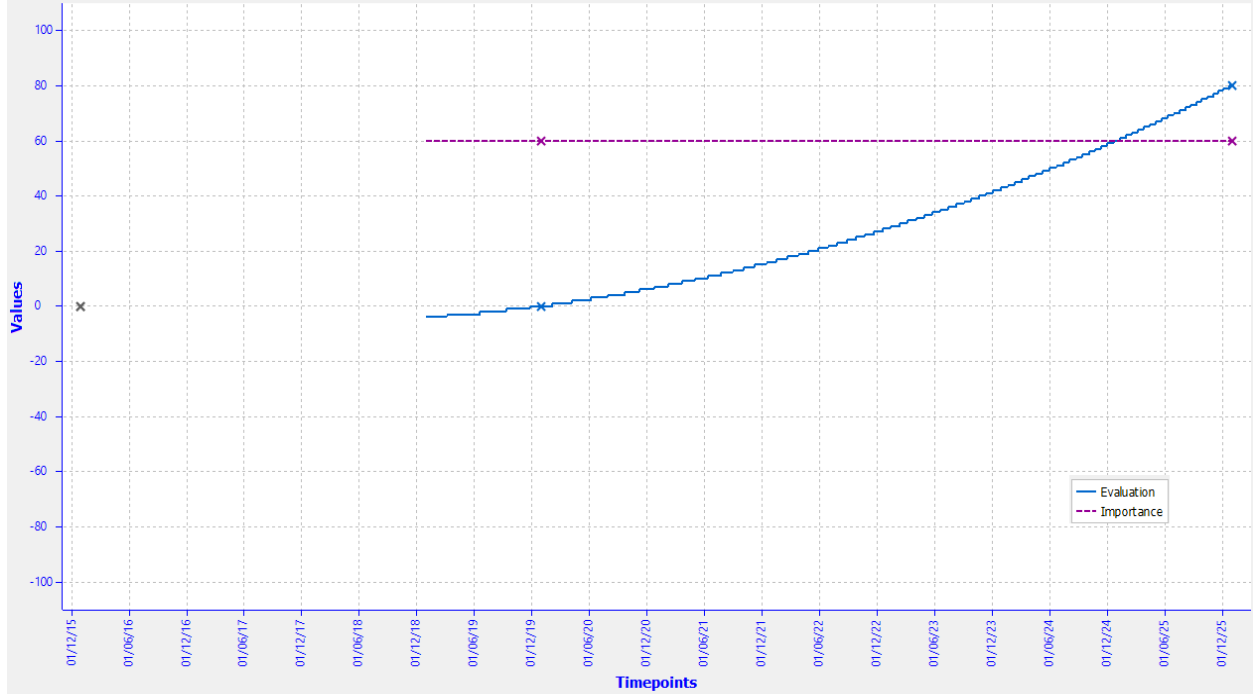


Figure 15: Variation in Evaluation and Importance of the actor "Renewable Nuclear Fusion Research Group"

4.4 Summary

This chapter introduces a quantitative algorithm for TimedGRL, which adds a preprocessing phase to the existing quantitative goal model analysis technique to enable this technique to work on timed goal models. Moreover, this chapter discusses how TimedGRL facilitates the visualization of evaluations at a single timepoint as well as over a timepoint group, giving us a powerful tool to visualize the changes over time using a single timed goal model. The example from the sustainability domain presented in Section 3.2 is used to demonstrate this evaluation process. Furthermore, the option to include dynamic contexts is explained, which provides a clean way to compare different solutions. A prototype implementation of all discussed concepts of TimedGRL has been integrated into the jUCMNav tool [12] and is discussed in the next chapter using a real-life case study.

CHAPTER 5: Validation using Case Study

This case study has been performed to assess the sufficiency of TimedGRL concepts in real world scenarios as well as to showcase the implementation and visualization of these concepts in the tool, jUCMNav. This chapter first describes the chosen case study and converts it into a goal model. It, then, provides a step by step explanation of how to go about adding changes to the model, and how to perform analysis on the model over time using the implemented tool. Finally, a discussion based on the results is presented.

5.1 The Case Study: Audit Report on Ontario Power Generation's HR

Every year the office of the Auditor General of Ontario [20], which is an independent office of the Legislative Assembly, conducts value-for-money and financial audits of the provincial government, its ministries and agencies. The office also audits organizations in the broader public sector that receive provincial funding, such as hospitals and long-term care homes, universities and colleges, and school boards. The main aim behind these audits is to help taxpayers receive value for their tax dollars. In order to achieve that, the Auditor carries out a thorough inspection of government spending and then produces Annual and Special reports that provide Members of Provincial Parliament with the information they need to assess the appropriate usage of public resources.

One of the fundamental tasks of the Auditor General is the value-for-money audits. These audits are performed to determine whether money was spent taking economy and efficiency into consideration, and whether government programs took relevant steps to measure and report on their effectiveness. Each year, the office selects certain ministry's or agency's programs and

activities, based on factors such as the results of previous audits, the total revenues or expenditures at risk, the impact of the program or activity on the public and the costs of performing the audit in relation to the benefits, and audits them. According to the audit results, the Auditor General provides some recommendations to the selected programs. All these results and recommendations are published in the Annual Report.

Along with a chapter on value-for-money audits, the Annual Report also contains a chapter of follow-up reviews of all the value-for-money audits from the Annual Report that was published two years ago. The Auditor General reviews the status of all the recommendations provided to the selected programs from two years ago, and gets an in-depth information about their completion dates along with the reasons for delay, if any, or other issues that the ministry or agency might be facing.

Thus, these value-for-money audits and their follow-up reviews after 2 years provide us a perfect library from which to choose a case study as they represent an evolving system. The recommendations provided by the Auditor General are the goals that the selected ministry/agency wants to achieve before the follow-up review 2 years later, if possible. Hence, changes can be added to represent this development over 2 years. Moreover, the follow-up reviews provide the actual status of those goals after two years, which can be used to compare with the expected status defined two years ago to determine if the goals are on track. In addition, these follow-up reviews can be used to estimate the actual completion time frame for all the recommendations.

Out of all these different audits to choose from, we select the audit of Ontario Power Generation Human Resources conducted in 2013 and published in Section 3.05 of the 2013 Annual Report [21]. The follow-up review for this audit was published in Section 4.05 of the 2015 Annual Report [22]. The main reason behind choosing this specific program is the detail with which the audit and the follow-up were reported. The analysis done and recommendations provided by the Auditor General are in depth and sometimes, expressed quantitatively, which helps with the conversion of the data into goals and changes. Moreover, Ontario Power Generation has published a summary of the audit report in 2013 [23] followed by a public 6-months status update of all the key actions on June 30, 2014 [24]. Having these extra in-depth information, along with the actual audit report and the follow-up review, provides us with a strong data set for both definition and validation of changes and analysis.

5.1.1 System Description

Ontario Power Generation (OPG) is one of the five successor companies to Ontario Hydro. It was established in April 1999. Ontario Energy Board regulates most of the revenue of OPG. Any excess revenue, exceeding the expenses of OPG, is used to pay the stranded debt caused by the splitting up of Ontario Hydro. At the time the audit was conducted, OPG had been dealing with major challenges in trying to improve its operational efficiency and reduce its operating costs, especially labor costs.

The purpose of the audit of the Human Resources department of OPG, conducted in 2013, was:

- To evaluate if OPG took appropriate measures to ensure that economy and efficiency was considered while acquiring and managing its human resources, and in accordance with applicable policies, legislative requirements, contractual agreements, and sound business practices; and
- To assess the findings and report them.

The Audit Report stated that various reasons like reduced demand, closing of coal plants, and increase in private-sector involvement in power generation had led to a decrease in the amount of electricity generated by OPG over the last ten years (as per the report's publication date). In spite of that, electricity prices had been rising in Ontario. Since, OPG generated about 60% of Ontario's electricity at that time, the cost of electricity depended considerably on its operating costs. Moreover, the repayment of the debt of the Ontario Hydro also relied partially on OPG's profitability, giving another reason to keep OPG's expenses in check.

The audit reported that about two-thirds of OPG's operating costs were human resources-related and, therefore it was crucial to handle expenditures on OPG's human resources efficiently. The Auditor General conducted detailed investigation on reasons behind this high expenditure on human resources in OPG and suggested a total of six comprehensive recommendations to overcome them. In the following paragraphs, we discuss all the recommendations from the 2013 Annual Report for OPG Human Resources. Moreover, we try to explore how the OPG breaks down these main recommendations into sub-goals and assign relevant tasks or resources to achieve them with the help of the summary and the 6-months status update published by the OPG. Finally, all these recommendations are converted into goal

models. These goal models are built using the tool jUCMNav, and all the figures, hereafter, are the actual screenshots from the tool.

Recommendation 1

Description: To ensure that staffing levels are reasonable and that it has the right people in the right positions to meet its business needs, Ontario Power Generation should:

- evaluate and align the size of its executive and senior management group with its overall staffing levels;
- address the imbalances between overstaffed and understaffed areas in its nuclear operations; and
- review and monitor compliance with its recruitment and security clearance processes.

OPG's Response and Measures offered in 2013: OPG reported to the Auditor General after its audit in 2013 that in order to reduce labor costs, create a viable cost structure, and balance consumer electricity prices, it started a multi-year Business Transformation initiative in 2010. This Business Transformation addressed issues like the number of executive and senior management positions, and overall staffing levels. OPG also informed that it had conducted extensive benchmarking of its nuclear and other operations, depending on which, OPG's teams were implementing various initiatives designed to address opportunities for efficiencies, cost reductions, and staff imbalances in nuclear operations. Moreover, OPG concurred with the other suggestions of this recommendation provided by the Auditor General and informed the office of the Auditor General that it would review and observe conformity with its recruitment and

security clearance processes. They also decided to conduct an internal audit of their hiring practices.

Figure 16 represents the goal model for Recommendation 1. This recommendation is divided into three parts. These three are represented as goals “Evaluate & Align Senior Management Group”, “Address Imbalance between Overstaffed & Understaffed Areas”, and “Review compliance with Recruitment & Security Clearance processes”. These goals together contribute to the top-level goal “Ensure Reasonable Staff Levels”. The sole actor of this particular goal model is “OPG_HumanResources”, and it is assumed that the top-level goal has an importance of “100”, as indicated in the bracket after its name in the model. Moreover, in this goal model, everywhere it is assumed that all the sub-goals contribute equally as none of the recommendations or its parts are assigned any preference in the reports. In the figure, the circular TimedGRL symbol (⌚) is overlaid on some of its elements. This symbol conveys that a change has been added to that particular element. The actual definition of changes to those elements will be discussed in later sections.

To achieve part 1 of the recommendation, i.e., “Evaluate and align the size of its executive and senior management group with its overall staffing levels”, OPG needed to reduce the number of members in its executive and senior management group at the same rate as the number of employees were decreasing. As per OPG’s multi-year Business Transformation, in 2013, they were planning to reduce the number of employees by 2000, i.e., at a rate of 7.3% by 2015. Hence, OPG would have to decrease the size of the executives and senior management group at the same rate. Thus, the goal “Evaluate & Align Senior Management Group” receives

contributions from the goals “Decrease staff levels by 7.3%” and “Decrease Senior Management by 7.3%” (see Figure 16).

Part 2 of the recommendation stated “address the imbalances between overstaffed and understaffed areas in its nuclear operations”. To address this, OPG planned to reduce its nuclear staffing levels from 8% above the benchmark to 0%, i.e., complete elimination of the benchmark gap. This has been represented as the goal “Decrease staffing level below benchmark from 8% to 0%”, which solely contributes to the goal “Address Imbalance between Overstaffed & Understaffed Areas” (see Figure 16).

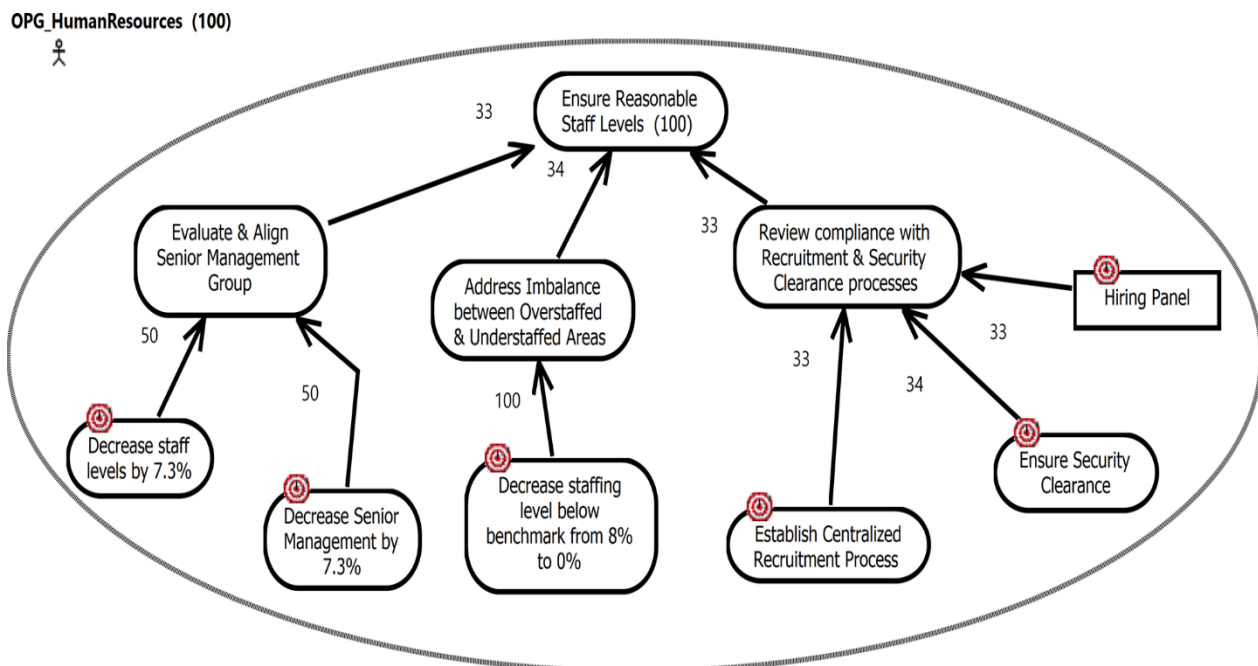


Figure 16: Goal Model for Recommendation 1

Part 3 of the recommendation stated “review and monitor compliance with its recruitment and security clearance processes”. To achieve this, OPG decided to centralize the requirements process to improve efficiency, controls, and compliance with the hiring process, represented by

the goal “Establish Centralized Recruitment Process” (see Figure 16). In addition, it was planned to establish Hiring Panels by 2015, represented by the Resource “Hiring Panel” (see Figure 16). Moreover, security clearance processes needed to be updated to ensure industry compliance, represented by the goal “Ensure Security Clearance” (see Figure 16). These two goals and one resource contribute equally to the goal “Review compliance with Recruitment & Security Clearance processes”.

Thus, OPG addressed all parts of Recommendation 1 and planned to achieve all key actions by 31st July, 2015 (estimated date as the 2015 Annual Report mentioned that the OPG handed in the data around the end of summer in 2015).

Recommendation 2

Description: To ensure that employees receive appropriate and reasonable compensation in a fair and transparent manner, Ontario Power Generation should:

- make its Annual Incentive Plan (AIP) more effective by creating a stronger link between awards and staff performance based on documented annual evaluations; and
- review salary levels and employee benefits, including pensions, to ensure that they are reasonable in comparison to other similar and broader-public-sector organizations and that they are paid out in accordance with policy, adequately justified, and clearly documented.

OPG’s Response and Measures offered in 2013: OPG reported to the Auditor General that Annual Incentive Plan (AIP) awards in OPG were offered, depending upon individual,

business unit, and corporate performance. Hence, OPG acknowledged the significance of strongly linking individual incentive awards with performance and, it was ready to assess measures to strengthen this linkage, as per the suggestions by the Auditor General. At the time of the audit, management compensation of OPG was at the 50th percentile (i.e., median) relative to the benchmark based on data from Canadian organizations in both general and specific industries. Total management compensation had declined since 2008, and OPG froze the compensation for OPG's executives, including vice presidents. Moreover, measures were taken to make sure that the employee salaries, benefits, and pensions followed OPG policy, Canada Revenue Agency taxation requirements, and other legislation. As recommended by the Auditor General, OPG agreed to continue observing and improving means to defend and clearly report the compensation. In addition, OPG acknowledged that its pension and benefits were higher than the market average, which led OPG to review its plans in 2011 in order to make them more viable. OPG also participated in a 2012 pension reform committee established by the government, and informed that it would be participating in the electricity sector working group, consisting of employer and employee representatives, as announced in the 2013 Ontario Budget.

Figure 17 represents the goal model for Recommendation 2. This recommendation is divided into two parts. These two are represented as goals "Create stronger links between Performance & Awards", and "Review Salary Levels & Employee Benefits". These goals together contribute to the top-level goal "Ensure Appropriate & Reasonable Compensation". Similar to the goal model of Recommendation 1, "OPG_HumanResources" is the only actor of this particular goal model, and the top-level goal has been assigned an importance of 100. The importance of the Recommendation 2 top-level goal is same as the Recommendation 1 top-level

goal as the audit report does not prioritize any of the six recommendations. Hence, all the recommendations are modeled to be equally important for this actor. Moreover, as before, everywhere it is assumed that all the sub-goals contribute uniformly. The TimedGRL symbol is visible in the model, however the actual definition of changes to those elements will be discussed in later sections.

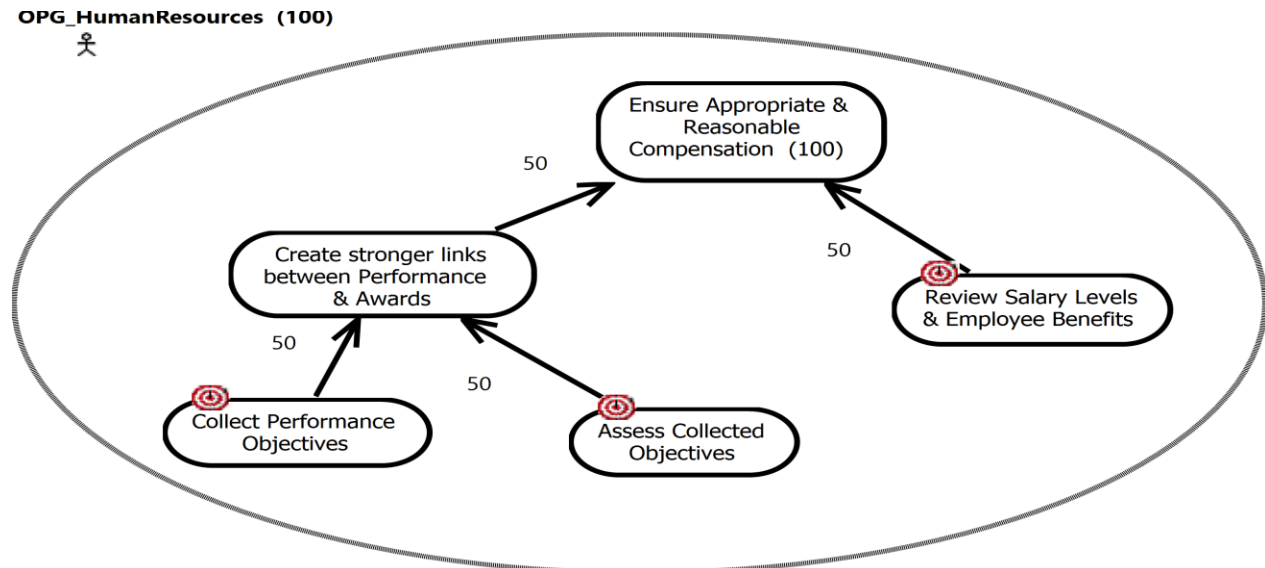


Figure 17: Goal Model for Recommendation 2

Part 1 of the recommendation can be summarized as “creating a stronger link between awards and staff performance based on documented annual evaluations”. To achieve this OPG planned to ensure that all the employees and management staff were required to enter performance objectives into an electronic system, allowing for compliance monitoring. In addition, proper assessment of these objectives was outlined in order to meet the expectation of creating stronger linkage between awards and performance. Thus, the goal “Create stronger links between Performance & Awards” receives contributions from the goals “Collect Performance Objectives” and “Assess Collected Objectives” (see Figure 17).

Part 2 of this recommendation deals with reviewing salary levels and employee benefits, including pensions to ensure they meet industry standard. To achieve this goal, OPG planned to perform various activities such as “Compensation and Pension Benchmarking”, and “Reviewing Compensation Levels for Management Staff”. All of these activities are captured by a single goal “Review Salary Levels & Employee Benefits” in the goal model (see Figure 17).

Thus, OPG addressed all parts of Recommendation 2 and planned to achieve all key actions by 31st July, 2015.

Recommendation 3

Description: To ensure that its non-regular and contract resources are used cost-efficiently, Ontario Power Generation should:

- improve its succession planning, knowledge retention, and knowledge transfer processes to minimize the need to rehire retired employees for extended periods;
- conduct an open competitive process for outsourcing its information technology services before the current contract expires; and
- manage and monitor closely the hours reported by the contractors to avoid the risk of overpayment.

OPG’s Response and Measures offered in 2013: OPG reported to the Auditor General that it followed contracting practices that were conformant with nuclear industry practices. However, OPG agreed to review its practices related to rehiring retired employees, as suggested by the Auditor General. It also informed that they outsourced their information technology

services in 2001 by conducting a competitive process, and the same contract was renewed and restructured in 2007 after thorough assessment and third-party validation. However, OPG agreed to evaluate all possible options before the expiry of the previous contract by conducting an open competitive process as recommended in the audit. Moreover, OPG agreed to enhance controls over accurate contractor payments by investigating alternatives to observe and regulate contractor hours.

Figure 18 represents the goal model for Recommendation 3. This recommendation is divided into three parts. These three parts are represented as goals “Strengthen succession planning”, “Improve IT contract renewal process”, and “Monitor contractor hours closely”. These goals together contribute to the top-level goal “Ensure cost-efficient use of Contract resources”. Similar to the previous goal models, “OPG_HumanResources” is the only actor of this particular goal model, and the top-level goal has been assigned an importance of 100. Moreover, as before, everywhere it is assumed that all the sub-goals contribute uniformly. The TimedGRL symbol is visible in the model, however the actual definition of changes to those elements will be discussed in later sections.

Part 1 of the recommendation can be summarized as “improving succession planning knowledge retention and knowledge transfer processes to minimize the need to rehire retired employees”. To achieve this OPG planned succession planning and knowledge transfer for strengthening critical/at-risk roles. In addition, OPG decided to introduce a new policy for re-hiring former employees which would try to increase the timeframe required before rehiring a previous contractor. These activities are encompassed into a single goal “Strengthen succession planning” in the goal model (see Figure 18).

OPG_HumanResources (100)

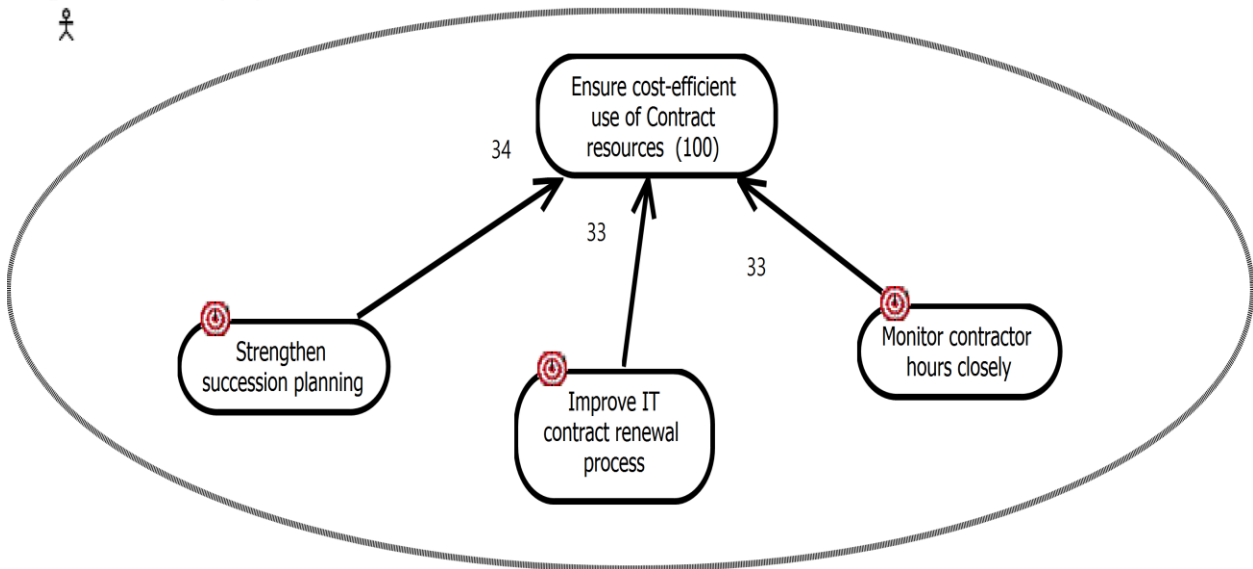


Figure 18: Goal Model for Recommendation 3

Part 2 of this recommendation deals with improving the IT contract renewal process by conducting an open competitive process. To achieve this goal, OPG agreed to put IT Outsourcing Agreement out for competitive bidding in May 2014. This action is represented as the goal “Improve IT contract renewal process” in the goal model (see Figure 18).

Part 3 of this recommendation stated “Manage and monitor closely the hours reported by the contractors to avoid the risk of overpayment”. To accomplish this, OPG planned to conduct a comprehensive assessment of the contractor control framework, including contract structures, time capture and approval processes and tools. In addition, OPG planned to implement a time tracking system for contractors at nuclear sites. Both of these actions are represented by a single goal “Monitor contractor hours closely” in the goal model (see Figure 18).

Thus, OPG addressed all parts of Recommendation 3 and planned to achieve all key actions by 31st July, 2015.

Recommendation 4

Description: To ensure that overtime hours and costs are minimized and monitored, Ontario Power Generation should:

- decrease overtime costs for outages by planning outages and arranging staff schedules in a more cost-beneficial way; and
- review other ways to minimize overtime.

OPG's Response and Measures offered in 2013: OPG reported to the Auditor General that the majority of overtime costs were because of emerging issues, unforeseen equipment conditions, and changes in regulatory requirements of nuclear outages, which were planned and resourced two years in advance because of their complex nature. OPG regularly adjusted the use of overtime versus contractors and its overtime cost percentage was comparable to large utility companies in the United States. However, OPG agreed to conduct a cost-benefit analysis to examine various options, including scheduling and hiring staff and/or contractors, to reduce overtime cost.

Figure 19 represents the goal model for Recommendation 4. This recommendation is divided into two parts. These two are represented as goals “Decrease overtime costs for outages”, and “Review other ways to minimize overtime”. These goals together contribute to the top-level goal “Ensure minimization of overtime hours & costs”. Similar to the previous goal models, “OPG_HumanResources” is the only actor of this particular goal model, and the top-level goal has been assigned an importance of 100. Moreover, as before, everywhere it is assumed that all

the sub-goals contribute uniformly. The TimedGRL symbol is visible in the model, however the actual definition of changes to those elements will be discussed in later sections.

Part 1 of the recommendation stated “decrease overtime costs for outages by planning outages and arranging staff schedules in a more cost-beneficial way”. To achieve this OPG planned to conduct a cost benefit analysis of specific station outage critical path work to identify opportunities to reduce overtime. In addition, crew shift changes were planned, which would lead to reduced overtime during outages. Thus, the goal “Decrease overtime costs for outages” receives contributions from the goals “Implement shift changes” and “Conduct cost benefit analysis” (see Figure 19).

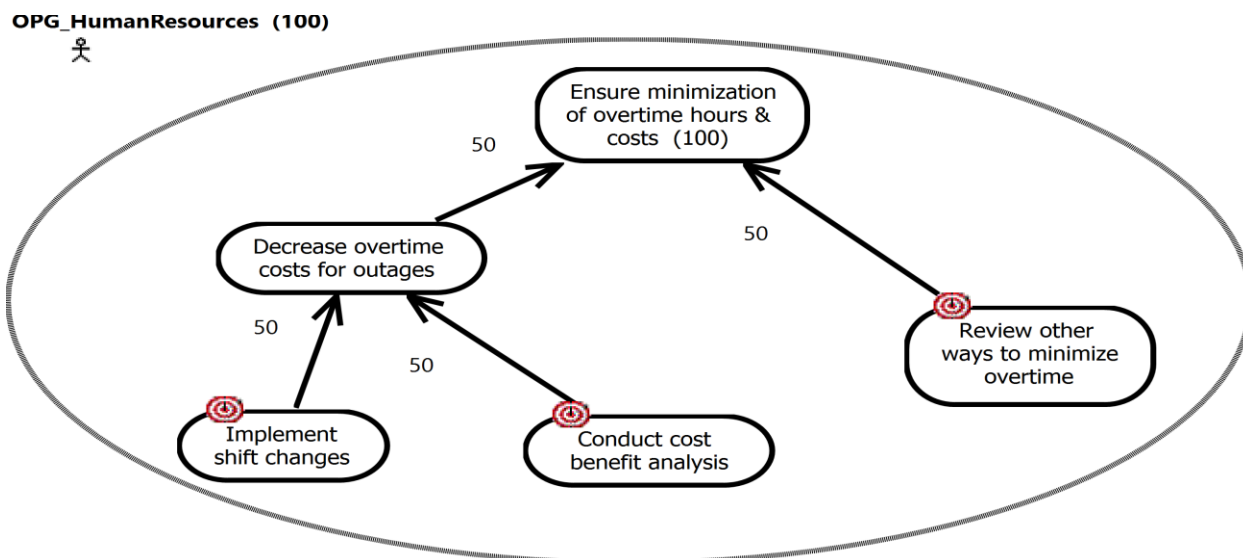


Figure 19: Goal Model for Recommendation 4

Part 2 of the recommendation asked to review other ways to minimize overtime. For this purpose, OPG planned to enhance management processes, approvals, and controls to limit individual overtime in its nuclear operations including the overtime ceiling. These actions are

represented by the goal “Review other ways to minimize overtime” in the goal model (see Figure 19).

Thus, OPG addressed all parts of Recommendation 4 and planned to achieve all key actions by 31st July, 2015.

Recommendation 5

Description: To minimize the cost of sick leaves and avoid potential misuses or abuses of sick leave entitlements, Ontario Power Generation should:

- review its sick leave plan for staff who joined prior to 2001; and
- monitor the results of sick leave management programs to identify and manage unusual sick leave patterns.

OPG’s Response and Measures offered in 2013: OPG reported to the Auditor General that it was trying to reduce sick leave costs, while promoting a healthy and productive workforce. OPG agreed to review its sick leave plans and assess the costs and benefits of any required changes, as recommended by the Auditor General. Moreover, OPG’s Business Transformation efforts were already dealing with implementing measures and tools such as the automated employee absence calendar to aid managers in effectively managing sick leave issues.

Figure 20 represents the goal model for both Recommendations 5 and 6. However, here we will focus on the part of the goal model that deals with Recommendation 5. This recommendation is divided into two parts. These two are represented as goals “Review sick leave plan for staffs who joined before 2001”, and “Identify & manage unusual sick leave

patterns”. These goals together contribute to the top-level goal “Minimize misuse of sick leaves”. Similar to the previous goal models, “OPG_HumanResources” is the only actor of this particular goal model, and the top-level goal has been assigned an importance of 100. Moreover, as before, everywhere it is assumed that all the sub-goals contribute uniformly. The TimedGRL symbol is visible in the model, however the actual definition of changes to those elements will be discussed in later sections.

Part 1 of the recommendation stated “review its sick leave plan for staff who joined prior to 2001”. To achieve this OPG planned to review and assess the sick leave plans for staff that joined prior to 2001 in the context of overall benefits and compensation. These actions are represented by the goal “Review sick leave plan for staffs who joined before 2001” in the goal model (see Figure 20).

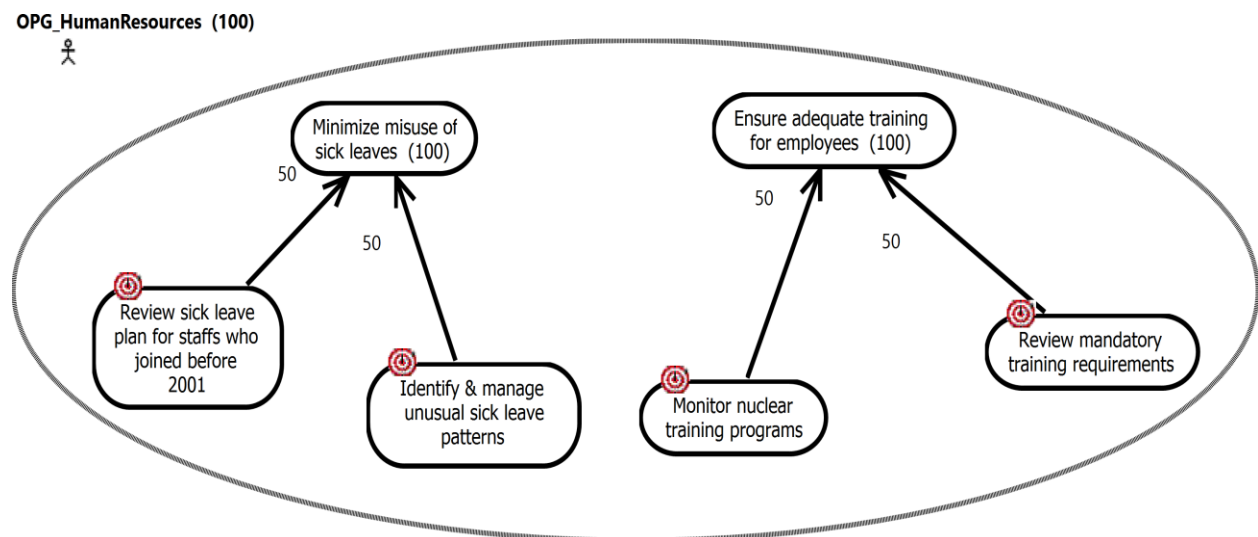


Figure 20: Goal Model for Recommendations 5 and 6

Part 2 of the recommendation suggested observing the results of sick leave management programs to identify and manage unusual sick leave patterns. For this purpose, OPG planned to

provide training and support for managers to address attendance issues, reduce the number of days required after which the employee needed to submit Medical Absence Reports, and implement a new centralized disability management process, utilizing a third-party provider for sick leave management. These actions are represented by the goal “Identify & manage unusual sick leave patterns” in the goal model (see Figure 20).

Thus, OPG addressed all parts of Recommendation 5 and planned to achieve all key actions by 31st July, 2015.

Recommendation 6

Description: To ensure that its employees are adequately trained for their jobs, Ontario Power Generation should:

- continue to review and monitor the adequacy, quality, and completion rates of its nuclear training programs in order to identify areas for improvement, and address the areas that have already been identified; and
- review the nature and timing of its mandatory training requirements as well as its delivery methods for hydro/thermal staff to ensure they are meeting business needs cost-effectively.

OPG’s Response and Measures offered in 2013: OPG reported to the Auditor General that its nuclear training programs were already extensively benchmarked against industry best practices and were routinely audited by the Canadian Nuclear Safety Commission and the World Association of Nuclear Operators. However, OPG agreed to review nature, timing, and delivery

methods of compulsory training requirements for hydro/thermal staff, as recommended by the Auditor General.

As mentioned before, Figure 20 represents the goal model for both Recommendations 5 and 6. However, here we will focus on the part of the goal model that deals with Recommendation 6. This recommendation is divided into two parts. These two are represented as goals “Monitor nuclear training programs”, and “Review mandatory training requirements”. These goals together contribute to the top-level goal “Ensure adequate training for employees”. Similar to the previous goal models, “OPG_HumanResources” is the only actor of this particular goal model, and the top-level goal has been assigned an importance of 100. Moreover, as before, everywhere it is assumed that all the sub-goals contribute uniformly. The TimedGRL symbol is visible in the model, however the actual definition of changes to those elements will be discussed in later sections.

Part 1 of the recommendation suggested continuous improvement of adequacy, quality and completion rates of OPG’s nuclear training programs. To achieve this OPG continued the implementation of previously planned enhancements to its nuclear training programs, wherever there were favorable circumstances for improvement while continuing to build on identified strengths. This is represented by the goal “Monitor nuclear training programs” in the goal model (see Figure 20).

Part 2 of the recommendation focused on reviewing the nature and timing of OPG’s mandatory training requirements as well as its delivery methods for hydro/thermal staff. For this purpose, OPG planned to perform various tasks such as implementing the previously identified

action plan for improving training programs, centralizing Hydro/Thermal training programs, and reviewing mandatory training requirements for Hydro/Thermal staff. These actions are represented by a single goal “Review mandatory training requirements” in the goal model (see Figure 20).

Thus, OPG addressed all parts of Recommendation 6 and planned to achieve all key actions by 31st July, 2015.

Comprehensive Goal Model

We have already identified the elements required and relationships among them for each Recommendation. Thus, the actor “OPG_HumanResources” has six equally important high-level goals that it wants to achieve. Using the current goal models, we can observe the satisfaction values of each individual recommendation, if analyzed. However, having a comprehensive goal model which can analyze and report the overall completion status would provide a valuable perspective on the overall system.

Figure 21 represents such a comprehensive goal model. It introduces a new actor “Office of The Auditor General”. The main objective of this actor is to ensure whether the recommendations provided to the OPG Human Resources are fulfilled, represented by the goal “Ensure completion of recommendations” in the goal model (see Figure 21). Each of the high-level goals of the actor “OPG_HumanResources” contributes uniformly to the main goal of the actor “Office of The Auditor General”. The uniform contributions are assumed because the recommendations were not prioritized in the report. Both the actors have been assigned an importance value of 100, assuming that they both are equally significant for the system as a whole.

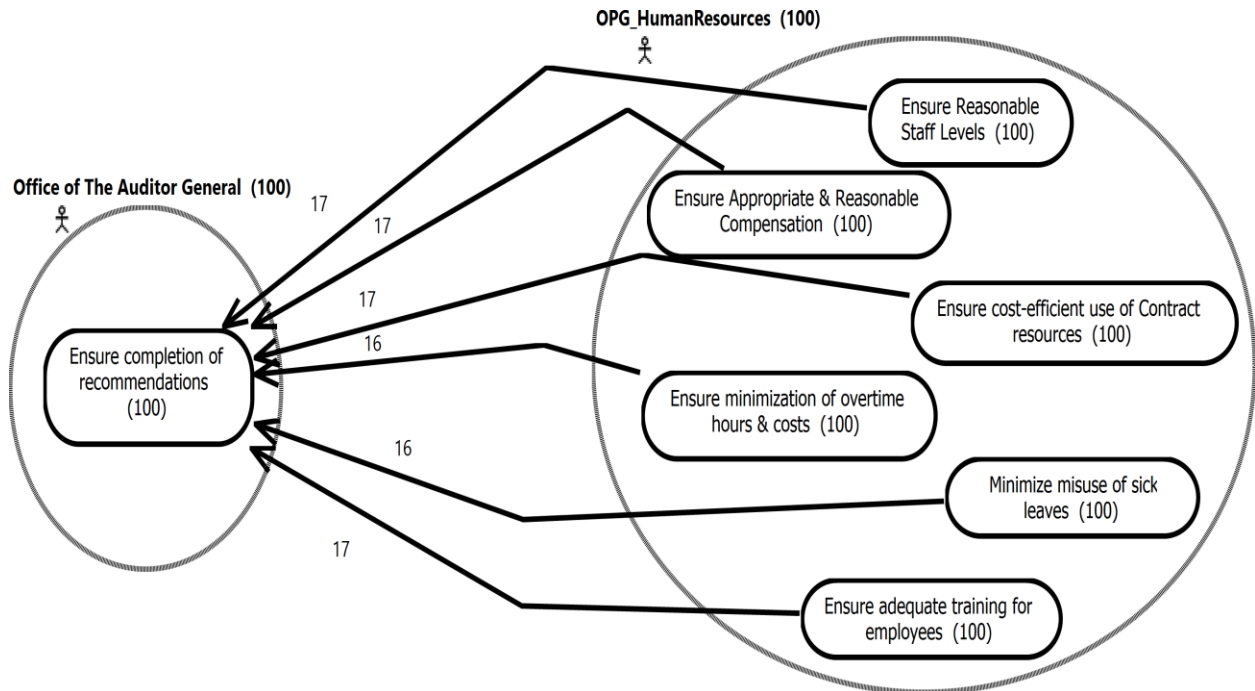


Figure 21: Comprehensive Goal Model for OPG Audit Report

Hence, we have modeled all the elements and their relationships required to portray this system correctly (along with some basic assumptions). Once we have a set of goal models, the next steps are to identify required changes for each element according to the expected development as well as the original development, and finally add those changes to the relevant elements, as is explained in the next section.

5.2 Identification and Addition of Changes to Model Elements

Currently, we have a set of goal models that represent the system completely. Before TimedGRL, it was already possible to initialize satisfaction values of leaf goals to desired values and analyze the system to observe the overall evaluation values of each model element. However, it was not possible to specify how the behavior of these leaf goals varied over a period of time, even if that behavior was known. So, the only option to evaluate the goal model at

different points in time was to manually calculate the values as per the expected behavior, update the model, and then analyze it.

As discussed in Chapter 3, TimedGRL allows the modeler to integrate such changes directly into the relevant model elements. However, before adding those changes, we first need to identify them, and then group them in different dynamic contexts according to the changes we want to apply together. In our case-study, both the Annual Reports by the Auditor General, and the summary and six-months status update provided by Ontario power Generation (OPG) provide us with enough data to identify changes in behavior over time for the relevant model elements corresponding to recommendations.

The available data sets suggest that the changes for this case study can be categorized into three groups:

1. **Expected Changes as of 10th December, 2013:** When the Auditor General generated the audit report for OPG Human Resources in 2013, OPG summarized the recommendations provided in the audit along with the measures they were going to take in order to achieve them. As a follow-up review was due in 2015, OPG expected to fulfill all recommendations before 31st July, 2015, when they were expected to provide the status report to the Auditor General. Thus, this set of expected changes should have been added to the goal model on 10th December, 2013, and it should show how OPG expected the system to change in order to achieve their target.

2. **Changes after Status Update as of 30th June, 2014:** In 2014, OPG published a public status report of all key actions suggested in the audit report of 2013. This report provided the actual state of all goal model elements along with updated target dates to achieve those actions, if required. Thus, this set of changes should have been added to the goal model on 30th June, 2014, and it should show what was the actual status of the system in 2014 and how OPG expected the system to change. If any of the changes from the first group was on track according to this status update, then that change could be reused in this group by including the dynamic context of the first group in the dynamic context for this group.
3. **Changes after Follow-up Review as of 31st July, 2015:** Around the end of summer in 2015, OPG provided the status of all the recommendations to the Auditor General as a part of follow-up review. This follow-up review report provided the actual completion status of each model element as of 31st July, 2015, along with the tentative completion dates for the recommendations that could not be fulfilled. Thus, this set of changes should have been added to the goal model on 31st July, 2015, and it should show what was the actual status of the system at the time of follow-up review (when all recommendations should actually have been fulfilled according to the expectation in 2013) and how OPG expected the recommendations, that were yet not fulfilled, to achieve their target. If any of the changes from the first or second group was on track according to this status update, then that change could be reused in this group by including the dynamic

context of the second group (and transitively the dynamic context of the first group) in the dynamic context for this group.

Once we have identified the three groups of changes, i.e., three dynamic contexts, the next step is to identify specific changes in those dynamic contexts and then, add them to the relevant model elements. The following sub-sections perform this step.

5.2.1 Expected Changes as of 10th December, 2013

Once the recommendations provided in the audit [21] have been converted into a goal model, let us try to identify changes that were applicable to the model elements at the time of the audit. To achieve the main six goals of the actor OPG_HumanResources, OPG had to satisfy all the leaf goals, i.e., the low-level goals whose contribution ultimately fed the main goals. Initially, none of those goals were satisfied (i.e., on 10th December, 2013). Thus, an initial solution is defined by creating a new evaluation strategy called “EvaluationStrategy6”, in which satisfaction values for all the leaf goals were assigned zero, meaning none of them are yet fulfilled. The already existing Quantitative GRL evaluation mechanism then propagates these initial satisfaction values to higher-level goal model elements (see Figure 22 and Figure 23 (similar evaluations for other goal models)).

OPG was targeting to implement all recommendations before the follow-up review after 2 years, i.e., by 31st July, 2015. Converting this into quantitative terms, OPG wanted to reach a satisfaction value of 100 for all the leaf goals, which would then be propagated to higher-level model elements, by 31st July, 2015, in order to achieve its target. Let us walk through each goal model and try to recognize the set of behaviors to be followed by all model elements.

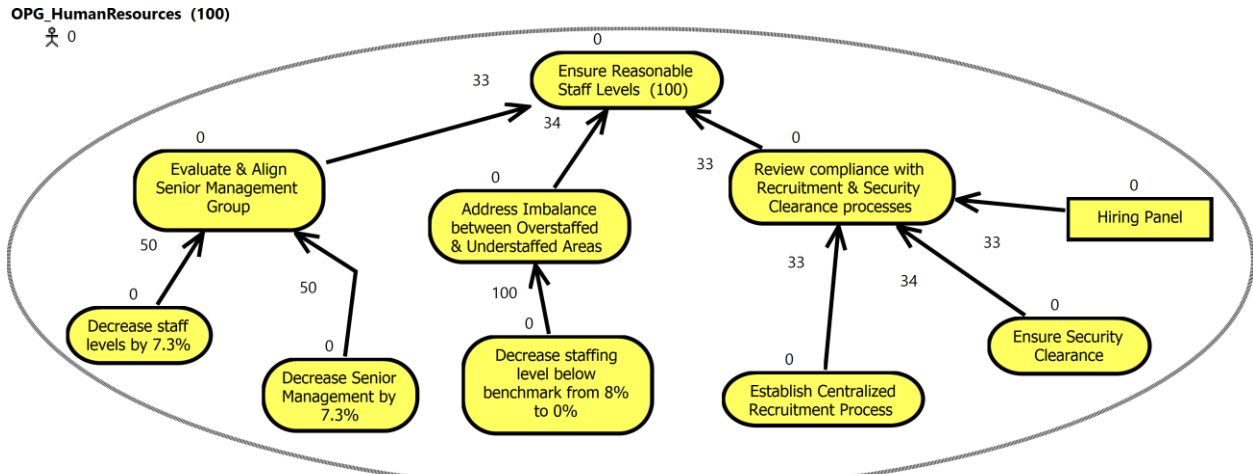


Figure 22: Initial evaluation (EvaluationStrategy6) of goal model of Recommendation 1

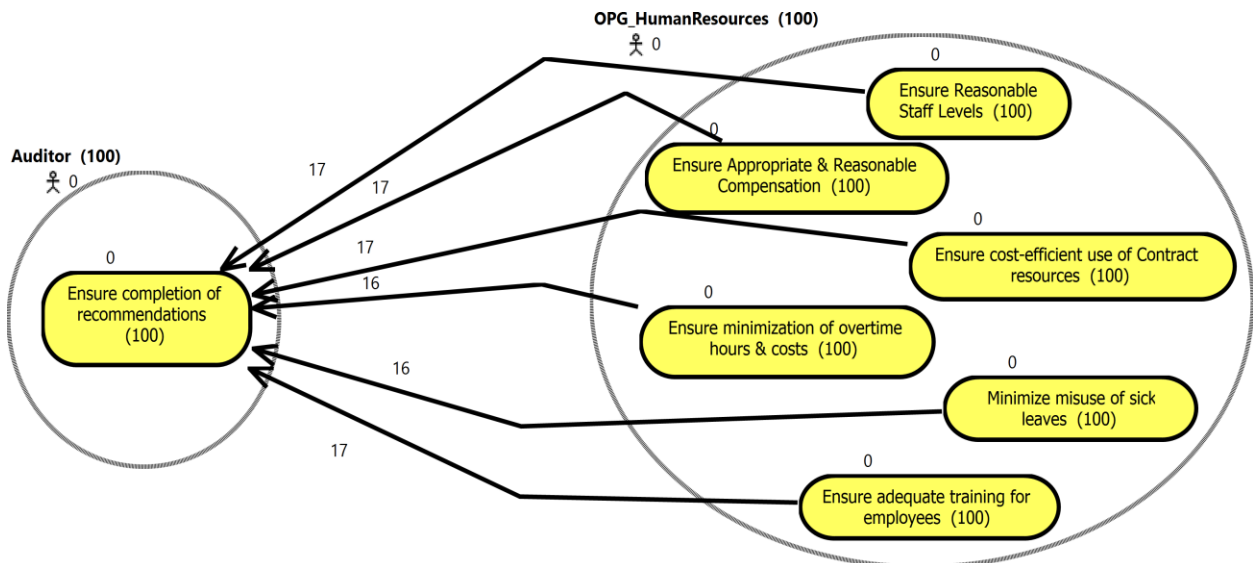


Figure 23: Initial evaluation (EvaluationStrategy6) of the comprehensive goal model

In the goal model representing Recommendation 1 (Figure 16), the leaf elements that OPG had to work on are “Decrease staff levels by 7.3%”, “Decrease Senior Management by 7.3%”, “Decrease staffing level below benchmark from 8% to 0%”, “Establish Centralized Recruitment Process”, “Ensure Security Clearance”, and “Hiring Panel”. On 10th December, 2013, satisfaction value for all the mentioned elements was zero, which needed to grow to 100 by 31st July, 2015, except the resource “Hiring Panel”, which was only going to be established

and fully functional towards the end (can be represented by adding a Deactivation change). However, the exact behavior in which these model elements were going to grow was not defined in any of the referred documents. Hence, we have decided to assume that all the changes portraying growth are going to be linear until specified. Thus, the changes that needed to be applied for Recommendation 1 goal model's leaf elements are LinearChanges on the satisfaction values of the five leaf goals growing from 0 to 100, a DeactivationChange on "Hiring Panel" resource from the beginning until 30th June, 2015 (estimated), and finally a ConstantChange of 100 on the satisfaction value of "Hiring Panel" from the day of its activation until indefinitely (any future date outside the evaluation range can be selected for this).

Similarly, for the goal models of the other recommendations, the satisfaction values of all leaf goals also grow from 0 to 100, without any exception. Hence, LinearChanges are applied to the satisfaction value of each leaf goal.

In jUCMNav, a dynamic context called "DynamicContext (Expectation)" is created to group the mentioned changes, so that they can act simultaneously on the system. Moreover, a set of initial satisfaction values (EvaluationStrategy) should be included in this dynamic context in order to facilitate evaluation. For this purpose, "EvaluationStrategy6", defined previously, is included in "DynamicContext (Expectation)" (see the selected second dynamic context in DynamicContextGroup13 in Figure 24(a)).

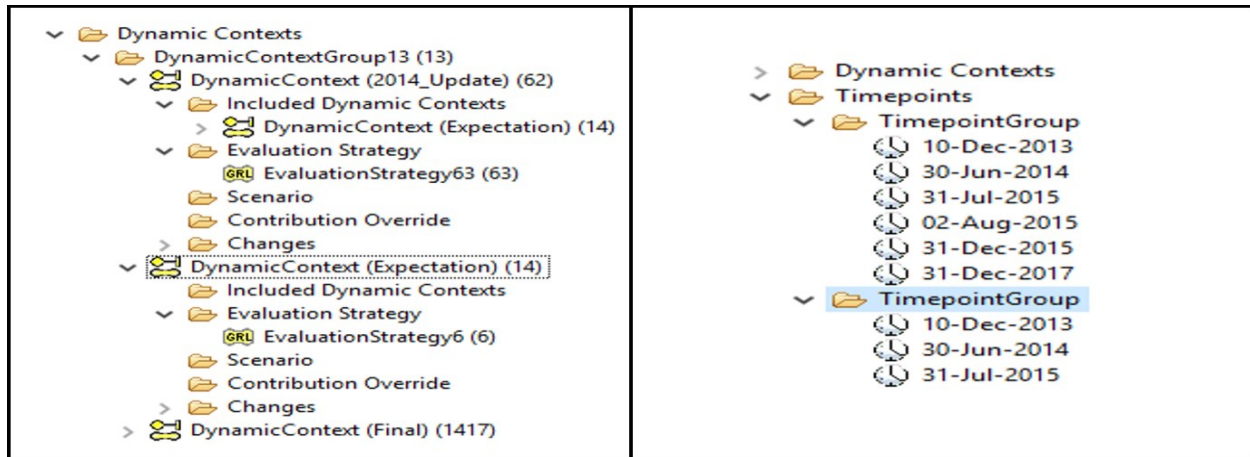


Figure 24: (a) Specification of Dynamic Contexts and (b) Specification of Timepoints

The next step now is to define all the required changes to the relevant model elements. A separate “Manage Change” dialog box has been introduced in jUCMNav to support addition or update of changes, which can be opened from the drop-down menu appearing after a right-click on a valid goal model element. The “Manage Change” box is always opened in the context of the selected element and allows to manage changes for the different attributes of that element. Using this functionality, we will start adding the discussed changes to our goal model.

Figure 25 depicts the addition of a LinearChange to the satisfaction value (also known as evaluation) of the leaf goal “Decrease staff levels by 7.3%” of Recommendation 1. This change needs to be added to the following dynamic context: DynamicContext (Expectation). The affectedProperty for this change is the element’s evaluation. The start date is 10th December, 2013, the end date is 31st July, 2013, and the new value is 100. According to the metamodel (Figure 3), the mentioned data is enough to create a LinearChange. Hence, clicking on “Add Change” will add a linear change for this element in the selected dynamic context. The added linear change will be visualized in the “Available Changes” table at the right side of the dialog box (see Figure 25).

Manage Change

Manage Change

Select Dynamic Context:

DynamicContextGroup13:DynamicContext (Expectation)

What to Change :

Element's Evaluation

Select Type of Change :

Linear Change

Select Start Date :

2013-12-10

Select End Date :

2015-07-31

Enter New value :

100

Select Decomposition Type

Enter Quadratic Coefficient

Enter Linear Coefficient

Enter Constant

Enter Formula

Add Change

Update Change

Delete Change

Deselect

Available Changes :

| Affected Property | Start Date | End Date | Change Type | New Value | Formula |
|-------------------|------------|----------|-------------|-----------|---------|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Reset

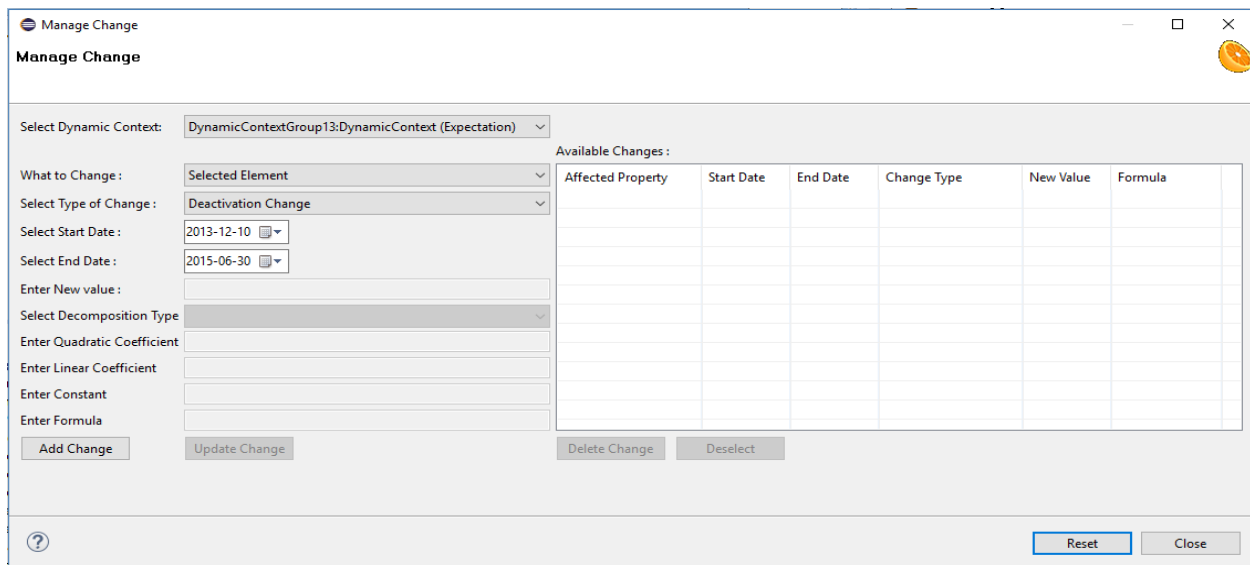
Close

Figure 25: Addition of a Linear Change through "Manage Change" dialog box

Linear changes for all the leaf goals are defined in a similar manner. Now, the only changes that remain to be specified are the DeactivationChange and ConstantChange for the resource “Hiring Panel” in Recommendation 1. Figure 26 depicts the addition of the DeactivationChange to the resource, while Figure 27 shows the final change list of the resource “Hiring panel” including the constant change. All the changes can also be visualized in their specific dynamic context. Moreover, the activation of fields in the Manage Change dialog box happens as per the selected change type. E.g., DeactivationChange does not require any input other than a start date and an end date, and hence all other fields are greyed out in Figure 26.

Similarly, to add a QuadraticChange, the modeler would need to enter the quadratic coefficient, the linear coefficient, and the constant of the quadratic formula. To add a FormulaChange, a valid formula must be typed in the provided space. The mXparser tool [34], which is a parser and evaluator of mathematical expressions/formulae provided as plain text/string, is being used for parsing the formula entered in the required box. mXparser supports

a large number of arbitrarily complex mathematical expressions. Two examples of possible valid formulae are $((2 \cdot t) + \sin(t))$ and $(1.5 + \ln(t))$. Finally, the decomposition type can be selected from the drop-down box to specify an EnumerationChange. As these changes are not required for this case study, they will not be added to the goal model.



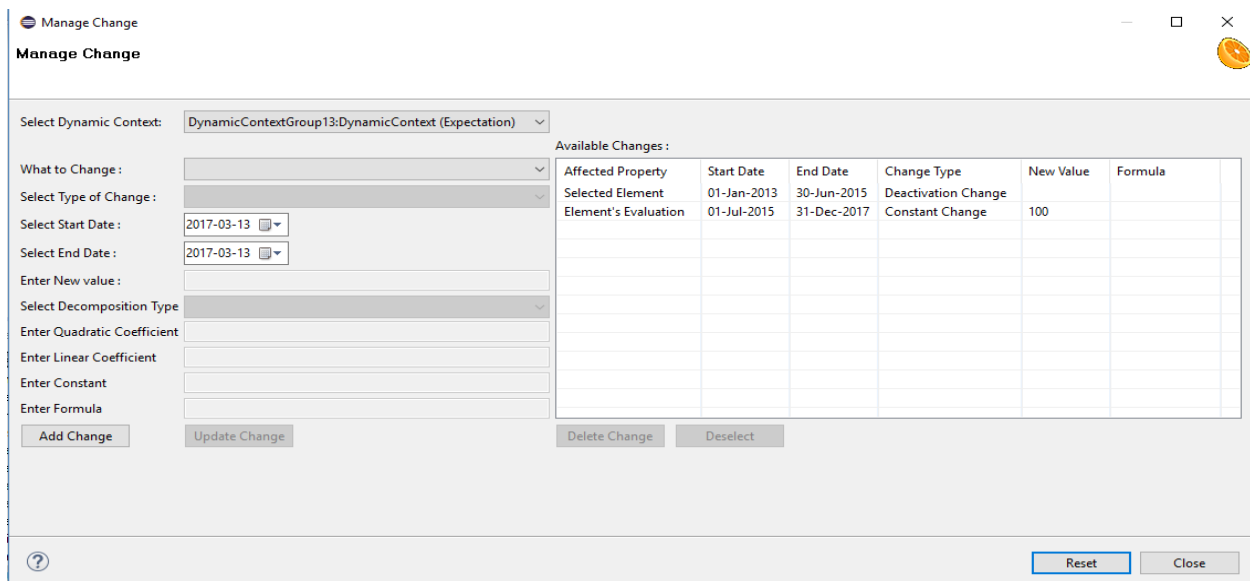
The 'Manage Change' dialog box is shown with the following fields and controls:

- Select Dynamic Context:** DynamicContextGroup13:DynamicContext (Expectation)
- What to Change:** Selected Element
- Select Type of Change:** Deactivation Change
- Select Start Date:** 2013-12-10
- Select End Date:** 2015-06-30
- Enter New value:** (empty field)
- Select Decomposition Type:** (empty dropdown)
- Enter Quadratic Coefficient:** (empty field)
- Enter Linear Coefficient:** (empty field)
- Enter Constant:** (empty field)
- Enter Formula:** (empty field)
- Buttons:** Add Change, Update Change, Delete Change, Deselect, Reset, Close

Available Changes Table:

| Affected Property | Start Date | End Date | Change Type | New Value | Formula |
|-------------------|------------|----------|-------------|-----------|---------|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Figure 26: Addition of a Deactivation Change through "Manage Change" dialog box



The 'Manage Change' dialog box is shown with the following fields and controls:

- Select Dynamic Context:** DynamicContextGroup13:DynamicContext (Expectation)
- What to Change:** (empty dropdown)
- Select Type of Change:** (empty dropdown)
- Select Start Date:** 2017-03-13
- Select End Date:** 2017-03-13
- Enter New value:** (empty field)
- Select Decomposition Type:** (empty dropdown)
- Enter Quadratic Coefficient:** (empty field)
- Enter Linear Coefficient:** (empty field)
- Enter Constant:** (empty field)
- Enter Formula:** (empty field)
- Buttons:** Add Change, Update Change, Delete Change, Deselect, Reset, Close

Available Changes Table:

| Affected Property | Start Date | End Date | Change Type | New Value | Formula |
|----------------------|-------------|-------------|---------------------|-----------|---------|
| Selected Element | 01-Jan-2013 | 30-Jun-2015 | Deactivation Change | | |
| Element's Evaluation | 01-Jul-2015 | 31-Dec-2017 | Constant Change | 100 | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Figure 27: Changes visualized in the table for the resource "Hiring panel" for the selected Dynamic Context

Hence, all the changes for this particular category have been identified and added to the relevant model elements, and the goal model is now ready to be evaluated at different timepoints and as a whole over time according to this set of changes. These analyses are discussed in later sections.

5.2.2 Changes after Status Update as of 30th June, 2014

OPG conducted a six-month follow-up of the recommendations provided, and published a public status update report on 30th June, 2014 [24]. This report informed the status at the time of the leaf goals required to achieve the target as well as the future forecast for the yet incomplete goals. Some of the leaf goals were still following the same behavior as expected on 10th December, 2013, while others' behaviors changed. Status of a few of the goals is not reported in this update. We assume here that these goals were still on track as expected and defined on 10th December, 2013. In the following paragraphs, we discuss the leaf goals whose behavior and forecast changed from the expectation.

Recommendation 1:

- The satisfaction value of the leaf goal “Decrease staffing level below benchmark from 8% to 0%” was 50 (as the benchmark at the time was 4%) on 30th June, 2014. However, OPG forecasted that this goal could only be achieved at the end of December 2017. Hence, two linear changes are required to illustrate that: (i) satisfaction value to 50 from 10th December, 2013, to 30th June, 2014; (ii) satisfaction value to 100 from 1st July, 2014, to 31st December, 2017.

- The leaf goal “Establish Centralized Recruitment Process” was already achieved on 30th June, 2014. Thus, the following two changes are required to illustrate that: (i) a LinearChange on its satisfaction value to 100 from 10th December, 2013, to 30th June, 2014; (ii) a ConstantChange of 100 on its satisfaction value from 1st July, 2014, to 31st December, 2017 (any future date outside the evaluation range would be fine).
- The leaf goal “Ensure Security Clearance” was already achieved on 30th June, 2014, and hence changes similar to the previous point need to be added to its satisfaction value.

Recommendation 2:

- The leaf goal “Collect Performance Objectives” was already achieved on 30th June, 2014, and hence, changes similar to the goal “Establish Centralized Recruitment Process” of Recommendation 1 need to be added to its satisfaction value.
- OPG discovered that the leaf goal “Assess Collected Objectives” could be activated only in March 2016, as assessment of that year could only be done after collecting objectives. Thus, the following two changes are required to illustrate that: (i) a DeactivationChange from 10th December, 2013, to 31st March, 2016; (ii) a ConstantChange of 100 on its satisfaction value from 1st April, 2016, until the end (any future date outside the evaluation range can be selected for this).

Recommendation 3:

- The leaf goal “Improve IT contract renewal process” was already achieved on 30th June, 2014, and hence, changes similar to the goal “Establish Centralized Recruitment Process” of Recommendation 1 need to be added to its satisfaction value.

Recommendation 4:

- The leaf goals “Implement shift changes” and “Conduct cost benefit analysis” were already achieved on 30th June, 2014, and hence, changes similar to the goal “Establish Centralized Recruitment Process” of Recommendation 1 need to be added to their satisfaction value.

Recommendation 5:

- OPG forecasted to achieve the leaf goal “Review sick leave plan for staffs who joined before 2001” by the end of June 2015. Thus, the following two changes are required to illustrate that: (i) a LinearChange on its satisfaction value to 100 from 10th December, 2013, to 30th June, 2015; (ii) a ConstantChange of 100 on its satisfaction value from 1st July, 2015, until the end (any future date outside the evaluation range can be selected for this).
- OPG forecasted to achieve the leaf goal “Identify & manage unusual sick leave patterns” by the end of December 2014. Thus, the following two changes are required to illustrate that: (i) a LinearChange on its satisfaction value to 100 from

10th December, 2013, to 31st December, 2014; (ii) a ConstantChange of 100 on its satisfaction value from 31st December, 2014, until the end (any future date outside the evaluation range can be selected for this).

Recommendation 6:

- OPG forecasted to achieve the leaf goal “Review mandatory training requirements” by the end of June 2015. Thus, following are the two changes required to illustrate that: (i) a LinearChange on its satisfaction value to 100 from 10th December, 2013, to 30th June, 2015; (ii) a ConstantChange of 100 on its satisfaction value from 1st July, 2015, until the end (any future date outside the evaluation range can be selected for this).

In jUCMNav, a dynamic context called “DynamicContext (2014_Update)” is created to group the changes in this category. “EvaluationStrategy6”, defined previously, is included in this dynamic context as well to define the set of initial satisfaction values. Moreover, the dynamic context of previous category (DynamicContext (Expectation)) is included in this context to reuse the changes that were still on track (see the first dynamic context in DynamicContextGroup13 in Figure 24(a)).

- ▼ Changes
 - ⊗ Decrease staffing level below benchmark from 8% to 0% (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Establish Centralized Recruitment Process (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Ensure Security Clearance (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Collect Performance Objectives (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Assess Collected Objectives (Deactive) 10-Dec-2013 -> 31-Mar-2016
 - ⊗ Assess Collected Objectives (Element's Evaluation) 01-Apr-2016 -> 31-Dec-2017
 - ⊗ Improve IT contract renewal process (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Implement shift changes (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Conduct cost benefit analysis (Element's Evaluation) 30-Jun-2014 -> 31-Dec-2017
 - ⊗ Review sick leave plan for staffs who joined before 2001 (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2015
 - ⊗ Review sick leave plan for staffs who joined before 2001 (Element's Evaluation) 01-Jul-2015 -> 31-Dec-2017
 - ⊗ Identify & manage unusual sick leave patterns (Element's Evaluation) 10-Dec-2013 -> 31-Dec-2014
 - ⊗ Identify & manage unusual sick leave patterns (Element's Evaluation) 31-Dec-2014 -> 31-Dec-2017
 - ⊗ Review mandatory training requirements (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2015
 - ⊗ Review mandatory training requirements (Element's Evaluation) 30-Jun-2015 -> 31-Dec-2017
 - ⊗ Decrease staffing level below benchmark from 8% to 0% (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014
 - ⊗ Establish Centralized Recruitment Process (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014
 - ⊗ Ensure Security Clearance (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014
 - ⊗ Collect Performance Objectives (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014
 - ⊗ Improve IT contract renewal process (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014
 - ⊗ Implement shift changes (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014
 - ⊗ Conduct cost benefit analysis (Element's Evaluation) 10-Dec-2013 -> 30-Jun-2014

Figure 28: Changes in the Dynamic Context "DynamicContext (2014_Update)"

Finally, all the discussed changes are added to this dynamic context for the relevant model elements. Figure 28 shows all the changes in the dynamic context by expanding the “Changes” folder shown in Figure 24(a). The goal model is now ready to be evaluated at different timepoints and as a whole over time according to this set of changes. These analyses are discussed in later sections.

5.2.3 Changes after Follow-up Review as of 31st July, 2015

At the time of the follow-up review, OPG reported the status of all its recommendations along with the forecasts of any incomplete recommendations to the Auditor General, which was in turn published in the 2015 Audit Report [22]. Some of the leaf goals were still following the same behavior as expected on 30th June, 2014, while others’ behaviors changed. In the following paragraphs, we discuss the leaf goals whose behavior and forecast changed from the expectation.

Recommendation 2:

- OPG discovered that the leaf goal “Review Salary Levels & Employee Benefits” was still incomplete at the time of reporting. However, it forecasted that it would be able to achieve this goal by the end of December 2015. Thus, the following new change is required to illustrate that: (i) a LinearChange on its satisfaction value to 100 from 10th December, 2013, to 31st December, 2015.

Recommendation 3:

- OPG discovered that the leaf goal “Monitor contractor hours closely” was still incomplete at the time of reporting. However, it forecasted that it would be able to achieve this goal by the end of December 2015. Thus, it requires a linear change similar to the leaf goal “Review Salary Levels & Employee Benefits” of Recommendation 2.

Recommendation 5:

- OPG discovered that both the leaf goals “Review sick leave plan for staffs who joined before 2001” and “Identify & manage unusual sick leave patterns” were still incomplete at the time of reporting. However, it forecasted that it would be able to achieve these goals by the end of December 2015. Thus, both goals require a linear change similar to the leaf goal “Review Salary Levels & Employee Benefits” of Recommendation 2.

In jUCMNav, a dynamic context called “DynamicContext (Final)” is created to group the changes in this category. “EvaluationStrategy6”, defined previously, is included in this dynamic context as well to define the set of initial satisfaction values. Moreover, the dynamic context of previous category (DynamicContext (2014_Update)) is included in this context to reuse the changes that were still on track (this also includes the context included in the included context). Finally, all the discussed changes are added to this dynamic context for the relevant model elements. The goal model is now ready to be evaluated at different timepoints and as a whole over time according to this set of changes. These analyses are discussed in the following sections.

5.3 Evaluation of Goal Model at Timepoints

After adding the corresponding changes to the model elements, we now have a set of timed goal models that represent the system and its behavior over time completely. This section deals with the evaluation of those timed goal models at required timepoints by selecting the needed dynamic context. Before TimedGRL, the only way to do this was to manually update each element of the goal model and then, run the analysis, which was a cumbersome task.

To evaluate the goal models at certain dates, the modeler first needs to define timepoints corresponding to those dates. These timepoints must be added to a timepoint group, which can in turn be used to provide an overall evaluation. For this case study, 10th December, 2013, 30th June, 2014, and 31st July, 2015, can be considered as perfect timepoints as the model was updated on those dates (See Figure 24(b)). Along with these, after analyzing all the changes, we see that one of the model elements “Decrease staffing level below benchmark from 8% to 0%” in Recommendation 1 was expected to be satisfied by 31st December, 2017 (for DynamicContext

(2014_Update) and DynamicContext (Final)). Hence, we include the timepoints 31st December, 2015, and 31st December, 2017, as well (see Figure 24(b)) to be able to visualize the actual completion behavior.

“Timed GRL Strategy Algorithm” needs to be selected as the GRL Evaluation Algorithm in the preferences of the jUCMNav tool. The following sub-sections show the analysis of the system’s goal model at different timepoints for all the defined dynamic contexts. For comparison purposes, we show the evaluations of the goal model of Recommendation 1 and the comprehensive goal model for all sets of dynamic contexts and timepoints.

5.3.1 Evaluations on 10th December, 2013

On 10th December, 2013, none of the leaf goals are satisfied for any of the dynamic contexts. Hence, all the three dynamic contexts give the same evaluation at this timepoint (see Figure 29 and Figure 30).

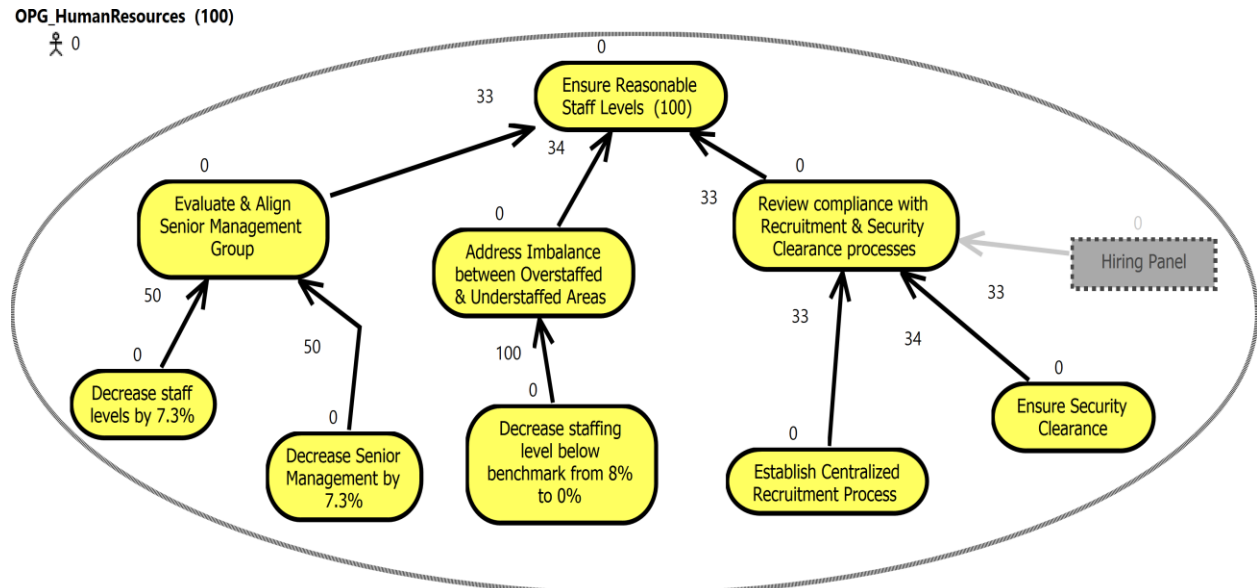


Figure 29: Evaluation of Recommendation 1 Model on 10th December, 2013

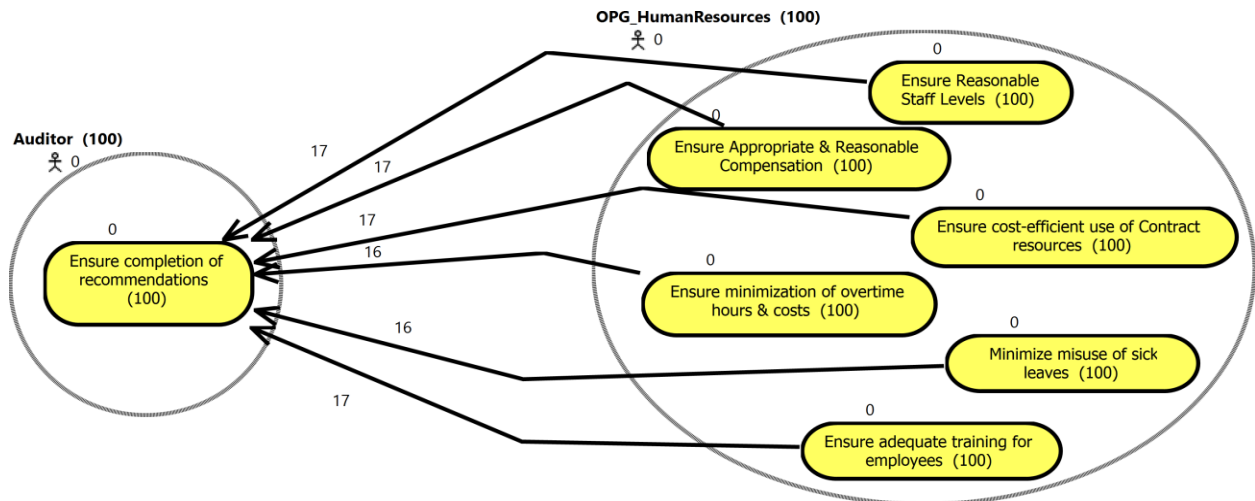


Figure 30: Evaluation of Comprehensive Model on 10th December, 2013

5.3.2 Evaluations on 30th June, 2014

Evaluations at this timepoint vary for different dynamic contexts due to the existence of disparate behavior among them for some of the recommendations. However, the evaluations of Recommendation 1 Model for the contexts “DynamicContext (2014_Update)” and “DynamicContext (Final)” are alike as no new changes are introduced in the latter for this particular recommendation. This evaluation is hence only shown once (see Figure 33).

DynamicContext (Expectation):

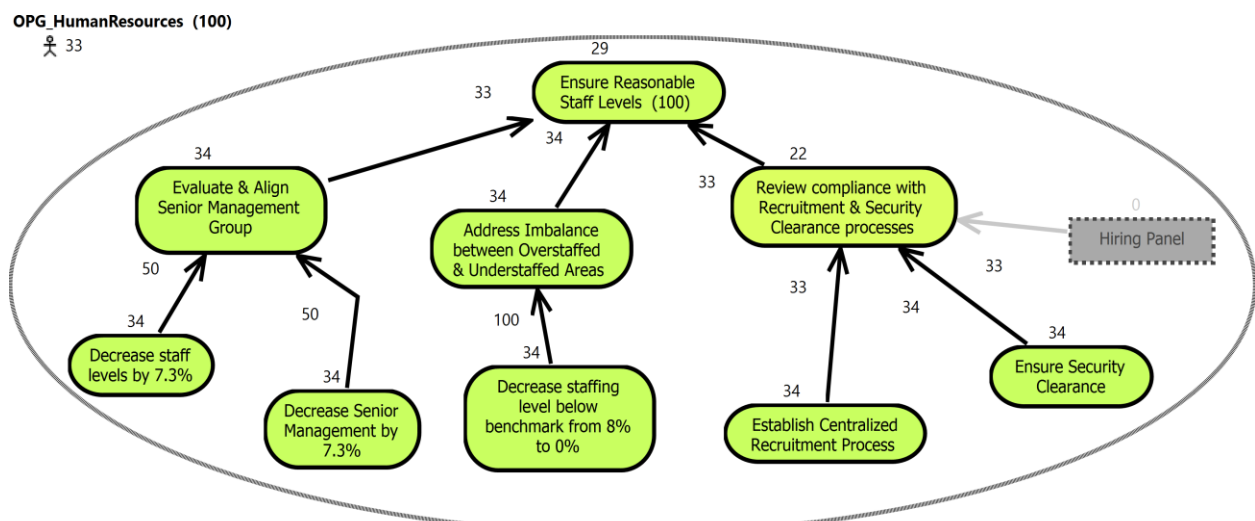


Figure 31: Evaluation of Recommendation 1 Model on 30th June, 2014 (DynamicContext (Expectation))

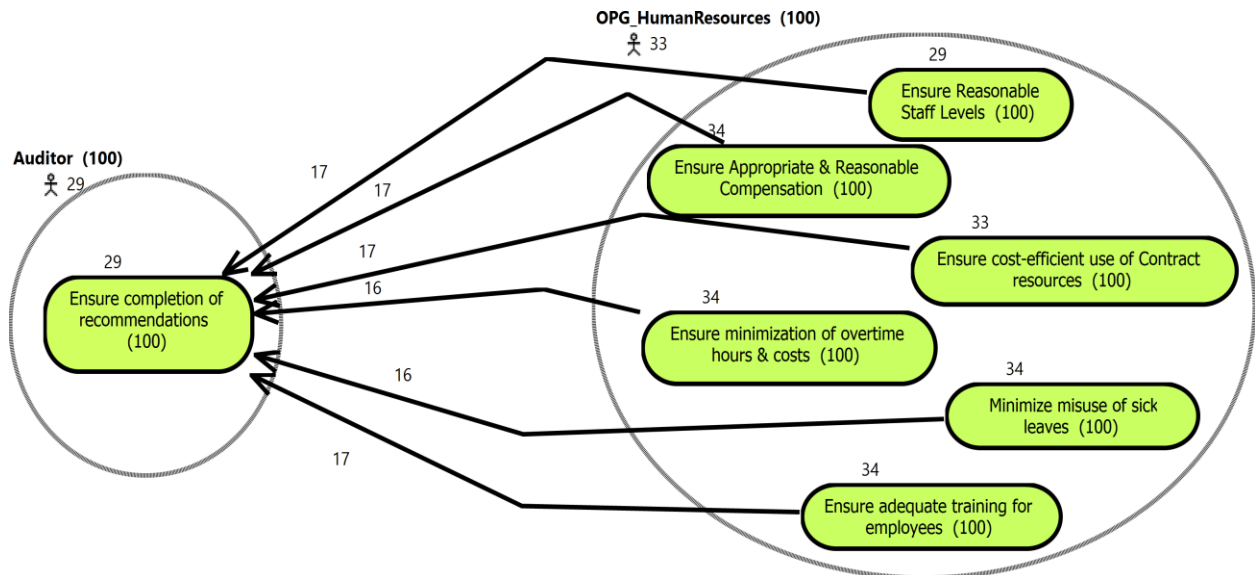


Figure 32: Evaluation of Comprehensive Model on 30th June, 2014 (DynamicContext (Expectation))

DynamicContext (2014_Update):

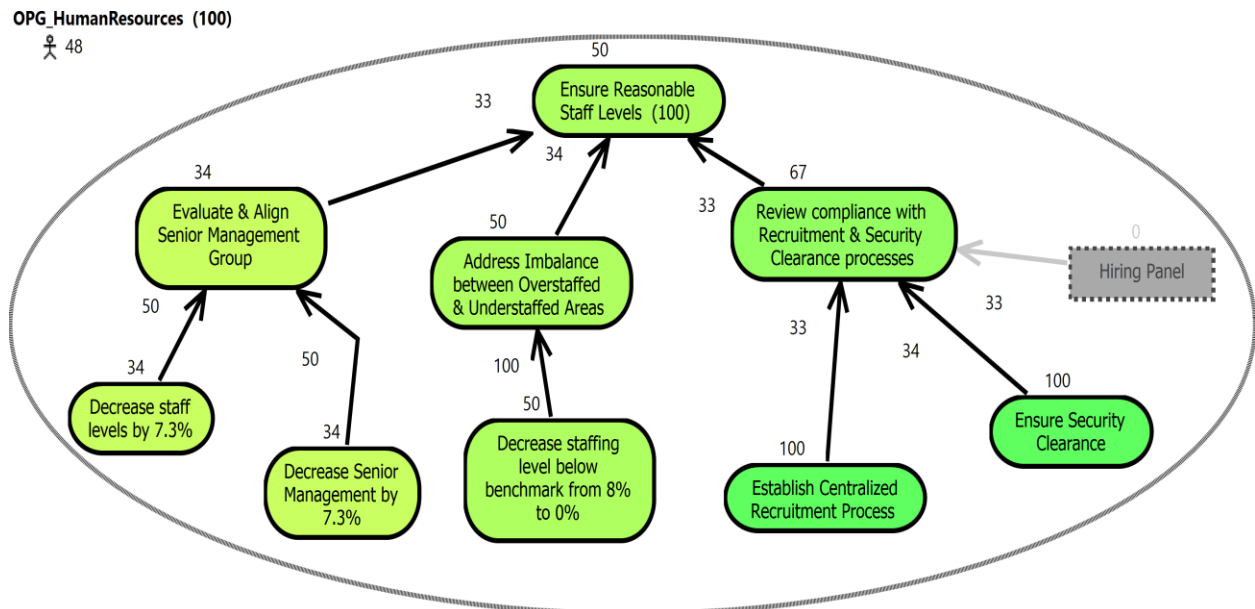


Figure 33: Evaluation of Recommendation 1 Model on 30th June, 2014 (DynamicContext (2014_Update))

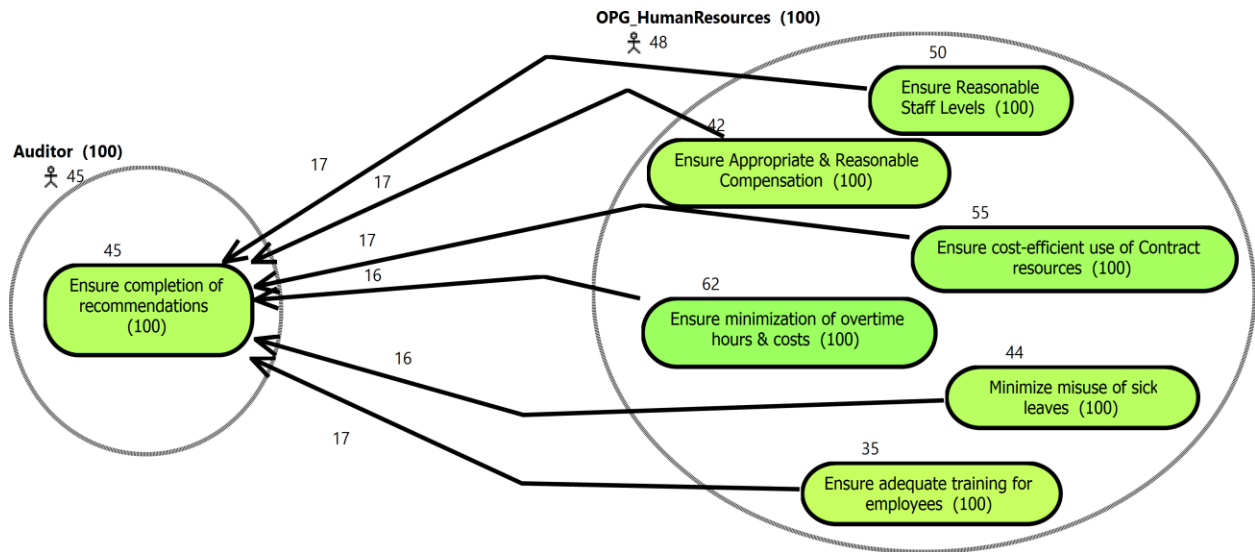


Figure 34: Evaluation of Comprehensive Model on 30th June, 2014 (DynamicContext (2014_Update))

DynamicContext (Final):

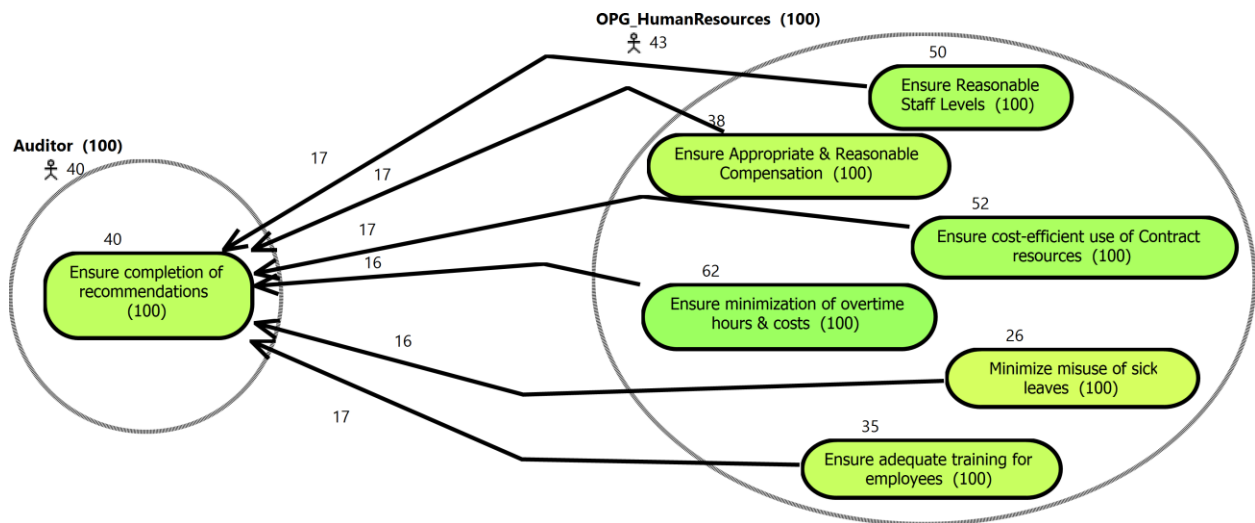


Figure 35: Evaluation of Comprehensive Model on 30th June, 2014 (DynamicContext (Final))

5.3.3 Evaluations on 31st July, 2015

Evaluations at this timepoint vary for different dynamic contexts. As per expectation, the evaluation for “DynamicContext (Expectation)” shows a fully satisfied system, and this evaluation is going to be consistent for every timepoint after this date. Similar to the previous timepoint, the evaluations of Recommendation 1 Model for the contexts “DynamicContext (2014_Update)” and “DynamicContext (Final)” are alike. This evaluation is hence only shown once (see Figure 38).

DynamicContext (Expectation):

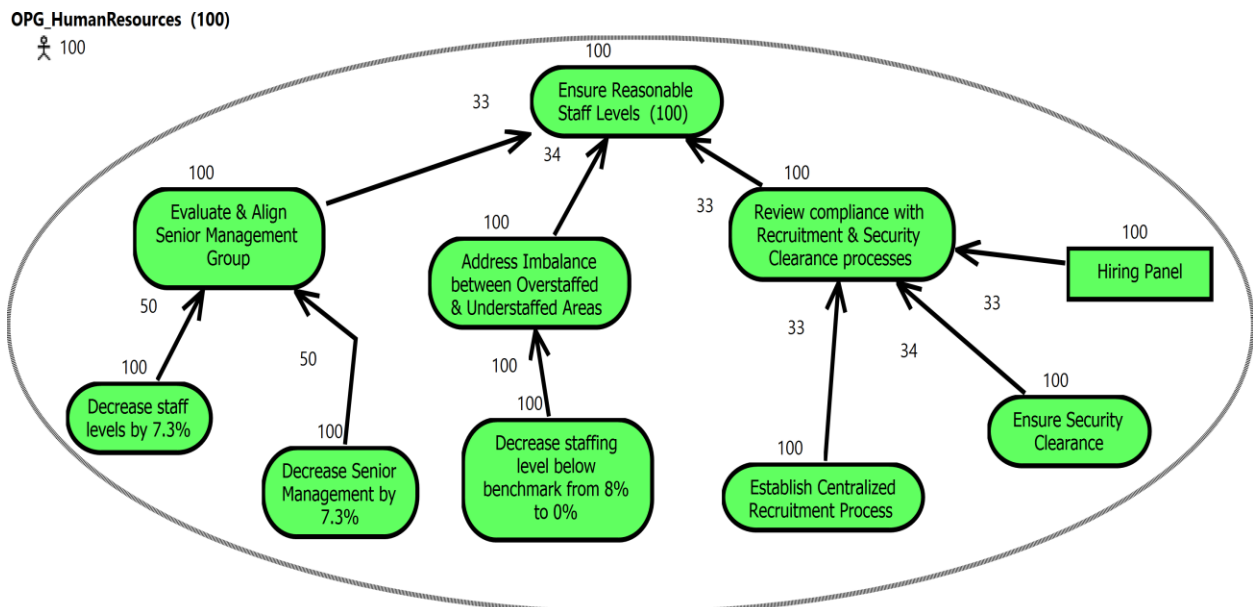


Figure 36: Evaluation of Recommendation 1 Model on 31st July, 2015 (DynamicContext (Expectation))

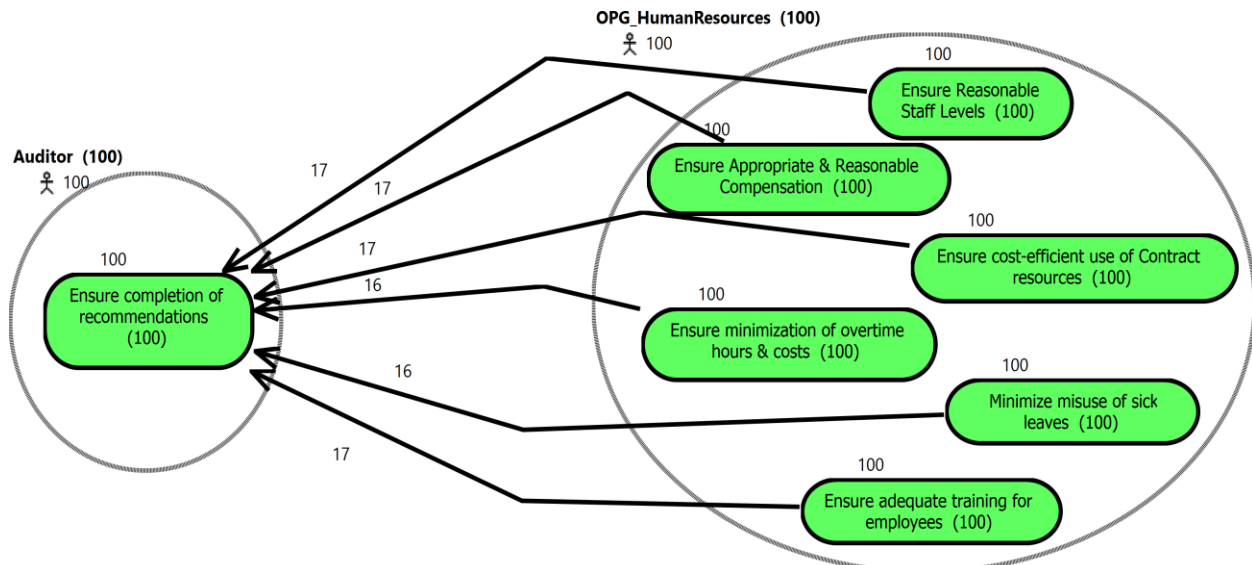


Figure 37: Evaluation of Comprehensive Model on 31st July, 2015 (DynamicContext (Expectation))

DynamicContext (2014_Update):

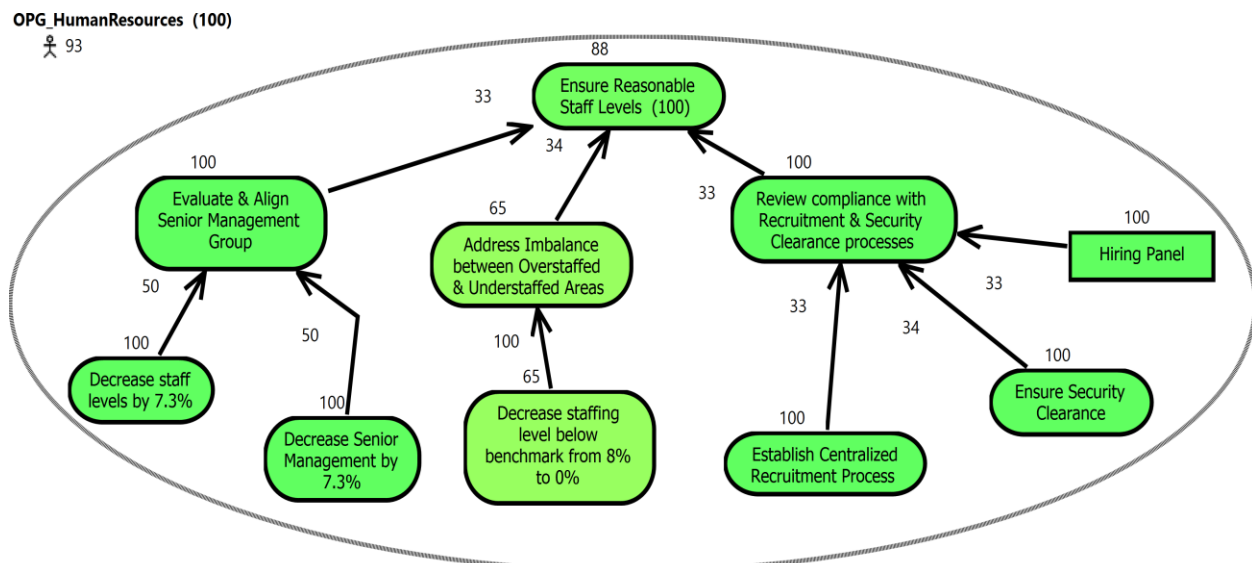


Figure 38: Evaluation of Recommendation 1 Model on 31st July, 2015 (DynamicContext (2014_Update))

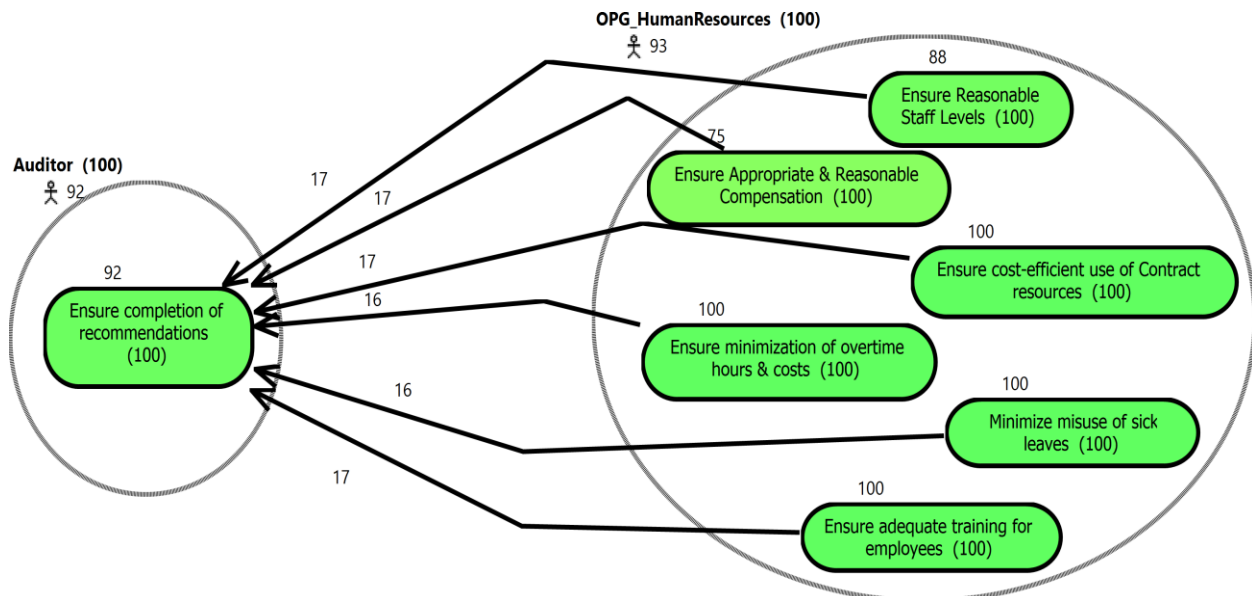


Figure 39: Evaluation of Comprehensive Model on 31st July, 2015 (DynamicContext (2014_Update))

DynamicContext (Final):

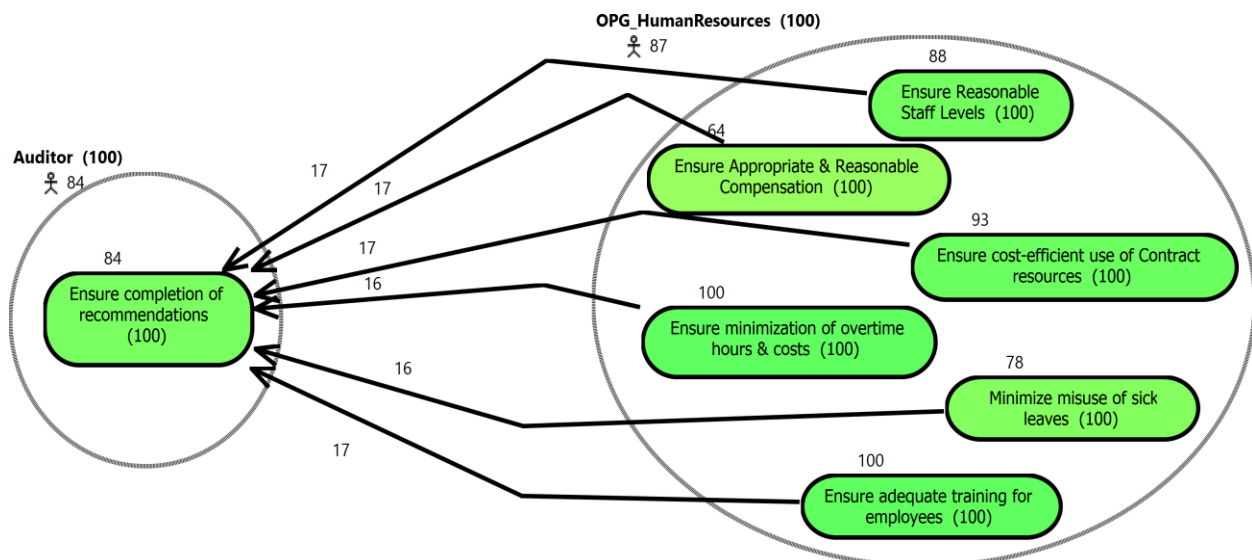


Figure 40: Evaluation of Comprehensive Model on 31st July, 2015 (DynamicContext (Final))

5.3.4 Evaluations on 31st December, 2015

Evaluations for the contexts “DynamicContext (2014_Update)” and “DynamicContext (Final)” give the same result as all the differences in their behavior were only up to this date.

DynamicContext (2014_Update) and DynamicContext (Final):

OPG_HumanResources (100)

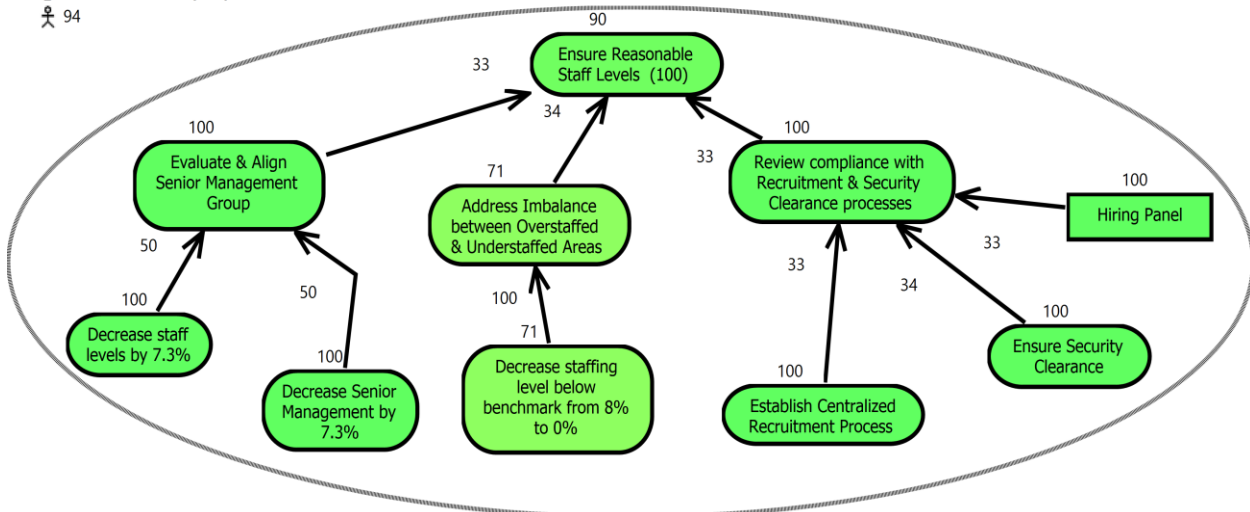


Figure 41: Evaluation of Recommendation 1 Model on 31st December, 2015

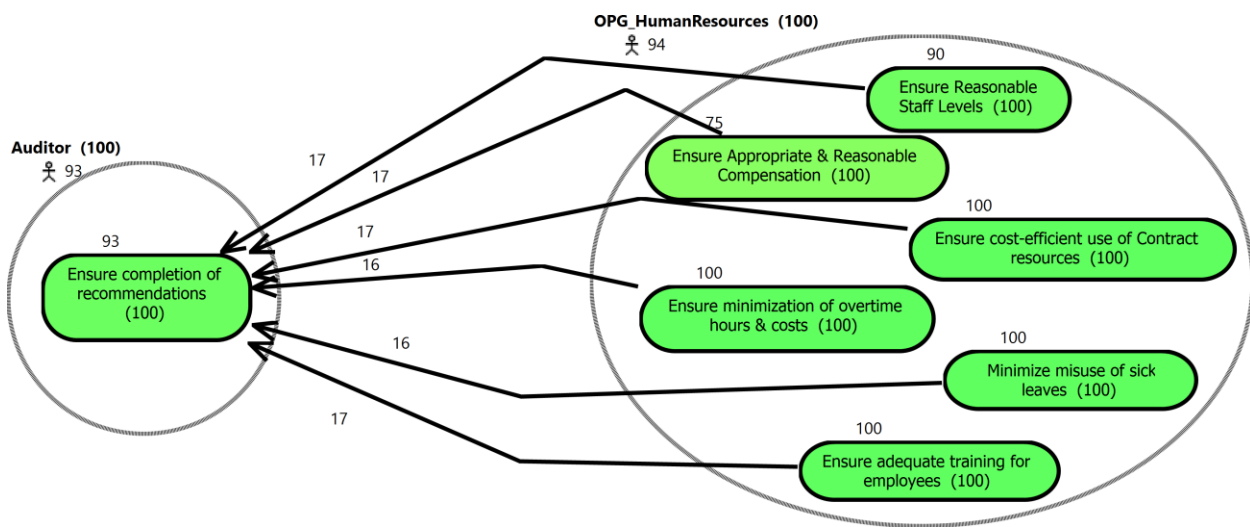


Figure 42: Evaluation of Comprehensive Model on 31st December, 2015

5.3.5 Evaluations on 31st December, 2017

This timepoint marks the full satisfaction of the system for the two dynamic contexts “DynamicContext (2014_Update)” and “DynamicContext (Final)” and hence, shows evaluations similar to Figure 36 and Figure 37.

5.3.6 Discussion

The satisfaction level of the goal “Ensure completion of recommendations” of the comprehensive goal model may give an overall idea of the actual status of the system as all the recommendations contribute to it in a balanced way. The observations at different timepoints for different dynamic contexts focusing on the mentioned goal can be summarized as follows:

- In the beginning, the satisfaction level is 0 as expected (see Figure 30).
- After 6 months, the satisfaction level is expected to be 29 (as per changes defined in DynamicContext (Expectation)) (see Figure 32). However, the actual satisfaction level reported on 30th June, 2014, is 46 (as per changes defined in DynamicContext (2014_Update)), which shows OPG was ahead of its target (see Figure 34) at the six months’ mark. On the other hand, OPG had new forecasts for some of the recommendations, which is updated in the model under a new category.
- At the time of the follow-up, the satisfaction level should have been 100 as per the expectation defined at the time of the audit (see Figure 37), while it should have been 92 according to OPG’s updated model of 2014 (see Figure 39). However, the actual satisfaction level is found to be 84, which shows that OPG was behind its expected schedule (see Figure 40). In addition, OPG had some new assumptions for its recommendations, which is updated in the model under a new category.

- Currently, the satisfaction level is expected to be 100 by 31st December, 2017 as per expectations on both 30th June, 2014, and 31st July, 2015.

Hence, this single model is able to encapsulate information about expectation as well as actual findings. It allows us to add behaviors in the model in different contexts, and then it can be used for various purposes such as reporting, validation, and comparison. Moreover, the option to reuse changes from included dynamic contexts provides a more organized, faster, and less-faulty way to define similar behaviors in different contexts. This allows modelers to concentrate on the new changes rather than describing existing changes again and again for different contexts. Additionally, using a single model at all stages of a project requires less maintenance than creating and handling multiple models representing different stages. This case study showcases all these advantages of using TimedGRL for the modeling of evolving systems.

However, until now, we have seen the analysis of the system taking only one date into account at a time. In the next section, we will see how TimedGRL supports visualization of system behavior over a period of time, hence providing us with a whole picture.

5.4 Overall Evaluation of Goal Model over a Timepoint Group

Along with the evaluation of a system at different timepoints using the same goal model, TimedGRL also introduces the concept of overall visualization of a system's status over a period of time. This has been made possible by the introduction of TimepointGroup, which is a collection of timepoints. This notion is also integrated in the jUCMNav tool.

To evaluate the goal models over time, the modeler first needs to define a timepoint group, which in turn has a collection of timepoints. As discussed in Section 4.3, a model is

evaluated from the earliest date in the timepoint group to the latest date. E.g., if a timepoint group contains the following timepoints: 1st January, 2010; 1st January, 2020; 31st December, 2031; it means selecting this timepoint group will return an evaluation over the period ranging from 1st January, 2010, to 31st December, 2031. The timepoints in concern for this case study are 10th December, 2013, 30th June, 2014, 31st July, 2015, 31st December, 2015, and 31st December, 2017 (Section 5.3). Studying the time-period covered by these specified timepoints should give a comprehensive review of how the system was expected to behave, and how it actually behaved until it reached its maximum satisfaction level. Hence, we choose the TimepointGroup of the mentioned timepoints as our input (See Figure 24(b)).

The next step is to choose an evaluation frequency, i.e., the rate at which we prefer the model to be analyzed. By default, this is 1 day, which means that analysis is done for all the days that lie between the earliest and the latest dates of the chosen timepoint group and the same is visualized in the heatmap and the charts. However, in this case study, the overall observation period ranges from 10th December, 2013, to 31st December, 2017, which gives a period of 4 years. So, a daily analysis is not preferable in this case as it would be time consuming and result in a large heatmap. Thus, we decide to use a frequency (also known as granularity) of 30 days, and update the same in the preferences of the jUCMNav tool (see Figure 43). The following subsections show the comprehensive evaluation of the system's goal model for different dynamic contexts in the selected timepoint group.

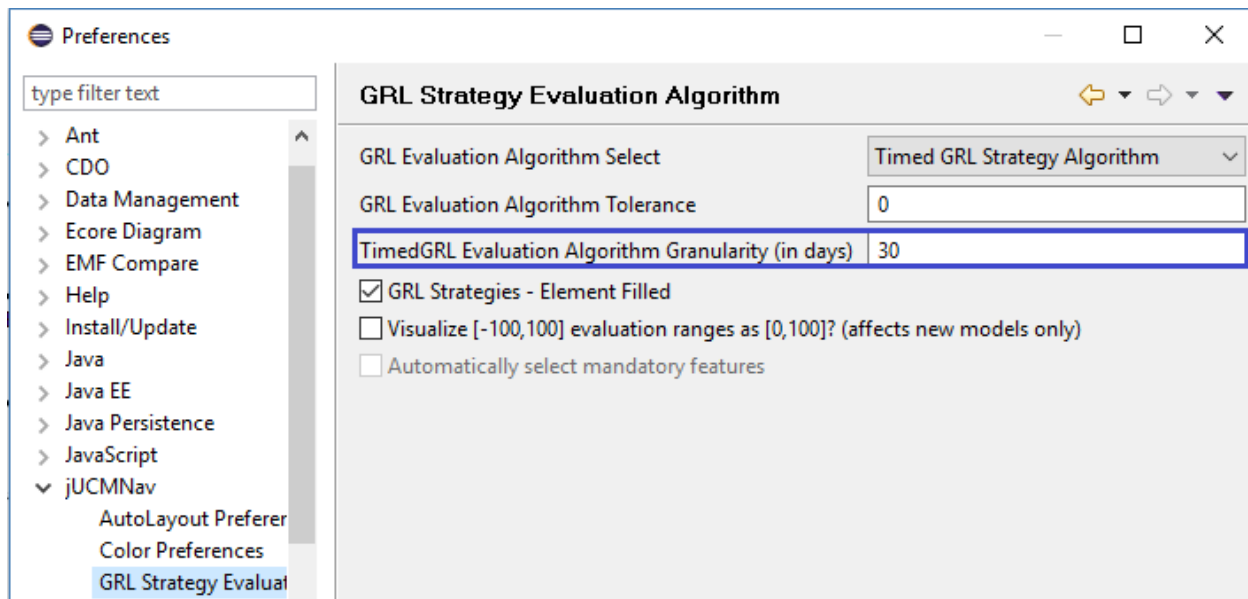


Figure 43: Updating Granularity in jUCMNav Preferences

5.4.1 Overall System Behavior for “DynamicContext (Expectation)”

The changes in this DynamicContext are based on the expectation of OPG as on 10th December, 2013, as discussed in Section 5.2.1. To summarize, OPG expected all the recommendations to be completed by 31st July, 2015. Running an overall evaluation using this dynamic context over the observation period defined by the timepoint group ranging from 10th December, 2013, to 31st December, 2017, generates the heatmap as shown in Figure 44. In the heatmap, all the intentional elements and actors are listed as per their hierarchies (parent-child relationship). Thus, the two actors, i.e., OPG_HumanResources and Auditor, are visualized in this heatmap. Moreover, all their children intentional elements are also listed. Finally, the topmost level is the system itself representing this full case study. As discussed in Section 4.3, the color-coded bar next to each element shows the variation in that element’s satisfaction value. This visualization gives a comprehensive idea of how the status of the system is changing over time.

To get a closer and more-detailed look of this variation, the chart visualization option, provided by jUCMNav to depict the variation in the satisfaction value of each individual intentional element, each actor, and the overall system, is used. Moreover, in this case study, these charts demonstrate the participation of contribution links in the computation of an intentional element's satisfaction value. The arrow button before each element, actor, or system in the heatmap provides access to its corresponding charts.

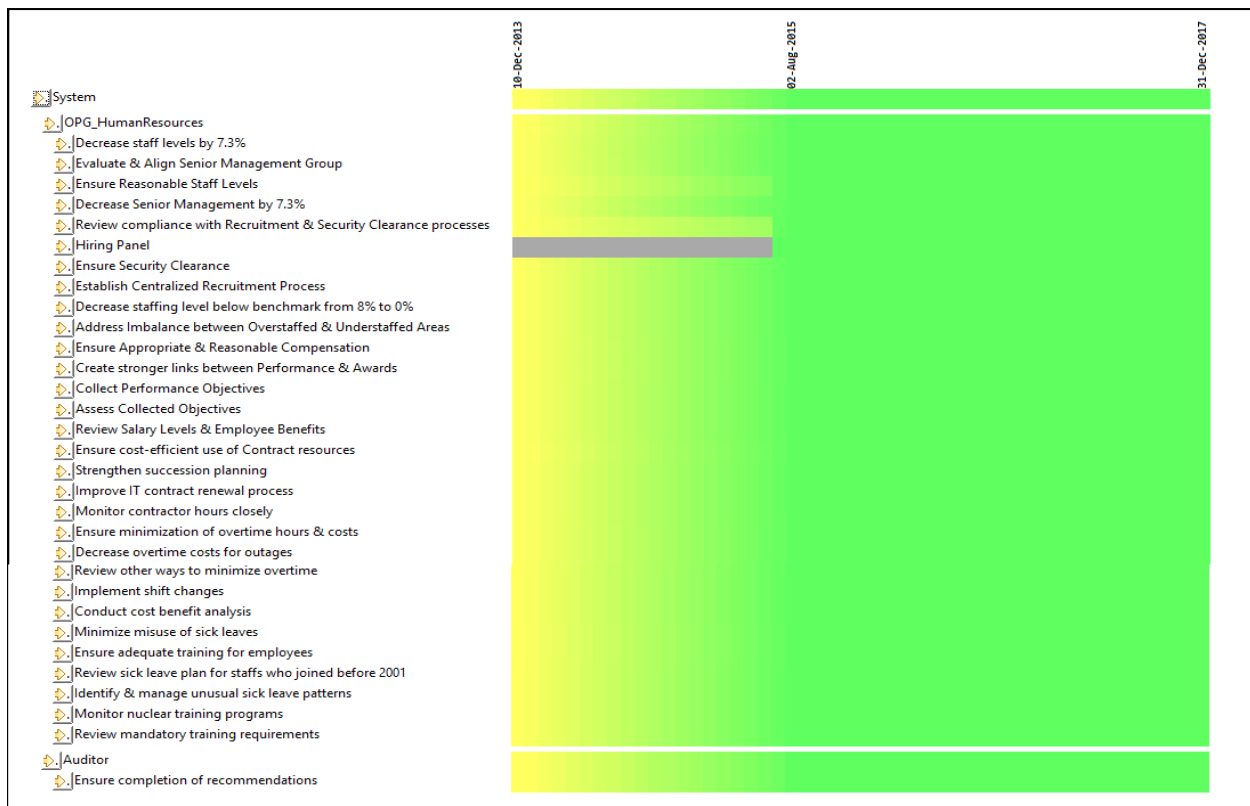


Figure 44: Heatmap for DynamicContext (Expectation)

Figure 45 depicts the chart for the variation in the satisfaction value and importance of the goal “Ensure completion of recommendations” over the same timepoint group as the heatmap. The importance of the goal to its parent actor does not change, however the satisfaction value increases linearly at first until it is satisfied fully by 31st July, 2015, and then remains fulfilled for the rest of the period. To further investigate this, we look at the chart shown in

Figure 46, which shows the effects of the individual contribution links on this goal's satisfaction value. All the six goals corresponding to six recommendations, that are feeding this goal, contribute equally, and the same is reflected in this chart.

Similarly, it is possible to look into the charts of individual elements, actors, and the system itself. Figure 47 shows the overall behavior of this system as expected at the time of audit.

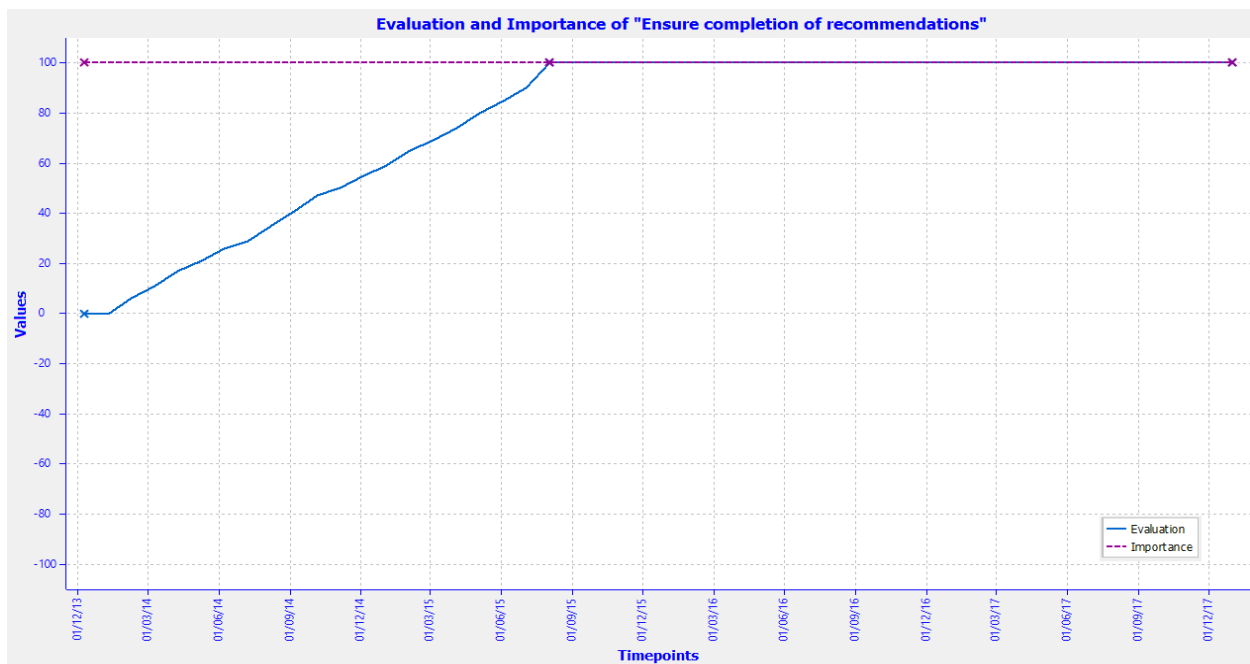


Figure 45: Variation in Evaluation and Importance for DynamicContext (Expectation) (for mentioned goal)

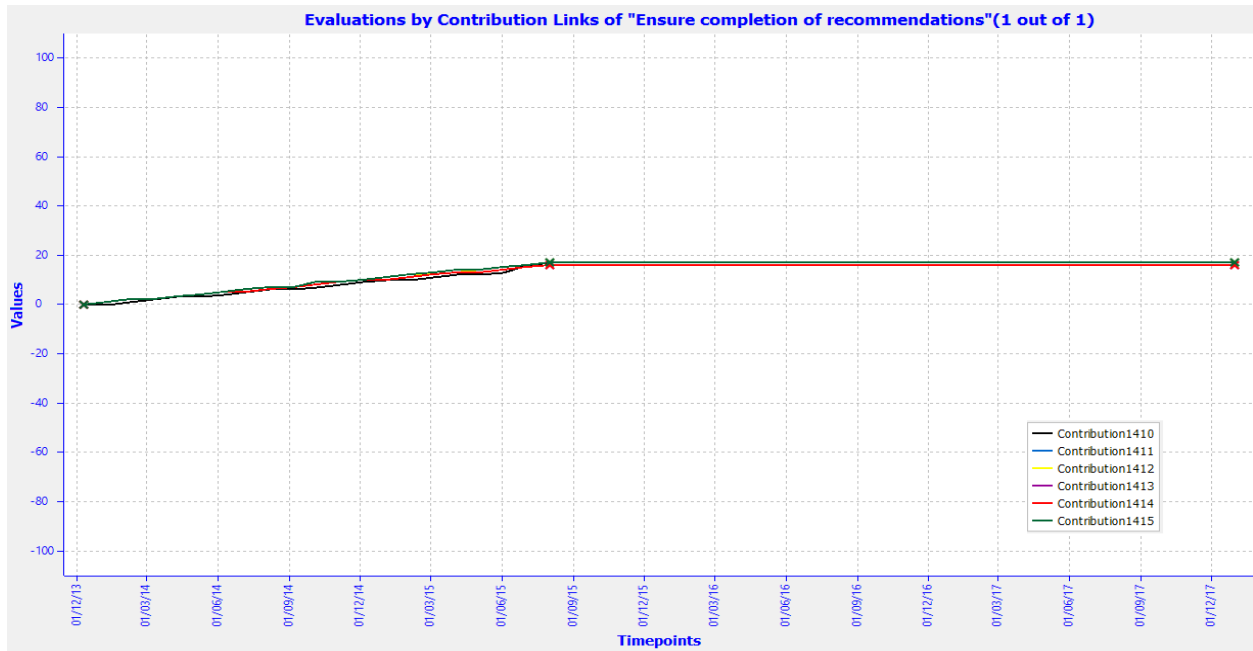


Figure 46: Variation in Evaluation by Contribution Links for DynamicContext (Expectation) (for mentioned goal)

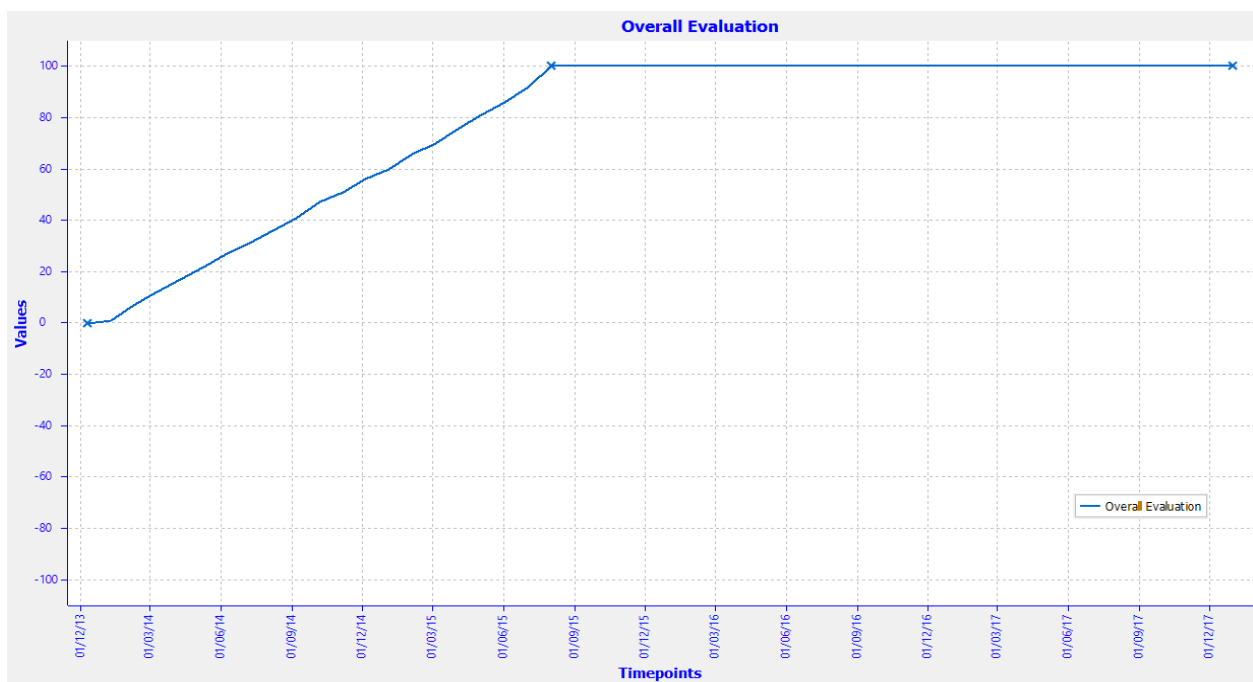


Figure 47: Behavior of the system for DynamicContext (Expectation)

5.4.2 Overall System Behavior for “DynamicContext (2014_Update)”

The changes in this DynamicContext are based on the actual status as well as forecasts of OPG as of 30th June, 2014, as discussed in Section 5.2.2. To summarize, some of the recommendations were on track according to previous expectations, while others were expected to behave differently. Running an overall evaluation using this dynamic context over the observation period defined by the timepoint group ranging from 10th December, 2013, to 31st December, 2017, generates the heatmap as shown in Figure 48.

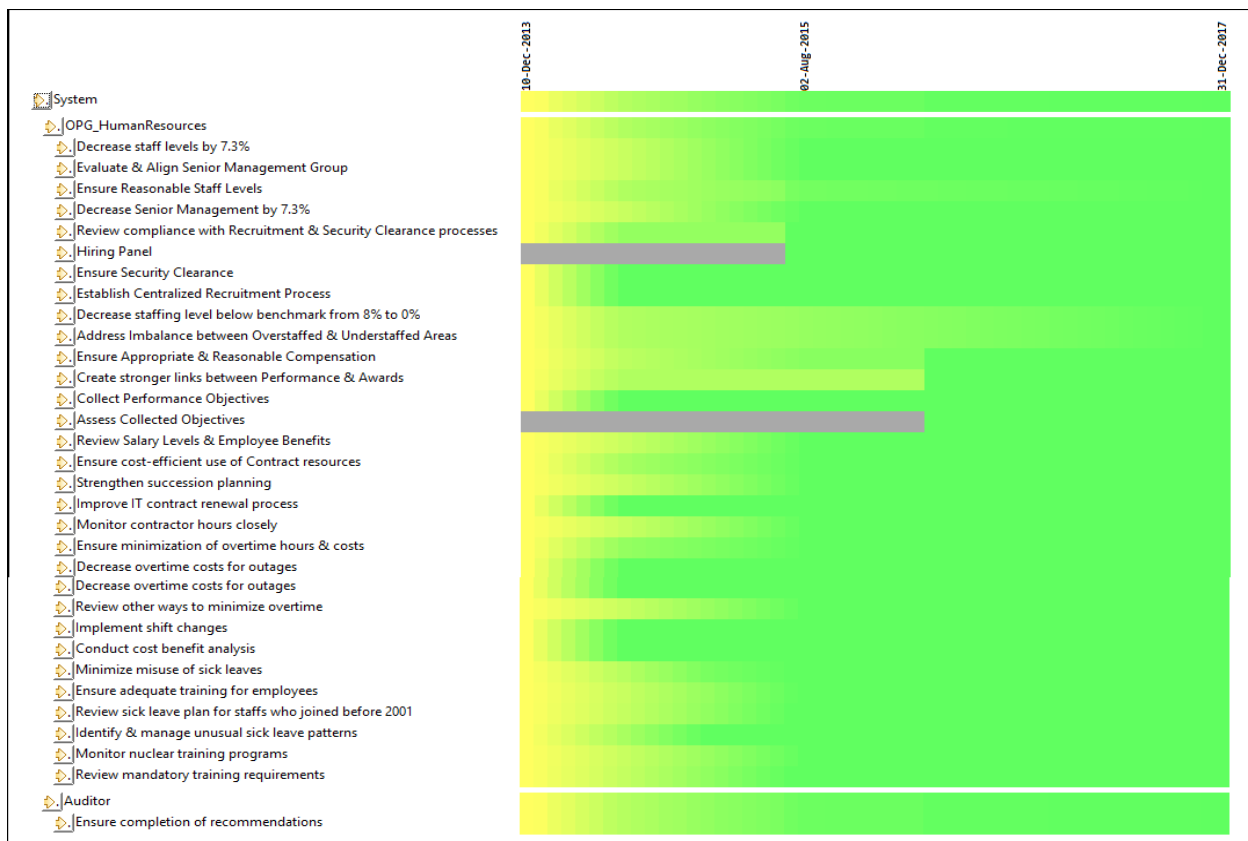


Figure 48: Heatmap for DynamicContext (2014_Update)

Figure 49 depicts the chart for the variation in satisfaction value and importance of the goal “Ensure completion of recommendations” over the same timepoint group as the heatmap.

The importance of the goal to its parent actor does not change, however the satisfaction value increases at first at a faster rate, then, the growth rate becomes slower around 31st July, 2015, and only reaches a full satisfaction value on 31st December, 2017. To further investigate this, we look at the chart shown in Figure 50, which shows the effects of the individual contribution links on this goal's satisfaction value. Figure 51 shows the overall behavior of this system as expected after the 6-months status report.

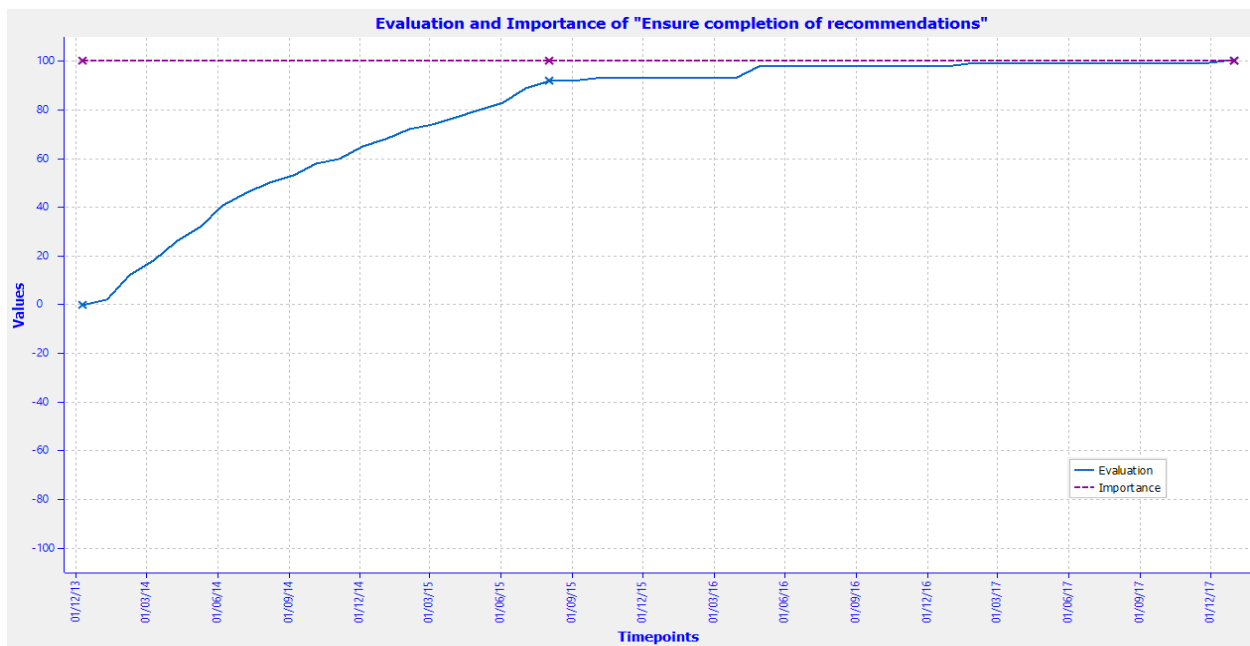


Figure 49: Variation in Evaluation and Importance for DynamicContext (2014_Update) (for mentioned goal)

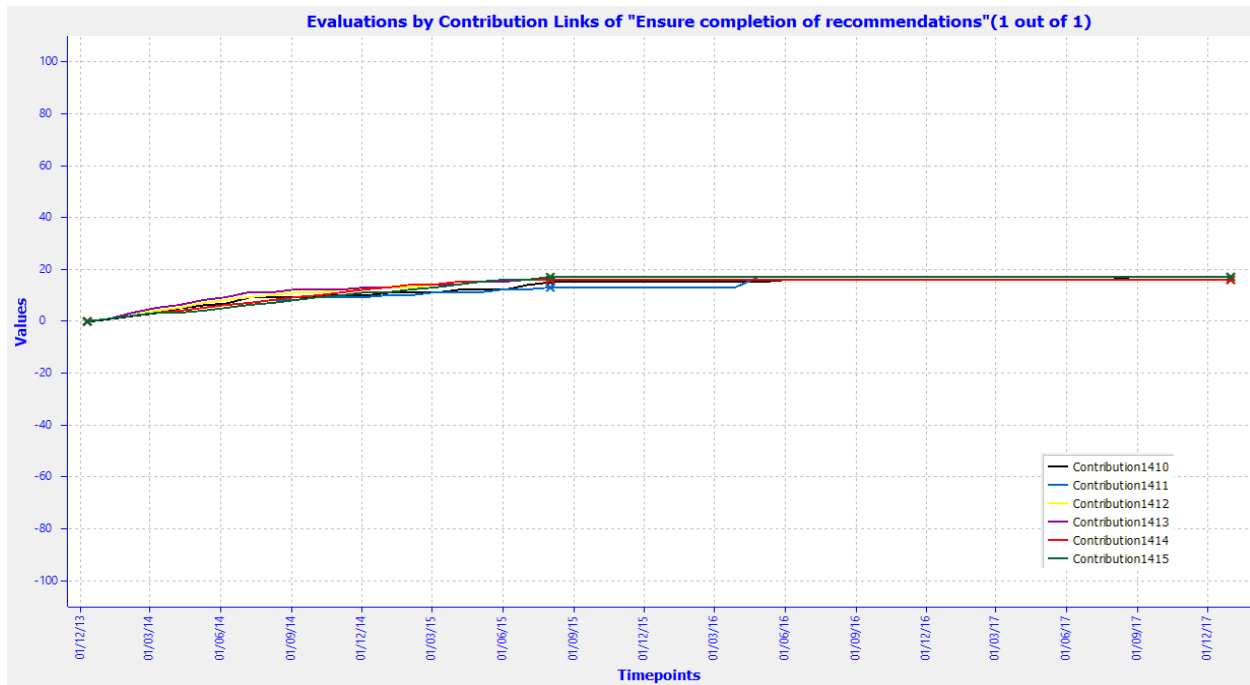


Figure 50: Variation in Evaluation by Contribution Links for DynamicContext (2014_Update) (for mentioned goal)

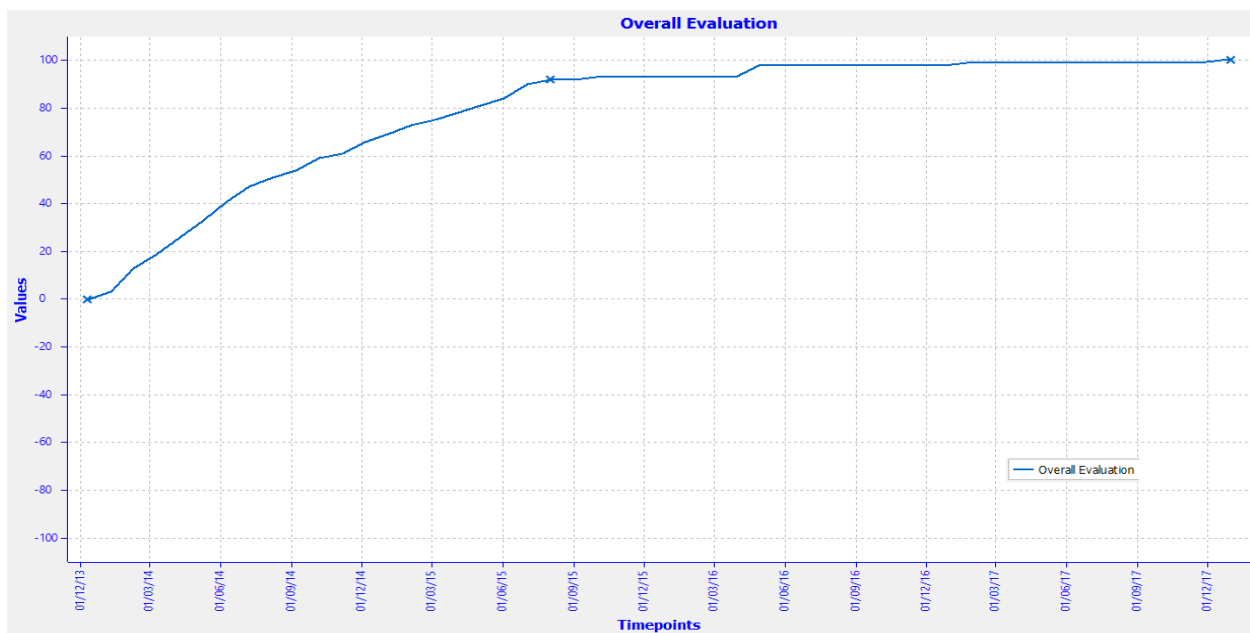


Figure 51: Behavior of the system for DynamicContext (2014_Update)

5.4.3 Overall System Behavior for “DynamicContext (Final)”

The changes in this DynamicContext are based on the actual status as well as forecasts of OPG as of 31st July, 2015, as discussed in Section 5.2.3. These changes reflect the actual status of the system at the time of the follow-up review along with the updated forecasts for uncompleted recommendations. Running an overall evaluation using this dynamic context over the observation period defined by the timepoint group ranging from 10th December, 2013, to 31st December, 2017, generates the heatmap as shown in Figure 52.



Figure 52: Heatmap for DynamicContext (Final)

Figure 53 depicts the chart for the variation in satisfaction value and importance of the goal “Ensure completion of recommendations” over the same timepoint group as the heatmap.

This chart is similar to the one generated using DynamicContext (2014_Update) as only a few low-level goals changed their behavior from the forecasts added in 2014. To further investigate this, we look at the chart shown in Figure 54, which shows the effects of individual contribution links on this goal's satisfaction value. Figure 55 shows the overall expected behavior of this system at the time of follow-up review, which is almost similar to the chart generated using DynamicContext (2014_Update).

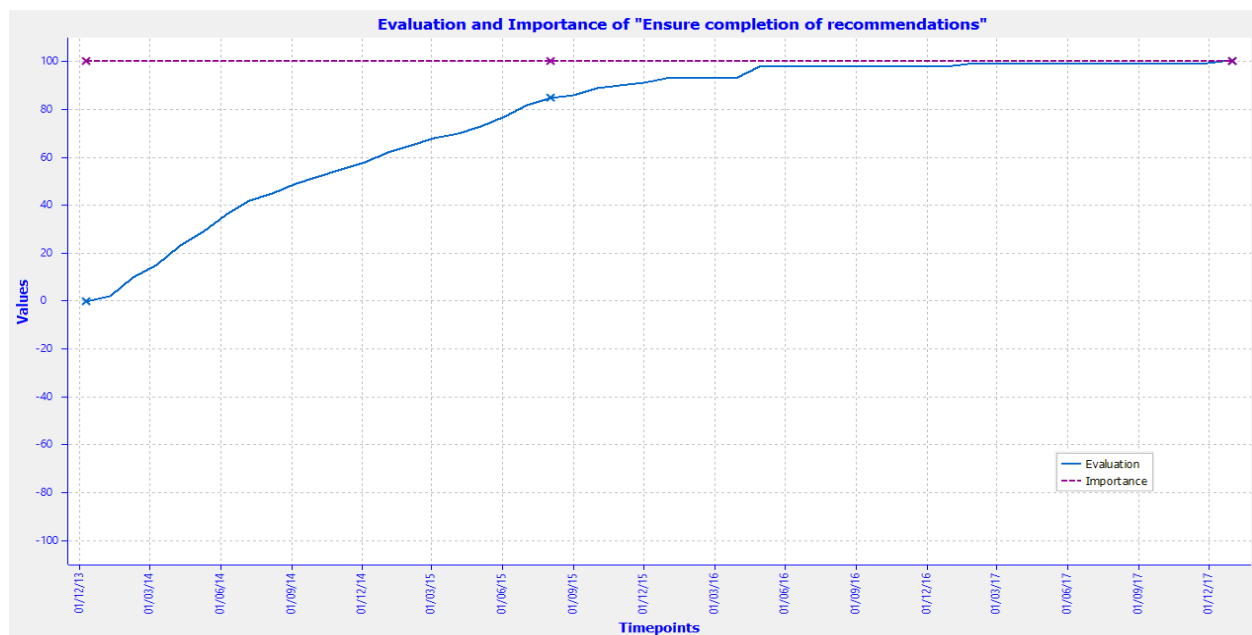


Figure 53: Variation in Evaluation and Importance for DynamicContext (Final) (for mentioned goal)

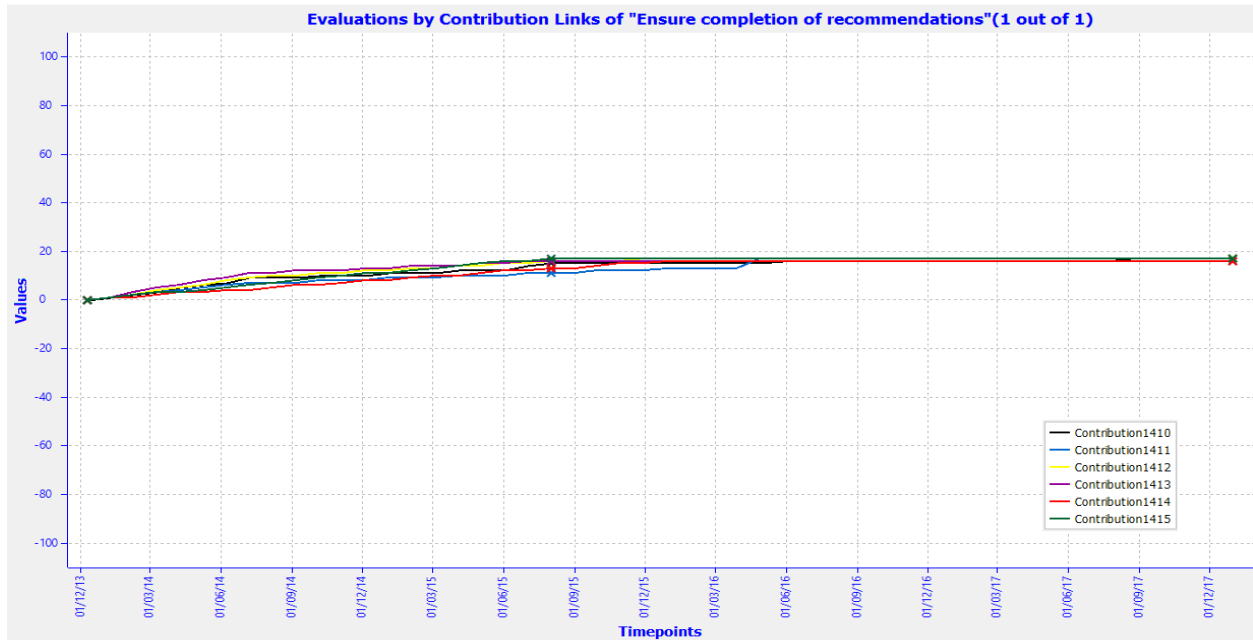


Figure 54: Variation in Evaluation by Contribution Links for DynamicContext (Final) (for mentioned goal)

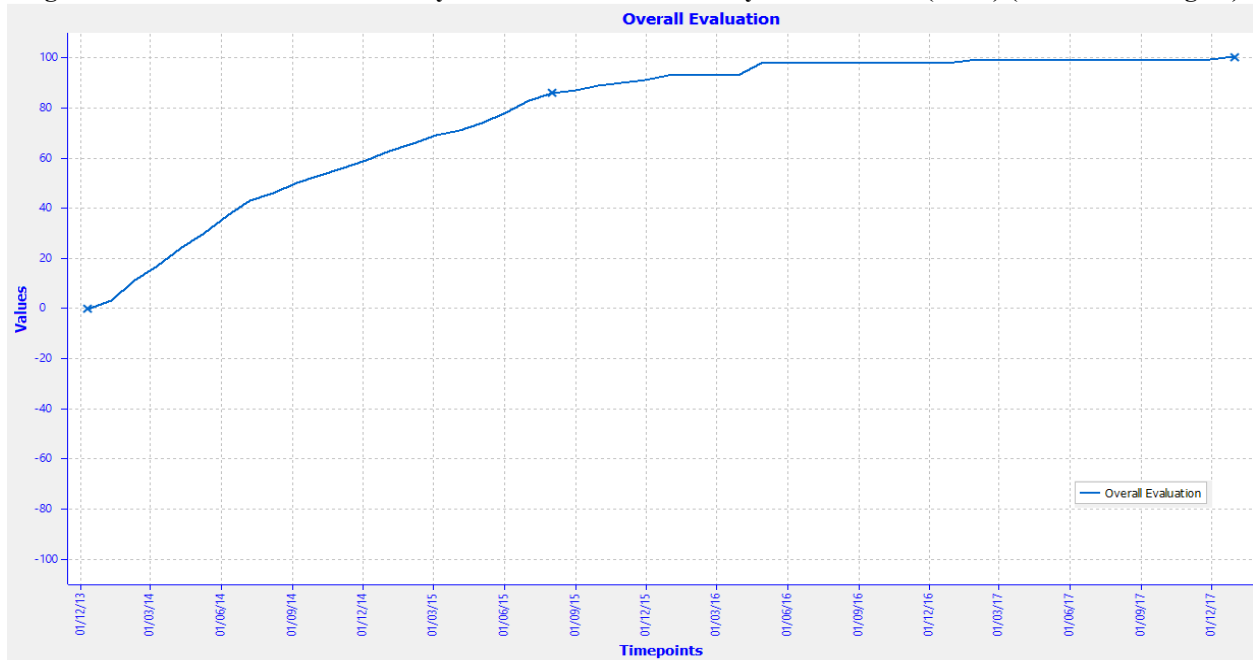


Figure 55: Behavior of the system for DynamicContext (Final)

5.4.4 Discussion

The heatmaps and the individual charts convey the overall changing behavior of the system represented by the goal model. After running the evaluation over different dynamic contexts for the concerned period and comparing them, modelers can analyze where the system

is expected to be and where it actually stands. At the time of the audit, OPG expected to fulfill all the recommended items before the follow-up review and hence, the overall evaluation shows a linear increase until the maximum by 31st July, 2015 (Figure 47). However, a six-month status report updated the expected behavior, which is shown in Figure 51 as an increase at a faster rate than expected at the time of the status update (June, 2014), yet almost 1.5 years shift in the date, when full satisfaction is forecasted. At the time of follow-up review, except for a few low-level goals, the system followed the behavior expected after six-month status update. The same is represented in Figure 55, which is almost similar to Figure 51, except in the time period ranging from 1st June, 2015, to 31st December, 2015.

Hence, a timed goal model provides the ability to visualize the represented system's behavior over a period of time. This enables the modelers to test their solutions and see how these solutions are going to affect the system in the future. Not only that, it allows the modelers to compare how different solutions are going to pan out by defining different dynamic contexts. Moreover, there are two ways offered to visualize the system's behavior over time: A heatmap, providing a color-coded abstract representation of variation in system's behavior, and a line chart, providing a more detailed representation of that variation. As seen in this case study, having access to both these visualization techniques gives the modeler a comprehensive outlook on the system and facilitates the discussion of the validity of the proposed solution. Thus, it can prove to be a light-weight, yet powerful way to describe the behavior of an evolving system in the requirements phase.

5.5 Summary

This chapter describes a real-life case study based on the Audit Report on Ontario Power Generation's Human Resources and converts it into a goal model by identifying actors, high-level goals, tasks, resources, and dependencies amongst them. Next, the changes are identified for this case study and a step by step explanation is provided to demonstrate how changes are added to the model using the jUCMNav tool. Moreover, these timed goal models are analyzed over time using jUCMNav and the evaluation results are presented in detail. Additionally, a discussion is provided explaining the evaluation results and how those results help in better understanding of the overall system's behavior. It discusses the advantages of using a single model to capture expectations and actual findings, and how this model can be used for reporting, validation, and comparison purposes. It also examines how visualizing system's behavior over time provides a comprehensive view of how the expected solution is going to pan out. Hence, this case study demonstrates the successful implementation of TimedGRL in jUCMNav as well as its usefulness in real world scenarios.

The next chapter explores the existing goal modeling languages closest to GRL, and the general analysis approaches used today. Moreover, it looks into some of the work that support model analysis over time and their comparison with TimedGRL.

CHAPTER 6: Related Work

There are many approaches that deal with the analysis of goal models [10]. Currently, however, most of those analyses focus on a single snapshot in time. Only a few among them address the issue of evolving systems by capturing changes and analyzing goal models over longer periods of time. This chapter first discusses goal modeling languages closest to GRL, and then, categorizes and discusses some of the general analysis approaches present today. Finally, it explores some of the work that is most closely related to TimedGRL.

Goal modeling languages, that are closest to GRL, include i^* , the NFR Framework, and KAOS. The i^* framework [29] is based on the intentional characteristics of the participants, which are known as agents. Similar to actors in GRL, agents possess intentional properties such as goals, beliefs, abilities, and commitments. Agents can depend on each other to achieve goals, to perform tasks, and to supply resources. This framework proposes the use of two types of models: the Strategic Dependency (SD) model represents the intentional level and the Strategic Rationale (SR) model represents the rational level, obtained by refining the SD model with reasoning capabilities. The OpenOME tool [28] is available for creating and analyzing i^* models. A qualitative reasoning approach is used to analyze dependency networks. However, this evaluation is static in nature and currently, there is no support to analyze a model over time as TimedGRL does.

The NFR Framework [4] provides structured graphical facilities for expressing NFRs (Non-Functional Requirements). These graphs capture the interdependencies of NFRs, which are represented as ‘soft-goals’. A qualitative propagation mechanism is used to analyze the impact of

decisions by determining the satisfaction level of NFRs. However, it is not possible to analyze and manage the changes in these graphs over time.

KAOS [30] is a goal-oriented requirements engineering language that provides a method for goal-driven requirements elaboration. This method allows a combination of different levels of expression and rationale: semiformal for modeling and structuring goals, qualitative for alternative selections and formal for the critical elements [31]. A quantitative algorithm can be used to evaluate the partial satisfaction of goals by computing the weighted average of the sub-goals' satisfaction. This is the quantitative version of the NFR qualitative algorithm. However, similar to i^* and the NFR Framework, this language also does not support specification and analysis of system evolution. TimedGRL introduces the concept of changes over time to support evolution specification in a goal model.

There are two categories in which the general goal model analysis approaches can be classified:

- Bottom-up approach: The analyses following this approach require a set of initial satisfaction values for the leaf elements in the goal model, which in turn is used to calculate high-level satisfaction values of stakeholder goals and system qualities. This approach is ideal to answer “what if” questions.
- Top-down approach: As the name suggests, the analyses following this approach operate in the opposite direction of the previous approach. In this case, possible satisfaction values for leaf elements are determined, that can achieve the desired satisfaction value of high-level elements. Usually, this approach requires

constraint-based techniques. E.g., Luo and Amyot [14] propose a declarative semantics for GRL based on a constraint-oriented interpretation of goal models that provides a more generic method to facilitate the analysis and optimization of goal models by using constraint solvers.

In some of the work, goals are defined with temporal logic. E.g., Letier et al. [13] extend the existing temporal logic to be able to model timed properties of discrete-time event-based models as well as make model checking of timed properties possible. Other work looks at conditional non-monotonic temporal beliefs and goals [7].

Work done by Grubb and Chechik [8][9] is most closely related to TimedGRL. It introduces the concept of dynamism in goal models by proposing an approach which allows specification of changes in intentions over time to capture evolution. Moreover, the functionality to query that model is introduced to answer a variety of “what if” questions. In contrast to TimedGRL, this approach uses qualitative values and relative time instead of quantitative values and concrete dates. In this technique, the goal model is translated into a constraint solving problem instead of analyzing the model with a propagation-based approach, as done in TimedGRL. Moreover, the language is not formally defined with the help of a metamodel, and it does not consider model management via included contexts.

Ernst, Borgida, and Jureta [19] propose an approach for requirements evolution, which focuses on finding possible solutions given an original requirements problem, its solutions, and a modified requirements problem by modifying/reusing solutions with minimal changes to the already existing solutions. Moreover, metrics are provided that provide reasons to prefer changed

solutions. As opposed to TimedGRL, this approach does not use propagation-based reasoning, but uses an assumption-based truth maintenance system and it does not explicitly define changes, but requires a full description of the modified requirements problem.

Hartmann et al. [32] introduce a native versioning concept to support historized models. This modeling technique integrates versioning on a model element level, which in turn allows every element in the model to evolve independently. A version identifier (VI) along with the path for each model element is used for identification purposes. It is possible to define model element relationships, however, the version information is not utilized in these cases and requires definition of the navigation context. Moreover, three basic operations (*shift*, *previous*, and *next*) are introduced to allow navigation in versions of model elements. All versions of all model elements are stored by converting them into *traces* and hence are accessible. The proposed approach is implemented in the Kevoree Modeling Framework [33]. However, this approach requires addition of each version of a model element separately, even if the version evolution follows a predictable pattern. Moreover, the modeler needs to have prior knowledge of the version of the model element that he needs to access. TimedGRL tackles these issues by allowing the addition of different types of changes directly in the model and introducing timepoints and timepoint groups to evaluate the model at a certain time.

Cicchetti et al. [41] present a metamodel independent technique to represent model differences. They propose that the difference between two models, which follow the same metamodel (say MM), conforms to another metamodel (say MMD), automatically derived from MM. This derived metamodel defines the constructs required to illustrate needed adjustments, that may occur on metamodels and which can be categorized as *additions*, *deletions*, and

changes. Hence, any metamodel can be updated to be able to visualize and define difference models. This approach can be used to construct the final model, given the original version and respective difference models. It is also possible to generate an inverse difference model to obtain the original model from the final model. Moreover, the approach proposes sequential merging (merging of consecutive difference model into a single model) and parallel merging (to enable simultaneous manipulation of the same artifacts). Another work by Cicchetti et al. [42] uses this technique for model co-evolution to automatically adapt a model to its updating metamodel. However, the modifications allowed in this approach seem discrete in nature. It does not specify how to handle continuous changes, specifically continuous numeric changes. TimedGRL allows the modeler to define both discrete and continuous changes in their model. Moreover, this approach deals with changes using different model versions, while TimedGRL uses time as a parameter to define changes.

Jürjens et al. [35] propose a security modelling notation to capture evolving security requirements for long-running security-critical software systems. They introduce UMLseCh, which is an extension of the UML profile UMLsec, to support system evolution. The proposed notation uses change stereotypes, obtained by extending current UMLsec stereotypes, to model changes. It is possible to add, delete, or / and substitute elements using these stereotypes and use these to reason about evolving security. Unlike TimedGRL, this approach deals with a specific domain (security) in modeling and only allows operations on the elements themselves, not on their properties. Wenzel et al. [36] also deal with model changes to support security verification of evolving systems in their work.

Yskout et al. [37] propose change patterns as a framework to capture and handle co-evolution between related software artifacts, like the requirements specification and the architectural design. This framework facilitates evolution of a software artifact based on changes taking place in another artifact at a different level of abstraction. Thus, the designer can prepare an artifact for changes expected in the future and whenever, the change actually takes place, the artifact is evolved as per the design. This approach deals with co-evolution of artifacts, while TimedGRL is currently based on a single artifact, i.e., the goal model, in which any change can be defined independent of other artifacts. Our future work includes extending TimedGRL concepts to scenario models, which will help in getting insight into evolving relationships between goal and scenario models. In contrast to TimedGRL, Yskout et al. do not use a real physical timescale with timepoints and timepoint ranges.

There are some other notable approaches that are similar to TimedGRL. Work done by Breu et al. [38] focuses on model based management, design and operation of dynamically evolving systems. The proposed approach provides a process model for change and change propagation. In this framework, any change is an event, which can trigger change propagation and change handling. UML state diagrams are used for this approach. These state machines are attached to the metamodel elements and their states show the major milestones in the lifetime of the corresponding model element. While this framework includes time events, it does not explain how concrete dates (i.e., timepoints in TimedGRL) are handled, nor does it illustrate how an evolving system is visualized and analyzed over time (e.g., with heatmaps in TimedGRL). Bergmann et al. [39] discuss change-driven model transformations, that are caused by changes in model due to random transactions on the model. R  th et al. [40] propose a live model

transformation framework, which continuously identifies any model changes and then, propagate their effects in the model. Similarly to other related work and in contrast to TimedGRL, real physical timescales are also not discussed by the work of Bergman et al. and R  th et al..

CHAPTER 7: Conclusions and Future Work

This thesis proposes an extension to the current Goal-oriented Requirement Language (GRL) called TimedGRL. It allows modelers to capture expected changes to goal model elements over time. Moreover, it provides a platform to reason about the goal model at different points in time. As a result, instead of performing trade-off analysis that is only based on a snapshot in time, it is now possible to do the same based on a description of the system's evolution over time. TimedGRL also enables the modeler to visualize the specified changes to goal model elements and the results of an evaluation over a given time-period.

TimedGRL is defined as a metamodel extension to GRL. An evaluation algorithm has also been proposed that preprocesses a TimedGRL goal model so that current GRL evaluation algorithms can still be used. While it is possible to define the majority of changes to goal models directly with TimedGRL, there still exist a few situations that cannot be modeled directly using this approach, because doing so would result in an invalid GRL model. For such situations, workaround solutions have been provided, that introduce only a few additional model elements. Moreover, the concepts and evaluation algorithm of TimedGRL have been implemented in the jUCMNav tool. A real-life situation has been modeled and evaluated to validate the sufficiency of the proposed TimedGRL concepts and their integration in the jUCMNav tool.

Currently, it is only possible to define numerical changes with respect to time. In future work, we will examine in detail how to best specify changes which are dependent not only on time but also on other system variables such as satisfaction values. E.g., it may be possible to make an element behave based on another element's satisfaction level. This will allow modelers

to explore relationships between different elements and use that to define their evolution over time. Another extension to TimedGRL could explore system changes that are triggered by the occurrence of a specific pattern or by the observation of aggregate changes.

Furthermore, the introduced TimedGRL concepts could be extended to scenario/workflow models, so that the evolution of a system can be captured and analyzed using both model views, hence providing a comprehensive modeling environment for evolving systems. Such an extension would be able to provide insight into evolving relationships between goal and scenario models. We will examine how TimedGRL concepts can be reused for scenario models and investigate the possible new concepts required for scenario models. We will also look into the visualization of evaluation over time for timed scenario models and integrated goal/scenario models.

Moreover, we will explore the possibility of restructuring the TimedGRL metamodel in a way that would allow other modeling languages to reuse TimedGRL concepts to support model analysis over time. We will identify the elements and TimedGRL concepts in the metamodel, which can be generalized and reused, and then reconstruct the metamodel using the abstract identified concepts. Additionally, we will apply this restructured metamodel to another modeling language in order to prove its validity and practicality.

REFERENCES

- [1] Amyot, D. and Mussbacher, G., “User Requirements Notation: The First Ten Years, The Next Ten Years”, *Journal of Software (JSW)*, 6(5):747–768, 2011. DOI: 10.4304/jsw.6.5.747-768.
- [2] Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., and Yu, E., “Evaluating Goal Models within the Goal-oriented Requirement Language”, *Intl. J. of Intelligent Systems (IJIS)*, Wiley, 25(8):841–877, 2010. DOI: 10.1002/int.20433.
- [3] Amyot, D., Rashidi-Tabrizi, R., Mussbacher, G., Kealey, J., Tremblay, E., and Horkoff, J., “Improved GRL Modeling and Analysis with jUCMNav 5”, 6th International i* Workshop (iStar 2013), 2013. CEUR-WS 978:137–139.
- [4] Chung, L., Nixon, B.A., Yu, E., and Mylopoulos, J., “Non-Functional Requirements in Software Engineering”, Kluwer Academic Publishers, 2000.
- [5] Duran, M.B. and Mussbacher, G. (2016), “Investigation of Feature Run-Time Conflicts on Goal Model-Based Reuse”, *Information Systems Frontiers (ISF)*, Springer 18(5):855-875. DOI: 10.1007/s10796-016-9657-7.
- [6] Duran, M.B., Mussbacher, G., Thimmegowda, N., and Kienzle, J., “On the Reuse of Goal Models”, 17th International System Design Languages Forum (SDL 2015), Springer, LNCS 9369:141–158, 2015. DOI: 10.1007/978-3-319-24912-4_11.
- [7] Gonçalves, R., Knorr, M., Leite, J., and Slota, M., “Non-monotonic Temporal Goals”, 12th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2013), Springer, LNCS 8148:374–386, 2013. DOI: 10.1007/978-3-642-40564-8_37.
- [8] Grubb, A.M., “Adding Temporal Intention Dynamics to Goal Modeling: A Position Paper”, 7th Intl. Workshop on Modeling in Software Engineering (MiSE 2015), IEEE, 66–71, 2015.
- [9] Grubb, A.M. and Chechik, M., "Looking into the Crystal Ball: Requirements Evolution over Time," 2016 IEEE 24th International Requirements Engineering Conference (RE), Beijing, 2016, pp. 86-95.doi: 10.1109/RE.2016.45
- [10] Horkoff, J. and Yu, E., “Comparison and Evaluation of Goal-Oriented Satisfaction Analysis Techniques,” *Requirements Engineering Journal (REJ)*, Springer, 18(3):199–222, 2013. DOI: 10.1007/s00766-011-0143-y
- [11] International Telecommunication Union, “Recommendation Z.151 (10/12), User Requirements Notation (URN) – Language definition”, 2012. <http://www.itu.int/rec/T-REC-Z.151/en>
- [12] jUCMNav website, version 7.0, University of Ottawa.
<http://jucmnav.softwareengineering.ca/jucmnav>
- [13] Letier, E., Kramer, J., Magee, J., and Uchitel, S., “Fluent Temporal Logic for Discrete-time Event-based Models”, *SIGSOFT Software Engineering Notes*, ACM, 30(5):70–79, 2005. DOI: 10.1145/1095430.1081719

- [14] Luo, H. and Amyot, D., “Towards a Declarative, Constraint-Oriented Semantics with a Generic Evaluation Algorithm for GRL”, 5th International i* Workshop (iStar 2011), 2013. CEUR-WS 766:26–31.
- [15] Lamsweerde, A.V., “Requirements Engineering: From System Goals to UML Models to Software Specifications”, John Wiley & Sons Ltd, 2009.
- [16] Yu, E., “Modeling Strategic Relationships for Process Reengineering”, Ph.D. thesis, Department of Computer Science, University of Toronto, Canada, 1995.
- [17] Nuseibeh, B. and Easterbrook, S., “Requirements engineering: a roadmap”, In Proceedings of the Conference on The Future of Software Engineering (ICSE '00). ACM, New York, NY, USA, 35-46. DOI=<http://dx.doi.org/10.1145/336512.336523>
- [18] Aprajita and Mussbacher, G., “TimedGRL: Specifying Goal Models Over Time”, 6th International Model-Driven Requirements Engineering Workshop (MoDRE 2016), Beijing, China, September 2016. IEEE CS, 125-134. DOI: 10.1109/REW.2016.035.
- [19] Ernst, N.A., Borgida, A., and Jureta, I., “Finding incremental solutions for evolving requirements”, 19th International Requirements Engineering Conference (RE 2011), IEEE, 15–24, 2011. DOI: 10.1109/RE.2011.6051656.
- [20] Office of the Auditor General of Ontario
<http://www.auditor.on.ca/>
- [21] 2013 Annual Report (Section 3.05), Office of the Auditor General of Ontario
http://www.auditor.on.ca/en/content/annualreports/arreports/en13/2013ar_en_web.pdf
- [22] 2015 Annual Report (Section 4.05), Office of the Auditor General of Ontario
http://www.auditor.on.ca/en/content/annualreports/arreports/en15/2015AR_en_final.pdf
- [23] Ontario Power Generation Summary of Key Actions from 2013 Auditor General’s Report on Human Resources
<http://www.opg.com/news-and-media/news-releases/Documents/131210BackgrounderAGReport.pdf>
- [24] Ontario Power Generation Key Actions Update (as of June 30, 2014) on 2013 Auditor General’s Report on Human Resources
http://www.opg.com/about/management/open-and-accountable/Documents/AGReport_OPGKeyActionsReport.pdf
- [25] Lamsweerde, A.V., “Goal-Oriented Requirements Engineering: A Guided Tour”, Proceedings Fifth IEEE International Symposium on Requirements Engineering, Toronto, Ont., 2001, pp. 249-262. doi: 10.1109/ISRE.2001.948567
- [26] Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J., “Tropos: An agent-oriented software development methodology”, Autonomous Agents and Multi-Agent Systems 2004; 8(3): 203–236.
- [27] Amyot, D., Shamsaei, A., Kealey, J., Tremblay, E., Miga, A., Mussbacher, G., Alhaj, M., Tawhid, R., Braun, E., Cartwright, N., “Towards advanced goal model analysis with jUCMNav”, In: RIGiM’12, ER Workshops. LNCS 7518, Springer, pp. 201–210 (2012)
- [28] OpenOME website, an open-source requirements engineering tool
<https://se.cs.toronto.edu/trac/ome>

- [29] Yu, E., "Towards modelling and reasoning support for early-phase requirements engineering", In: Proc 3rd IEEE Int Symp on Requirements Engineering, Washington, DC: IEEE CS;1997. pp 226-235.
- [30] Darimont, R., Delor, E., Massonet, P., Lamsweerde, A.V., "GRAIL/KAOS: An Environment for Goal-Driven Requirements Engineering", Proc. ICSE'98 - 20th Intl. Conf. on Software Engineering, Kyoto, April 1998, vol. 2, 58-62. (Earlier and shorter version found in Proc. ICSE'97 - 19th Intl. Conf. on Software Engineering, Boston, May 1997,612-613.)
- [31] Lamsweerde, A.V., Letier, E., "From Object Orientation to Goal Orientation: A Paradigm Shift for Requirements Engineering", Proc. Radical Innovations of Software and Systems Engineering, LNCS, (2003).
- [32] Hartmann, T., Fouquet, F., Nain, G., Morin, B., Klein, J., Barais, O., Traon, Y. L., "A native versioning concept to support historized models at runtime", in Model-Driven Engineering Languages and Systems - 17th International Conference, MODELS 2014, Valencia, Spain, September 28 - October 3, 2014. Proceedings, 2014, pp. 252–268.
- [33] Kevoree Modeling Framework website. <http://kevoree.org/kmf>
- [34] mXparser website. <http://mathparser.org/>
- [35] Jürjens, J., Ochoa, M., Schmidt, H., Marchal, L., Houmb, S. H., Islam, S., "Modelling Secure Systems Evolution: Abstract and Concrete Change Specifications", In Proceedings of the 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM), volume 6659 of Lecture Notes in Computer Science, pages 504–526. Springer, 2011.
- [36] Wenzel, S., Poggenpohl, D., Jürjens, J., Ochoa, M., "Specifying model changes with UMLchange to support security verification of potential evolution", Computer Standards & Interfaces, v.36 n.4, p.776-791, June, 2014.
- [37] Yskout, K., Scandariato, R., Joosen, W., "Change patterns - Co-evolving requirements and architecture", Software and System Modeling 13(2): 625-648 (2014).
- [38] Breu, R., Agreiter, B., Farwick, M., Felderer, M., Hafner, M., Innerhofer-Oberperfler, F., "Living Models - Ten Principles for Change-Driven Software Engineering", Int. J. Software and Informatics 5(1-2): 267-290 (2011)
- [39] Bergmann, G., Ráth, I., Varró, G., Varró, D., "Change-driven model transformations - Change (in) the rule to rule the change", Software and System Modeling 11(3): 431-461 (2012)
- [40] Ráth, I., Bergmann, G., Ökrös, A., Varró, D., "Live Model Transformations Driven by Incremental Pattern Matching", In: Vallecillo A., Gray J., Pierantonio A. (eds) Theory and Practice of Model Transformations. ICMT 2008. Lecture Notes in Computer Science, vol 5063. Springer, Berlin, Heidelberg.
- [41] Cicchetti, A., Ruscio, D. D., Pierantonio, A., "A Metamodel Independent Approach to Difference Representation", Journal of Object Technology, 6(9):165–185, October 2007.
- [42] Cicchetti, A., Ruscio, D. D., Eramo, R., Pierantonio, A., "Automating co-evolution in model-driven engineering," in 12th International IEEE Enterprise Distributed Object Computing Conference, ECOC 2008, 15-19 September 2008, Munich, Germany. IEEE Computer Society, 2008, pp. 222-231.