



Measurement and classification of inter-actor dependencies in goal models

Jameleddine Hassine^{1,2} · Muhammad Tukur¹

Received: 15 November 2020 / Revised: 4 October 2021 / Accepted: 22 November 2021
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract

Goal-oriented requirements engineering approaches aim to capture desired goals and strategies of relevant stakeholders during early requirements engineering stages, using goal models. Socio-technical systems (STSs) involve a rich interplay of human actors (traditional stakeholders, described as actors in goal models) and technical systems. Actors may depend on each other for goals to be achieved, activities to be performed, and resources to be supplied. These dependencies create new opportunities by extending actors' capabilities but may make the actor vulnerable if the dependee fails to deliver the dependum (knowingly or unintentionally). This paper proposes a novel quantitative metric, called Actor Interaction Metric (AIM), to measure inter-actor dependencies in Goal-oriented Requirements Language (GRL) models. The metric is used to categorize inter-actor dependencies into positive (beneficial), negative (harmful), and neutral (no impact). Furthermore, the AIM metric is used to identify the most harmful/beneficial dependency for each actor. The proposed approach is implemented in a tool targeting the textual GRL language, part of the User Requirements Notation (URN) standard. We evaluate experimentally our approach using 13 GRL models, with positive results on applicability and scalability.

Keywords Goal-oriented requirements · Metric · Inter-actor dependencies · GRL · URN

1 Introduction

Goal modeling has been recognized as a popular way of representing and reasoning about the objectives of socio-technical systems (STS). Socio-technical systems (STSs) are organizational systems consisting of people, business processes, software applications, and hardware components [40].

In a goal model, goals are used to depict business objectives and system requirements [43]. These goals are connected to other goals and/or other model elements such as tasks that the system is expected to perform or resources

that are required. Goal model elements can be allocated to stakeholders or systems, often referred to as actors or agents. Over the years, several common goal modeling languages and notations have been developed in the Requirements Engineering (RE) community, including popular ones such as *i** [60], Keep All Objects Satisfied (KAOS) [55], the NFR Framework [15], Tropos [22], and the Goal-oriented Requirement Language (GRL) [36] part of ITU-T's User Requirements Notation (URN) standard. Several techniques have been proposed to analyze goal models. These techniques differ in their targeted language and in their purpose. Goal model analysis areas include the qualitative or/and quantitative evaluation of satisfaction levels of goals and actors composing the model given some initial satisfaction levels [3,9,19,31,32,41], the evaluation of design alternatives [53,56], the prioritization of requirements [49], the checking of goal model's formal properties [17,21], the computation of different types of metrics [19,26,27], and the detection and resolution of conflicts [4,28,29,38,44,45,57]. Socio-technical systems (STSs) involve a rich interplay of human actors (the traditional stakeholders, described as actors in goal models) and technical systems [16]. Actors may depend on each other for goals to be achieved, activities to be per-

Communicated by J. Araujo, A. Moreira, G. Mussbacher, and P. Sánchez.

✉ Jameleddine Hassine
jhassine@kfupm.edu.sa
Muhammad Tukur
g201708390@kfupm.edu.sa

- ¹ Department of Information and Computer Science, KFUPM, Dhahran 31261, Kingdom of Saudi Arabia
- ² Interdisciplinary Research Center for Intelligent Secure Systems, KFUPM, Dhahran 31261, Kingdom of Saudi Arabia

formed, and resources to be supplied. These dependencies will open new opportunities by extending actors' capabilities, since an actor will be able to achieve goals that he was not able to realize (or not as easily or as efficiently) without these dependencies. By following the chains of dependencies, one could explore the expanded possibilities that are open to an actor. However, these dependencies may make the actor (dependor) vulnerable. Indeed, if the dependee fails to deliver the dependum (knowingly or unintentionally), the dependor would be adversely affected in its ability to fulfill its goals [61], which may lead to delays or increased costs, and may hinder the overall project success.

Although there is no common agreement about a definition of conflict between requirements [57], in goal models a conflict often refers to a situation where the satisfaction of a goal may preclude the satisfaction of another [57]. For example, a conflict occurs when different goals provide contradictory contributions to the same goals [22]. In addition, in real-world competitive environments, the goals of different stakeholders may be of a conflicting or opposing nature. Such conflicts may not be avoided, however, it is essential to analyze inter-actor dependencies and answers should be provided to many questions such as (1) To what extent inter-actor dependencies can have an impact on the intervening actors? (2) How to measure effectively the absolute magnitude of these dependencies (beneficial or undesirable), regardless of the strategies adopted by the interacting actors? (3) Can such measurement be refined/adjusted in presence of constraints related to the satisfaction of some model elements? (4) How can an actor identify his most harmful dependency (making him the most vulnerable) and his most beneficial dependency (providing him with the most valuable opportunity)?

Existing work provides insights to answer, partially, some of these questions. However, to the best of our knowledge, no comprehensive approach has been proposed to address all these concerns. The main goal of this paper is to propose a novel, simple, and flexible approach to measure quantitatively the impact of inter-actor dependencies in goal models. Furthermore, our aim is to propose a scalable technique, since scalability is considered as one of the most important problems associated with i^* -based frameworks [42,48] and many goal-oriented quantitative techniques suffer from scalability issues [13,14,30,51].

Our proposed approach is meant to be applied during the early stages of the requirements engineering phase. It is based on goal model satisfaction analysis, which has been used as a decision-making tool, helping modelers to choose between alternative system functionalities or design configurations, before the requirements specification is produced [31]. We have selected the Goal-oriented Requirement Language (GRL) [36] as our target language, given its status as an ITU-T (International Telecommuni-

cation Union Telecommunication Standardization Sector) international standard and because it supports quantitative evaluation strategies. In particular, we have chosen the textual representation of GRL, part of the TURN (Textual User Requirements Notation) standard to automate our approach [36]. The textual GRL has been introduced in order to improve the usability, productivity, and scalability of GRL models [1,2,39].

In this paper, we make the following contributions:

- Propose a novel metric, called *Actor Interaction Metric (AIM)*, to measure inter-actor dependencies in GRL models. AIM is a quantitative metric since it is based on the computation of GRL actors' quantitative satisfaction intervals. It is also syntactic in the sense that it depends only on the model structure, not on the model semantics, i.e., we don't consider the model domain, the types of intentional elements (e.g., goal vs. softgoal vs. task), and the meaning of the text enclosed within the intentional elements. The AIM metric is used to categorize inter-actor dependencies into positive, negative, and neutral. Furthermore, the AIM metric is used to identify the most harmful/beneficial dependency for each actor, given any user-defined strategy.
- Automate the GRL actor interaction analysis approach and the proposed AIM within the textual GRL language, part of the TURN (Textual User Requirements Notation) standard [36].
- Evaluate experimentally the applicability and scalability of the proposed approach and prototype tool by applying it to a running example as well as to 13 Textual GRL models of different sizes, configurations, and complexity (8 models were taken from literature and 5 models were created to cover specific configurations).

The remainder of this paper is organized as follows. The next section provides the background of this research. In the following section, we discuss the problem at hand, the motivation, and the rationale of this research. Section 4 presents and discusses existing work related to the analysis of actor interactions and conflict detection in goal-oriented models. In Sect. 5, we present our proposed inter-actor interaction analysis approach and we apply it to our running example (Sect. 5.8). We evaluate experimentally our approach in Sect. 6, using 13 models, eight of which are publicly available. A presentation of the main benefits of our approach, a comparison with related work, a presentation of potential threats to validity, and a description of some practical considerations are provided in Sect. 7. Finally, conclusions and future work are presented in Sect. 8.

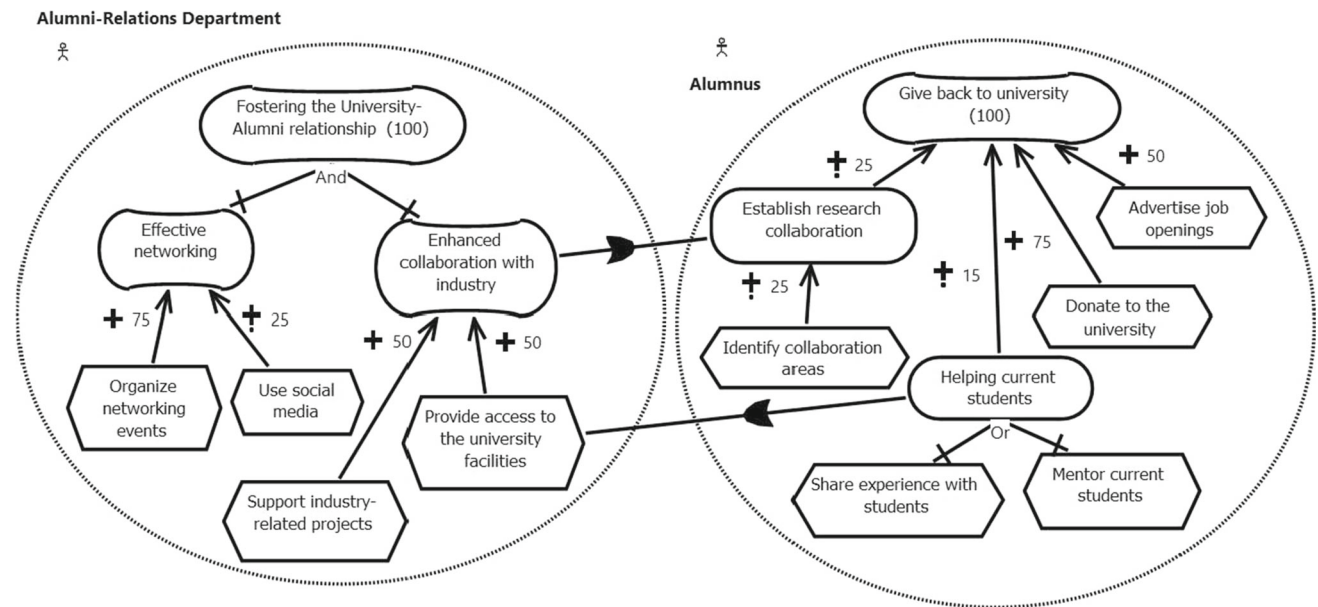






Fig. 1 University–Alumni GRL model

2 Research background


To make the paper self-contained, we start by introducing the GRL language constructs using a simplified and modified GRL specification of the University–Alumni model introduced by Hassine and Amyot [28]. This example will be used as our motivational example in order to define the research problem (see Sect. 3) and to illustrate the applicability of the proposed approach (see Sect. 5).









2.1 GRL in nutshell: University–Alumni GRL model


The University–Alumni GRL model, produced using the *jUCMNav* [37] tool and illustrated in Fig. 1, is composed of two GRL actors “Alumni-Relations Department” and “Alumnus.”

Actors (illustrated as ) represent holders of intentions. They are often used to represent stakeholders (as in this running example) as well as systems. GRL actors may enclose intentional elements describing their intentions and capabilities. For example, the actor “Alumni-Relations Department” wants to foster the relationship between the university and its alumni. This is represented as a *softgoal* (illustrated as ) named “Fostering the University–Alumni relationship.” Softgoals distinguish themselves from goals (illustrated as , e.g., “Helping current students” in actor “Alumnus”) in that there are no clear-cut criteria with respect to its satisfaction. Decomposition links (illustrated as ) allow an intentional element to be decomposed into subelements (using AND, OR, or XOR relationships). For example, softgoal “Fostering the University–Alumni rela-

tionship” is refined using an AND-decomposition into two softgoals “Effective networking” and “Enhanced collaboration with industry.”

Actor capabilities are often described using tasks (illustrated as , representing activities and possible solutions to (or operationalizations to) achieve goals or softgoals. For instance, goal “Helping current students” (in actor “Alumnus”) is decomposed, using an OR-decomposition, into two tasks “Share experience with students” and “Mentor current students.” Actor “Alumni-Relations Department” has four leaf tasks “Organize networking events,” “Use social media,” “Support industry-related projects,” and “Provide access to the university facilities.”

Furthermore, intentional elements, within an actor, may be connected using contribution links (illustrated as ) expressing the impact (positive or negative, at different levels of sufficiency) of one intentional element on another one. A contribution link has a qualitative contribution type (Make (, Help (, SomePositive (, Unknown (, SomeNegative (, Break (, or Hurt () or a quantitative contribution weight (e.g., an integer value within [− 100, 100]). For example, task “Organize networking events” contributes positively (i.e., *SomePositive* as qualitative value and +75 as quantitative value) to softgoal “Effective networking.”

The interaction between actors “Alumni-Relations Department” and “Alumnus” are expressed using two explicit dependency links (illustrated as ): (1) softgoal “Enhanced collaboration with industry” depends on task “Establish research collaboration.” We refer to this dependency as #1 in Table 2 (Sect. 5.8), and (2) goal “Helping current students” depends on task “Provide access to the

university facilities.” We refer to this dependency as #2 in Table 2 (Sect. 5.8).

2.2 GRL satisfaction analysis

Many analysis techniques can be applied to GRL models. The URN standard [36] introduces GRL satisfaction analysis [9], which assigns some initial satisfaction values (called *evaluation strategy*) to a subset of the intentional elements (often leaf elements) and then propagates these values to the other intentional elements of the model via the various model links, e.g., decompositions, contributions, dependencies, etc.

2.2.1 GRL strategies

Strategies can be: (1) *qualitative*: seven satisfaction labels are considered, namely, Satisfied (✓), Weakly Satisfied (✓◊), Denied(✗), Weakly Denied (✗◊), Neutral (●), Unknown (?), and Conflict (⊗), (2) *quantitative*: satisfaction values are computed within the [−100, 100] range (−100 for sufficiently denied, 100 for sufficiently satisfied, [1, 99] for weakly satisfied, and [−99, −1] for weakly denied) [36], or (3) *hybrid*: uses a combination of qualitative and quantitative values.

The (Partially) Satisfied value represents the presence of evidence which is (Insufficient) Sufficient to satisfy an intentional element [34]. Partially denied and denied values have the same definition with respect to negative evidence [34].

Figure 2 illustrates examples of propagation of the satisfaction values through various GRL links. For example, in Fig. 2a, the satisfaction level of goal G (decomposed through AND-decomposition link) is the minimum value of the quantitative evaluation values of its source elements G1 (+75), G2(+25), and G3(−50). For an OR-decomposition link (see Fig. 2b), the satisfaction level is the maximum value of the quantitative evaluation of its source elements; hence the satisfaction of goal G is +75.

For contribution links, the total quantitative contribution is the sum of the products of the quantitative evaluation of each source element by its quantitative contribution level to the element. For example, in Fig. 2c, the satisfaction of goal G is computed as follows: $(75 (G1) \times 100\%) + (25 (G2) \times 75\%) + (-50 (G3) \times 50\%) = 68$. It is worth noting that if the satisfaction value of an element exceeds +100, it will be set to +100. Similarly, if the satisfaction value of an element goes below −100, it will be set to −100. For dependency links, the source element of the dependency links cannot have an evaluation value higher than those of the target elements of the dependency links. A simple example is shown in Fig. 2d, with a strategy that initializes the satisfaction of G1 to +75 and the satisfaction of G2 to +25. The satisfaction value of G is computed as follows: $\min(75 \times 50\%, 25) = 25$.

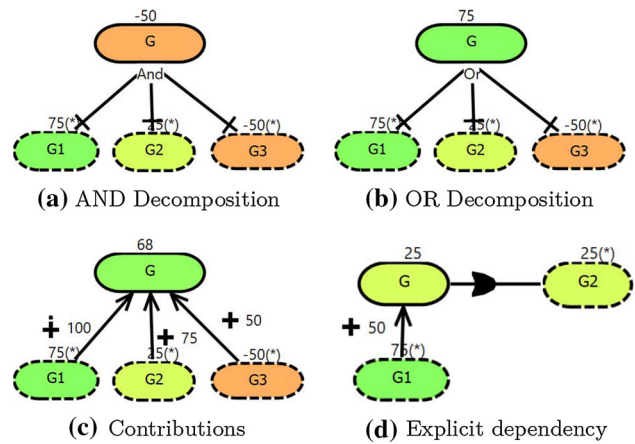


Fig. 2 Quantitative propagation of satisfaction values through GRL links

In addition, a consolidated satisfaction value can be derived for the containing GRL actor based on the satisfaction of its intentional elements. To this end, the analyst should identify which intentional elements are important to the containing actor (can be root or non-root elements). At least one intentional element bound to the actor, to be evaluated, should have an importance level. The qualitative importance (shown between parentheses in intentional elements) can be (H)igh, (M)edium, (L)ow, or None (default). The quantitative importance is specified as an integer between (0) and (100). For example, in Fig. 1, all root elements, i.e., “Fostering the University–Alumni relationship” and “Give back to university,” have high importance, denoted by (100). The satisfaction of an actor is computed as follows:

$$Sat(Actor) = \frac{\sum_i Sat(element_i) \times Imp(element_i)}{\sum_i Imp(element_i)}$$

For example, in Fig. 1, the computed satisfaction of actor “Alumni-Relations Department” is the same as the satisfaction of the softgoal “Fostering the University–Alumni relationship.”

In GRL, we distinguish between explicit and implicit dependencies [26]. Contribution, correlations, and decomposition links are considered as implicit dependencies (as opposed to explicit dependency, illustrated as \rightarrow). For example, the satisfaction of goal G in Fig. 2a–c depends on the satisfaction of its children G1, G2, and G3.

2.2.2 Evaluation of alternative strategies

Depending on the context and goals of the intervening stakeholders, analysts design and evaluate several strategies in order to select the one representing the best trade-off. The most appropriate strategy provides rationale and documentation for decisions leading to requirements.

In the running example (see Fig. 1), actor “Alumni-Relations Department” wants to achieve his top goal “Fostering the University–Alumni relationship.” To attain his goal, the actor has 4 activities (GRL tasks: “Organize networking events,” “Use social media,” “Support industry-related projects,” “Provide access to the university facilities”) from which he can satisfy one or many. Depending on the university regulations and the budgetary constraints in place, actor “Alumni-Relations Department” may consider, among others, the following strategies:

1. *D1: Sufficient budget and no facility restrictions:* actor “Alumni-Relations Department” may decide to fully satisfy (i.e., 100) all his GRL tasks.
2. *D2: Limited budget and no facility restrictions:* actor “Alumni-Relations Department” may decide to fully satisfy (i.e., 100) “Use social media” and “Provide access to the university facilities,” while weakly satisfy (i.e., 50) “Organize networking events” and “Support industry-related projects.”
3. *D3: Sufficient budget and strict facility restrictions:* actor “Alumni-Relations Department” may decide to fully satisfy (i.e., 100) “Use social media,” “Organize networking events,” and “Support industry-related projects,” while fully deny (i.e., –100) “Provide access to the university facilities.”
4. *D4: Sufficient budget and partial facility restrictions:* actor “Alumni-Relations Department” may decide to fully satisfy (i.e., 100) “Use social media,” “Organize networking events,” and “Support industry-related projects,” while weakly satisfy (i.e., 50) “Provide access to the university facilities.”

Similarly, actor “Alumnus” wants to achieve his top goal “Give back to university” and he has 5 activities (GRL tasks: “Identify collaboration areas,” “Share experience with students,” “Mentor current students,” “Donate to the university,” “Advertise job openings”) from which he can satisfy one or many.

Depending on his willingness to make monetary vs. non-monetary contributions and the amount of effort he can spend to achieve “Give back to university,” actor “Alumnus” may consider, among others, the following strategies:

1. *A1: Non-monetary contributions only:* actor “Alumnus” may decide to fully deny (i.e., –100) “Donate to the university” and “Mentor current students,” while fully satisfy “Identify collaboration areas,” “Share experience with students,” and “Advertise job openings.”
2. *A2: Monetary contributions only:* actor “Alumnus” may decide to fully satisfy (i.e., 100) “Donate to the university” and fully deny (i.e., –100) the remaining GRL tasks.

3. *A3: Moderate non-monetary and monetary contributions:* actor “Alumnus” may decide to weakly satisfy (i.e., 50) all his tasks.

Combining the four Alumni-Relations Department strategies and the three Alumnus strategies results in 12 strategies, summarized in Table 1. For each alternative solution, we show the resulting actor satisfaction values.

Given the 12 GRL strategies, the best trade-offs to satisfy both actors are: D1 & A3, D2 & A3, and D4 & A3, all leading to 12 and 73 as actors’ satisfaction values. The next step would be to develop the requirements of the selected alternative solution. For example, if D2 & A3 alternative is selected, the “Alumni-Relations Department” has to:

- Create a sound social media strategy to fully satisfy “Use social media.” For example, a sound strategy involves setting clear goals, pick the right social media channels, create content, and track meaningful metrics, e.g., click-throughs, cost-per-click, etc.
- Provide visitors an easy access to the university premises in order to fully satisfy “Provide access to the university facilities.”
- Ideally, the “Alumni-Relations Department” would like to organize 2 networking events per year. However, given the allocated budget, the department will plan for one single event to partially satisfy “Organize networking events.”
- Put in place a procedure to provide partial support for industry-related projects, e.g., limited travel grants, limited summer assignments, etc., in order to weakly satisfy “Support industry-related projects.”

Similarly, actor “Alumnus” should detail all requirements that lead to a partial satisfaction all his tasks. For example, to partially satisfy “Donate to the university,” an alumnus can decide to participate and donate to one single fund-raising event (the university organizes at least two fund-raising events every year).

2.2.3 GRL circular dependencies

GRL models may contain circular dependencies (a.k.a. cycles). As stated in the URN standard [36], a dependency cycle is not evaluated (i.e., satisfaction analysis is blocked) unless the satisfaction of one of its elements is manually overridden. Manually setting a satisfaction value is similar to breaking the cycle because manual evaluations cannot be overridden by the automatic propagation algorithm. Figure 3 illustrates a GRL model with a circular dependency: (1) G2 depends on G1 (explicit dependency), (2) G1 depends on G3 (explicit dependency), and G3 depends on G1 (implicit dependency, i.e., contribution). In presence of this cycle, the

Table 1 Running example: assessment of alternative solutions

GRL Model strategies	Actor satisfaction Alumni-Relations Department	Alumnus
D1 & A1	25	-3
D1 & A2	-25	3
D1 & A3	12	73
D2 & A1	25	-3
D2 & A2	-25	3
D2 & A3	12	73
D3 & A1	0	-33
D3 & A2	-25	3
D3 & A3	0	50
D4 & A1	25	-11
D4 & A2	-25	3
D4 & A3	12	73

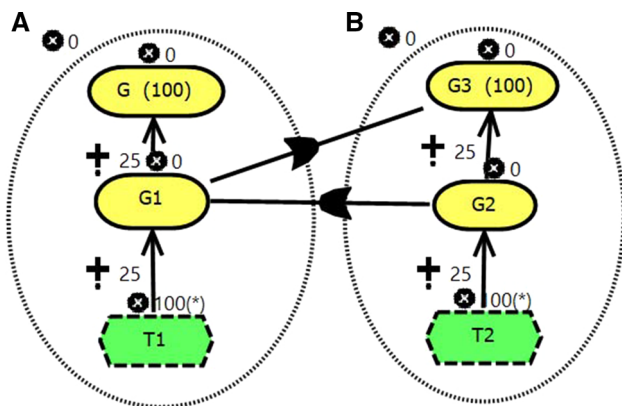


Fig. 3 Example of circular dependency

initial satisfaction values of tasks T1 and T2 (fully satisfied), were not propagated to the rest of the model.

For a complete description of the GRL language and the satisfaction algorithms, interested readers are referred to the URN standard [36].

2.3 Textual GRL (TGRL)

In addition to the concrete graphical syntax supported by the *jUCMNav* tool [37], textual representations of GRL (TGRL) have been proposed in order to improve the usability, productivity, and scalability of GRL models [1,2,39]. The textual representation of GRL has been standardized by the International Telecommunication Union (ITU-T), as part of TURN (Textual User Requirements Notation). TURN is defined as an Xtext grammar, from which an editor and the TURN metamodel are automatically generated [39], covering a large subset of URN [36].

In this research, we implement our proposed approach for the textual representation of GRL, as defined in the ITU-T

```
urnModel UniversityAlumni {
  description "University Alumni TGRL Model"
  urnVersion "29.0"
}
actor AlumniRelations#"Alumni-Relations Department"{
  softgoal FosteringUnivAlumniRel#"Fostering the University Alumni relationship"{
    importance high
  }
  softgoal EffectiveNetworking#"Effective Networking"{
    and decomposes FosteringUnivAlumniRel
  }
  softgoal EnhancedCollIndustry#"Enhanced collaboration with industry"{
    and decomposes FosteringUnivAlumniRel
    dependsOn Alumnus.EstablishResearchCollaboration
  }
  task OrganizeNetworkingEvents#"Organize networking events"{
    contributesTo EffectiveNetworking with 75
  }
  task UseSocialMedia#"Use social media"{
    contributesTo EffectiveNetworking with 25
  }
  task SupportIndustryProjects#"Support industry-related projects"{
    contributesTo EnhancedCollIndustry with 50
  }
  task ProvideAccessFacilities#"Provide access to the university facilities"{
    contributesTo EnhancedCollIndustry with 50
  }
}
actor Alumnus#"Alumnus"{
  goal GivebackUniversity#"Give back to University"{
    importance high
  }
  task EstablishResearchCollaboration#"Establish research collaboration"{
    contributesTo GivebackUniversity with 25
  }
  task DonateToUniverity#"Donate to the university"{
    contributesTo GivebackUniversity with 75
  }
  task IdentifyCollaborationAreas#"Identify collaboration areas"{
    contributesTo EstablishResearchCollaboration with 25
  }
  task AdvertiseJobOpenings#"Advertise job openings"{
    contributesTo GivebackUniversity with 50
  }
  goal HelpingCurrentStudents#"Helping current students"{
    dependsOn AlumniRelations.ProvideAccessFacilities
    contributesTo GivebackUniversity with 15
  }
  task ShareExpWithStudents#"Share experience with students"{
    or decomposes HelpingCurrentStudents
  }
  task MentorCurrentStudents#"Mentor current students"{
    or decomposes HelpingCurrentStudents
  }
}
```

Fig. 4 TGRL specification of the University–Alumni GRL model

standard [36]. Figure 4 illustrates the textual GRL specification corresponding to the GRL model of Fig. 1.

Cycles in graphical GRL models may not be easily visually recognized since they might include links of different types (dependency, decomposition, and contribution), as illustrated in Fig. 3. This issue is even more problematic in

Textual GRL. In our proposed approach, we tackle this problem by detecting whether a TGRL specification is acyclic or not.

3 Problem definition: GRL actor interactions

In GRL models, actors depend on each other for the satisfaction of their goals, through explicit/implicit dependencies. Ideally, an actor should benefit from his outgoing dependencies to maximize the satisfaction of his goals. However, outgoing dependencies may have a negative impact, reducing the satisfaction levels of the actors' goals. For example, if the dependee intentionally or unintentionally fails to deliver the dependum, the depender becomes vulnerable and would be adversely affected in its ability to fulfill its goals. Such situation should be avoided since it represents a situation of conflict in the sense defined in [22], i.e., a conflict occurs when different goals provide contradictory contributions to the same goals [22].

Hence, the requirements engineer/analyst has to assess the magnitude and the impact of such inter-actor dependencies and make the required adjustments to achieve an acceptable trade-off between the interacting actors.

In what follows, we explore the problem by examining two concrete scenarios of interactions between GRL actors. Note that the colors of the intentional elements indicate their satisfaction levels (the greener the better, and the redder the worse).

3.1 Interaction scenario 1

In this scenario, we assume that actor "Alumni-Relations Department" decides to fully satisfy all his 4 leaf tasks. We also assume that actor "Alumnus" decides to fully deny (i.e., -100 value) task "Identify collaboration areas" (may be because the Alumnus may not have access to information about potential collaboration areas (e.g., classified information) or he/she knows that his/her employer policy is not to engage in collaborations with universities), and fully satisfy all his remaining leaf elements (see Fig. 5a). As a result of this strategy, actor "Alumnus" becomes fully satisfied (i.e., 100 value), while actor "Alumni-Relations Department" becomes weakly denied (i.e., -25 value).

Based on this scenario, we can make two important observations:

1. Although actor "Alumni-Relations Department" has fully satisfied all his leaf elements (to maximize his satisfaction), he ended up with a weakly denied satisfaction (i.e., -25). The strategy of "Alumnus" caused the weakly denial of softgoal "Establish research collaboration" (i.e., -25) and hence the weakly denial of the

dependent softgoal "Enhanced collaboration with industry" (since its satisfaction cannot surpass the one of its dependee, i.e., "Establish research collaboration"). The negative payoff of "Alumni-Relations Department" (i.e., -25) is caused by one single dependency link between softgoal "Enhanced collaboration with industry" and task "Establish research collaboration." In absence of that dependency link, actor "Alumni-Relations Department" will be fully satisfied (i.e., $+100$ value) (see Fig. 6).

2. From actor "Alumnus" perspective, although task "Identify collaboration areas" is fully denied, the actor was able to achieve full satisfaction value (i.e., 100). Hence, if actors are competing against each other (which is not the case in this particular university context), "Alumnus" wins, while "Alumni-Relations Department" loses.

Based on this scenario, we conclude that:

- Actor "Alumni-Relations Department" is vulnerable, and he is at the mercy of actor "Alumnus," that can impact him negatively (i.e., by denying task "Identify collaboration areas").
- Only one single dependency (between softgoal "Enhanced collaboration with industry" and task "Establish research collaboration") is responsible for the huge reduction in satisfaction value of "Alumni-Relations Department," i.e., from 100 to -25 (see Fig. 6).
- Actor "Alumnus" has the ability to impact "Alumni-Relations Department," without compromising his overall satisfaction.

3.2 Interaction scenario 2

In this scenario (see Fig. 5b), we assume that actor "Alumnus" fully satisfies his leaf elements, except task "Identify collaboration areas" (fully denied, i.e., -100 value), while actor "Alumni-Relations Department" satisfies all his leaf elements, except task "Provide access to the university facilities" (fully denied, i.e., -100 value). This strategy results in the weakly denial of actor "Alumni-Relations Department" (i.e., -25 value) and to the full satisfaction of actor "Alumnus" (i.e., $+100$ value).

Based on this scenario, we can make two important observations:

1. From actor "Alumnus" perspective, although task "Identify collaboration areas" was fully denied (voluntarily), while goal "Helping current students" was also fully denied as a result of the dependency link between "helping current students" and "Provide access to the

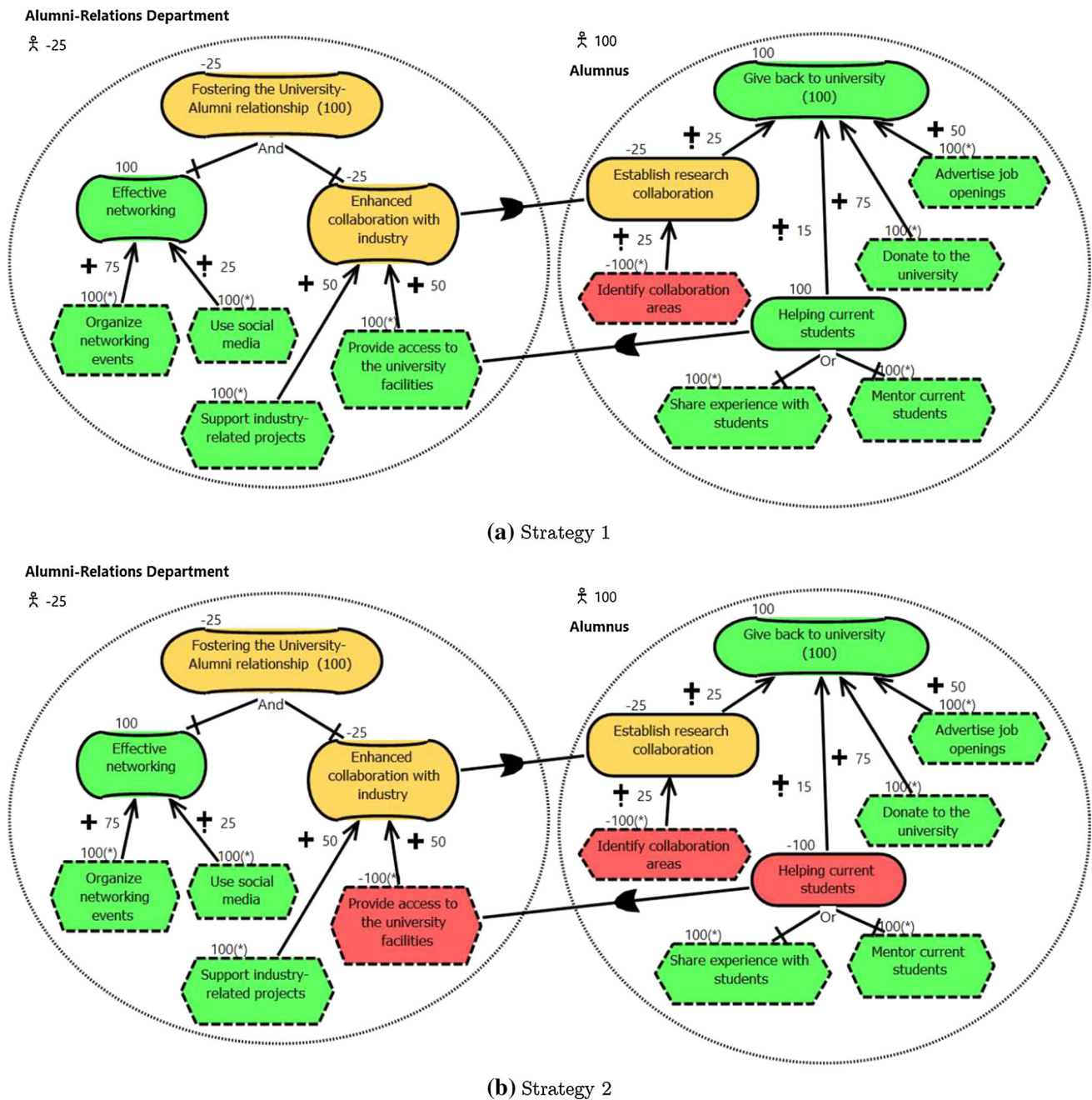


Fig. 5 University–Alumni GRL model: two inter-actor interaction scenarios

university facilities” (since its satisfaction cannot surpass the one of its dependee, i.e., “*Provide access to the university facilities*”), Alumnus was able to achieve full satisfaction (i.e., +100 value). Hence, the denial of “*Provide access to the university facilities*” had no impact on its satisfaction.

2. From actor “Alumni-Relations Department” perspective, the denial of task “*Provide access to the university facilities*” did not impact negatively the satisfaction of actor “Alumnus.”

Based on this scenario, we conclude that:

- Actor “Alumnus” is not vulnerable to “Alumni-Relations Department,” since the latter can avoid any negative impact (caused by the external dependency) even intentionally.
- The dependency (between “*helping current students*” and “*Provide access to the university facilities*”) has no impact on the satisfaction of “Alumnus,” if he chooses the right strategy.

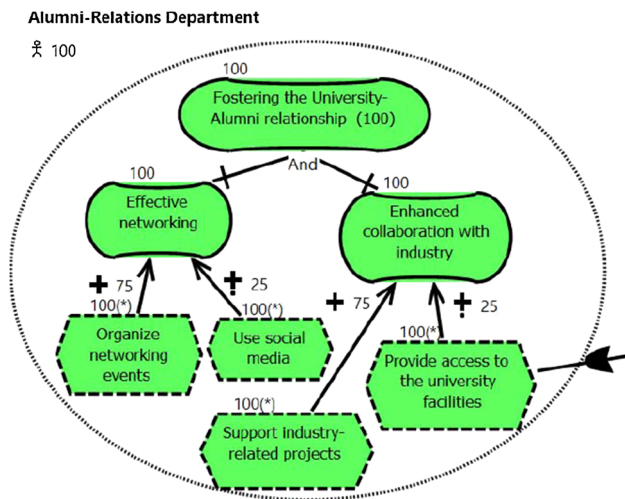


Fig. 6 Strategy 1: Alumni-Relations Department with no outgoing dependencies

Based on the observations from both interaction scenarios, there is a strong need to assess the effect of inter-actor dependencies on GRL actors. Furthermore, it is necessary to assess this impact, not only for one strategy (e.g., each scenario represents one strategy) but also when we have a range of strategies.

4 Related work

In what follows, we review existing approaches about the analysis of actor interactions and conflict detection in goal-oriented models.

van Lamsweerde et al. [57] proposed formal heuristics to identify divergences among different stakeholders' viewpoints or within a single viewpoint. Divergences are considered as a weak form of conflict, where divergent goals are not contradictory (can be simultaneously satisfied), but become inconsistent when certain conditions, called *boundary conditions*, hold. These boundary conditions are expressed in temporal logic and follow a pre-determined set of patterns. The proposed methodology was presented in the KAOS [55] framework. Contrary to this technique, our approach is syntactic as it does not consider goal model context and semantics.

In the context of Goal-oriented Requirements Language (GRL) [36], Hassine and Amyot [28] proposed an empirical approach that allows for early detection of potential conflicts among intervening stakeholders of the system. These conflicts are quantitatively detected using a questionnaire and examined using statistical analysis. Later, Hassine and Amyot [29] proposed an empirical approach based on concept analysis in order to fix goal model artifacts that are subject to conflicts. These approaches [28,29] do not focus

on inter-actor interactions as a source of conflicts, rather they define a conflict as a situation where stakeholders have different views of the GRL sub-model under analysis. In our proposed approach, we assume that stakeholders are responsible for their local decisions (i.e., local GRL actor elements) and potential conflicts come exclusively from the inter-actor dependencies.

Ali et al. [6] proposed an approach for the detection of conflicting context changes. These changes emerge as the result of the simultaneous execution of two or more system executable processes (i.e., tasks in a goal model) aiming to change the same object in the system environment into different states (in order to meet different requirements). To this end, the authors have enriched their previous work on contextual goal modeling [5] using Tropos, by introducing two additional information that the designer have to specify explicitly: (1) the effect of tasks execution on the system operational environment, and (2) the sequence/parallelism operators between tasks. The introduction of these new annotations, leads to many execution variants of the goal model. Next, the consistency of the context of each variant is checked and inconsistent variants are discarded from future processing. The tasks of each resulting consistent variant are then extracted and partitioned into sets, according to their parallel execution. Each partition of tasks is checked to know if it includes tasks changing an object in the system environment into different states. The approach is supported by a CASE tool. There are two key differences between our proposed work and the one by Ali et al. [6]. First, we focus on actor satisfaction analysis without considering the sequencing of tasks. Second, we focus on inter-actor interactions, while in Ali et al. [6], their focus was on the detection of conflicts within the system itself.

DeVries and Cheng [18] proposed a Feature Interaction (FI) detection technique, called *Phorcys*. *Phorcys* analyzes systems composed of a set of features described hierarchically using a AND/OR goal model. *Phorcys* symbolically analyzes each feature with respect to different possible feature combinations (from the goal model) and uses a constraint solver to check for conflicting combination of features (a.k.a. n-way feature interaction). Contrary to this work [18] where the authors have used the goal model to describe dependencies within features (without referring to actors), we consider interactions between different entities expressed as separate actors.

Sutcliffe and Minocha [52] proposed an approach to analyze dependency coupling to detect excessive interactions among users and systems. The interactions are modeled using *i** [60] and scenarios expressed using use case interaction diagrams. The authors use expert judgment to analyze the couplings of the model, then define a metric that is based on an 11-points scale in order to compare the couplings of different alternative scenarios. However, their approach is solely

based on the i^* model structure and does not consider goal satisfaction analysis.

Bryl et al. [13,14] proposed a framework, called *P-tool*, to generate a space of actor dependency network configurations that satisfy stakeholders' goals. These dependency configurations are evaluated based, among others, on the length of the obtained plan, the overall plan cost, and the degree of satisfaction of non-functional requirements. In addition, the authors introduced some guidelines to evaluate each retained dependency configuration in order to determine whether it represents a game-theoretic equilibrium [47]. An equilibrium is defined, as an i^* [60] goal model where no actor can do better with respect to its own goals by adopting a different strategy for delegating and accepting delegations. Sumesh et al. [51] proposed a Game Theory (GT) [47]-based approach to analyze i^* Strategic Rationale (SR) models having goals with opposing objective functions and connected through inter-actor dependencies. The authors modeled the game as a two-person, zero-sum game, where the players are the two top softgoals of each actor and the alternative options represent the available strategies for each actor. Outcome matrices (satisfaction values) are created for the top softgoals, based on the objective function values. The resulting matrices are then used to find the Nash equilibrium, representing the best trade-off between the intervening players. While their approach [51] aims to find the best initial strategy, our proposed approach focus on measuring the degree and impact of the model inter-actor dependencies. In addition, in [51] satisfaction values are created for all possible strategies, with all leaf elements having a binary satisfaction value (achieved or not achieved). Instead, our approach is more flexible, allowing for the use of interval-based strategies covering ranges of satisfaction values.

More recently, Hassine et al. [30] proposed a game-theoretic approach to reconcile the interactions that can occur between two GRL actors, without the need to modify the model structure. The authors [30] formalized the situation of interaction as a two-player, non-cooperative, nonzero-sum game. The computation of a Nash equilibrium produces a reconciliation strategy that represents a reasonable trade-off solution for the interacting actors.

Subramanian et al. [50] proposed a fuzzy-based approach to evaluate goals using inter-actor dependencies in an i^* framework. The authors claimed that fuzzy numbers are more appropriate than exact numerical values, when it comes to translating vague stakeholder's linguistic terms into quantitative values assigned to goal model links and intentional elements. In our proposed approach, we use intervals of integers to characterize the intentional elements/actors satisfactions.

Franch et al. [20] proposed a framework to analyze actor dependency models (i^* Strategic Dependency models) with respect to a given set of model properties, such

as privacy, accuracy, and efficiency. Actors are classified into two classes, namely, human (H) and software (S), while dependencies are categorized into three categories, namely, human-human (H-H), software-software (S-S), and human-software (S-H). The evaluation of a given dependency d , part of a model M with respect to a certain property P , is computed as the product of the dependency weight (value between 0 and 1) and the dependency adjusted weight for P considering its depender and dependee (value between 0 and 1). The global structural dependency metric of the model is then computed as the sum of the evaluations of its elements, and normalized the value considering the number of dependencies. This approach has been applied by Grau et al. [23,24] to assess the effectiveness of alternative architectures using a coupling metric over i^* SD models. Coupling is measured as the number of incoming and outgoing dependencies (multiplied by a weight factor relative to each actor) an actor is associated with. Xavier Franch [19] has proposed an i^* -based framework to measure actor predictability using Object Constraint Language (OCL). An i^* actor would be concerned about how other actors, that he is depending on, would behave. Hence, inter-actor dependencies represent the main component in the definition of the predictability metric. The author [19] considers task and resource dependencies to be totally predictable (evaluated to 1), while goal and softgoal are not, hence, requiring expert judgment. The predictability of goals and softgoals is a function of the depender expertise and the know-how of the dependees. This metric does not focus on the actor satisfaction, instead, it is more interested in the analysis of structural properties of the model.

In the GRL [36] context, jUCMNav [37] tool (GRL modeling and analysis framework) captures many simple structural GRL metrics (e.g., number of actors, goals, tasks, intentional elements, links, etc.) and allows for the definition of additional user-defined metrics using OCL, such as number of explicit/implicit inter-actor dependencies. Furthermore, Hassine and Alshayeb [26] proposed a quantitative metric to measure the Actor External Dependencies (AED) in GRL goal models. Given an actor A having k intentional elements, AED metric is defined as follows:

$$AED(A) = \frac{\sum_{i=1}^{i=k} \frac{nAED_i}{nSE - k}}{k}$$

where $nAED$ (Number of Actor External Dependencies), denotes the number of external dependencies (incoming contribution and decomposition links, and outgoing explicit dependencies) for each element enclosed in actor "A," and nSE denotes the total number of intentional elements in the GRL model. AED is purely structural metric as it relies only on the structure of the GRL model (e.g., number of inter-

actor dependencies and number of intentional elements) and does not consider actor satisfactions analysis.

A summary of the comparison of our work with related work is provided in Sect. 7.2.

5 Measuring GRL inter-actor interactions

In this section, we start by describing our research methodology, then we present our research questions (see Sect. 5.2), and finally we describe our proposed approach.

5.1 Research methodology

The research methodology adopted in this work is composed of four main phases (see Fig. 7). First, we start by surveying existing goal-oriented techniques that focus on characterizing and quantifying stakeholders' interactions in socio-technical systems (STS). The motivational example, presented in Sect. 3, to understand how GRL actors interact under different evaluation strategies and why such interactions cannot be characterized using solely the inter-actor dependencies. Then, the research goals are formulated using two research questions (see Sect. 5.2). As per our proposed solution, we proposed a quantitative metric to measure and classify GRL inter-actor dependencies into three categories: beneficial (positive impact on actor satisfaction), harmful (negative impact on actor satisfaction), and neural (no impact on actor satisfaction). Finally, we have evaluated our approach by applying it to several publicly available GRL models as well models created to satisfy specific configurations, e.g., models containing circular dependencies, many actors, etc.

5.2 Research questions

The main goal of this research is to come up with a sound characterization and analysis of the impact of GRL actors' interactions on their satisfaction. More particularly, we aim to address the following research questions:

1. *Research question RQ1*: How to measure the impact of dependencies between GRL actors?

Rationale for RQ1: Interactions between GRL actors can be measured using a simple metric that consists of counting the number of external dependencies, regardless of actors' structures and sizes. However, as shown in scenarios 1 and 2 (see Sects. 3.1 and 3.2), although each actor has one single external dependency link, the first dependency is crucial to "Alumni-Relations Department," while the second dependency has no impact on "Alumnus" satisfaction. Another metric to measure actors' interactions, proposed by Hassine and Alshayeb

[26], is to compute AED (Actor External Dependency). The AED values for each actor in our running example are computed as follows:

$$AED(\text{Alumni Relations Department}) = \frac{1}{15 - 7}$$

$$= 0.018$$

$$AED(\text{Alumnus}) = \frac{1}{15 - 8}$$

$$= 0.018$$

We notice that both actors have the same AED value. Hence, AED does not reflect the real impact of each dependency link on the satisfaction of the actors. Therefore, these two metrics (i.e., counting external dependencies and AED) fail to measure accurately the impact of the dependencies between the GRL actors. Therefore, there is a need for a metric that takes into account:

- The topology/structure of each actor (intentional elements and link types).
- The explicit and implicit external dependencies of each actor.
- The impact (positive or negative) of explicit/implicit inter-actor dependencies on the overall satisfaction of interacting actors.

2. *Research question RQ2*: How to identify which external dependency has the highest negative/positive impact on a given actor

Rationale for RQ2: Inter-actor dependencies may have either a positive (i.e., beneficial interactions) or a negative (i.e., harmful interactions) impact on actor satisfaction. It would be useful for an analyst to assess the impact of adding/removing inter-actor dependencies on the actors' satisfactions. For example, if the current inter-actor dependencies have a negative impact on an actor, it would be convenient to recognize which dependency, among all, is the most harmful for him. That is, which dependency when removed, would result in the best improvement of the actor's satisfaction. Similarly, it would be helpful to recognize which dependency, among all, is the most beneficial for a given actor. That is, which dependency when removed, would result in the highest deterioration of the actor's satisfaction.

5.3 Proposed approach

An overview of the proposed approach is depicted as an UML activity diagram in Fig. 8. The main building block of the

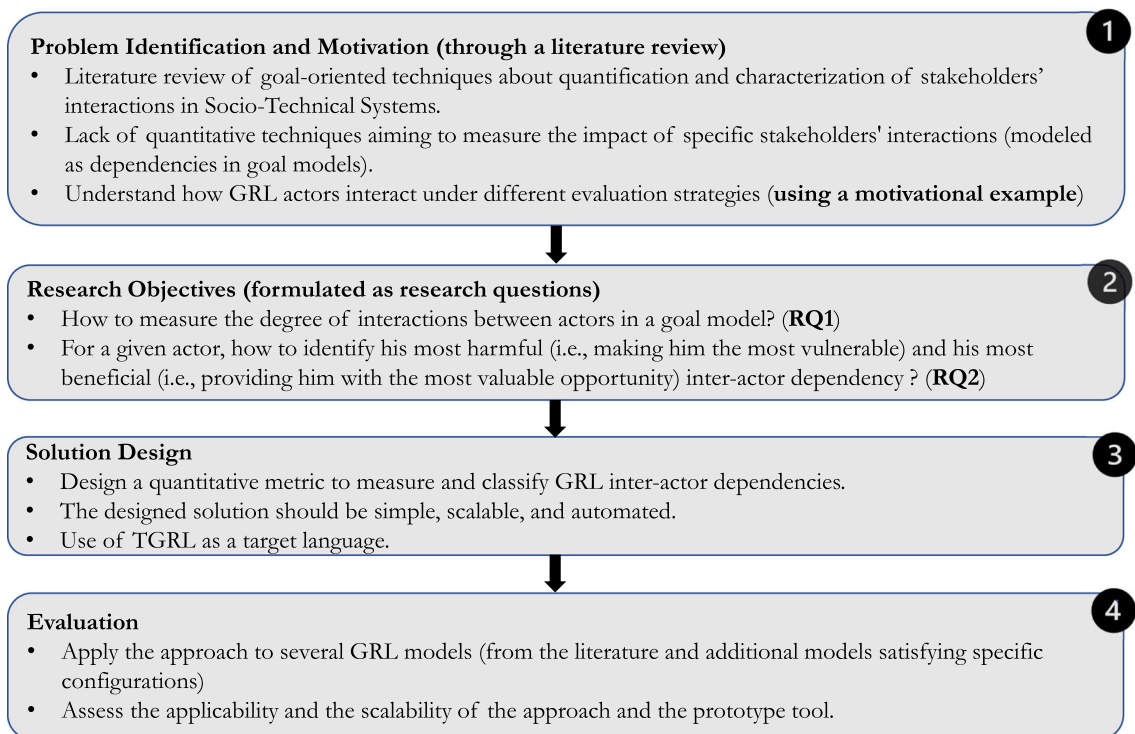


Fig. 7 Overview of the research methodology

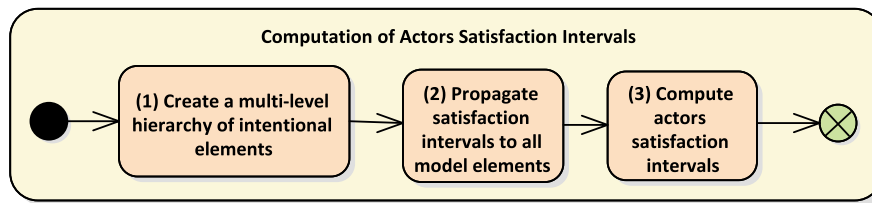
approach is the computation of actors satisfaction intervals, described as a UML Activity (called Computation of Actors Satisfaction Intervals, see Fig. 8a) and composed of three main actions: (1) create a multi-level hierarchy of the model intentional elements, (2) propagate satisfaction intervals to all model elements, (3) compute actors satisfaction intervals. This behavior will be used as a classifier and will be instantiated to compute actors satisfaction intervals in the following three configurations: (1) with all inter-actors dependencies, (2) without all inter-actor dependencies, and (3) without a specific dependency D (described using the comment construct in Fig. 8b).

In GRL satisfaction analysis [9], initial satisfaction values, part of a strategy, are assigned to the intentional elements (often leaf elements) and then these values propagate to the other intentional elements of the model via the various model links. In our approach, instead of assigning one single satisfaction value per leaf element, we consider an interval of values $[a, b] = \{x \in \mathbb{Z} : a \leq x \leq b\}$, where a and b are integers ranging from -100 to $+100$. This choice is motivated by the fact that in the early stages of quantitative satisfaction analysis, accurate satisfaction levels (as one single value) for each leaf element (representing the input to the model) may not be available. Furthermore, the use of satisfaction intervals would allow for more flexibility by exploring ranges of values at once and by computing the worst/best (i.e., lower bound/upper bound) case satisfaction for each intentional

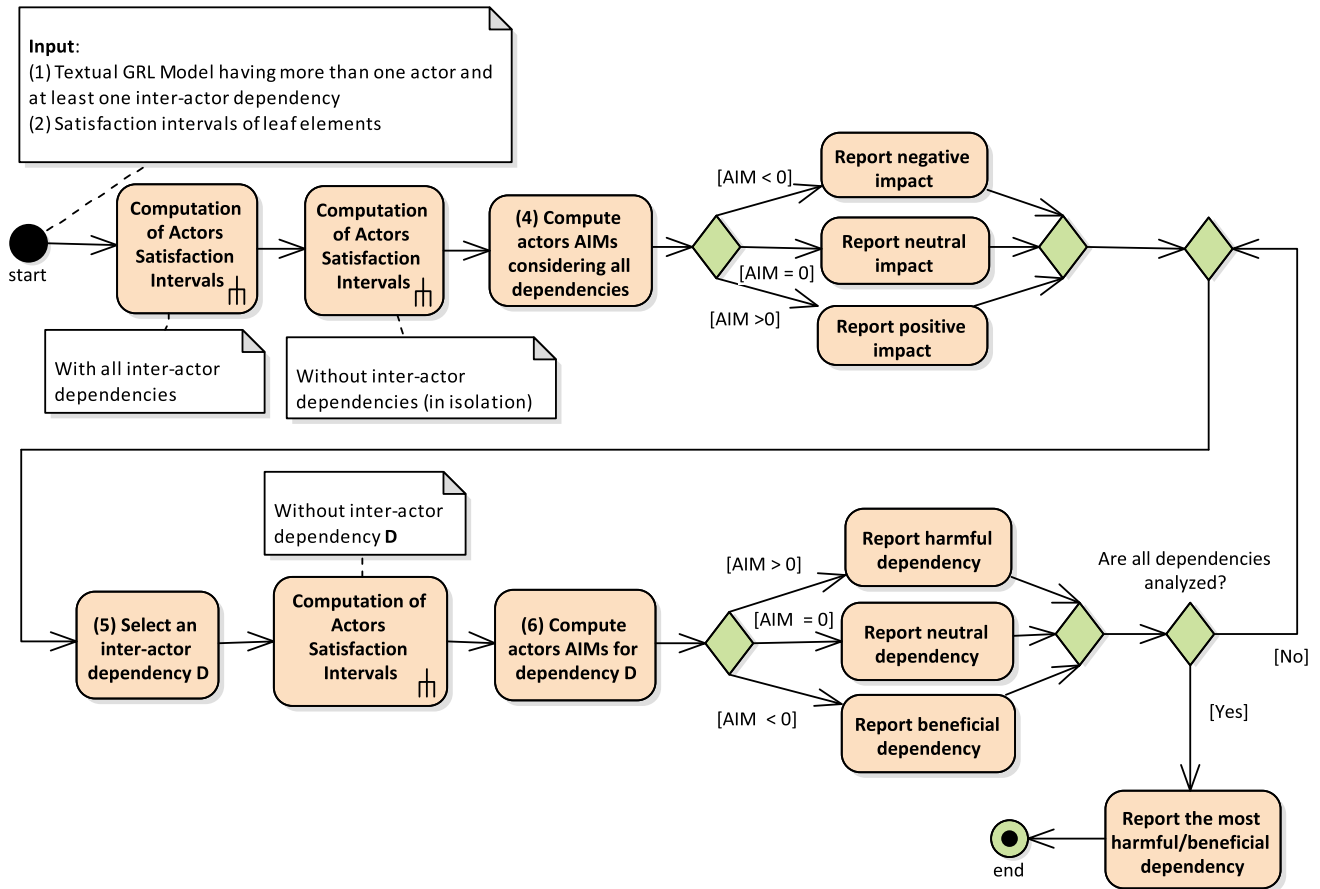
element. The analyst can specify input intervals based on the model semantics and the domain/organizational knowledge (see Sect. 7.3 for more details on the practical use of the proposed approach).

It is worth noting, that an analyst may still enter single values as input (like in the standard GRL satisfaction analysis), by having intervals with the same lower and upper bounds.

As shown in Fig. 8b, the input of the approach is composed of: (1) A TGRL model (having at least 2 actors and one inter-actor dependency), and (2) the satisfaction intervals of the TGRL model leaf elements. In the first phase of the approach, actor satisfaction intervals are computed, with and without inter-actor dependencies, followed by a computation of the actors AIMs (action 4 in Fig. 8b). The overall impact of all inter-actor dependencies is then reported to the user as negative, positive, or neutral. The second phase of the approach starts with the selection of an inter-actor dependency (action 5 in Fig. 8b), then a computation of the actors' satisfaction intervals is performed on the initial model after discarding the selected dependency, and the actors AIMs are obtained (action 6 in Fig. 8b). Next, the impact of the selected inter-actor dependency is reported to the user as harmful, beneficial, or neutral. The second phase is repeated for all inter-actor dependencies. Once completed, the most harmful and the most beneficial dependencies are reported.



(a) Activity: Computation of Actors Satisfaction Intervals



(b) Analysis of GRL actor interactions

Fig. 8 UML activity diagram describing the proposed approach

5.4 Model multi-level hierarchy

The first step to compute actor satisfaction intervals is to rank model intentional elements to create a multi-level hierarchy (action 1 in Fig. 8a). First, the leaf nodes (i.e., elements that have no incoming contribution or decomposition links from within or outside an actor) are placed in level 0. Next, unranked nodes having its children in level 0 are ranked in level 1 (i.e., level is incremented). The process continues until all nodes are ranked. Figure 9 illustrates how nodes of the running example are ranked in a four-level hierarchy.

The multi-level hierarchy can be built for any GRL model structure, not only tree-like (a tree is a connected graph with-

out cycles) models. Indeed, our running example (Fig. 9) is not a tree, since the sequence of elements (“Provide Access to the university facilities,” “Enhanced collaboration with industry,” “Establish research collaboration,” “Give back to university,” “Help current students”) represents an undirected cycle (not a dependency cycle blocking the satisfaction analysis). Furthermore, in the same example of Fig. 9, goal “Help current students” (level 1) is linked to two elements of different levels, i.e., “Give back to university” (level 2) and “Provide Access to the university facilities” (level 0). In addition, the satisfaction interval of “Provide Access to the university facilities” (level 0) contributes to the computation of the satisfaction intervals of two elements belonging

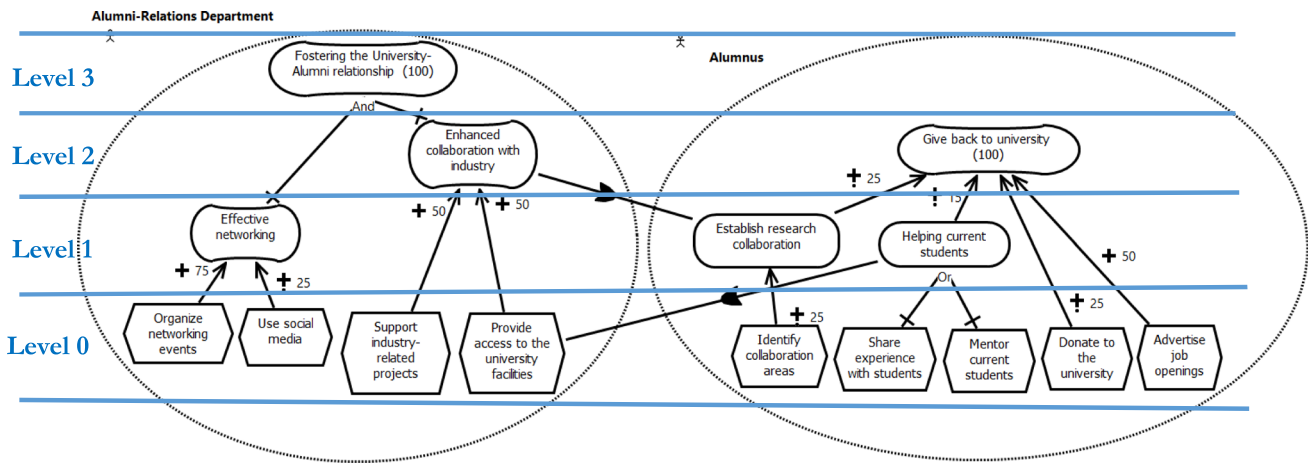


Fig. 9 Multi-level hierarchy of the running example

to two different levels, i.e., “*Help current students*” (level 1) and “*Enhanced collaboration with industry*” (level 2).

The multi-level hierarchy has the following benefits:

- It will help propagate satisfaction intervals one level at a time. The satisfaction interval of an element in level n can only be computed if all satisfaction intervals of the previous levels are computed. For example, in Fig. 9 soft-goal “*Enhanced collaboration with industry*” is placed in level 2 (instead of level 1), because its satisfaction interval requires the computation of the satisfaction interval of goal “*Establish research collaboration*” in level 1.
- The multi-level hierarchy allows us to detect circular dependencies within an actor or between actors. As explained in Sect. 2.2, circular dependencies in goal models are deemed as a bad smell because their presence would block the propagation of satisfaction values.

Algorithm 1 describes the multi-level hierarchy creation procedure.

5.5 Propagation of satisfaction intervals

In what follows, we define how satisfaction intervals propagate through various GRL links, i.e., contributions, decompositions, explicit dependencies, and aggregation of links.

Definition 1 (Decomposition Satisfaction Interval Propagation) Let D_t be a decomposition link of type $t \in \{AND, OR, XOR\}$, refining an intentional element E into k intentional elements E_1, E_2, \dots, E_k . Let $I_{E_j} = [lb_{E_j}, ub_{E_j}]$ be the satisfaction interval of intentional element E_j , where lb denotes the lower bound and ub denotes the upper bound). The satisfaction interval of E , denoted by I_E is computed as follows:

Algorithm 1: Mapping GRL nodes to the multi-level hierarchy

```

Input : TGRL model
Output: Multilevel hierarchy
    ▷ set unRanked contains initially all model nodes
level = 0;
foreach node  $n$  in unRanked do
    if leaf( $n$ ) then
        rank( $n$ , level);
        unRanked ← unRanked \  $n$ ;
        ▷ Increment the level
    level++;
    ▷ newlyRanked contains ranked nodes at each iteration
    newlyRanked ← ∅;
    while (not(empty(unRanked))) do
        foreach node  $n \in$  unRanked do
            ▷ Ranked returns whether a set of nodes is already ranked
            if not(ranked(children( $n$ ))) then
                rank( $n$ , level);
                add( $n$ , newlyRanked);
            if empty(newlyRanked) then
                print(“Model has a cycle!”);
                exit;
            else
                unRanked ← unRanked \ newlyRanked;
                newlyRanked ← ∅;
        if empty(unRanked) then
            ▷ Increment the level
            level++;
    
```

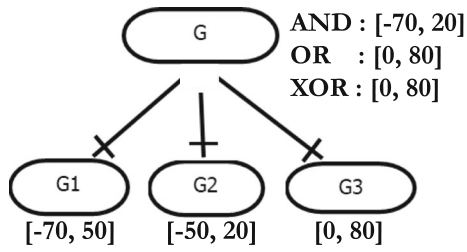


Fig. 10 Example of propagation of satisfaction intervals through decomposition links

- $t = \text{AND} : I_E = [\min_{\forall lb_{E_j}, \forall ub_{E_j}}, \min_{\forall lb_{E_j}, \forall ub_{E_j}}]$.
- $t = \text{OR} : I_E = [\max_{\forall lb_{E_j}, \forall ub_{E_j}}, \max_{\forall lb_{E_j}, \forall ub_{E_j}}]$.
- $t = \text{XOR} : I_E = [\max_{\forall lb_{E_j}, \forall ub_{E_j}}, \max_{\forall lb_{E_j}, \forall ub_{E_j}}]$.

Figure 10 illustrates an example of the propagation of the satisfaction intervals of three goals G1, G2, and G3 through a decomposition link.

The resulting interval of G is computed as follows:

- AND-decomposition:

$$I_G = [\min(lb_{G1}, lb_{G2}, lb_{G3}), \min(ub_{G1}, ub_{G2}, ub_{G3})]$$

$$= [\min(-70, -50, 0), \min(50, 20, 80)]$$

$$= [-70, 20]$$

- OR and XOR decomposition:

$$I_G = [\max(lb_{G1}, lb_{G2}, lb_{G3}), \max(ub_{G1}, ub_{G2}, ub_{G3})]$$

$$= [\max(-70, -50, 0), \max(50, 20, 80)]$$

$$= [0, 80]$$

Definition 2 (Contribution Interval Propagation) Let C_{val} be a GRL contribution link having of quantitative value val over $[-100, +100]$. Let $I_{source} = [lb_{source}, ub_{source}]$ be the satisfaction interval of the GRL intentional element source of C_{val} and $I_{destination} = [lb_{destination}, ub_{destination}]$ be the satisfaction interval of the GRL intentional element destination of C_{val} , resulting from the propagation.

- $val \geq 0 : I_{destination} = [lb_{source} \times val, ub_{source} \times val]$.
- $val \leq 0 : I_{destination} = [ub_{source} \times val, lb_{source} \times val]$.

Figure 11 illustrates an example of the propagation of the satisfaction intervals of two goals G1 and G2, through a Help contribution and a Hurt contribution, respectively.

The propagation of the satisfaction interval of G1 produces:

$$I_G = [lb_{G1} * 25\%, ub_{G1} * 25\%]$$

$$= [-70 * 25\%, 50 * 25\%] = [-17, 12]$$

Fig. 11 Example of propagation of satisfaction intervals through contribution links

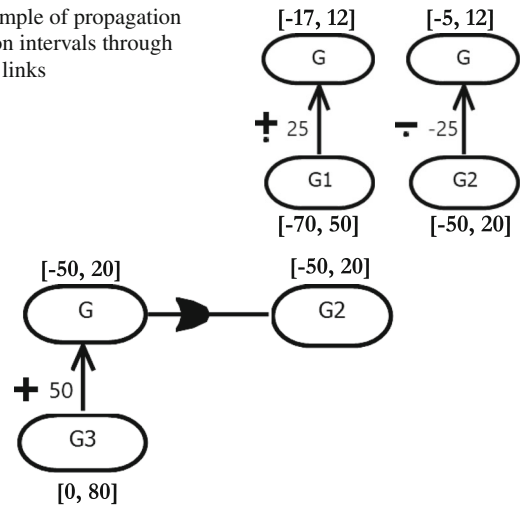


Fig. 12 Example of propagation of satisfaction intervals through dependencies

The propagation of the satisfaction interval of G2 produces:

$$I_G = [lb_{G2} * -25\%, ub_{G2} * -25\%]$$

$$= [20 * -25\%, -50 * -25\%] = [-5, 12]$$

Definition 3 (Dependency Interval Propagation) Let D be a GRL explicit dependency link between *source* and *destination* elements. Let $I_{destination} = [lb_{destination}, ub_{destination}]$ be the satisfaction interval of the GRL intentional element destination of D . The satisfaction interval of the GRL intentional element source of D is defined as: $I_{source} = [\min(lb_{source}, lb_{destination}), \min(ub_{source}, ub_{destination})]$, where $[lb_{source}, ub_{source}]$ represents the satisfaction interval of the source element in the absence of the dependency.

Figure 12 illustrates an example of the impact of a dependency of the satisfaction interval of goal G. Initial satisfaction values of G3 and G2 are $[0, 80]$ and $[-50, 20]$, respectively.

The satisfaction interval of G is computed as follows:

$$I_G = [\min(lb_{G3} * 50\%, lb_{G2}), \min(ub_{G3} * 50\%, ub_{G2})]$$

$$= [\min(0 * 50\%, -50), \min(80 * 50\%, 20)]$$

$$= [\min(0, -50), \min(40, 20)] = [-50, 20].$$

Note that the satisfaction interval of G2 is $[-50, 20]$ and the satisfaction of the depender G cannot exceed the one of the dependee, then the satisfaction interval of G becomes $[-50, 20]$.

Contribution links are additive and convey their satisfaction to the target intentional element. The aggregation of satisfaction intervals is computed as follows:

Definition 4 (Aggregation of satisfaction intervals through contribution links) Let "target" be an intentional element

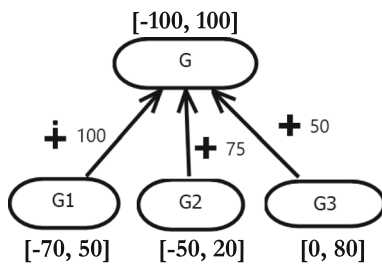


Fig. 13 Example of propagation of satisfaction intervals through an aggregation of contribution links

having n incoming links each carrying a satisfaction interval $I_k = [lb_k, ub_k]$, $k=1..n$. The satisfaction interval of destination is the aggregation of these n intervals and is computed as follows:

$$I_{target} = [\max(\sum_{k=1}^n lb_k, -100), \min(\sum_{k=1}^n ub_k, 100)]$$

Figure 13 illustrates an example of the impact of propagating satisfaction intervals through an aggregation of contribution links. Goal G1 will contribute with the interval $[lb_{G1} * 100\%, ub_{G1} * 100\%] = [-70, 50]$, goal G2 will contribute with the interval $[lb_{G2} * 75\%, ub_{G2} * 75\%] = [-37, 15]$, and goal G3 will contribute with the interval $[lb_{G3} * 50\%, ub_{G3} * 50\%] = [0, 40]$. Hence, the satisfaction interval of goal G is computed as follows (making sure that the interval lower bound does not go below -100 and the upper bound do not exceed 100):

$$I_G = [\max(-70 - 37 + 0, -100), \min(50 + 15 + 40, 100)] \\ = [\max(-107, -100), \min(105, 100)] = [-100, 100]$$

5.6 Computation of actor satisfaction interval

The satisfaction interval of a GRL actor is computed as a weighted average of the satisfaction intervals of its contained intentional elements having non-null importance values.

Definition 5 (Actor Satisfaction Interval) Let A be a GRL actor having n intentional elements with non-null importance values. The satisfaction intervals of these n elements are denoted as: $I_k = [lb_k, ub_k]$, $k=1..n$. The satisfaction interval of A = $[a1, a2]$ is computed as follows:

$$a1 = \frac{\sum_k lb_k \times Importance(element_k)}{\sum_k Importance(element_k)} \\ a2 = \frac{\sum_k ub_k \times Importance(element_k)}{\sum_k Importance(element_k)}$$

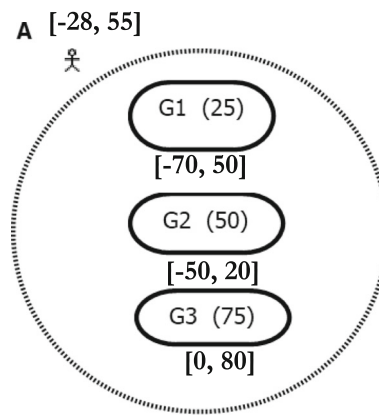


Fig. 14 Example of the satisfaction interval of an actor

Figure 14 illustrates an example of computation of actor A satisfaction interval $[a1, a2]$:

$$a1 = \frac{(-70) * 25 + (-50) * 50 + 0 * 75}{25 + 50 + 75} = -28 \\ a2 = \frac{(50) * 25 + (20) * 50 + 80 * 75}{25 + 50 + 75} = 55$$

5.7 Computation of actor interaction metric (AIM)

In order to address research question RQ1, we define the Actor Interaction Metric (AIM). The AIM metric uses the notion of interval midpoint, which is defined as follows:

Definition 6 (Interval Midpoint) Given an interval $I = [a, b]$, the midpoint of I is defined as:

$$Mp(I) = \frac{a + b}{2}$$

Interval midpoint is a simple mathematical operator of interval arithmetic [46], that helps represent an interval as an expression of its midpoint plus/minus an error bound (i.e., interval radius expressed as $(b - a)/2$); hence $I = Mp(I) \pm radius(I)$.

The actor Interaction Metric (AIM) is then defined as follows:

Definition 7 (Actor Interaction Metric) Let A be an actor interacting with one or many actors. Let $A_{AllDeps}$ be the satisfaction interval of actor A (with all current dependencies) and let A_{isol} be the satisfaction interval of actor A in isolation (without external dependencies). The actor A interaction metric is computed as follows:

$$AIM(A) = Mp(A_{AllDeps}) - Mp(A_{isol})$$

The Actor Interaction Metric (AIM) is designed based on the fact that a satisfaction interval can be extended/reduced

(after the removal of the external dependencies) in the positive direction, negative direction, or in both directions. The magnitude and direction of the extension/reduction will determine whether the external dependencies are harmful, beneficial, or neutral.

It is worth noting, that the interval midpoint value has no semantic meaning outside the context of AIM metric computation and the measurement of the variation of actors' satisfactions with and without inter-actor dependencies. Hence, it should not be linked to the satisfiability of the selected alternative solution. Indeed, different actor satisfaction intervals (resulting from different alternative solutions) can have the same interval midpoint value.

Figure 15a–c illustrates three examples of interval inclusion. Assume that intervals $[a, b]$ and $[c, d]$ represent actor satisfaction intervals with and without all external dependencies, respectively (similar reasoning is applied in the opposite case). In Fig. 15a, compared to $[a, b]$, interval $[c, d]$ represents a deterioration of the upper bound b by “ $b-d$ ” and an improvement of the lower bound a by “ $c-a$,” with “ $b-d > c-a$ ”; hence the actor external dependencies, in this case, have a positive impact on the actor satisfaction. In Fig. 15b, compared to $[a, b]$, interval $[c, d]$ represents a deterioration of the upper bound b by “ $b-d$ ” and an improvement of the lower bound a by “ $c-a$,” with “ $b-d < c-a$ ”; hence the actor external dependencies, in this case, have a negative impact on the actor satisfaction. Figure 15c illustrates a case of neutral impact since the improvement of $[c, d]$ over $[a, b]$, i.e., “ $c-a$ ” is equal to the deterioration, i.e., “ $b-d$.”

The same reasoning applies to intersecting (Fig. 15e) and disjoint (Fig. 15d) intervals. In these two examples, compared to $[a, b]$, interval $[c, d]$ represents an improvement of both the lower bound and the upper bound by “ $c-a$ ” and “ $d-b$,” respectively. Hence the actor external dependencies, in both cases, have a negative impact on the actor satisfaction.

The AIM computation of the initial GRL model is represented by action (4) in the UML activity diagram of Fig. 8b. Based on the value of the AIM for a given actor A, the inter-actor interaction is classified (see Fig. 8b) as positive (i.e., $AIM(A) > 0$), neutral (i.e., $AIM(A) = 0$), or negative (i.e., $AIM(A) < 0$). Hence, RQ1 is addressed.

Furthermore, based on the assessment of the impact of all inter-actor dependencies, an analyst may want to assess the impact of the removal of a dependency. To achieve this goal, we repeat the computation of AIM, while removing one dependency at a time. The resulting AIM relative to a given dependency D is represented by action (4) in the UML activity diagram of Fig. 8b and derived as follows:

$$AIM(A, D) = Mp(A_{AllDepsExceptD}) - Mp(A_{AllDeps})$$

where $Mp(A_{AllDepsExceptD})$ denotes the midpoint of the satisfaction interval of actor A, when the dependency D is

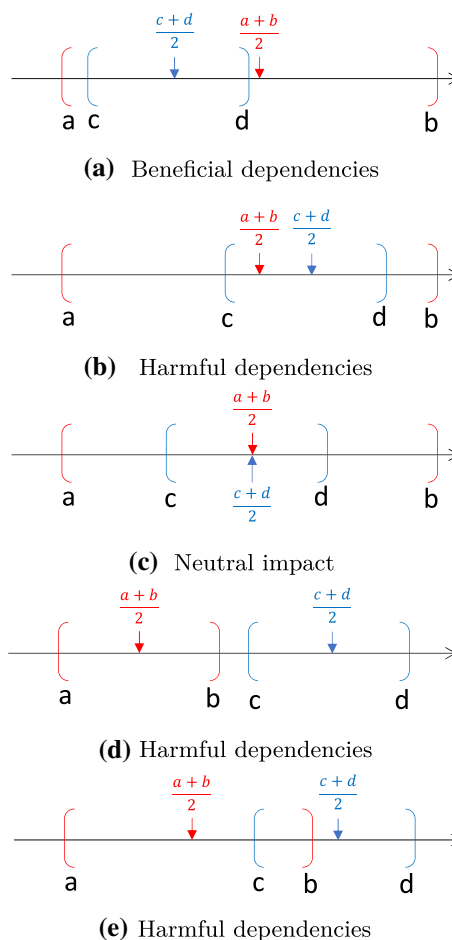


Fig. 15 Examples of dependencies impact

removed. The dependency D is declared as harmful if AIM is positive, beneficial if AIM is negative, or neutral if AIM is equal to zero.

The most harmful dependency for actor A, denoted as $D_{most\ harmful}$, is defined as follows:

$$\forall \text{ dependencies } D, \\ AIM(A, D_{most\ harmful}) > AIM(A, D)$$

Similarly, the most beneficial dependency for actor A, denoted as $D_{most\ beneficial}$, is defined as follows:

$$\forall \text{ dependencies } D, \\ AIM(A, D_{most\ beneficial}) < AIM(A, D)$$

Hence, RQ2 is addressed.

5.8 Applying the approach to the running example

In this section, we present the results of applying our proposed approach to the University–Alumni GRL model of Fig. 1.

It is worth noting that since we compute the AIM for each actor, we have omitted the actor’s name when referring to its AIM, e.g., instead of writing AIM(A), we write AIM. Similarly, we have omitted the actor’s name when referring to the AIM of one specific dependency, e.g., instead of AIM(A, d), we write AIM(d). This has been applied to all result tables throughout the paper. Furthermore, to improve the readability of result tables, harmful dependencies are colored in light coral, the most harmful dependency is colored in red, beneficial dependencies are colored in light green, and the most beneficial dependencies are colored in green.

We have considered two strategies; each corresponds to a set of satisfaction intervals of the model’s leaf elements:

1. Default satisfaction intervals (i.e., $[-100, 100]$) for all leaf elements. In this scenario, the full interval $[-100, 100]$ is used to perform a dry-run simulation in order to evaluate the satisfiability of the intervening actors and get a general assessment of the impact of the inter-actor dependencies, regardless of the chosen alternatives. The results (shown in Table 2) show that the satisfaction interval of actor “Alumni-Relations Department” is $[-100, 25]$, which indicates that regardless of the chosen alternative, the satisfaction of “Alumni-Relations Department” cannot exceed 25 (this has been observed when designing the 12 strategies depicted in Table 1). The overall impact of the two inter-actor dependencies is negative (AIM = -37.5) for “Alumni-Relations Department.” In particular, the first dependency (i.e., *Enhanced collaboration with industry* → *Establish research collaboration*) is deemed to be harmful for “Alumni-Relations Department” with an AIM equal to 37.5 (marked with a red color in Table 2. Both inter-actor dependencies have no impact on the satisfaction interval of actor “Alumnus.”
2. Strategy D3 & A2 (presented in Sect. 2.2.2) with expanded input intervals: this scenario shows the ability of the approach to handle both single and interval-based satisfaction values. We have assigned the $[100, 100]$ interval for the fully satisfied tasks and the $[30, 70]$ interval to the weakly satisfied tasks (instead of the value 50). Like the scenario with default intervals, results (see Table 3) show that the first dependency (i.e., *Enhanced collaboration with industry* → *Establish research collaboration*) is deemed to be harmful for “Alumni-Relations Department” with an AIM equal to -50 . However, the magnitude of its negative impact is even higher (i.e., 50

vs. 37.5). In addition, both inter-actor dependencies have no impact on the satisfaction interval of actor “Alumnus.”

5.9 Automation of the approach

The manual computation of the AIM metric for the entire textual GRL model as well as for every inter-actor dependency is an error-prone activity since it is very difficult to manipulate purely textual specifications. Converting a textual GRL specification into a graphical GRL model then manually propagates the satisfaction intervals may also introduce errors during both the model conversion and the interval propagation. This issue becomes more challenging when dealing with large TGRL models. In addition, the process is often repetitive as the analyst may want to recompute the metric after each model adjustment (see Sect. 7.3). Hence reaching a satisfactory model may be very time-consuming.

To cope with this problem, we have automated the proposed approach as a stand-alone command line tool. The tool is developed in java. It accepts as input: (1) a textual GRL specification file (.turn as extension), and (2) the initial satisfaction interval ranges ($[-100, 100]$, $[0, 100]$, or user defined), and computes the global AIM of each actor of the model (see Part A in Fig. 16a), the impact and the classification of each dependency (see Part B in Fig. 16b), and the most beneficial/harmful dependency if any (Part C is not shown). The tool is publicly available, and it is free to download and use ¹.

6 Evaluation

In this section, we evaluate our proposed approach and prototype tool using several TGRL models. We follow the templates and recommendations presented by Wohlin et al. [58].

6.1 Experiment planning

The general goal of the evaluation is: “To analyze the applicability and scalability of our proposed approach and tool in measuring GRL actor interactions.”

The subjects of the experiment are 13 TGRL models of different sizes, structures, and complexity. Table 4 provides the characteristics of these models in terms of:

- The number of GRL actors in the model.
- The number of intentional elements (goal, softgoal, task, etc.).
- The number of explicit inter-actor dependencies.

¹ <https://github.com/jhassine74/AIM>

Table 2 Running example: default satisfaction interval ([-100, 100])

AIM and interval computation	Actors	
	Alumni-Relations Department	Alumnus
All dependencies	[-100, 25]	[-100, 100]
Isolation	[-100, 100]	[-100, 100]
AIM	-37.5 (negative)	0
Remove dep. #1 (<i>Enhanced..</i> → <i>Establish..</i>)	[-100, 100]	[-100, 100]
AIM (<i>Enhanced..</i> → <i>Establish..</i>)	50	0
Remove dep. #2 (<i>Helping..</i> → <i>Provide..</i>)	[-100, 25]	[-100, 100]
AIM (<i>Helping..</i> → <i>Provide..</i>)	0	0

Table 3 Running example: strategy D2 & A3 with expanded input intervals

Running example input intervals		
Alumni-Relations Department:		
Use social media: [100, 100]		
Provide access to the university facilities: [100, 100]		
Organize networking events: [30, 70]		
Support industry-related projects: [30, 70]		
Alumnus:		
Identify collaboration areas: [30, 70]		
Share experience with students: [30, 70]		
Mentor current students: [30, 70]		
Donate to the university: [30, 70]		
Advertise job openings: [30, 70]		

AIM and interval computation	Actors	
	Alumni-Relations Department	Alumnus
All dependencies	[7, 17]	[43, 100]
Isolation	[47, 77]	[43, 100]
AIM	-50 (negative)	0
Remove dep. #1 (<i>Enhanced..</i> → <i>Establish..</i>)	[47, 77]	[43, 100]
AIM (<i>Enhanced..</i> → <i>Establish..</i>)	37.5	0
Remove dep. #2 (<i>Helping..</i> → <i>Provide..</i>)	[7, 17]	[43, 100]
AIM (<i>Helping..</i> → <i>Provide..</i>)	0	0

- The number of implicit inter-actor dependencies (i.e., inter-actor contribution and decomposition links).

Eight GRL models (#1 to #8) are taken from the literature, while the remaining 5 models are generic models that have been created to address the scalability of the approach (models #9 and #13) and to test the ability of the tool to detect circular dependencies (i.e., models #10, #11, #12). The 8 publicly available models have 2 to 3 interacting actors, while the generic model #9 has 6 actors with a total of 8 inter-actor dependencies.

For a given GRL model, the propagation of the satisfaction intervals and the computation of AIMs are performed [2 (i.e., with and without all dependencies) + number of

inter-actor dependencies] times. Hence, in addition to the size of the model (number of intentional elements and links) and the number of actors (AIMs are computed for each actor), which affect one single run, the number of dependencies in the model is crucial to demonstrate the scalability of the approach. To this end, in addition to Model #9 (6 actors and 8 inter-actor dependencies), we have introduced Model #13 (Fig. 30) that encloses 14 inter-actor dependencies (although, in practice, such high number of dependencies is less likely).

It is worth noting that models #1 [39], #2 [51], #3 [25], #5 [32], #7 [35], and #8 [33] are originally *i** [60] models that were converted to GRL notation.

```

=====
PART A:(All Dependencies vs Isolation)
=====
Actors' satisfaction intervals:
Actor AlumniRelations:
All Dependencies:      [-100.0, 25.0]
Isolation:             [-100.0, 100.0]
AIM (AIM(A) = Mp(A_AllDeps) - Mp(A_isol)): -37.5
Interaction type: Negative

Actor Alumnus:
All Dependencies:      [-100.0, 100.0]
Isolation:             [-100.0, 100.0]
AIM (AIM(A) = Mp(A_AllDeps) - Mp(A_isol)): 0.0
Interaction type: Neutral

```

(a) AIM Tool: Part A output: Global Actor AIM

```

=====
PART B:(All Dependencies Except D vs All Dependencies)
=====
D1) Remove dependency link between EnhancedCollIndustry
--> EstablishResearchCollaboration
Important Elements satisfaction intervals:
FosteringUnivAlumniRel:
All Dependencies:      [-100, 100]
Isolation:             [-100, 100]

GiveBackUniversity:
All Dependencies:      [-100, 100]
Isolation:             [-100, 100]

Actors' satisfaction intervals:
Actor AlumniRelations:
All Dependency Except D1: [-100.0, 100.0]
All Dependencies:        [-100.0, 25.0]
AIM (AIM(A,D) = Mp(A_AllDepsExceptD) - Mp(A_AllDeps)) : 37.5
Dependency impact: Harmful

Actor Alumnus:
All Dependency Except D1: [-100.0, 100.0]
All Dependencies:        [-100.0, 100.0]
AIM (AIM(A,D) = Mp(A_AllDepsExceptD) - Mp(A_AllDeps)) : 0.0
Dependency impact: Neutral
=====
D2) Remove dependency link between helpingcurrentstudents
--> provideaccessfacilities
Important Elements satisfaction intervals:
FosteringUnivAlumniRel:
All Dependencies:      [-100, 25]
Isolation:             [-100, 100]

GiveBackUniversity:
All Dependencies:      [-100, 100]
Isolation:             [-100, 100]

Actors' satisfaction intervals:
Actor AlumniRelations:
All Dependency Except D2: [-100.0, 25.0]
All Dependencies:        [-100.0, 25.0]
AIM (AIM(A,D) = Mp(A_AllDepsExceptD) - Mp(A_AllDeps)) : 0.0
Dependency impact: Neutral

Actor Alumnus:
All Dependency Except D2: [-100.0, 100.0]
All Dependencies:        [-100.0, 100.0]
AIM (AIM(A,D) = Mp(A_AllDepsExceptD) - Mp(A_AllDeps)) : 0.0
Dependency impact: Neutral

```

(b) AIM Tool: Part B output: Impact of each dependency

Fig. 16 Excerpt of the AIM tool output

The presence of circular dependencies (i.e., cycles) in GRL models hinders the propagation of satisfaction values throughout the model [10]. Similarly, in our inter-actor dependency analysis context, cycles would prevent satisfaction intervals from propagating to all elements of the model. To test the ability of our tool in detecting cycles, we have designed three generic models: (1) model #10: cycle composed of explicit dependencies, (2) model #11: cycle composed of implicit dependencies, and (3) model #12: cycle composed of both explicit and implicit dependencies. It is worth noting that, although based on i^* [60], the GRL language is permissive when it comes to the use of explicit/implicit dependencies intra/inter-actors, and with respect to the types of leaf/roots intentional elements and how they are linked together.

6.2 Experimental tasks

Two experiments have been conducted using the 13 TGRL specifications described in Table 4. For experimental purposes and given the lack of specific candidate strategies that are supported by real design rationales, we have opted for two default intervals $[-100, 100]$ and $[0, 100]$ (as satisfaction input assigned to the leaf elements of the model). In the first experiment (referred to as Experiment 1), the $[-100, 100]$ interval was used to perform a dry-run simulation in order to evaluate the satisfiability of the model's actors and get an overall assessment of the type and magnitude of the impact of the inter-actor dependencies, regardless of the chosen alternatives.

Generally, an actor doesn't tend to deny its designed leaf elements that were obtained from the operationalization of the actor's goals (i.e., tasks) or through the refinement of coarse-grained goals. Hence, in the second experiment (referred to as Experiment 2), we have used $[0, 100]$ as input satisfaction intervals, discarding both the full denial (i.e., satisfaction equal to -100) and the weak denial (satisfaction within $[-100, -1]$) of the model's leaf elements.

The AIM metric is computed for the whole model and for each inter-actor dependency link.

6.3 Experiment results

In this section, we present and discuss the results of both experiments.

6.3.1 Experiment 1 results

Tables 5, 6, and 7 report the results of the first experiment for models having only two actors: model #1 (Fig. 18), #2 (Fig. 19), #3 (Fig. 20), #6 (Fig. 23), #7 (Fig. 24), and #13 (Fig. 30). Table 8 shows the results of the first experiment for models having three actors (models #4 (Fig. 21), #5 (Fig. 22),

Table 4 Characteristics of the selected GRL models

#	Model	Number of GRL actors	Number of intentional elements	Number of explicit inter-actor dependencies	Number of implicit inter-actor dependencies
1	Telecom system [39] (Fig. 18)	2	9	1	0
2	Tele-medicine system [51] (Fig. 19)	2	14	1	0
3	Waste management system [25] (Fig. 20)	2	20	2	1
4	Wireless system [10] (Fig. 21)	3	16	1	6
5	Youth counseling [32] (Fig. 22)	3	23	4	0
6	Hybrid car system [11] (Fig. 23)	2	21	0	4
7	Technology system [35] (Fig. 24)	2	9	3	0
8	PC system [33] (Fig. 25)	3	12	4	0
9	Large # of actors (Generic model) (Fig. 26)	6	21	4	4
10	Generic cyclic model (cycle composed of explicit dependencies) (Fig. 27)	2	7	2	0
11	Generic cyclic model (cycle composed of implicit dependencies) (Fig. 28)	2	7	0	2
12	Generic cyclic model (cycle composed of both explicit and implicit dependencies) (Fig. 29)	2	7	1	1
13	Large model with large number of dependencies (Generic model) (Fig. 30)	2	110	10	4

Table 5 Experiment 1 results (models having 2 actors)—Part 1

Model 1 [39]	Actors	
	Telecom provider	Technician
All dependencies	[− 67, 67]	[− 100, 100]
Isolation	[− 67, 67]	[− 100, 100]
AIM	0 (neutral)	0 (neutral)
Remove <i>VoiceConn</i> → <i>LoggEquip</i>	[− 67, 67]	[− 100, 100]
AIM (<i>VoiceConn</i> → <i>LoggEquip</i>)	0 (neutral)	0 (neutral)
Model 2 [51]	Patient	Healthcare provider
All dependencies	[− 75, 43]	[− 75, 75]
Isolation	[− 75, 75]	[− 75, 75]
AIM	− 16 (negative)	0 (neutral)
Remove <i>QualityOfCare</i> → <i>ViableHealthcare</i>	[− 75, 75]	[− 75, 75]
AIM (<i>QualityOfCare</i> → <i>ViableHealthcare</i>)	16	0 (neutral)
Model 3 [25]	City	Citizen
All dependencies	[− 41, 41]	[− 50, 50]
Isolation	[− 41, 41]	[− 53, 53]
AIM	0 (neutral)	0 (neutral)
Remove <i>Process Green Waste</i> → <i>Quality of Waste</i>	[− 41, 41]	[− 53, 53]
AIM (<i>Process Green Waste</i> → <i>Quality of Waste</i>)	0 (neutral)	0 (neutral)
Remove <i>Willingness to Separate Waste</i> → <i>GW Education</i>	[− 41, 41]	[− 53, 53]
AIM (<i>Willingness to Separate Waste</i> → <i>GW Education</i>)	0 (neutral)	0 (neutral)
Remove <i>Positive City Image</i> → <i>Enjoy City</i>	[− 41, 41]	[− 50, 50]
AIM (<i>Positive City Image</i> → <i>Enjoy City</i>)	0 (neutral)	0 (neutral)

and #8 (Fig. 25)). Table 9 illustrates the results of the first experiment for the model #9 (Fig. 26), having 6 actors. Models #10 (Fig. 27), #11 (Fig. 28), and #12 (Fig. 29) contain circular dependencies and these cycles were successfully detected by the prototype tool. The results of the first experiment are summarized as follows:

- In models #1 (Fig. 18), #3 (Fig. 20), #6 (Fig. 23), #7 (Fig. 24), and #8 (Fig. 25), inter-actor dependencies have no global impact on the satisfaction of the actors (i.e., AIM for all actors and for all dependencies is 0, which is classified as *neutral*).
- Model #2 has one single dependency between actors Patient and Healthcare Provider. This dependency has a negative impact on the satisfaction of actor Patient (i.e., AIM = -16). The removal of this dependency will increase the midpoint of the satisfaction of Patient by 16 points.
- In Model #4 (Fig. 21), actor Vendor has no outgoing dependencies (AIM = 0, as expected). Actor System has only one outgoing dependency with no impact on its overall satisfaction (AIM = 0). Actor Service Provider has 6 incoming dependencies, having a positive impact on his satisfaction (AIM = 6). The removal of the

implicit dependency (correlation) between *Service in SCP* and *Min Switch Load*, would improve the satisfaction of the model by 2 points (AIM = 8). However, the best improvement for actor Service Provider is realized by removing the implicit dependency (correlation) between *Service in SCP* and *MinMessage Exchange* (AIM = 19, cell colored in red in Table 8). The removal of the explicit dependency, or the implicit dependency between *Data in SCP* and *MinChanges* have no impact (AIM = 0). The most valuable dependency for actor Service Provider is the one between *Service in ControlSwitch* and *MinMessage Exchange*, since its removal will impact negatively his satisfaction by 24.5 points (AIM = -24.5, cell colored in green in Table 8).

- Model #5 (Fig. 22) has 4 explicit inter-actor dependencies. These dependencies have no impact on actors Kids & Youth and Counselors, i.e., AIM = 0, while they impact negatively actor Organization, i.e., AIM = -6.5. The dependency between *highQualityCounseling(Org)* and *HighQualityCounseling* is the harmful dependency since its removal would improve the interaction by 6.5 points, i.e., AIM = 6.5, while the removal of other dependencies would not impact the actor satisfaction, i.e., AIM = 0.

Table 6 Experiment 1 results (models having 2 actors)—Part 2

Model 6 [11]	Actors	
	User	System
All dependencies	[- 25, 25]	[- 100, 100]
Isolation	[- 25, 25]	[- 100, 100]
AIM	0 (neutral)	0 (neutral)
Remove <i>Electric Engine</i> → <i>Comfortable</i>	[- 50, 50]	[- 100, 100]
AIM (<i>Electric Engine</i> → <i>Comfortable</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>Electric Engine</i> → <i>Reduce</i>	[0, 0]	[- 100, 100]
AIM (<i>Electric Engine</i> → <i>Reduce</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>Fuel Engine</i> → <i>Comfortable</i>	[- 100, 100]	[- 100, 100]
AIM (<i>Fuel Engine</i> → <i>Comfortable</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>Fuel Engine</i> → <i>Reduce</i>	[- 50, 50]	[- 100, 100]
AIM (<i>Fuel Engine</i> → <i>Reduce</i>)	0 (neutral)	0 (neutral)
Model 7 [35]	Technology provider	Technology user
All dependencies	[- 100, 25]	[- 100, 100]
Isolation	[- 100, 25]	[- 100, 100]
AIM	0 (neutral)	0 (neutral)
Remove dependency <i>Sell tech for profit</i> → <i>Purchase Technology</i>	[- 100, 25]	[- 100, 100]
AIM (<i>Sell tech for profit</i> → <i>Purchase Technology</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>Produce tech</i> → <i>Purchase Technology</i>	[- 100, 25]	[- 100, 100]
AIM (<i>Produce tech</i> → <i>Purchase Technology</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>tech users abide</i> → <i>Abide by Licensing</i>	[- 100, 25]	[- 100, 100]
AIM (<i>tech users abide</i> → <i>Abide by Licensing</i>)	0 (neutral)	0 (neutral)

- Model #9 (Fig. 26) has 6 actors and 8 inter-actor dependencies. The dependencies of the model have a negative impact on Actors A1, A2, A3, A4, and A5, while they have no impact on A6. A1 is the most negatively impacted actor (i.e., AIM=-30.5), while A2 is the least impacted (i.e., AIM=-2). The dependency between G2 and G4 was found to be the most harmful dependency to A1, since its removal would lead to an AIM of 30.5. The dependency between G4 and G6 is the most beneficial dependency to A1, since its removal deteriorates the AIM of A1, i.e., AIM = -7. Actor A2 has two harmful dependencies of the same magnitude, i.e., AIM = 2 for both. For actor A3, the most harmful dependency is between G11 and G12, since its removal improves the AIM by 19 points (cell colored in red in Table 9).
- Model #13 (Fig. 30) has two actors and 14 inter-actor dependencies. These dependencies have a global negative impact on both actors A and B, with B being the most affected (AIM of -50 for B vs. an AIM of -15 for A). The model accounts for 5 harmful dependencies for both actors plus one harmful dependency that affects actor B only. The dependency between GA26 and GB14 was found to be the most harmful dependency to actor A (its removal improves the AIM by 5.5 points), while the

dependency between GA21 and GB14 was found to be the most harmful to actor B (its removal improves the AIM by 7.5 points). The positive contribution between SB2 and GA1 was found to be beneficial for A (the only one), while the positive contribution between GA5 and SB1 was found to be beneficial for B only. The most beneficial dependency for both actors A and B (i.e., between GB11 and GA26) provides an improvement of 1.5 and 3.5 points for actors A and B, respectively (cell colored in dark green in Table 7).

6.3.2 Experiment 2 results

In the second experiment, we use intervals [0, 100] instead of [- 100, 100]. Tables 5, 6 and 7 report the results of the second experiment for models having only two actors: models #1 (Fig. 18), #2 (Fig. 19), #3 (Fig. 20), #6 (Fig. 23), #7 (Fig. 24), and #13 (Fig. 30). Table 8 shows the results of the second experiment for models having three actors (models #4 (Fig. 21), #5 (Fig. 22), and #8 (Fig. 25)). Table 9 illustrates the results of the second experiment for the model #9 (Fig. 26), having 6 actors. Models #10 (Fig. 27), #11 (Fig. 28), and #12 (Fig. 29) contain circular dependencies and these cycles

Table 7 Experiment 1 results (models having 2 actors)—Part3

Model 13	Actors	
	A	B
All dependencies	[− 100, 64]	[− 100, 36]
Isolation	[− 68, 62]	[− 45, 81]
AIM	− 15 (negative)	− 50 (negative)
Remove <i>GA5</i> → <i>GB6</i>	[− 100, 68]	[− 100, 40]
AIM (<i>GA5</i> → <i>GB6</i>)	2	2
Remove <i>GA11</i> → <i>GB6</i>	[− 100, 66]	[− 100, 39]
AIM (<i>GA11</i> → <i>GB6</i>)	1.0	1.5
Remove <i>GA21</i> → <i>GB14</i>	[− 100, 64]	[− 85, 36]
AIM (<i>GA21</i> → <i>GB14</i>)	0 (neutral)	7.5
Remove <i>GA26</i> → <i>GB14</i>	[− 100, 75]	[− 100, 45]
AIM (<i>GA26</i> → <i>GB14</i>)	5.5	4.5
Remove <i>SB2</i> → <i>GA9</i>	[− 100, 66]	[− 100, 39]
AIM (<i>SB2</i> → <i>GA9</i>)	1.0	1.5
Remove <i>GB6</i> → <i>GA16</i>	[− 100, 64]	[− 100, 36]
AIM (<i>GB6</i> → <i>GA16</i>)	0 (neutral)	0 (neutral)
Remove <i>GB10</i> → <i>GA16</i>	[− 100, 64]	[− 100, 36]
AIM (<i>GB10</i> → <i>GA16</i>)	0 (neutral)	0 (neutral)
Remove <i>GB10</i> → <i>GA21</i>	[− 100, 64]	[− 100, 36]
AIM (<i>GB10</i> → <i>GA21</i>)	0 (neutral)	0 (neutral)
Remove <i>GB14</i> → <i>TA6</i>	[− 100, 75]	[− 100, 45]
AIM (<i>GB14</i> → <i>TA6</i>)	5.5	4.5
Remove <i>TB1</i> → <i>GA26</i>	[− 100, 64]	[− 100, 36]
AIM (<i>TB1</i> → <i>GA26</i>)	0 (neutral)	0 (neutral)
Remove <i>GA1</i> → <i>SB2</i>	[− 87, 49]	[− 100, 36]
AIM (<i>GA1</i> → <i>SB2</i>)	-1.0	0 (neutral)
Remove <i>GA5</i> → <i>SB2</i>	[− 100, 64]	[− 100, 36]
AIM (<i>GA5</i> → <i>SB2</i>)	0 (neutral)	0 (neutral)
Remove <i>SB1</i> → <i>GA5</i>	[− 100, 64]	[− 100, 34]
AIM (<i>SB1</i> → <i>GA5</i>)	0 (neutral)	-1.0
Remove <i>GB11</i> → <i>GA26</i>	[− 100, 61]	[− 100, 29]
AIM (<i>GB11</i> → <i>GA26</i>)	-1.5	-3.5

were successfully detected by the prototype tool. The results of the second experiment are summarized as follows:

- In models #1, #5, #7, and #8, inter-actor dependencies have no global impact on the satisfaction of the actors (i.e., AIM=0 (neutral) for all actors and for all removals of dependencies).
- Model #2 has one single dependency between actors Patient and Healthcare Provider. This dependency has a negative impact on the satisfaction of actor Patient (i.e., AIM = − 16). The removal of this dependency will

increase the midpoint of the satisfaction of Patient by 16 points.

- Model #3 has three dependencies between actors City and Citizen. These three dependencies have no impact on the satisfaction of City (i.e., AIM=0 (neutral)), while they exhibit a slightly positive impact on Citizen (i.e., AIM = 1.5). The most beneficial dependency for actor Citizen is the one between *Positive City Image*→*Enjoy City*, since its removal would have a negative impact on his satisfaction (AIM = − 1.5).
- In model #4, actor Vendor has no outgoing dependencies (AIM = 0 as expected). Actor System has only one

Table 8 Experiment 1 results (models having 3 actors)

Model 4 [10]	Service provider	System	Vendor
All dependencies	[− 68, 14]	[− 100, 100]	[− 100, 100]
Isolation	[− 68, 2]	[− 100, 100]	[− 100, 100]
AIM	6 (positive)	0 (neutral)	0 (neutral)
Remove <i>Data in New Service Node</i> → <i>ServiceNode</i>	[− 68, 14]	[− 100, 100]	[− 100, 100]
AIM (<i>Data in New Service Node</i> → <i>ServiceNode</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove dependency <i>Data in SN</i> → <i>MinChanges</i>	[− 70, 16]	[− 100, 100]	[− 100, 100]
AIM (<i>Data in SN</i> → <i>MinChanges</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove dependency <i>Data in SCP</i> → <i>MinChanges</i>	[− 66, 12]	[− 100, 100]	[− 100, 100]
AIM (<i>Data in SCP</i> → <i>MinChanges</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>Service in ControlSwitch</i> → <i>MinMessageExchange</i>	[− 68, − 35]	[− 100, 100]	[− 100, 100]
AIM (<i>Service in ControlSwitch</i> → <i>MinMessageExchange</i>)	−24.5	0 (neutral)	0 (neutral)
Remove dependency <i>Service in SCP</i> → <i>MinMessageExchange</i>	[− 68, 52]	[− 100, 100]	[− 100, 100]
AIM (<i>Service in SCP</i> → <i>MinMessageExchange</i>)	19	0 (neutral)	0 (neutral)
Remove dependency <i>Service in SCP</i> → <i>MinSwitchLoad</i>	[− 68, 30]	[− 100, 100]	[− 100, 100]
AIM (<i>Service in SCP</i> → <i>MinSwitchLoad</i>)	8	0 (neutral)	0 (neutral)
Remove <i>Service in ControlSwitch</i> → <i>MinSwitchLoad</i>	[− 68, − 2]	[− 100, 100]	[− 100, 100]
AIM (<i>Service in ControlSwitch</i> → <i>MinSwitchLoad</i>)	−8	0 (neutral)	0 (neutral)
Model 5 [32]	Organization	Kids & Youth	Counselors
All dependencies	[− 29, 16]	[− 65, 65]	[− 47, 47]
Isolation	[− 16, 16]	[− 65, 65]	[− 47, 47]
AIM	− 6.5 (negative)	0 (neutral)	0 (neutral)
Remove <i>highQualityCounseling(Org)</i> → <i>highQualityCounseling</i>	[− 16, 16]	[− 65, 65]	[− 47, 47]
AIM (<i>highQualityCounseling(Org)</i> → <i>highQualityCounseling</i>)	6.5	0 (neutral)	0 (neutral)
Remove dependency <i>textMessage(Org)</i> → <i>textMessage(C)</i>	[− 29, 16]	[− 65, 65]	[− 47, 47]
AIM (<i>textMessage(Org)</i> → <i>textMessage(C)</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>kidsUseCyberCafe(Org)</i> → <i>kidsUseCyberCafe(C)</i>	[− 29, 16]	[− 65, 65]	[− 47, 47]
AIM (<i>kidsUseCyberCafe(Org)</i> → <i>kidsUseCyberCafe(C)</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove dependency <i>kidsUseCyberCafe3</i> → <i>kidsUseCyberCafe</i>	[− 29, 16]	[− 65, 65]	[− 47, 47]
AIM (<i>kidsUseCyberCafe3</i> → <i>kidsUseCyberCafe</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Model 8 [33]	PC User	Data Pirate	PC Provider
All dependencies	[− 100, 100]	[− 100, 100]	[− 100, 31]
Isolation	[− 100, 100]	[− 100, 100]	[− 100, 31]
AIM	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>purchase PC product</i> → <i>produce PC products</i>	[− 100, 100]	[− 100, 100]	[− 100, 31]
AIM (<i>purchase PC product</i> → <i>producepcproducts</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>obtain PC from data pirate</i> → <i>make content available</i>	[− 100, 100]	[− 100, 100]	[− 100, 31]
AIM (<i>obtain PC from data pirate</i> → <i>make content available</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>make content available</i> → <i>allow p2p</i>	[− 100, 100]	[− 100, 100]	[− 100, 31]
AIM (<i>make content available p2p</i> → <i>allow p2p</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>pc users abide by licensing</i> → <i>Abide by licensing</i>	[− 100, 100]	[− 100, 100]	[− 100, 31]
AIM (<i>cuser abide by rules</i> → <i>abide by rules</i>)	0 (neutral)	0 (neutral)	0 (neutral)

Table 9 Experiment 1 results (Model having 6 GRL actors)

Model 9	Actors					
	A1	A2	A3	A4	A5	A6
All dependencies	[- 100, 39]	[- 43, 39]	[- 75, 18]	[- 87, 65]	[- 100, 75]	[- 25, 25]
Isolation	[- 100, 100]	[- 25, 25]	[- 75, 75]	[- 87, 87]	[- 75, 75]	[- 25, 25]
AIM	- 30.5 (negative)	- 2 (negative)	- 28.5 (negative)	- 11 (negative)	- 12.5 (negative)	0 (neutral)
Remove G2—G4	[- 100, 100]	[- 43, 39]	[- 75, 18]	[- 87, 65]	[- 100, 75]	[- 25, 25]
AIM (G2—G4)	30.50	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove T3 — T7	[- 100, 39]	[- 43, 39]	[- 75, 18]	[- 87, 65]	[- 100, 75]	[- 25, 25]
AIM (T3 — T7)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G7 — G9	[- 100, 43]	[- 43, 43]	[- 75, 18]	[- 87, 87]	[- 100, 75]	[- 25, 25]
AIM (G7 — G9)	2	2	0 (neutral)	11	0 (neutral)	0 (neutral)
Remove G9 — T8	[- 100, 39]	[- 43, 39]	[- 75, 18]	[- 87, 65]	[- 75, 75]	[- 25, 25]
AIM (G9 — T8)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	12.50	0 (neutral)
Remove G4 — G6	[- 100, 25]	[- 25, 25]	[- 75, 18]	[- 87, 65]	[- 100, 75]	[- 25, 25]
AIM (G4 — G6)	- 7	2	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G2—G5	[- 100, 39]	[- 43, 39]	[- 75, 18]	[- 87, 65]	[- 100, 75]	[- 25, 25]
AIM (G2—G5)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G11—G8	[- 100, 39]	[- 43, 39]	[- 75, 18]	[- 87, 65]	[- 100, 75]	[- 25, 25]
AIM (G11—G8)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G11—G12	[- 100, 39]	[- 43, 39]	[- 75, 56]	[- 87, 65]	[- 100, 75]	[- 25, 25]
AIM (G11—G12)	0 (neutral)	0 (neutral)	19	0 (neutral)	0 (neutral)	0 (neutral)

outgoing dependency with no impact on its overall satisfaction ($AIM = 0$). Actor *ServiceProvider* has 6 incoming dependencies, having a positive impact on his satisfaction ($AIM = 6$). The removal of the implicit dependency (correlation) between *Service in SCP* and *Min Switch Load*, would improve the satisfaction of the model by 2 points ($AIM = 8$). However, the best improvement for actor *Service Provider* is realized by removing the implicit dependency (correlation) between *Service in SCP* and *MinMessage Exchange* ($AIM = 19$). The removal of the explicit dependency, or the implicit dependency between *Data in SCP* and *MinChanges* will have no impact ($AIM = 0$). The most valuable dependency for actor *Service Provider* is the one between *Service in ControlSwitch* and *MinMessage Exchange*, since its removal will impact negatively his satisfaction by 24.5 points ($AIM = -24.5$).

- Model #6 has 4 inter-actor dependencies, having no impact on actor *System*, i.e., $AIM = 0$ (neutral). The most harmful dependencies for actor *System* are (1) *Electric Engine* → *Comfortable* and (2) *Fuel Engine* → *Reduce*, since the removal of each one of them would improve the satisfaction of *System* by 12.5 points ($AIM = 12.5$). The most beneficial dependencies for actor *System* are (1) *Electric Engine* → *Reduce* and (2) *Fuel Engine* → *Comfortable*, since the removal of each one of them would impact negatively the satisfaction of *System* by 12.5 points ($AIM = -12.5$).
- Model #9 has 6 actors and 8 dependencies. The dependencies of the model have a negative impact on Actors A1, A2, A3, and A4, while they have no impact on A5 and A6. A1 is the most negatively impacted actor (i.e., $AIM = -30.5$), while A2 is the least impacted (i.e., $AIM = -2$). The dependency between *G2* and *G4* was found to be the most harmful dependency to A1, since its removal would lead to an AIM of 30.5. The dependency between *G4* and *G6* is the most beneficial dependency to A1 and A2, since its removal deteriorates the AIM s of A1 and A2, i.e., $AIM = -7$. For actor A3, the most harmful dependency is between *G11* and *G12*, since its removal improves the AIM by 19 points.
- For Model #13 (Fig. 30), the 14 inter-actor dependencies have an overall positive impact on actor A (AIM of 1) and an overall negative impact on actor B (AIM of -22.5). Similar to Experiment 1 (see Table 7), the model accounts for 5 harmful dependencies for both actors. The dependency between *GA26* and *GB14* was found to be the most harmful dependency to both A (its removal improves the AIM by 5.5 points) and B (its removal improves the AIM by 4.5 points). The positive contribution between *GB11* and *GA26* was found to be beneficial for A (its removal deteriorates the AIM by 1.5 points), while the positive contribution between *GA5* and *SB1* was found to be ben-

eficial for B. The most beneficial dependency for A (i.e., between *GA1* and *SB2*) provides an improvement of 7.5 points (cell colored in dark green in Table 12). Furthermore, the most beneficial dependency for B (i.e., between *GB11* and *GA26*) provides an improvement of 3.5 points (cell colored in dark green in Table 12).

6.4 Results analysis

For both experiments, all computed intervals and metrics produced by the tool (for the acyclic GRL models, i.e., model #1 to #9 and model #13) and the resulting inter-actor dependency classifications were checked manually, revealing a 100% correctness rate. Furthermore, the tool was able to detect successfully the presence of circular dependencies in the cyclic GRL models, i.e., models #10, #11, and #12.

Table 15 summarizes the impact of inter-actor dependencies for both experiments. We notice that for models #1, #2, #7, #8, and #13 the number of dependencies in each category is the same. For models #3, #5, and #9, we see small improvements in experiment 2 compared with experiment 1, i.e., for model #3: 5 neutral and 1 beneficial vs. 6 neutral, for model #5: 12 neutral vs. 1 harmful and 11 neutral, and for model #9: 7 harmful, 1 beneficial, and 40 neutral vs. 5 harmful, 2 beneficial and 41 neutral. For models #4 and #6, we cannot conclude whether there is an improvement or a deterioration. Hence, based on these results, we cannot conclude that moving the initial intervals to the positive territory, i.e., from $[-100, 100]$ to $[0, 100]$ in our case, would benefit the interacting actors. Therefore, results depend on both initial intervals and the model structure. The analyst may experiment with different interval ranges to reach satisfactory inter-actor interactions. Otherwise, he may change the model structure by removing harmful dependencies and repeat the process again. Practical considerations have been discussed in Sect 7.3.

The propagation of the satisfaction intervals is based on breadth-first graph traversal algorithm and has $O(n + e)$ time complexity, where n is the number of intentional elements and e is the number of links. Table 15 shows the computation time of each model, measured on a PC running 64-bit windows 10 Enterprise, with a processor Intel(R) Core(TM) i7-7600U CPU@2.8 Ghz and 8GB of memory. Each recorded time measurement represents the average time of 5 runs. The computation times for all models vary from 16ms to 94ms. We notice that the larger models (i.e., #9 and #13) have the highest run times and they are comparable. These results confirm the linear time complexity of our algorithm.

Table 10 Experiment 2 results (models having 2 actors)—Part I

	Actors	
	Telecom Provider	Technician
Model 1 [39]		
All dependencies	[0, 67]	[0, 100]
Isolation	[0, 67]	[0, 100]
AIM	0 (neutral)	0 (neutral)
Remove <i>VoiceConn</i> → <i>LoggEquip</i>	[0, 67]	[0, 100]
AIM (<i>VoiceConn</i> → <i>LoggEquip</i>)	0 (neutral)	0 (neutral)
Model 2 [51]	Patient	Healthcare Provider
All dependencies	[0, 43]	[0, 75]
Isolation	[0, 75]	[0, 75]
AIM	− 16 (negative)	0 (neutral)
Remove <i>QualityOfCare</i> → <i>ViableHealthcare</i>	[0, 75]	[0, 75]
AIM (<i>QualityOfCare</i> → <i>ViableHealthcare</i>)	16	0 (neutral)
Model 3 [25]	City	Citizen
All dependencies	[0, 41]	[0, 53]
Isolation	[0, 41]	[0, 50]
AIM	0 (neutral)	1.5 (positive)
Remove <i>Process Green Waste</i> → <i>Quality of Waste</i>	[0, 41]	[0, 53]
AIM (<i>Process Green Waste</i> → <i>Quality of Waste</i>)	0 (neutral)	0 (neutral)
Remove <i>Willingness to Separate Waste</i> → <i>GW Education</i>	[0, 41]	[0, 53]
AIM (<i>Willingness to Separate Waste</i> → <i>GW Education</i>)	0 (neutral)	0 (neutral)
Remove <i>Positive City Image</i> → <i>Enjoy City</i>	[0, 41]	[0, 50]
AIM (<i>Positive City Image</i> → <i>Enjoy City</i>)	0 (neutral)	− 1.5

7 Discussion

In what follows, we discuss the benefits of our proposed approach, compare it with related work, provide some practical considerations on how to use the approach, and present the potential threats to validity.

7.1 General benefits of our approach

The presented approach has the following benefits:

- *Interval-based satisfaction analysis*: To the best of our knowledge, our technique is the first goal-based analysis approach that uses intervals to quantify the satisfaction of intentional elements and actors.
- *Support of various types of dependencies*: GRL is permissive (compared to other GORE languages like i^* [60]) when it comes to the types of inter-actor dependencies. Thus, our approach supports both implicit (i.e., contributions, correlations, and decompositions) and explicit inter-actor dependencies.

- *Completeness and scalability*: Our approach and prototype tool handles all GRL constructs and can be applied to any GRL model of any size and having any number of actors. Furthermore, our prototype tool handles both TGRL contribution types, i.e., quantitative (e.g., contributesTo G1 with 75) and qualitative (e.g., contributesTo G1 with somePositive).
- *Generality*: Although our tool is TGRL specific, it can be adapted to other goal-oriented languages with minimal modifications.
- *Fully automated*: Our prototype tool computes automatically the AIM for the entire TGRL model as well as the ones relative to each inter-actor dependency. In addition, it is capable of detecting cycles, i.e., circular dependencies.

7.2 Comparison with related work

In this section, we compare our approach with existing actor interaction analysis techniques, highlighted in Sect. 4, in the context of goal models.

Table 11 Experiment 2 results (models having 2 actors)—Part2

Model 6 [11]	Actors	
	User	System
All dependencies	[0, 25]	[0, 100]
Isolation	[0, 25]	[0, 100]
AIM	0 (neutral)	0 (neutral)
Remove <i>Electric Engine</i> → <i>Comfortable</i>	[0, 50]	[0, 100]
AIM (<i>Electric Engine</i> → <i>Comfortable</i>)	12.5	0 (neutral)
Remove dependency <i>Electric Engine</i> → <i>Reduce</i>	[0, 0]	[0, 100]
AIM (<i>Electric Engine</i> → <i>Reduce</i>)	-12.5	0 (neutral)
Remove dependency <i>Fuel Engine</i> → <i>Comfortable</i>	[0, 0]	[0, 100]
AIM (<i>Fuel Engine</i> → <i>Comfortable</i>)	-12.5	0 (neutral)
Remove dependency <i>Fuel Engine</i> → <i>Reduce</i>	[0, 50]	[0, 100]
AIM (<i>Fuel Engine</i> → <i>Reduce</i>)	12.5	0 (neutral)
Model 7 [35]	Technology Provider	Technology User
All dependencies	[0, 25]	[0, 100]
Isolation	[0, 25]	[0, 100]
AIM	0 (neutral)	0 (neutral)
Remove dependency <i>Sell tech for profit</i> → <i>Purchase Technology</i>	[0, 25]	[0, 100]
AIM (<i>Sell tech for profit</i> → <i>Purchase Technology</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>Produce tech</i> → <i>Purchase Technology</i>	[0, 25]	[0, 100]
AIM (<i>Produce tech</i> → <i>Purchase Technology</i>)	0 (neutral)	0 (neutral)
Remove dependency <i>tech users abide</i> → <i>Abide by Licensing</i>	[0, 25]	[0, 100]
AIM (<i>tech users abide</i> → <i>Abide by Licensing</i>)	0 (neutral)	0 (neutral)

Table 16 summarizes the comparison based on the following criteria:

- *Goal model Notation*: denotes the model type, e.g., GRL, KAOS, Tropos, i^* SR, etc.
- *Technique*: specifies the type of approach, e.g., rule-based, heuristics, satisfaction analysis, metric, game theory, empirical, etc.
- *Model aspect*: denotes which goal model aspect is used in the approach, e.g., quantitative/qualitative/hybrid, structural, etc.
- *Source of interactions*: denotes how the conflict is defined, e.g., conflicting view points, inter-actor dependencies, intra-actor dependencies, etc.
- *Automation*: denotes whether the approach is manual, or automated.

As shown in Table 16, five types of interactions/conflicts were addressed in the reviewed literature: *Inter-actor dependencies* [13,14,19,20,23,24,26,50–52], divergences (weak forms of conflict) [57], conflicting view points [28], feature interactions (FI) [18], and intra-actor dependencies [6].

It is worth noting that goal-oriented quantitative approaches based on Game Theory (GT) [47] suffer from scalability issues. Indeed, all GT-based surveyed approaches [13,14,30,51] model interactions in goal models as a two-person game, i.e., game with only two players (e.g., in Sumesh et al. [51] the players represent two top softgoals of each actor, in Hassine et al. [30] the players are two GRL actors), in order to come up with reconciliation strategies, representing Nash Equilibria. This scalability issue is not limited to the number of supported actors but also to the number of supported elements within an actor. For example, in [51] only two alternative options, representing the only two available strategies for each actor, are considered. The approach presented in [30] relaxes this constraint by supporting more strategies, but the proposed solution remains not scalable. Indeed, given two GRL actors with m and q leaf elements, the produced payoff bimatrix (composed of actors' satisfaction values) will be of size $5^m \times 5^q$. Therefore, if the numbers of leaves, i.e., m and q , become large, the size of the bimatrix will grow exponentially. Our proposed approach supports any number of actors and any type of dependencies (explicit/implicit) between actors.

Table 12 Experiment 2 results (models having 2 actors)—Part3

Model 13	Actors	
	A	B
All dependencies	[0, 64]	[0, 36]
Isolation	[0, 62]	[0, 81]
AIM	1.0 (positive)	-22.5 (negative)
Remove <i>GA5</i> → <i>GB6</i>	[0, 68]	[0, 40]
AIM (<i>GA5</i> → <i>GB6</i>)	2	2
Remove <i>GA11</i> → <i>GB6</i>	[0, 66]	[0, 39]
AIM (<i>GA11</i> → <i>GB6</i>)	1.0	1.5
Remove <i>GA21</i> → <i>GB14</i>	[0, 64]	[0, 36]
AIM (<i>GA21</i> → <i>GB14</i>)	0 (neutral)	0 (neutral)
Remove <i>GA26</i> → <i>GB14</i>	[0, 75]	[0, 45]
AIM (<i>GA26</i> → <i>GB14</i>)	5.5	4.5
Remove <i>SB2</i> → <i>GA9</i>	[0, 66]	[0, 39]
AIM (<i>SB2</i> → <i>GA9</i>)	1.0	1.5
Remove <i>GB6</i> → <i>GA16</i>	[0, 64]	[0, 36]
AIM (<i>GB6</i> → <i>GA16</i>)	0 (neutral)	0 (neutral)
Remove <i>GB10</i> → <i>GA16</i>	[0, 64]	[0, 36]
AIM (<i>GB10</i> → <i>GA16</i>)	0 (neutral)	0 (neutral)
Remove <i>GB10</i> → <i>GA21</i>	[0, 64]	[0, 36]
AIM (<i>GB10</i> → <i>GA21</i>)	0 (neutral)	0 (neutral)
Remove <i>GB14</i> → <i>TA6</i>	[0, 75]	[0, 45]
AIM (<i>GB14</i> → <i>TA6</i>)	5.5	4.5
Remove <i>TB1</i> → <i>GA26</i>	[0, 64]	[0, 36]
AIM (<i>TB1</i> → <i>GA26</i>)	0 (neutral)	0 (neutral)
Remove <i>GA1</i> → <i>SB2</i>	[0, 49]	[0, 36]
AIM (<i>GA1</i> → <i>SB2</i>)	-7.5	0 (neutral)
Remove <i>GA5</i> → <i>SB2</i>	[0, 64]	[0, 36]
AIM (<i>GA5</i> → <i>SB2</i>)	0 (neutral)	0 (neutral)
Remove <i>SB1</i> → <i>GA5</i>	[0, 64]	[0, 34]
AIM (<i>SB1</i> → <i>GA5</i>)	0 (neutral)	-1
Remove <i>GB11</i> → <i>GA26</i>	[0, 61]	[0, 29]
AIM (<i>GB11</i> → <i>GA26</i>)	-1.5	-3.5

Five metric-based approaches have been proposed [19,20,23,24,26,52] to measure inter-actor dependencies. They rely only on the configuration of the goal model (i.e., structural metrics) and some of them require human judgment [20,52]. It is worth noting that our proposed approach is the unique goal-oriented analysis approach that uses interval-based satisfactions.

7.3 Practical considerations

In this section, we describe how the proposed approach and metric can be used by requirements engineers/analysts to analyze GRL models from two perspectives: (1) evaluation of potential alternative solutions (through interval-

based satisfaction analysis), and (2) assessing the impact of dependencies between the intervening actors. However, assessing the usefulness of the approach for requirements engineers/analysts in a real-world context, requires a separate empirical study that is outside the scope of the paper.

Figure 17 illustrates the workflow of one possible deployment of the approach, as a UML activity diagram model. In the first step, the analyst selects one potential solution by defining the satisfaction intervals of the model leaf elements (that will be refined later into requirements). The choice of interval ranges is based on the analyst domain/organizational knowledge and the GRL model syntactic constraints (e.g., satisfy only one element in presence of a XOR decomposition). Interval ranges are specified based on the presence

Table 13 Experiment 2 results (models having 3 actors)

Model 4 [10]	Service Provider	System	Vendor
All dependencies	[0, 14]	[0, 100]	[0, 100]
Isolation	[0, 2]	[0, 100]	[0, 100]
AIM	6 (positive)	0 (neutral)	0 (neutral)
Remove <i>Data in New Service Node</i> → <i>ServiceNode</i>	[0, 14]	[0, 100]	[0, 100]
AIM (<i>Data in New Service Node</i> → <i>ServiceNode</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove dependency <i>Data in SN</i> → <i>MinChanges</i>	[0, 16]	[0, 100]	[0, 100]
AIM (<i>Data in SN</i> → <i>MinChanges</i>)	1	0 (neutral)	0 (neutral)
Remove dependency <i>Data in SCP</i> → <i>MinChanges</i>	[0, 12]	[0, 100]	[0, 100]
AIM (<i>Data in SCP</i> → <i>MinChanges</i>)	-1	0 (neutral)	0 (neutral)
Remove <i>Service in ControlSwitch</i> → <i>MinMessageExchange</i>	[0, -35]	[0, 100]	[0, 100]
AIM (<i>Service in ControlSwitch</i> → <i>MinMessageExchange</i>)	-24.5	0 (neutral)	0 (neutral)
Remove dependency <i>Service in SCP</i> → <i>MinMessageExchange</i>	[0, 52]	[0, 100]	[0, 100]
AIM (<i>Service in SCP</i> → <i>MinMessageExchange</i>)	19	0 (neutral)	0 (neutral)
Remove dependency <i>Service in SCP</i> → <i>MinSwitchLoad</i>	[0, 30]	[0, 100]	[0, 100]
AIM (<i>Service in SCP</i> → <i>MinSwitchLoad</i>)	8	0 (neutral)	0 (neutral)
Remove <i>Service in ControlSwitch</i> → <i>MinSwitchLoad</i>	[-2, 0]	[0, 100]	[0, 100]
AIM (<i>Service in ControlSwitch</i> → <i>MinSwitchLoad</i>)	-8	0 (neutral)	0 (neutral)
Model 5 [32]	Organization	Kids & Youth	Counselors
All dependencies	[0, 16]	[0, 65]	[0, 47]
Isolation	[0, 16]	[0, 65]	[0, 47]
AIM	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>highQualityCounseling(Org)</i> → <i>highQualityCounseling</i>	[0, 16]	[0, 65]	[0, 47]
AIM (<i>highQualityCounseling(Org)</i> → <i>highQualityCounseling</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove dependency <i>textMessage(Org)</i> → <i>textMessage(C)</i>	[0, 16]	[0, 65]	[0, 47]
AIM (<i>textMessage(Org)</i> → <i>textMessage(C)</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>kidsUseCyberCafe(Org)</i> → <i>kidsUseCyberCafe(C)</i>	[0, 16]	[0, 65]	[0, 47]
AIM (<i>kidsUseCyberCafe(Org)</i> → <i>kidsUseCyberCafe(C)</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove dependency <i>kidsUseCyberCafe3</i> → <i>kidsUseCyberCafe</i>	[0, 16]	[0, 65]	[0, 47]
AIM (<i>kidsUseCyberCafe3</i> → <i>kidsUseCyberCafe</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Model 8 [33]	PC User	Data Pirate	PC Provider
All dependencies	[0, 100]	[0, 100]	[0, 31]
Isolation	[0, 100]	[0, 100]	[0, 31]
AIM	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>purchase PC product</i> → <i>produce PC products</i>	[0, 100]	[0, 100]	[0, 31]
AIM (<i>purchase PC product</i> → <i>producepcproducts</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>obtain PC from data pirate</i> → <i>make content available</i>	[0, 100]	[0, 100]	[0, 31]
AIM (<i>obtain PC from data pirate</i> → <i>make content available</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>make content available</i> → <i>allow p2p</i>	[0, 100]	[0, 100]	[0, 31]
AIM (<i>make content available p2p</i> → <i>allow p2p</i>)	0 (neutral)	0 (neutral)	0 (neutral)
Remove <i>pc users abide by licensing</i> → <i>Abide by licensing</i>	[0, 100]	[0, 100]	[0, 31]
AIM (<i>cuser abide by rules</i> → <i>abide by rules</i>)	0 (neutral)	0 (neutral)	0 (neutral)

Table 14 Experiment 2 results (Model having 6 actors)

Model 9	Actors					
	A1	A2	A3	A4	A5	A6
All dependencies	[0, 39]	[0, 39]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
Isolation	[0, 100]	[0, 25]	[0, 75]	[0, 87]	[0, 75]	[0, 25]
AIM	-30.5 (negative)	7 (positive)	-28.5 (negative)	-11 (negative)	0 (neutral)	0 (neutral)
Remove G2—G4	[0, 100]	[0, 39]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
AIM (G2—G4)	30.50	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove T3—T7	[0, 39]	[0, 39]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
AIM (T3—T7)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G7—G9	[0, 43]	[0, 43]	[0, 18]	[0, 87]	[0, 75]	[0, 25]
AIM (G7—G9)	2	2	0 (neutral)	11	0 (neutral)	0 (neutral)
Remove G9—T8	[0, 39]	[0, 39]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
AIM (G9—T8)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G4—G6	[0, 25]	[0, 25]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
AIM (G4—G6)	-7	-7	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G2—G5	[0, 39]	[0, 39]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
AIM (G2—G5)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G11—G8	[0, 39]	[0, 39]	[0, 18]	[0, 65]	[0, 75]	[0, 25]
AIM (G11—G8)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)	0 (neutral)
Remove G11—G12	[0, 39]	[0, 39]	[0, 56]	[0, 65]	[0, 75]	[0, 25]
AIM (G11—G12)	0 (neutral)	0 (neutral)	19	0 (neutral)	0 (neutral)	0 (neutral)

Table 15 Summary of inter-actor interactions

	Experiment 1			Experiment 2			Average Execution Time (ms)
	Harmful	Beneficial	Neutral	Harmful	Beneficial	Neutral	
Model1	0	0	2	0	0	2	16
Model2	1	0	1	1	0	1	20
Model3	0	0	6	0	1	5	32
Model4	2	2	17	3	3	15	56
Model5	1	0	11	0	0	12	40
Model6	0	0	8	2	2	4	32
Model7	0	0	6	0	0	6	19
Model8	0	0	12	0	0	12	33
Model9	7	1	40	5	2	41	94
Model13	11	4	13	10	4	14	93

of evidence (positive or negative) to satisfy an intentional element and its level (fully or partially) [34]. For instance, the analyst may choose interval ranges near the higher end (e.g., [80, 95]) for leaf elements that will probably be satisfied, interval ranges near the lower end (e.g., [-95, -80]) for leaf elements that will probably be denied.

Next, the analyst computes and analyzes the produced AIM metric (step 2). In case the type and magnitude of all inter-actor dependencies (as a whole) are acceptable (e.g., beneficial, neutral, or may be slightly harmful, i.e., tolerable low AIM value), then the process ends, and the model

remains unchanged. Otherwise (i.e., guard condition [not satisfied] in Fig. 17), the analyst would analyze the computed AIM value for each inter-actor dependency (step3). The analyst may then mitigate the detected vulnerabilities by:

1. *Choose different satisfaction intervals (step 4):* The analyst may decide to try other alternative solutions using the initial GRL model. Changing the input intervals produces different AIM values (step 2).
2. *Remove one or multiple inter-actor dependency (step 5):* Given all AIM values for all inter-actor dependencies

Table 16 Comparison with related work

Approach	Goal Notation	Technique	Model aspect	Interactions	Automation
van Lamsweerde et al. [57]	KAOS	Formal heuristics	Structural	Divergence	No
Hassine and Amyot [28]	GRL	Empirical	Structural	Conflicting view points	No
Ali et al. [6]	Tropos	Goal-Context relations	Structural	Intra-actor dependencies	Yes
De Vries and Cheng [18]	Generic AND/OR	Constraint solver	Structural	Feature Interaction (FI)	Yes
Sutcliffe and Minocha [52]	i^*	Metric-based	Structural	Inter-actor dependencies	No
Bryl et al. [13,14]	i^*	Game theory	Quantitative	Inter-actor dependencies	Yes
Sumesh et al. [51]	i^* SR	Game theory	Quantitative	Inter-actor dependencies	Yes
Hassine et al. [30]	GRL	Game theory	Quantitative	Inter-actor dependencies	Yes
Subramanian et al. [50]	i^*	Satisfaction analysis (Fuzzy numbers)	Quantitative	Inter-actor dependencies	No
Franch et al. [20]	i^* SD	Metric-based	Structural	Inter-actor dependencies	No
Xavier Franch [19]	i^*	Metric-based	Structural	Inter-actor dependencies	No
Grau et al. [23,24]	i^* SD	Metric-based	Structural	Inter-actor dependencies	No
Hassine and Alshayeb [26]	GRL	Metric-based	Structural	Inter-actor dependencies	No
This paper	GRL	Metric-based & Satisfaction Analysis (interval-based)	Quantitative/ structural	Inter-actor dependencies	Yes

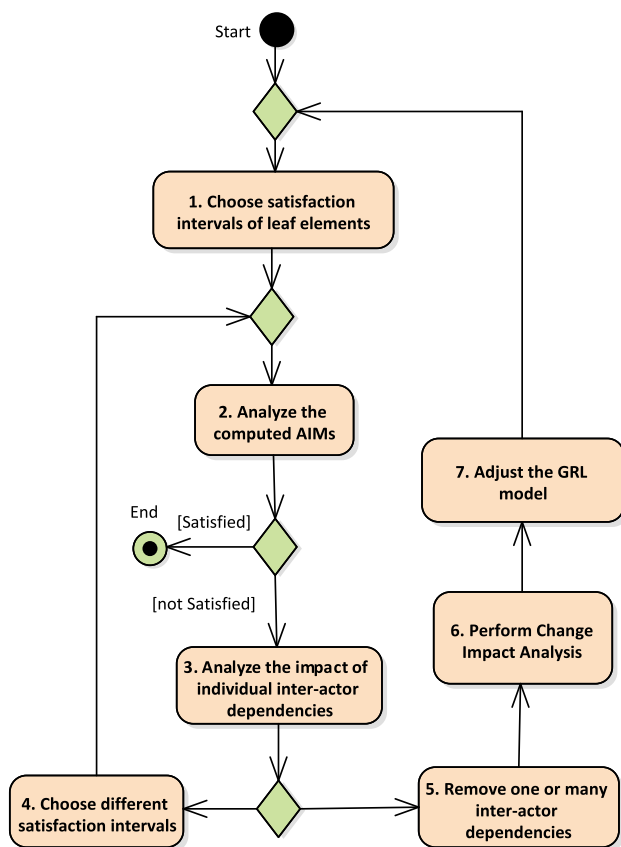


Fig. 17 Possible workflow describing a potential deployment of the approach

and all satisfaction intervals, the analyst may then select a candidate dependency to be removed, for instance the dependency that leads to the best improvement of the AIM value (i.e., the most harmful dependency). However, the analyst should perform a change impact analysis (CIA) [7,8] in order to assess the syntactic/semantic impact of removing the dependency (step 6). The evaluation of the syntactic impact involves studying how the removal of an inter-actor dependency would propagate to the rest of the GRL model (following the network of links) and determine the potentially impacted GRL elements. Alkaf et al. [7] proposed an automated approach to determine the impact of changes in GRL models by marking all impacted elements. The evaluation of the semantic impact aims to evaluate how the depender (of the deleted dependency) can mitigate the dependency deletion, which may result in some adjustments (step 7) to the GRL model (e.g., rephrase goals, introduction of a new task, etc.). The modification of the GRL model requires repeating the process starting from the first step.

It is worth noting that the acceptance of the GRL model may depend on the chosen AIM threshold (e.g., in case of slightly harmful dependencies, see Step 2). An AIM thresh-

old can be set by the analyst/organization based on the domain knowledge and the organizational tolerance level to inter-actor dependencies. However, the characterization of AIM thresholds requires the conduct of an empirical study in various real-world contexts, which is outside the scope of this paper.

Furthermore, the proposed approach should be mainly used during the early stages of the requirements engineering phase. At this stage, the modeler is not yet interested in the operational details of processes or system requirements, or component interactions [36]. However, changes to the goal model after the development of the solution (which are less likely to happen since goals are known to be stable compared to requirements [12,54]) may significantly impact system requirements/design and may be very costly.

7.4 Threats to validity

The proposed approach, and the experimental evaluation (Sect. 6) are subject to several limitations and threats to validity, categorized here according to three important types identified by Wright et al. [59].

- *Construct validity*: The first threat is that we have used quantitative interval values to model satisfaction levels of leaf elements. We could have used qualitative intervals such as [Denied..Satisfied] (instead of $[-100, 100]$), or [Neutral..Satisfied] (instead of $[0, 100]$), limiting the number of options for each leaf element. We believe that our metric is more precise in measuring inter-actor interactions and that the use of a qualitative metric would not be as sensitive as our quantitative metric. However, an experimental investigation is required to verify this hypothesis. This is out of the scope of this paper and will be addressed in future work.

A second threat is related to the validity of our interval-based propagation algorithm. It is worth noting that our interval-based forward propagation algorithm is inspired from the ITU-T quantitative forward propagation algorithm, described in [36]. Furthermore, although this algorithm would promote the adoption of our approach by analysts and researchers, who are familiar with the standard GRL and the jUCMNav [37] tool, it does inherit the same limitations. Indeed, in presence of many contributions links and with the use of wide initial interval ranges (see Fig. 13), the computed satisfaction interval may exceed the accepted range (e.g., $[-107, 105]$ in Fig. 13) resulting in the interval being capped to $[-100, 100]$. Such limitation may result in similar intervals (i.e., $[-100, 100]$) with/ without dependencies leading to more neutral cases, which may hamper the practicality of the approach. This limitation is inherited from the

GRL propagation algorithm, which limits the range of values to $[-100, 100]$. To mitigate this threat, the analyst may experiment with different initial interval ranges with preference for smaller ranges. Furthermore, like the quantitative forward propagation algorithm, our interval-based propagation algorithm does not support circular dependencies.

A third threat concerns the use of interval midpoint, as it abstracts the interval information and does not preserve the original range. For example, intervals $[-10, 10]$ and $[-50, 50]$ have the same midpoint, i.e., zero. To mitigate this threat, our prototype tool computes and displays all satisfaction intervals of all model elements.

- *Internal validity*: There is a potential criticism with respect to having a large number of neutral impact (i.e., $AIM = 0$), even in the presence of many inter-actor dependencies. Indeed, the interval computation does not depend only on the number of dependencies between the interacting actors, but also on the GRL structure and the selected strategies. Hence, having neutral impacts does not impact the validity of the computed metrics.

Another possible threat is related to the choice of two intervals, i.e., $[-100, 100]$ and $[0, 100]$, in our experimental evaluation. The first choice (i.e., $[-100, 100]$), presumes full uncertainty with respect to the satisfiability of leaf elements of the GRL model. The second choice (i.e., $[0, 100]$) presumes that each actor would try his best not to deny his leaf elements. However, our approach is flexible, allowing the analyst to use different initial satisfaction intervals for any intentional element. For instance, an analyst may not choose semantically invalid strategies (e.g., satisfying mutually exclusive alternatives).

- *External validity*: As for external validity, a possible threat is that our proposed approach is tailored to the textual GRL language [36] and to the GRL forward propagation algorithm [36]. Although the steps and the running example are illustrated using GRL, our approach can likely be adjusted and applied to other goal-oriented languages that support actors, intentional elements, and their relationships (including i^* [60]). Another possible external threat is related to the lack of evidence of the usefulness of the approach for requirements engineers/analysts in a real-world context. Providing such evidence requires the conduct of an empirical study that is outside the scope of this paper.

8 Conclusions and future work

We have proposed a novel quantitative metric, called *Actor Interaction Metric (AIM)*, to measure inter-actor dependen-

cies in GRL models. The AIM metric is based on the computation of satisfaction intervals and takes into account both the structural configurations of GRL actors and their quantitative satisfaction levels. The metric is used to categorize inter-actor dependencies into positive, negative, and neutral. In addition, the metric helps identify the most harmful/beneficial dependency for each actor, given any user-defined strategy. The approach is implemented in a prototype tool, that targets the inter-actor dependencies in textual GRL. We have evaluated experimentally our approach using 13 TGRL models, achieving 100% accuracy in terms of correctness, as well as detection and classification rate.

As future work, we plan to integrate our approach within jUCMNav, the most comprehensive tool available to date that supports the definition and analysis of GRL models. In addition, we will investigate the design of a qualitative inter-actor dependency metric by reusing the ITU-T standard GRL qualitative propagation algorithm.

Acknowledgements The authors would like to acknowledge the support provided by the Deanship of Scientific Research at King Fahd University of Petroleum & Minerals for funding this work through Project No. IN171027.

Appendix A: GRL models

See Figs. 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, and 30.

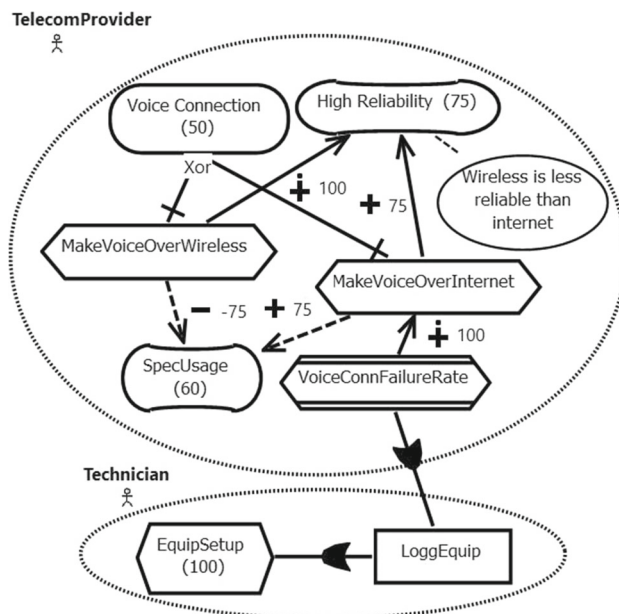


Fig. 18 Model 1: Telecommunication system (adapted from [39])

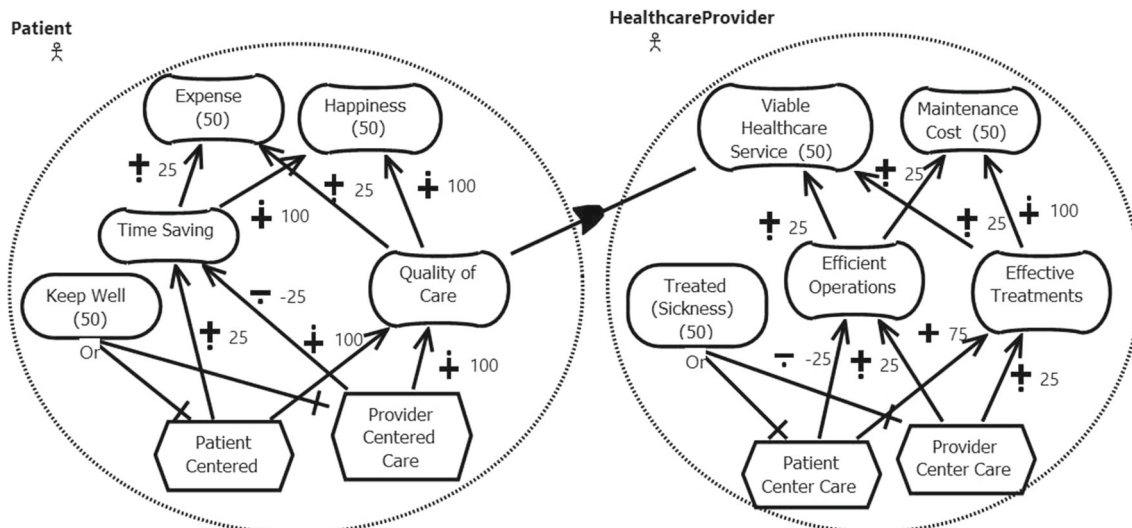


Fig. 19 Model 2: Tele-medicine system (adapted from [51])

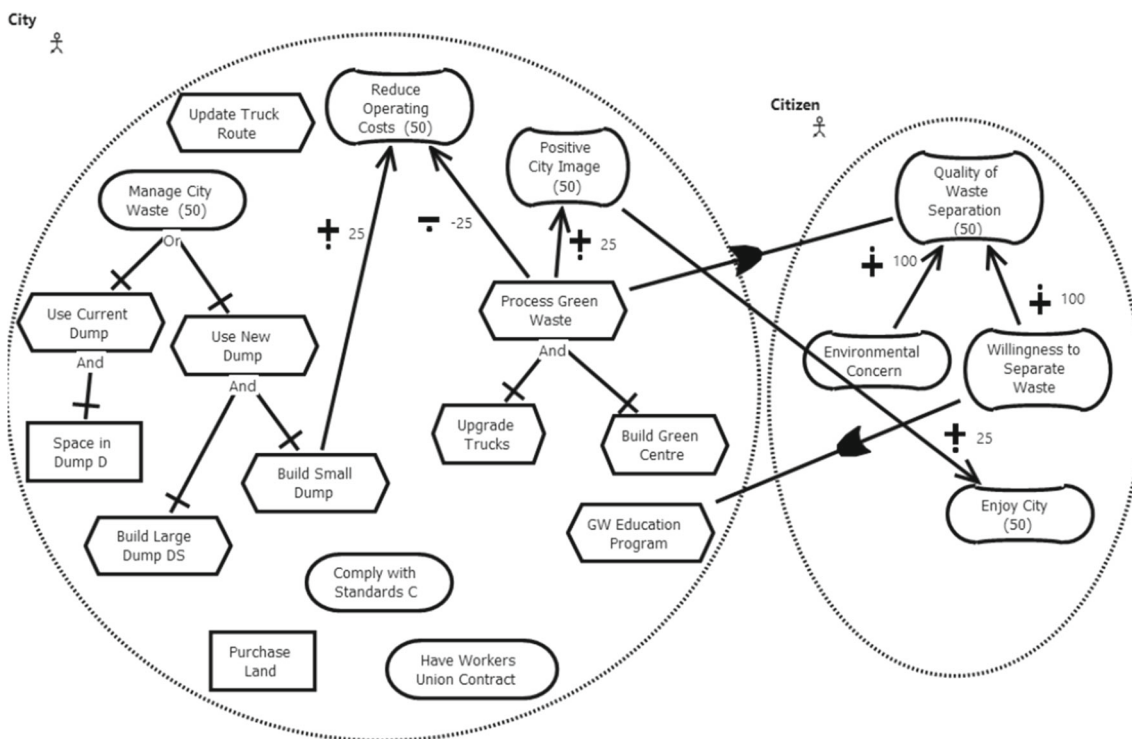


Fig. 20 Model 3: Waste management system (adapted from [25])

Fig. 21 Model 4: Wireless system (adapted from [10])

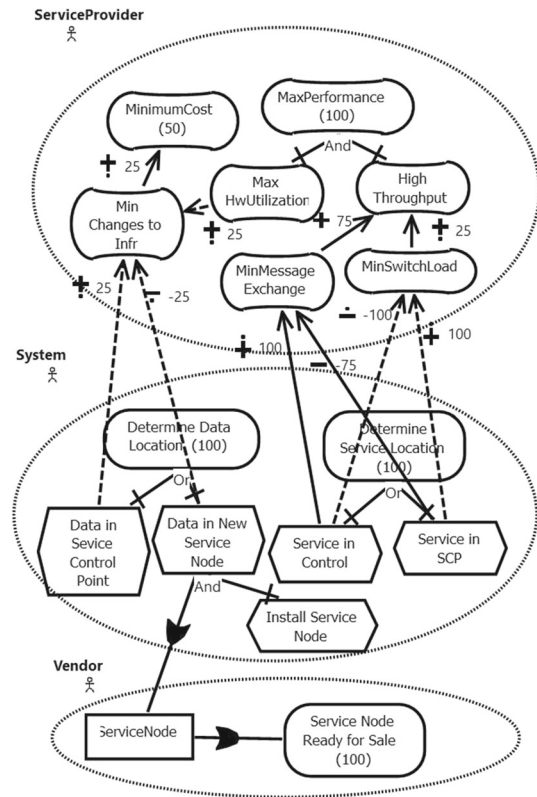
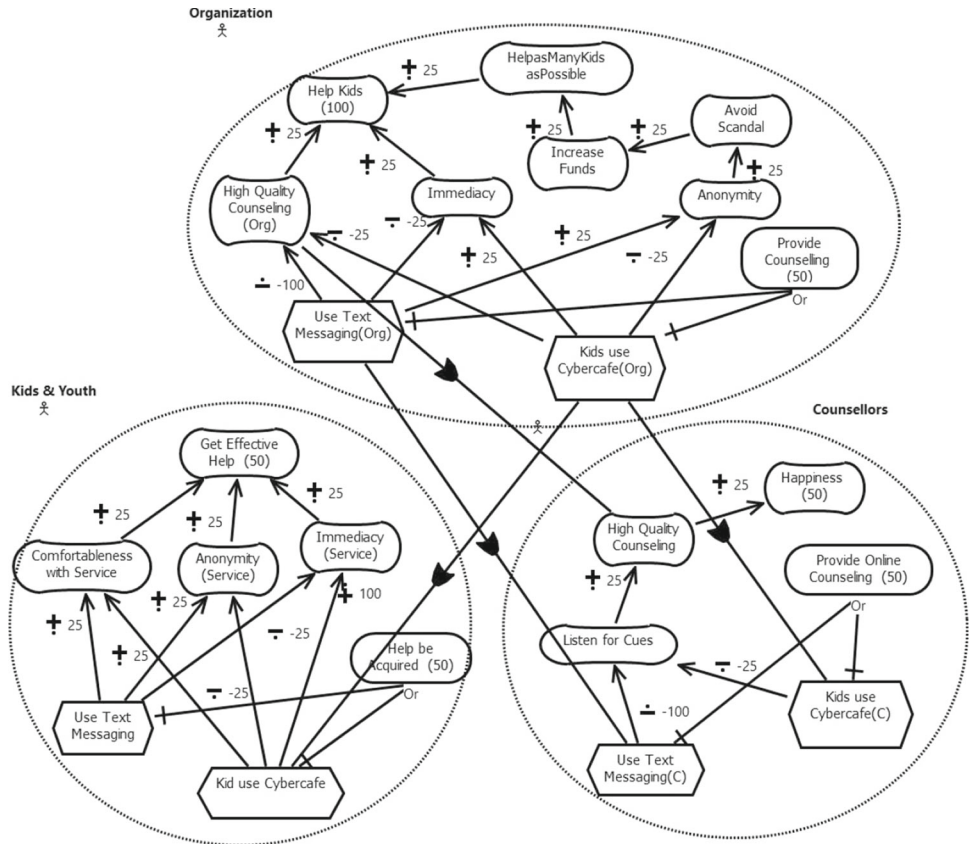


Fig. 22 Model 5: Youth counseling (adapted from [32])



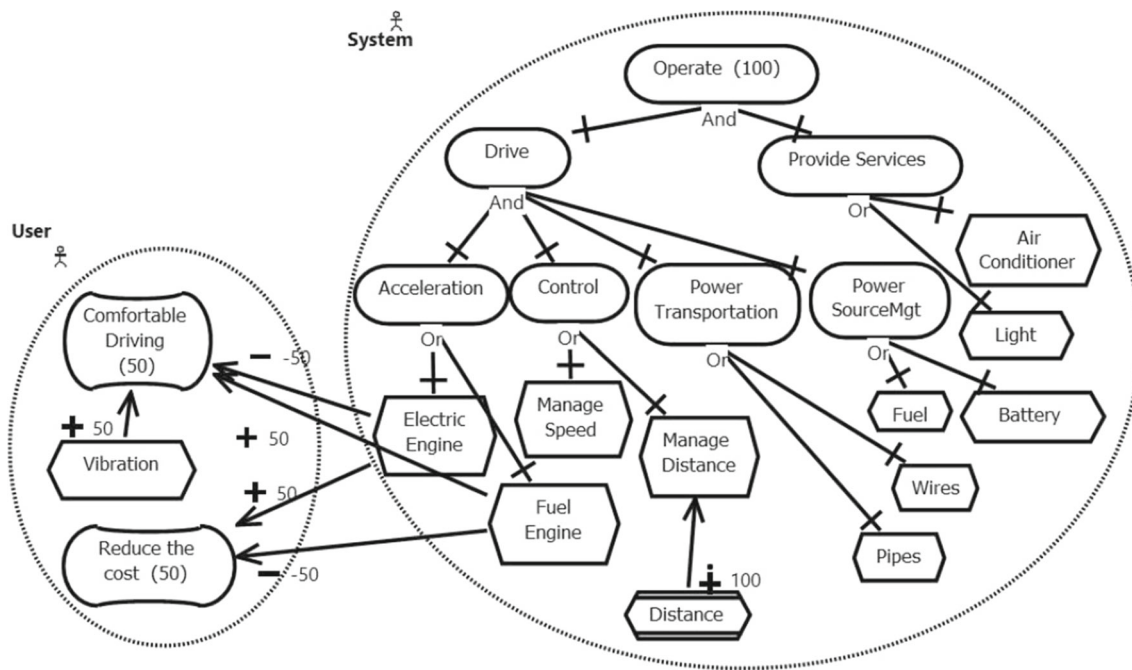


Fig. 23 Model 6: Hybrid car system (adapted from [11])

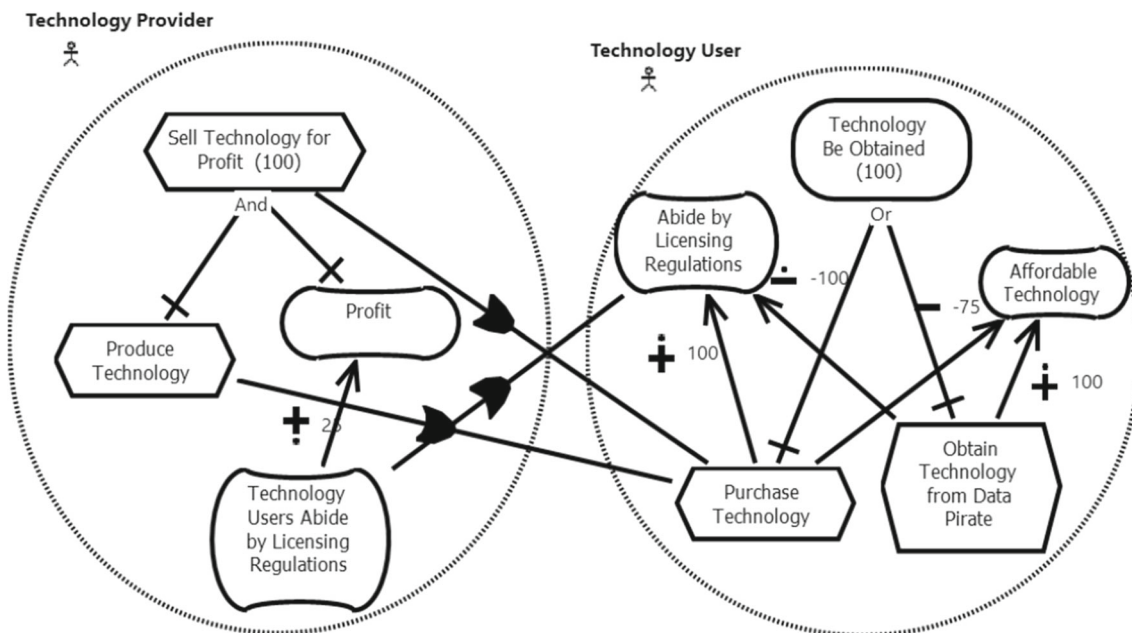


Fig. 24 Model 7: Technology system (adapted from [35])

Fig. 25 Model 8: PC system (adapted from [33])

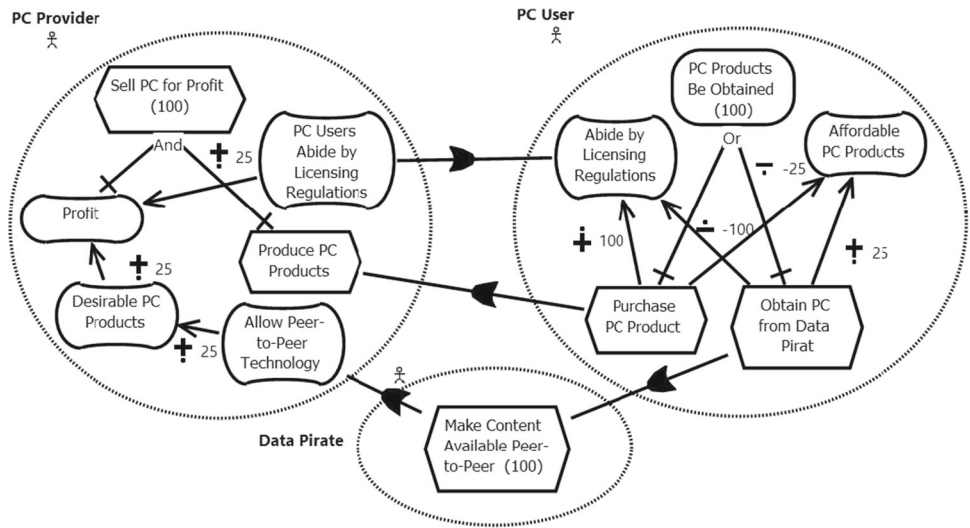


Fig. 26 Model 9: Generic example 1 (large model having a large number of actors)

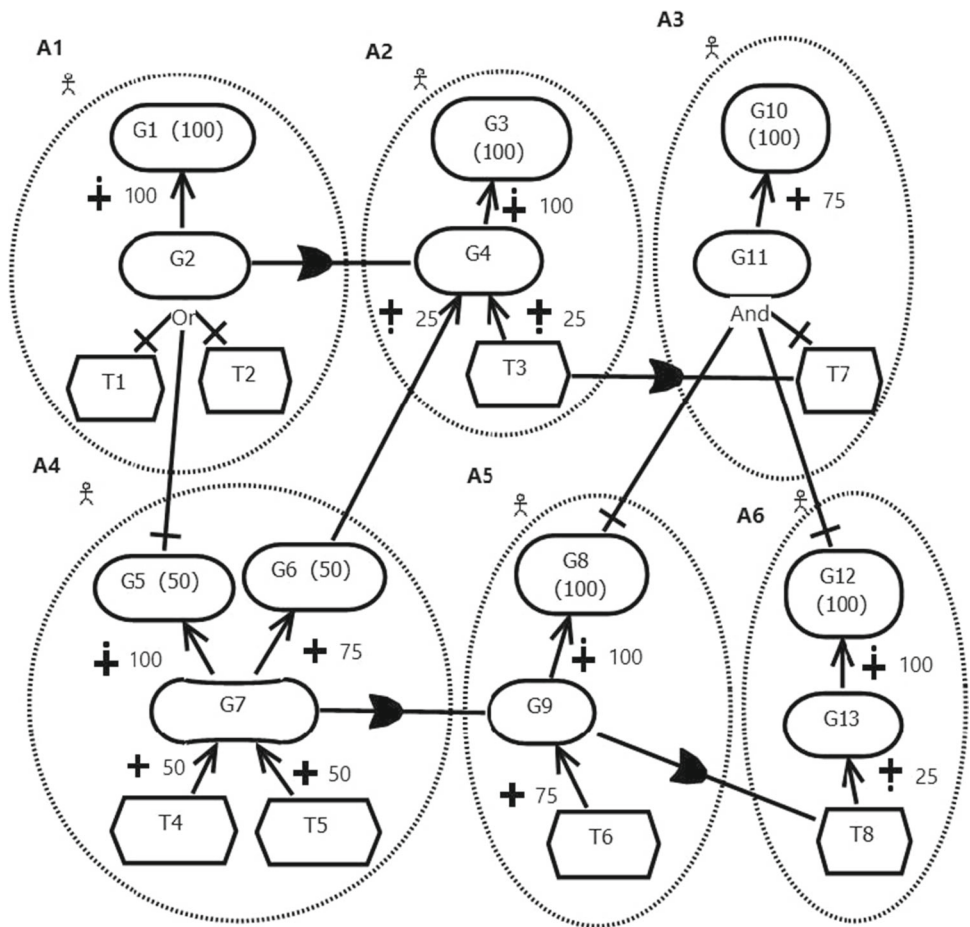


Fig. 27 Model 10: Generic cyclic model (cycle composed of explicit dependencies)

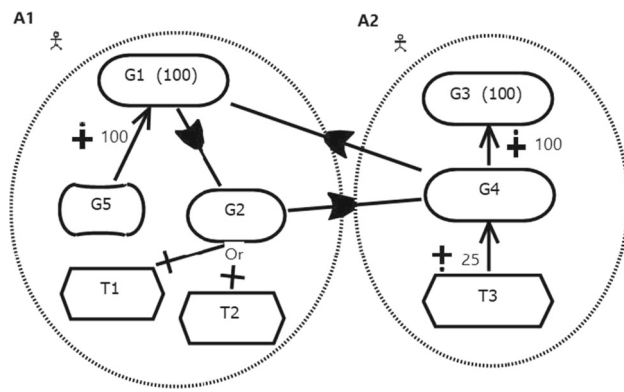


Fig. 28 Model 11: Generic cyclic model (cycle composed of implicit dependencies)

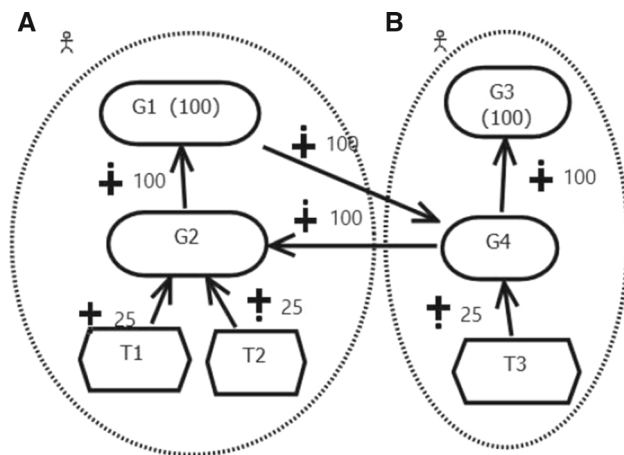


Fig. 29 Model 12: Generic cyclic model (cycle composed of a mixture of explicit and implicit dependencies)

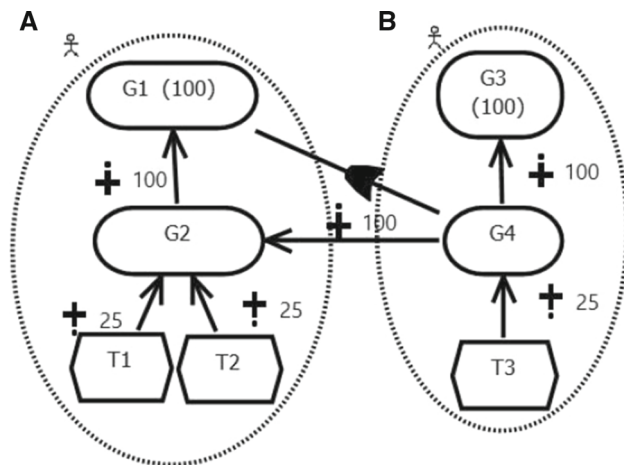
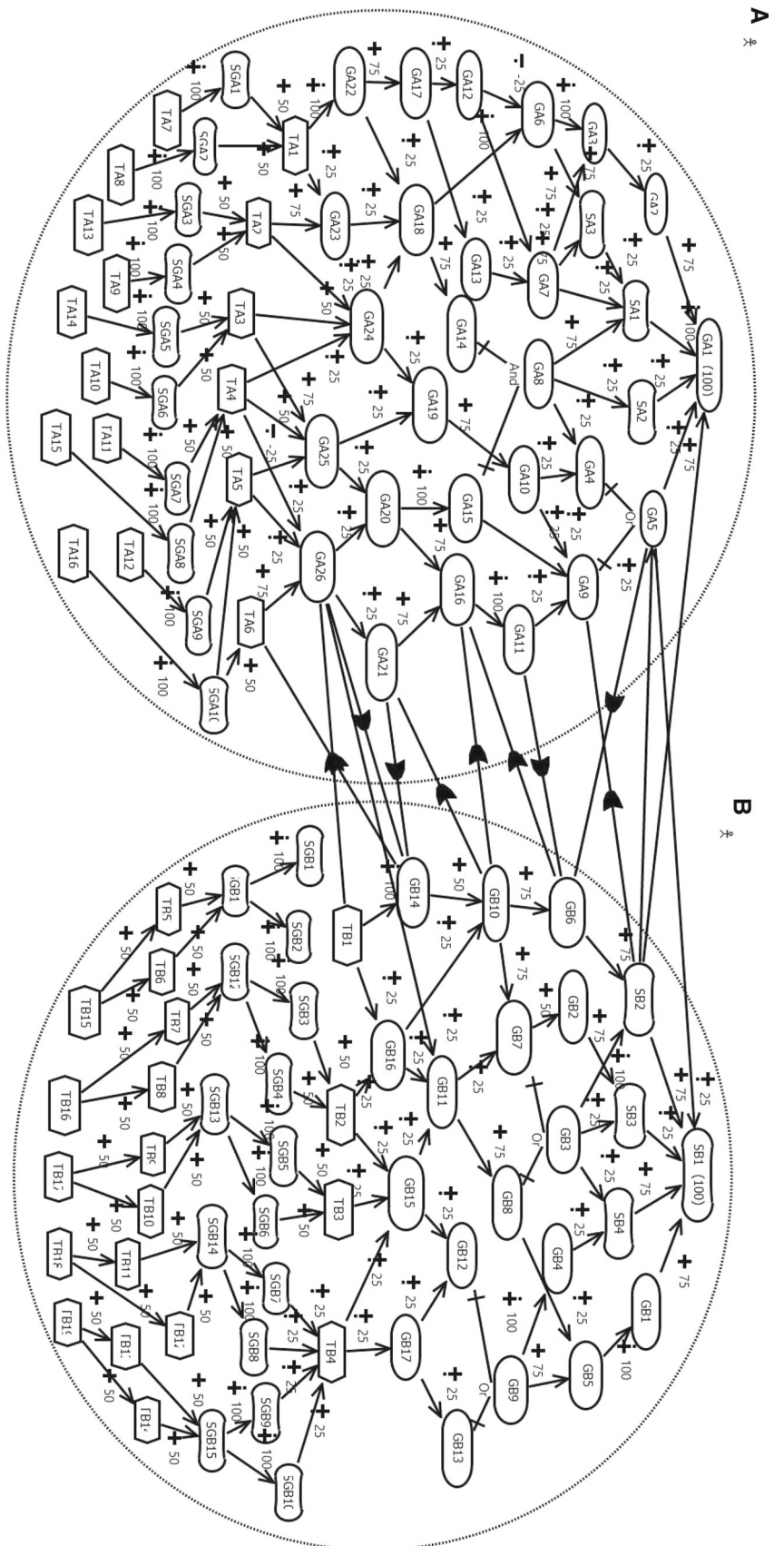


Fig. 30 Model 13: Generic model having a large number of dependencies



References

1. Abdelzad, V., Amyot, D., Alwidian, S.A., Lethbridge, T.: A textual syntax with tool support for the goal-oriented requirement language. In: Castro, J., Filho, G.A.C., Liaskos, S. (eds.) Proceedings of the Eighth International i*Workshop, iStar 2015, in Conjunction with the 23rd International Requirements Engineering Conference (RE 2015), Ottawa, Canada, August 24–25, 2015. CEUR Workshop Proceedings, vol. 1402, pp. 61–66. CEUR-WS.org (2015). <http://ceur-ws.org/Vol-1402/paper6.pdf>
2. Abdelzad, V., Amyot, D., Lethbridge, T.C.: Adding a textual syntax to an existing graphical modeling language: Experience report with GRL. In: Fischer, J., Scheidgen, M., Schieferdecker, I., Reed, R. (eds.) SDL 2015: Model-Driven Engineering for Smart Cities—17th International SDL Forum, Berlin, Germany, October 12–14, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9369, pp. 159–174. Springer (2015). https://doi.org/10.1007/978-3-319-24912-4_12
3. Affleck, A., Krishna, A.: Supporting quantitative reasoning of non-functional requirements: a process-oriented approach. In: Proceedings of the International Conference on Software and System Process, ICSSP '12, pp. 88–92. IEEE Press, Piscataway (2012). <https://doi.org/10.5555/2664360.2664375>
4. Akhigbe, O., Alhaj, M., Amyot, D., Badreddin, O., Braun, E., Cartwright, N., Richards, G., Mussbacher, G.: Creating quantitative goal models: Governmental experience. In: Yu, E., Dobbie, G., Jarke, M., Purao, S. (eds.) Conceptual Modeling. Lecture Notes in Computer Science, vol. 8824, pp. 466–473. Springer (2014). https://doi.org/10.1007/978-3-319-12206-9_40
5. Ali, R., Dalpiaz, F., Giorgini, P.: A goal-based framework for contextual requirements modeling and analysis. *Requir. Eng.* **15**(4), 439–458 (2010). <https://doi.org/10.1007/s00766-010-0110-z>
6. Ali, R., Dalpiaz, F., Giorgini, P.: Reasoning with contextual requirements: detecting inconsistency and conflicts. *Inf. Softw. Technol.* **55**(1), 35–57 (2013). <https://doi.org/10.1016/j.infsof.2012.06.013>
7. Alkaf, H.S., Hassine, J., Binalialhag, T., Amyot, D.: An automated change impact analysis approach for user requirements notation models. *J. Syst. Softw.* **157**, 110–397 (2019). <https://doi.org/10.1016/j.jss.2019.110397>
8. Alkaf, H.S., Hassine, J., Hamou-Lhadj, A., Alawneh, L.: An automated change impact analysis approach to GRL models. In: SDL 2017: Model-Driven Engineering for Future Internet—18th International SDL Forum, Budapest, Hungary, October 9–11, 2017, Proceedings, pp. 157–172 (2017). https://doi.org/10.1007/978-3-319-68015-6_10
9. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., Yu, E.: Evaluating goal models within the goal-oriented requirement language. *Int. J. Intell. Syst.* **25**, 841–877 (2010). <https://doi.org/10.1002/int.v25:8>
10. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., Yu, E.S.K.: Evaluating goal models within the goal-oriented requirement language. *Int. J. Intell. Syst.* **25**(8), 841–877 (2010). <https://doi.org/10.1002/int.20433>
11. Anda, A.A., Amyot, D.: Arithmetic semantics of feature and goal models for adaptive cyber-physical systems. In: Damian, D.E., Perini, A., Lee, S. (eds.) 27th IEEE International Requirements Engineering Conference, RE 2019, Jeju Island, Korea (South), September 23–27, 2019, pp. 245–256. IEEE (2019). <https://doi.org/10.1109/RE.2019.00034>
12. Antón, A.I., McCracken, W.M., Potts, C.: Goal decomposition and scenario analysis in business process reengineering. In: Wijers, G., Brinkkemper, S., Wasserman, A.I. (eds.) Advanced Information Systems Engineering, CAiSE'94, Utrecht, The Netherlands, June 6–10, 1994, Proceedings. Lecture Notes in Computer Science, vol. 811, pp. 94–104. Springer (1994). https://doi.org/10.1007/3-540-58113-8_164
13. Bryl, V., Giorgini, P., Mylopoulos, J.: Designing cooperative IS: exploring and evaluating alternatives. In: On the Move to Meaningful Internet Systems 2006, pp. 533–550 (2006). https://doi.org/10.1007/11914853_32
14. Bryl, V., Giorgini, P., Mylopoulos, J.: Requirements analysis for socio-technical systems: exploring and evaluating alternatives. Tech. rep., DIT-06-006. University of Trento, Italy (2006). <http://eprints.biblio.unitn.it/965/1/006.pdf>
15. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. The Kluwer International Series in Software Engineering, Kluwer Academic Publishers Group, Dordrecht (1999)
16. Dalpiaz, F., Giorgini, P., Mylopoulos, J.: Adaptive socio-technical systems: a requirements-based approach. *Requir. Eng.* **18**(1), 1–24 (2013). <https://doi.org/10.1007/s00766-011-0132-1>
17. Deb, N., Chaki, N., Ghose, A.K.: i*tonusmv: a prototype for enabling model checking of i* models. In: 24th IEEE International Requirements Engineering Conference, RE 2016, Beijing, China, September 12–16, 2016, pp. 397–398 (2016). <https://doi.org/10.1109/RE.2016.62>
18. DeVries, B., Cheng, B.H.C.: Automatic detection of feature interactions using symbolic analysis and evolutionary computation. In: 2018 IEEE International Conference on Software Quality, Reliability and Security, QRS 2018, Lisbon, Portugal, July 16–20, 2018, pp. 257–268. IEEE (2018). <https://doi.org/10.1109/QRS.2018.00039>
19. Franch, X.: On the quantitative analysis of agent-oriented models. In: Dubois, E., Pohl, K. (eds.) Advanced Information Systems Engineering, 18th International Conference, CAiSE 2006, Luxembourg, Luxembourg, June 5–9, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4001, pp. 495–509. Springer (2006). https://doi.org/10.1007/11767138_33
20. Franch, X., Grau, G., Quer, C.: A framework for the definition of metrics for actor-dependency models. In: 12th IEEE International Conference on Requirements Engineering (RE 2004), 6–10 September 2004, Kyoto, Japan, pp. 348–349. IEEE Computer Society (2004). <https://doi.org/10.1109/RE.2004.2>
21. Fuxman, A., Mylopoulos, J., Pistore, M., Traverso, P.: Model checking early requirements specifications in tropos. In: 5th IEEE International Symposium on Requirements Engineering (RE 2001), 27–31 August 2001, Toronto, Canada, pp. 174–181 (2001). <https://doi.org/10.1109/ISRE.2001.948557>
22. Giorgini, P., Mylopoulos, J., Sebastiani, R.: Goal-oriented requirements analysis and reasoning in the tropos methodology. *Eng. Appl. Artif. Intell.* **18**, 159–171 (2005). <https://doi.org/10.1016/j.engappai.2004.11.017>
23. Grau, G., Franch, X.: A goal-oriented approach for the generation and evaluation of alternative architectures. In: Oquendo, F. (ed.) Software Architecture, First European Conference, ECSA 2007, Aranjuez, Spain, September 24–26, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4758, pp. 139–155. Springer (2007). https://doi.org/10.1007/978-3-540-75132-8_12
24. Grau, G., Franch, X., Maiden, N.A.M.: Prim: An i*-based process reengineering method for information systems specification. *Inf. Softw. Technol.* **50**(1–2), 76–100 (2008). <https://doi.org/10.1016/j.infsof.2007.10.006>
25. Grubb, A.M., Chechik, M.: Modeling and reasoning with changing intentions: An experiment. In: Moreira, A., Araújo, J., Hayes, J., Paech, B. (eds.) 25th IEEE International Requirements Engineering Conference, RE 2017, Lisbon, Portugal, September 4–8, 2017, pp. 164–173. IEEE Computer Society (2017). <https://doi.org/10.1109/RE.2017.19>
26. Hassine, J., Alshayeb, M.: Measurement of actor external dependencies in GRL models. In: Dalpiaz, F., Horkoff, J. (eds.) Proceedings of the Seventh International i* Workshop co-located with the

- 26th International Conference on Advanced Information Systems Engineering (CAiSE 2014), Thessaloniki, Greece, June 16–17, 2014. CEUR Workshop Proceedings, vol. 1157. CEUR-WS.org (2014). <http://ceur-ws.org/Vol-1157/paper22.pdf>
27. Hassine, J., Alshayeb, M.: Measuring goal-oriented requirements language actor stability. *Inf. Softw. Eng. J.* **13**(1), 203–226 (2019). <https://doi.org/10.5277/e-Inf190106>
 28. Hassine, J., Amyot, D.: A questionnaire-based survey methodology for systematically validating goal-oriented models. *Requir. Eng.* **21**(2), 285–308 (2016). <https://doi.org/10.1007/s00766-015-0221-7>
 29. Hassine, J., Amyot, D.: An empirical approach toward the resolution of conflicts in goal-oriented models. *Softw. Syst. Model.* **16**(1), 279–306 (2017). <https://doi.org/10.1007/s10270-015-0460-6>
 30. Hassine, J., Kroumi, D., Amyot, D.: A game-theoretic approach to analyze interacting actors in GRL goal models. *Requir. Eng.* (2021). <https://doi.org/10.1007/s00766-021-00349-1>
 31. Horkoff, J., Yu, E.: Comparison and evaluation of goal-oriented satisfaction analysis techniques. *Requir. Eng.* **18**(3), 199–222 (2013). <https://doi.org/10.1007/s00766-011-0143-y>
 32. Horkoff, J., Yu, E.S.K.: Evaluating goal achievement in enterprise modeling—an interactive procedure and experiences. In: Persson, A., Stirna, J. (eds.) *The Practice of Enterprise Modeling, Second IFIP WG 8.1 Working Conference, PoEM 2009, Stockholm, Sweden, November 18–19, 2009. Proceedings. Lecture Notes in Business Information Processing*, vol. 39, pp. 145–160. Springer (2009). https://doi.org/10.1007/978-3-642-05352-8_12
 33. Horkoff, J., Yu, E.S.K.: A qualitative, interactive evaluation procedure for goal- and agent-oriented models. In: Yu, E.S.K., Eder, J., Rolland, C. (eds.) *Proceedings of the Forum at the CAiSE 2009 Conference, Amsterdam, The Netherlands, 8–12 June 2009. CEUR Workshop Proceedings*, vol. 453. CEUR-WS.org (2009). <http://ceur-ws.org/Vol-453/paper04.pdf>
 34. Horkoff, J., Yu, E.S.K., Ghose, A.: Interactive goal model analysis applied-systematic procedures versus ad hoc analysis. In: van Bommel, P., Hoppenbrouwers, S., Overbeek, S., Proper, E., Barjjs, J. (eds.) *The Practice of Enterprise Modeling—Third IFIP WG 8.1 Working Conference, PoEM 2010, Delft, The Netherlands, November 9–10, 2010. Proceedings. Lecture Notes in Business Information Processing*, vol. 68, pp. 130–144. Springer (2010). https://doi.org/10.1007/978-3-642-16782-9_10
 35. Horkoff, J., Yu, E.S.K., Liu, L.: Analyzing trust in technology strategies. In: *Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services, PST 2006, Markham, Ontario, Canada, October 30–November 1, 2006. ACM International Conference Proceeding Series*, vol. 380, p. 9. ACM (2006). <https://doi.org/10.1145/1501434.1501446>
 36. ITU-T: Recommendation Z.151 (10/18), User Requirements Notation (URN) language definition, Geneva, Switzerland (2018). <http://www.itu.int/rec/T-REC-Z.151/en>
 37. jUCMNav v7.0.0.: jUCMNav Project (tool, documentation, and meta-model) (2016). <http://softwareengineering.ca/jucmnav>
 38. Jureta, I., Faulkner, S., Schobbens, P.Y.: Clear justification of modeling decisions for goal-oriented requirements engineering. *Requir. Eng.* **13**(2), 87–115 (2008). <https://doi.org/10.1007/s00766-007-0056-y>
 39. Kumar, R., Mussbacher, G.: Textual user requirements notation. In: Khendek, F., Gotzhein, R. (eds.) *System Analysis and Modeling. Languages, Methods, and Tools for Systems Engineering—10th International Conference, SAM 2018, Copenhagen, Denmark, October 15–16, 2018, Proceedings. Lecture Notes in Computer Science*, vol. 11150, pp. 163–182. Springer (2018). https://doi.org/10.1007/978-3-030-01042-3_10
 40. Li, T., Horkoff, J., Mylopoulos, J.: Holistic security requirements analysis for socio-technical systems. *Softw. Syst. Model.* **17**(4), 1253–1285 (2018). <https://doi.org/10.1007/s10270-016-0560-y>
 41. Liaskos, S., McIlraith, S.A., Sohrabi, S., Mylopoulos, J.: Representing and reasoning about preferences in requirements engineering. *Requir. Eng.* **16**(3), 227–249 (2011). <https://doi.org/10.1007/s00766-011-0129-9>
 42. Lima, P., Vilela, J., Gonçalves, E.J.T., Pimentel, J., Holanda, A., Castro, J., Alencar, F.M.R., Lencastre, M.: An extended systematic mapping study about the scalability of i* models. *CLEI Electron. J.* **19**(3), 6 (2016). <https://doi.org/10.19153/cleiej.19.3.6>
 43. Liu, L., Yu, E.S.K.: Designing information systems in social context: a goal and scenario modelling approach. *Inf. Syst.* **29**(2), 187–203 (2004). [https://doi.org/10.1016/S0306-4379\(03\)00052-8](https://doi.org/10.1016/S0306-4379(03)00052-8)
 44. Mirbel, I., Villata, S.: Enhancing goal-based requirements consistency: an argumentation-based approach. In: Fisher, M., van der Torre, L., Dastani, M., Governatori, G. (eds.) *Computational Logic in Multi-Agent Systems. Lecture Notes in Computer Science*, vol. 7486, pp. 110–127. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-32897-8_9
 45. Murukannaiah, P.K., Kalia, A.K., Telang, P.R., Singh, M.P.: Resolving goal conflicts via argumentation-based analysis of competing hypotheses. In: Zowghi, D., Gervasi, V., Amyot, D. (eds.) *23rd IEEE International Requirements Engineering Conference, RE 2015, Ottawa, ON, Canada, August 24–28, 2015*, pp. 156–165. IEEE Computer Society (2015). <https://doi.org/10.1109/RE.2015.7320418>
 46. Neumaier, A.: Basic Properties of Interval Arithmetic. *Encyclopedia of Mathematics and Its Applications*, pp. 1–32. Cambridge University Press, Cambridge (1991). <https://doi.org/10.1017/CBO9780511526473.003>
 47. Osborne, M.: *An Introduction to Game Theory*. Oxford Univ. Press, New York (2004)
 48. Pastor, O., Estrada, H., Martínez, A.: Strengths and weaknesses of the i* framework: an empirical evaluation. In: Yu, E.S.K., Giorgini, P., Maiden, N.A.M., Mylopoulos, J. (eds.) *Social Modeling for Requirements Engineering, Cooperative Information Systems*, pp. 607–644. MIT Press (2011)
 49. Sadiq, M., Jain, S.K.: Applying fuzzy preference relation for requirements prioritization in goal oriented requirements elicitation process. *Int. J. Syst. Assur. Eng. Manag.* **5**(4), 711–723 (2014). <https://doi.org/10.1007/s13198-014-0236-3>
 50. Subramanian, C.M., Krishna, A., Kaur, A.: Reasoning about goal satisfaction for early requirements engineering in the i* framework using inter-actor dependency. In: Kankanhalli, A., Burton-Jones, A., Teo, T.S.H. (eds.) *19th Pacific Asia Conference on Information Systems, PACIS 2015, Singapore, July 5–9, 2015*, p. 89 (2015). <http://aisel.aisnet.org/pacis2015/89>
 51. Sumesh, S., Krishna, A., Subramanian, C.M.: Game theory-based reasoning of opposing non-functional requirements using inter-actor dependencies. *Comput. J.* **62**(11), 1557–1583 (2019). <https://doi.org/10.1093/comjnl/bxy143>
 52. Sutcliffe, A.G., Minocha, S.: Linking business modelling to socio-technical system design. In: Jarke, M., Oberweis, A. (eds.) *Advanced Information Systems Engineering, 11th International Conference CAiSE'99, Heidelberg, Germany, June 14–18, 1999, Proceedings. Lecture Notes in Computer Science*, vol. 1626, pp. 73–87. Springer (1999). https://doi.org/10.1007/3-540-48738-7_7
 53. Vinay, S., Aithal, S., Sudhakara, G.: A quantitative approach using goal-oriented requirements engineering methodology and analytic hierarchy process in selecting the best alternative. In: Kumar, A., Kumar, T.V.S. (eds.) *Proceedings of International Conference on Advances in Computing. Advances in Intelligent Systems and Computing*, vol. 174, pp. 441–454. Springer (2012). https://doi.org/10.1007/978-81-322-0740-5_54

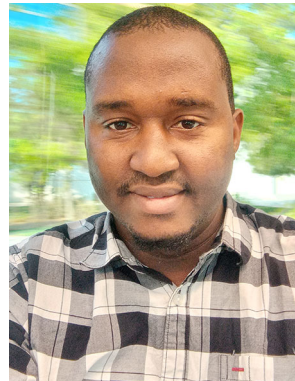
54. van Lamsweerde, A.: Goal-oriented requirements engineering: a guided tour. In: 5th IEEE International Symposium on Requirements Engineering (RE 2001), 27–31 August 2001, Toronto, Canada, pp. 249–262. IEEE Computer Society (2001). <https://doi.org/10.1109/ISRE.2001.948567>
55. van Lamsweerde, A.: Requirements engineering: from craft to discipline. In: Harrold, M.J. Murphy, G.C. (eds.) Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2008), Atlanta, Georgia, USA, pp. 238–249. ACM (2008)
56. van Lamsweerde, A.: Reasoning about alternative requirements options. In: Conceptual Modeling: Foundations and Applications—Essays in Honor of John Mylopoulos, pp. 380–397 (2009). https://doi.org/10.1007/978-3-642-02463-4_20
57. van Lamsweerde, A., Darimont, R., Letier, E.: Managing conflicts in goal-driven requirements engineering. *IEEE Trans. Softw. Eng.* **24**(11), 908–926 (1998). <https://doi.org/10.1109/32.730542>
58. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, Norwell (2000)
59. Wright, H.K., Kim, M., Perry, D.E.: Validity concerns in software engineering research. In: Roman, G.C., Sullivan, K.J. (eds.) *FoSER*, pp. 411–414. ACM, New York (2010)
60. Yu, E.S.K.: Towards modeling and reasoning support for early-phase requirements engineering. In: Proceedings of the 3rd IEEE International Symposium on Requirements Engineering, RE '97, pp. 226–235. IEEE Computer Society, Washington, DC (1997). <http://portal.acm.org/citation.cfm?id=827255.827807>
61. Yu, E.S.K., Mylopoulos, J.: An actor dependency model of organizational work: with application to business process reengineering. In: Proceedings of the Conference on Organizational Computing Systems, COOCS 1993, Milpitas, California, USA, November 1–4, 1993, pp. 258–268. ACM (1993). <https://doi.org/10.1145/168555.168584>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Jameleddine Hassine is an Associate Professor at the department of Information and Computer Science of King Fahd University of Petroleum and Minerals (KFUPM). Dr. Hassine holds a Ph.D. from Concordia University, Canada (2008) and an M.Sc. from the University of Ottawa, Canada (2001). Dr. Hassine has several years of industrial experience within worldwide telecommunication companies; Nortel Networks (Canada) and Cisco Systems (Canada). His main research interests

include requirements engineering (languages and methods), software testing, formal methods, software evaluation, and maintenance. He is actively involved in several funded research projects and has over 50 publications on various research topics in his field. Dr. Hassine published his research in many high-impact journals like Requirements Engineering Journal (REJ), Journal of Systems and Software (JSS), Information and Software Technology (IST), and Software and Systems Modeling (SoSyM).



Muhammad Tukur holds an M.Sc. degree in Software Engineering from King Fahd University of Petroleum and Minerals, Dhahran, KSA (2020). He has a B.Sc. degree in Computer Science from Gombe State University, Nigeria (2015). He is currently an Assistant Lecturer in Computer Science Department, Gombe State University, Nigeria. His main research interests include, among others, Software Engineering, Data Science and Visualization, Requirements Engineering, and IT project management.

agement.