Requirements Eng (2016) 21:29–61
DOI 10.1007/s00766-014-0209-8

ORIGINAL ARTICLE

Interactive goal model analysis for early requirements engineering

Jennifer Horkoff · Eric Yu

Received: 19 November 2012 / Accepted: 24 July 2014 / Published online: 24 August 2014

Abstract In goal-oriented requirements engineering, goal models have been advocated to express stakeholder objectives and to capture and choose among system requirement candidates. A number of highly automated procedures have been proposed to analyze goal achievement and select alternative requirements using goal models. However, during the early stages of requirements exploration, these procedures are difficult to apply, as stakeholder goals are typically high-level, abstract, and hard-to-measure. Automated procedures often require formal representations and/or information not easily acquired in early stages (e.g., costs, temporal constraints). Consequently, early requirements engineering (RE) presents specific challenges for goal model analysis, including the need to encourage and support stakeholder involvement (through interactivity) and model improvement (through iterations). This work provides a consolidated and updated description of a framework for iterative, interactive, agent-goal model analysis for early RE. We use experiences in case studies and literature surveys to guide the design of agent-goal model analysis specific to early RE. We introduce analysis procedures for the *i** goal-oriented framework, allowing users to ask "what if?" and "are certain goals achievable? how? or why not?" The *i** language and our analysis procedures are formally defined. We describe framework implementation, including model visualization techniques and scalability tests. Industrial, group, and individual case studies are applied to test framework effectiveness. Contributions, including limitations and future work, are described.

Keywords Goal-oriented requirements engineering · Goal modeling · Modeling · Model analysis · Model iteration · Interactive modeling · Satisfaction analysis

1 Introduction

Models focusing on stakeholder goals have been proposed for use in requirements engineering (RE) (e.g., [10, 11, 39, 51]). It has been suggested that such models are particularly suitable for elicitation and analysis in early RE as they can show the underlying motivations for systems, capture non-functional success criteria, and show the effects of high-level design alternatives on goal achievement for various stakeholders through a network of dependencies. We call this type of model, including agents with interdependent goals, agent-goal models. Example of agent-goal model frameworks include *i** [51, 52], GRL [3], and Tropos [6].

An agent-goal model can be used to answer "what if?" analysis by propagating the "satisfaction level" of goals onto other goals along the paths of contributions as defined in the model [10]. We refer to this as "forward" analysis. Conversely, one can start from the desired goals and work "backwards" along contribution paths to determine what combinations of choices (if any) will satisfy desired sets of objectives.

This work was undertaken while Jennifer Horkoff was a graduate student in the Department of Computer Science, University of Toronto, Canada.

J. Horkoff (✉)
Department of Information Engineering and Computer Science, University of Trento, Trento, Italy
e-mail: horkoff@disi.unitn.it

E. Yu
Faculty of Information, University of Toronto, Toronto, Canada
e-mail: eric.yu@utoronto.ca

Several procedures have been developed to perform forward and backward analysis on goal models (e.g., [4, 10, 20, 40, 41]). Most of these procedures aim for a high degree of automation, desirable especially for large and complex models. However, during the early stages of requirements exploration, stakeholder goals are typically high-level, abstract, and hard-to-measure. Automated procedures often require formal representations and/or information not easily acquired in early stages (e.g., costs, temporal constraints). Consequently, early RE presents specific challenges for goal model analysis, including the need to encourage and support stakeholder involvement (through interactivity) and model improvement (through iteration).

We address these needs by developing a framework for iterative, interactive analysis of agent-goal models in early requirements engineering. Our framework facilitates analysis through methods, algorithms, and tools. We summarize the contributions of this work as follows:

- Our framework provides *analysis power*, allowing users to ask "what if certain requirements alternatives are chosen?", "is it possible to achieve certain goal(s) in the model? If so, how? If not, why not?"
- Our analysis methods are *interactive*, allowing users to use their knowledge of the domain to make decisions over contentious areas of the model, encouraging stakeholder involvement in the analysis process.
- We provide a guiding *methodology* for goal model creation and analysis.
- Our interactive procedures and methodology aim to encourage model *iteration*, revealing unknown information, and potentially increasing the completeness and accuracy of the models.
- Our analysis procedures are appropriate for early, *high-level analysis*, as they do not require formal or quantitative information beyond what is captured by goal models.
- We provide a clear and formal *interpretation* of our example goal modeling notation ($i*$) and the analysis procedures.
- We place emphasis on procedure *usability*, tested as part of several studies.
- We assess *scalability* of the automated and interactive elements of the framework, showing the procedures scale to models of a reasonable size.

This work improves upon and unifies earlier work by the authors, presenting a cohesive and consistent framework for interactive and iterative early RE model analysis. Development of the backward analysis procedure [29, 31] has helped to clarify the forward analysis procedure, previously described informally in [28, 30]. The backward analysis procedure has evolved since its introduction in [29]—in this work, we include an updated description.

Previous work has described studies which evaluate components of the framework [24, 28, 30, 31, 33, 35]. Here, we present a consolidated view of study results, summarizing discovered strengths and limitations. We present recent scalability results over the framework implementation and compare the consolidated framework to related work.

The paper is organized as follows. After a motivating example in Sects. 1.1 and 2 provides an overview of the agent-goal model language used in our examples ($i*$), including a formal description of the language. Section 3 motivates and describes the analysis procedures, including examples. Section 4 provides a suggested modeling and analysis methodology using the running example. Section 5 describes implementation, including the OpenOME tool, procedural details, visualization techniques, and scalability tests. Section 6 describes the evaluation of the framework through several case studies. Section 7 reviews existing goal model analysis approaches. Section 8 evaluates the contributions of the framework, discussing limitations and future work.

1.1 Motivating example: youth counseling organization

Consider the challenges of a youth counseling organization, studied as part of a multi-year strategic requirements analysis project undertaken by the authors and other colleagues [13]. The not-for-profit organization focuses on counseling for youth over the phone, but must now expand their ability to provide counseling via the Internet. Online counseling could be viewed by multiple individuals and may provide a comforting distance which would encourage youth to ask for help. However, in providing counseling online, counselors lose the cues they would gain through live conversation, such as timing or voice tone. Furthermore, there are concerns with confidentiality, protection from predators, public scrutiny over advice, and liability over misinterpreted guidance. The organization must choose among multiple technical options to expand their internet counseling service, including a modification of their existing anonymous question and answer system, discussion boards, wikis, text messaging, chat rooms. In order to make strategic decisions, a high-level understanding of the organization, system users, and the tradeoffs among technical alternatives is needed.

Modeling methods described in previous work can be applied to understand the domain, producing agent-goal models which include systems, stakeholders, goals, contributions, and dependencies [51]. Figure 1 contains a simplified example of an agent-goal model created for this domain. In this model, the Counseling Organization must choose between several forms of online counseling. Their

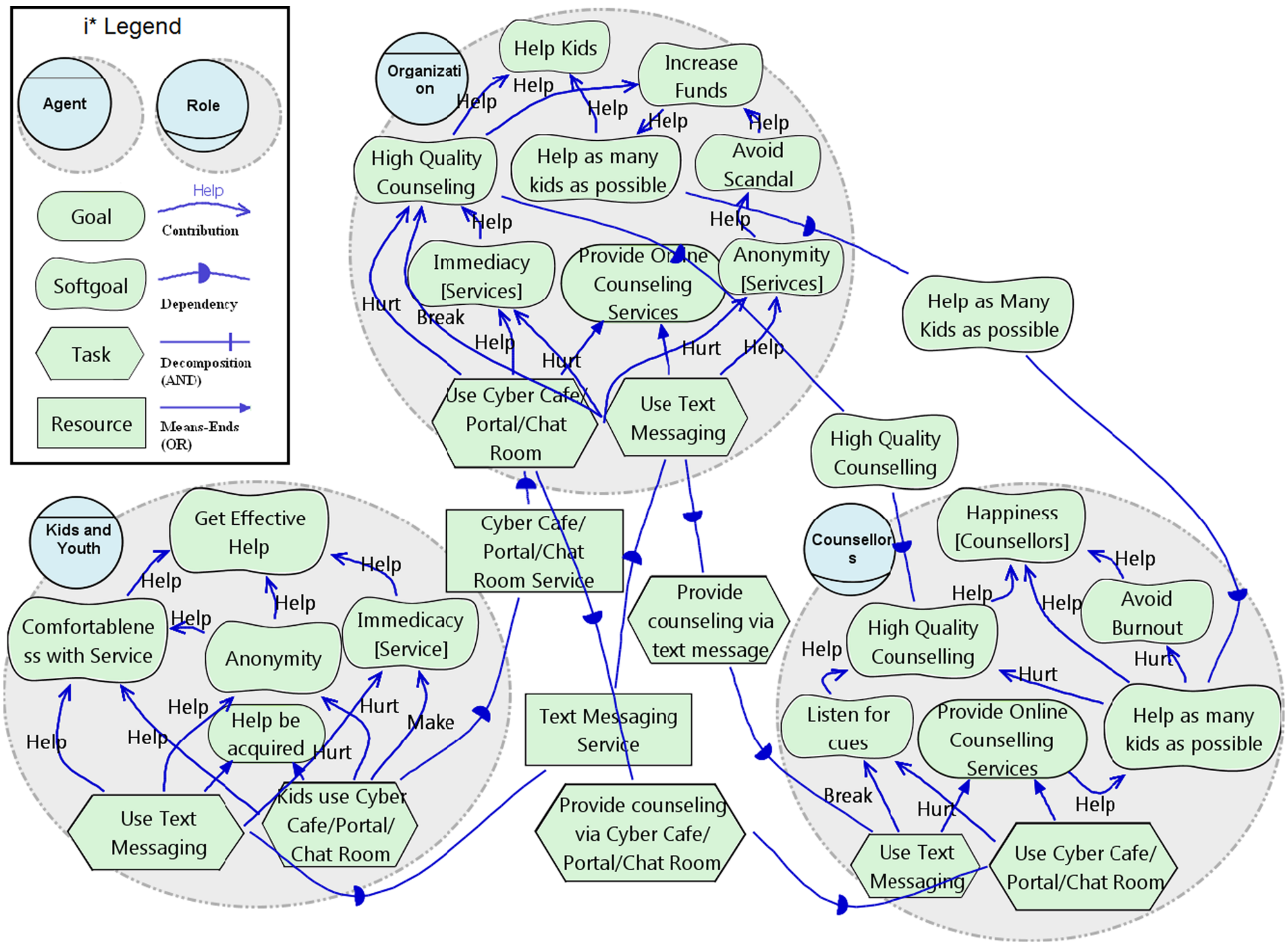


Fig. 1 *i** Model representing simplified relationships and alternatives for online counseling (adapted from [28, 30])

choices affect not only their goals, but also the goals of the Counselors and the Kids and Youth. The model contains three *actors*: the Organization (top), Kids and Youth (bottom left), and Counselors (bottom right). The Organization, an *agent*, wants to achieve several *softgoals*, including Helping Kids, Increasing Funds, and providing High Quality Counseling. These goals are difficult to precisely define, yet are critical to the organization. The Organization has the *hard* goal of Providing Online Counseling Services and explores two alternative tasks for this goal: Use Text Messaging and Use CyberCafé/Portal/ChatRoom. These alternatives contribute positively or negatively by various degrees to the Organization's goals, which in turn contribute to each other. For example, Use Text Message *hurts* Immediacy which *helps* High Quality Counseling.

The Organization *depends* on the Counselors to provide the alternative counseling services and for many of its softgoals, for example, High Quality Counseling. Kids and Youth depend on the Organization to provide various counseling services, such as CyberCafé/Portal/Chat Room. Both the Counselors and Kids have their own goals to achieve, also receiving contributions from the counseling alternatives. Although the internal goals of each actor may be similar, each actor is autonomous, including the meaning individually attributed to goal, e.g., High Quality Counseling may mean something different for the Counselor than for the Organization.

Examining this type of model raises several questions: Which counseling alternative is the most effective, and for whom? Are there alternatives which could achieve each actor's goals? If not, why not? What important information is missing from the model? Is the model sufficiently correct? Generally, how can such an organization explore and evaluate options for online counseling, balancing the needs of multiple parties, while dealing with the complexity of the model and domain?

Although some questions may be answered by studying the model, tracing effects consistently quickly becomes too complex for humans. The model in Fig. 1 is a simplified version of a larger model, tracing the effects of alternative functionality is especially difficult when the model becomes large. There is a need for systematic analysis procedures which help the modeler to trace effects in order to answer domain questions, evaluate alternative requirements, and explore the model. Such procedures should account for the early, high-level, and exploratory nature of the models and elicitation process. We return to our motivating example when illustrating analysis procedures in Sect. 3.

2 The *i** agent-goal modeling framework: variation and formalization

In order to aid comprehension, illustration, and implementation, analysis procedures introduced in our framework should be described concretely, over a specific language. Several possible goal-oriented languages are available (e.g., NFR [10], Tropos [6], KAOS [11]). The *i** framework, which builds upon the NFR framework, has been used as a basis for agent-goal modeling in the GRL and Tropos frameworks. As such, it includes many existing goal model language concepts. Other frameworks, such as KAOS, do not support informally or imprecisely defined softgoals, making them more suitable for later RE specification and analysis. We select the *i** framework as an underlying base for our analysis procedure (limitations of this selection are discussed in Sect. 8.2). This section provides a high-level description of *i**, discusses variation in *i** use, then provides a formal definition of *i** concepts, facilitating a formal description of our agent-goal model analysis, consolidating work presented in [26, 29, 31].

2.1 The *i** framework

*i** models are intended to facilitate exploration of the system domain with an emphasis on social aspects by providing a graphical depiction of system actors including their intentions, dependencies, and alternatives [51, 52]. The agent-oriented aspect of *i** is represented by *actors*, including *agents* and *roles*, and the associations between them.

Actors *depend* upon each other for the accomplishment of tasks, the provision of *resources*, the satisfaction of *goals* and *softgoals*. Softgoals are goals without clear-cut criteria for satisfaction; therefore, a softgoal is satisfied when it is judged to be sufficiently satisfied. *Dependency* relationships include the *depender*, the actor depending on another actor, the *dependum*, the intention being depended upon, and the *dependee*, the actor being depended upon.

The *intentions* which motivate dependencies are explored inside each actor, considering the goals, softgoals, tasks, and resources explicitly desired by the actors. Dependencies are linked to specific intentions within the dependee and depender. The intention depending on the dependum is referred to in this work as the *depender intention*, while the intention depended on to satisfy the dependum is referred to as the *dependee intention*.

The interrelationships between intentions inside an actor are depicted via three types of links. *Decomposition* links show the intentions which are necessary in order to accomplish a task. *Means-Ends* links show the alternative tasks which can accomplish a goal. *Contribution* links show the effects of softgoals, goals, and tasks on softgoals. Positive/negative contributions representing evidence which is sufficient enough to satisfice/deny a softgoal are represented by *Make/Break* links, respectively. Contributions with positive/negative evidence that is not in itself sufficient enough to satisfice/deny a softgoal are represented by *Help/Hurt* links. Positive/negative evidence of unknown strength can be represented by *Some+/Some-* links.

2.2 *i** Variations

The description of the *i** framework by Yu in [51] aimed to be flexible enough to facilitate modeling of early requirements, leaving the language open to a certain degree of interpretation and adaptation. Consequently, the core syntax of the *i** framework has often been modified (e.g., [3, 6]). We aim to support common variations from *i** syntax as introduced in [51]. Our previous work in [26] has surveyed *i** syntax variations in research papers and coursework. Commonly occurring syntactical structures are classified under “strict” and “loose” versions of *i** syntax, corresponding to syntax errors and warnings, respectively. We use this survey of *i** syntax variations to determine how broad or how flexible to make our formal definition, aiming to create a balance between clarity and flexibility. A full list of syntax variations supported by our definition can be found in [24].

2.3 Formalization

To facilitate partial automation of analysis, we introduce a more formal description of the *i** framework. In our description, we use the following notation:

- $\mapsto$ is used as a mapping from an intention or relation to a member of a set, so $i \mapsto \{a, b\}$ means that i maps to either a or b.
- $\rightarrow$ is used to represent relationships between elements, so if $(i_1, i_2) \in \mathcal{R}$ we write this as $\mathcal{R} : i_1 \rightarrow i_2$.

We express agent-goal model concepts formally as follows.

Definition 1 (*agent–goal model*) An agent-goal model is a tuple $\mathcal{M} = <\mathcal{I}, \mathcal{R}, \mathcal{A}>$, where $\mathcal{I}$ is a typed set of intentions, $\mathcal{R}$ is a set of relations between intentions, and $\mathcal{A}$ is set of actors.

Definition 2 (*intention type*) Each intention maps to one type in the *IntentionType* set, $\mathcal{I} \mapsto IntentionType$, where $IntentionType = \{Softgoal, Goal, Task, Resource\}$.

Definition 3 (*relation type*) [Relation Type] Each relation maps to one type in the RelationType set, $\mathcal{R} \mapsto RelationType$, where $RelationType = \{R^{me}, R^{dec}, R^{dep}, R^{c}\}$. These relationships correspond to means-ends, decomposition, dependency, and contribution links, respectively. R^c can be broken down into a further set $ContributionType = \{R^m, R^{hlp}, R^u, R^{hrt}, R^b\}$ where if $r \in R \mapsto R^c$ then $r \mapsto ContributionType$. The contribution link types correspond to make, help, unknown, hurt, and break, respectively.

Definition 4 (*relation behavior*) The following relationships are binary (one intention relates to one intention, $R : I \rightarrow I$): R^{dep}, R^c. The remaining relationships (R^{me}, R^{dec}) are $(n+1)$-ary (one to many intentions relate to one intention), $R : I \times \ldots \times I \rightarrow I$.

The formalism could be supplemented to include the mapping from intentions to actors, actor types, and actor association links. Currently, these types do not play a role in the automated portion of our framework. We leave their inclusion in the formalism to future work. For simplicity, we treat Some+/Some- as Help/Hurt, respectively. Thus, we exclude these links from *ContributionType*.

We define several other concepts useful for analysis, such as leaves, roots, and positive/negative links.

Definition 5 (*leaf/root intention*) An intention $i \in I$ is a leaf if there does not exist any relation, $r \in R$ such that $r : I \rightarrow i$ or $r : I \times \ldots \times I \rightarrow i$, it is a root if there does not exist any relation, $r \in R$ such that $r : i \rightarrow I$ or $r : i \times \ldots \times I \rightarrow I$.

Definition 6 (*positive/negative link*) A relation $r \in R$ is positive if $r \mapsto Pos = R^m, R^{hlp}$, it is negative if $r \mapsto Neg = R^{hrt}, R^b$.

3 Interactive analysis

This section describes qualitative, interactive evaluation procedures for goal- and agent-oriented models, allowing the user to compare alternatives in the domain, asking forward, "what if?" type questions, and finding satisfying solutions using backward, "are these goals achievable?" questions. The forward procedure has previously been described in [28, 30]. Here, the description is expanded and improved, described more precisely using the formalism from Sect. 2.3. The backward analysis procedure described in this section has appeared in [29]. Here, we improve upon the description, presenting a unifying description of forward and backward analysis, using the same illustrative, counseling service example.

In the rest of this section, we motivate the need for forward and backward analysis, provide a procedure overview, and required definitions and propagation rules. We end with concrete examples of both forward and backward analysis.

3.1 Challenges and motivation

In this section, we use the counseling service model from Sect. 1.1 (Fig. 1) to answer example "what if?" and "are certain goals achievable?" questions in an "ad hoc" manner, without using a systematic or semiautomated procedure. This experience reveals some of the more detailed challenges associated with analyzing goal models, motivating the need for systematic analysis as introduced in this section.

Forward analysis. In Sect. 1.1, we asked "Which counseling alternative is the most effective?" We could start this analysis by considering the alternative where Use Text Messaging (shortened hereafter to Text), represented as a task in the model, is implemented, and Use Cyber Café/Portal/Chat Room, another task (shortened hereafter to Chat), is not implemented. The reader can try to use their knowledge of *i** syntax provided in Sect. 2 to trace the effects of the satisfaction or denial of these tasks through the links in the model. In one path inside of the Kids and Youth actor, for example, Text would help Anonymity, which would help both Comfortableness with Service and Get Effective Help. In another path, Text would hurt Immediacy [Service], which, in turn helps Get Effective Help. In yet another path, Chat is not implemented, yet this task has a help effect on Comfortableness (with service), which in turn helps Get Effective Help again. Considering these multiple sources of incoming evidence (and there are more paths to trace) is Get Effective Help satisfied? Partially satisfied? Does it have conflicting evidence? How can we make use of stakeholder knowledge in order to combine and resolve multiple sources of evidence for softgoals?

When tracing the effects manually, it is cognitively difficult to follow all paths and make these decisions manually. In this example, we have not even left the boundaries of the Kids and Youth. When considering the effects of dependencies into and out from the actor, tracing the effects of alternatives through the paths of links becomes even more complicated.

Backward analysis. As a model may contain many alternatives, it is helpful to find key promising alternatives by asking questions in the backward direction. Given certain top-level goal targets, “Are the goals achievable?”, “If so, how?”, and “If not, why?” For example, is there an alternative which causes Get Effective Help in Kids and Youth to be partially satisfied? To answer this manually, we must trace the links backward until we find potential solutions.

As we have seen while manually propagating in the forward direction, some softgoals receive many sources of incoming evidence through contribution links. During backward analysis, we must work backward to determine the labels for contributing intentions, again making use of stakeholder domain knowledge. For example, to at least partially satisfy Get Effective Help, what level of satisfaction do the three contributing goals (Comfortableness with Service, Anonymity, Immediacy [Service]) need? In one combination, we could judge that it would be sufficient for these three softgoals to be at least partially satisfied. From this, we could continue to trace links backward down to the task alternatives. For Immediacy to be partially satisfied, we can judge that Text should be denied (not implemented), while Chat should be satisfied. The target label for Anonymity leads us to an opposite judgment. We can see that this selection of analysis results will not produce a consistent solution, we must return and re-evaluate our previous judgments, if possible.

We can see that the process of tracing branching backward paths, backtracking through judgments, is challenging to perform manually. What is needed is an automated process, tracing down the links to find contributing effects, finding areas requiring judgment, then backtracking to previous judgments when judgments result in contradictions (e.g., satisfied and not satisfied).

In formulating such an interactive backward procedure, we face some interesting questions and technical challenges. What types of questions could and should be posed to the user, and at what point in the procedure? How can we capture and make use of stakeholder knowledge through human judgments? When a choice does not lead to an acceptable solution, to what state does the procedure backtrack? How can we present information about conflicts to the user? Is there a computationally realistic approach? The backward analysis procedure introduced in this work represents one approach to answering these questions.

3.2 Procedure overview

The analysis procedure starts with an analysis question of the form “How effective is an alternative with respect to model goals?” or “Are certain goals achievable?” The procedure makes use of a set of qualitative evaluation labels assigned to intentions to express their degree of satisfaction or denial. The process starts by assigning labels to intentions related to the analysis question. These labels are propagated through the model links, either forward or backward, using defined rules. The procedure is interactive when the user must make judgments over conflicting or partial incoming or outgoing evidence for softgoals. The final satisfaction and denial labels for the intentions of each actor are analyzed in light of the original question. In the forward direction, an assessment is made as to whether the analysis alternative sufficiently achieved key goals. In the backward direction, the solution achieving key goals (if found) is examined. These results may stimulate further analysis and potential model refinement. We can summarize the procedure steps as follows:

1. *Initiation*: The evaluator decides on an analysis question and applies corresponding initial evaluation labels to the model. The initial labels are added to a set of labels to be propagated.
   Steps 2 and 3 are performed iteratively, until there is nothing new to propagate (forward) or a contradiction has been found and there are no new applicable judgments (backward).
2. *Propagation*: The evaluation labels are propagated through the model. Results propagated through contribution links are stored in the destination softgoal.

   2.b. *Backtrack*: (Backward) if a contradiction is found, the procedure backtracks to the last set of softgoal resolutions, if such a set exists.
3. *Softgoal resolution*: Sets of multiple labels are resolved by applying automatic cases or manual judgments, producing results which are incorporated back in to the propagation.
4. *Assessment*: The final results are examined in light of the initial analysis question. Model issues can be discovered, and further possibilities are evaluated.

3.3 Qualitative analysis labels and predicates

We adopt the qualitative labels used in NFR evaluation [10], replacing “weakly” with “partially.” The resulting labels are *satisfied*, *partially satisfied*, *conflict*, *unknown*, *partially denied*, and *denied*. The satisfied (✓) label represents the presence of evidence which is sufficient to satisfy a goal. Here, *evidence* comes from connected intentions, which themselves have evidence of the aforementioned types. Partially satisfied (✓.) represents the presence of positive evidence not sufficient to satisfy a goal. Partially denied (✗.) and denied (✗) have the same definition with respect to negative evidence. Conflict (↯) indicates the presence of both positive and negative evidence judged to have roughly the same magnitude.

Unknown (?) represents the situation where there is evidence, but its effect is unknown. We use partially satisfied and denied labels for tasks, resources, and goals, despite their clear-cut nature, to allow for greater expressiveness.

In order to express evaluation evidence as part of our formalism, we introduce analysis predicates, similar to those used in Tropos analysis [21].

Definition 7 (*analysis predicates*) Model analysis evidence is expressed using a set of predicates, $\mathcal{V} = \{S(i), PS(i), C(i), U(i), PD(i), D(i)\}$ over $i \in \mathcal{I}$. Here $S(i)/PS(i)$ represents evidence of full/partial satisfaction, $C(i)$ represents conflict, $U(i)$ represents unknown, and $D(i)/PD(i)$ represents full/partial denial.

It is important to note that analysis labels and predicates, although similar, are not handled in exactly the same way by our procedure. Typically, there is a one-to-one mapping between labels and predicates for an intention, and labels can be seen as the graphical representation of predicates, while predicates are the encoding of labels. However, in our implementation, it is possible for more than one analysis predicate to hold (be true) for an intention. Such situations are resolved through *human judgment*, with the output being a single label/predicate displayed on the intention (more detail provided in Sects. 3.5.2 and 3.6.2).

The predicates which hold for an intention tell us nothing about whether the other evaluation predicates hold for this intention. For example, a value of true for $S(\text{Text})$ does not imply that $D(\text{Text})$ is false, and a false value for $S(\text{Text})$ only means that S does not hold, not that $D(\text{Text})$ or any other predicate is true.

Similarly, in our framework, conflict predicates are not automatically derived from other, non-conflict predicates (unless there is a contribution link of the type *Conflict*). For example, $S(\text{Text})$ and $D(\text{Text})$ does not imply $C(\text{T}ext)$. This allows the user greater flexibility, giving the user the option to resolve conflicting evidence through human judgment. We still use the term *analysis predicate conflict* to indicate a situation such as $S(\text{Text})$ and $D(\text{Text})$, where more than one analysis predicate holds for an intention and those predicates represent conflicting evidence.

Definition 8 (*analysis predicate conflict*) When, for an intention $i \in \mathcal{I}$, a predicate from more than one of the following four sets is true: $\{S(i), PS(i)\}, \{U(i)\}, \{C(i)\}, \{PD(i), D(i)\}$

We also make use of the term *contradiction*, where an analysis predicate, $v(i)$, is both true and false $(v(i) \wedge \neg v(i))$.

3.4 Analysis runs and initial labels

Analysis is started by placing a set of initial labels reflecting an analysis question on the model. In our Fig. 1 counseling service model, we have asked in the forward direction "What if Text and not Chat is implemented?" We can express this question by labeling Text as satisfied and Chat as denied, expressed in our procedure by making the following analysis predicates true: $S(\text{Text})$ and $D(\text{C}hat)$. In the backward direction, we have asked "is it possible for Get Effective Help to be partially satisfied?" In backward analysis, initial labels are often called *targets*, as they are desired outcome of analysis. In this case, the target would be expressed using the predicate $PS(\text{GetEffectiveHelp})$.

Our example initial labels have been applied to a subset of our counseling service example in Fig. 2 (also showing final analysis results), where elements receiving forward and backward initial labels are highlighted green and blue (medium and dark gray), respectively.

We can express the selection of initial analysis labels as follows:

Definition 9 (*initial analysis labels*) For some subset of intentions within an agent-goal model, $i_1 \ldots i_n \in \mathcal{I}$, a selection of analysis labels is made and is encoded with the corresponding analysis predicates, $v(i_1)\ldots v(i_n) \in \mathcal{V}$. This selection represents an analysis question in the domain. We refer to the set of predicates representing initial labels, $v(i_1)\ldots v(i_n)$, as $\mathcal{IL}$.

In this work, the selection of initial labels in both the forward and backward procedure is called an *alternative*. Often, when referring to *i** models, an alternative is also used to mean the choice between means in a means-ends relationship. For example, in our counseling organization model, Provide Online Counseling can be achieved via one (or both) of Chat or Text. In order to produce evaluation results which take into account all connected intentions in the model, forward analysis typically places initial labels both over alternatives for goals and over other intentions, covering at least all leaves. Similarly, backward analysis targets cover intentions across the model, typically covering most root intentions. We often use the broader notion of an alternative in this work, using the narrower (means-ends) meaning only for specific model examples.

Together, we call the selection of initial labels, human judgments, and the corresponding analysis results an analysis run, defined more precisely as follows:

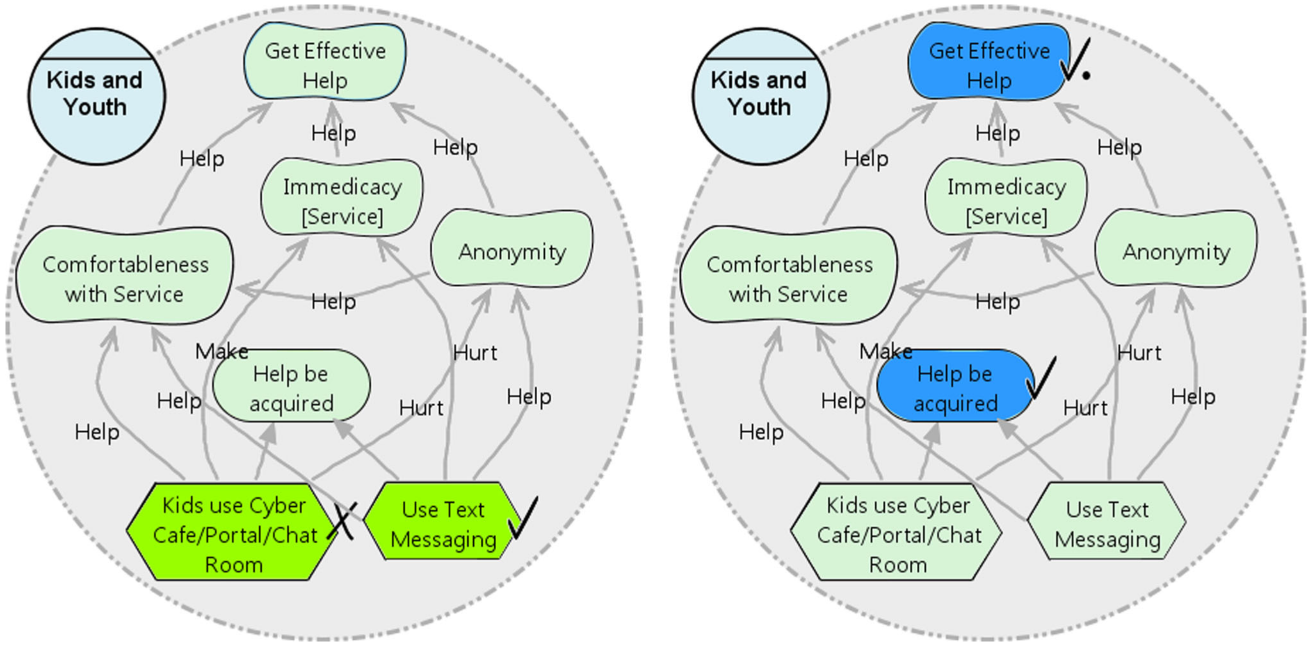


Fig. 2 Kids and Youth actor showing initial forward analysis labels and leaf highlighting (*left*) and initial backward analysis labels and root highlighting (*right*)

Definition 10 (*analysis run*) The results of a single run of the analysis procedure. Given a selection of initial analysis labels translated to predicates, $\mathcal{IL}$, for some subset of intentions, $i_1 \ldots i_n \in \mathcal{I}$, within an agent-goal model, and given a set of human judgments (see Sect. 3.5.2), the analysis algorithm produces analysis results for a set of intentions, $i_1 \ldots i_m \in \mathcal{I}$,
$v(i_1) \ldots v(i_m) \in \mathcal{V}$, visualized using analysis labels. If a different set of initial analysis labels or judgments were used, this would be a different analysis run, with potentially different results over $i_1 \ldots i_m$.

Any intention could be selected to receive an initial label as part of an analysis run (although leaf and root intentions are the most likely). Furthermore, each initial intention could be given one of six labels. If there are n intentions in the model, there are 6^n possible sets of initial analysis labels over the model, although the number of intentions given initial labels is usually far less than n. Generally, evaluating an analysis alternative is not helpful unless it reflects a realistic potential selection of requirements, i.e., a useful analysis question in the real world. Initial labels should be derived from domain-relevant questions or be selected to test the "sanity" of the model. The development of analysis questions is discussed in more detail while considering methodology in Sect. 4.

3.5 Forward analysis

In this section, we provide more technical details concerning forward analysis, including propagation rules and the resolution of multiple sources of evidence using human judgments.

3.5.1 Forward propagation rules

We present rules in order to facilitate a standard propagation of labels through agent-goal model relationships (links). We develop axioms which cover the propagation of each possible analysis label through each type of relation.

Generally, for an intention $i \in \mathcal{I}$, which is the destination of a relationship, $r \in \mathcal{R}, r : i_1 \times \ldots \times i_n \rightarrow i$ forward propagation predicates take on the form:

Forward propagation
(Some combination of $v(i_1) \ldots v(i_n)$, $v \in \mathcal{V}) \rightarrow v(i)$

We present propagation rules for dependency, decomposition, and means-ends relationships, with rules presented in Table 1.

Dependency links. The nature of a dependency indicates that if the *dependee intention* is satisfied then the intention depended for (the *dependum*) will be satisfied. If the dependum is satisfied, then the *depender intention* will be satisfied as well. Thus, the analysis label of the dependee intention is propagated directly to the depender intention through the dependum. We express this propagation by looking only at a piece of the dependency link at a time (from the dependee intention to the dependum, or from the dependum to the depender intention), supporting flexibility for syntax variations (e.g., sharing or omitting dependums). We express propagation for these relationships in the axiom below.

$$\text{Given}: r^{dep}: i_s \rightarrow i_d, \quad v(i_s) \in \mathcal{V} \\ v(i_s) \rightarrow v(i_d). \tag{1}$$

Recall that s is used to indicate the source of the relationship, while d indicates the destination (see top picture in Table 1). In this case, we are referring to the source and destination of the analysis label in forward propagation, not necessarily the source and destination of the dependency. It could be argued that as the depender intention is depending on something, it is the "source" of the dependency, but in forward analysis, it is the destination of the analysis label.

Decomposition links. Decomposition links depict the intentions necessary to accomplish a task, indicating the use of an AND relationship, selecting the "minimum" label among the source labels. In order to facilitate this type of propagation, we must provide an ordering over our set of analysis labels, $\mathcal{V}$, defining minimum and maximum. Unlike Tropos analysis [21], we are not able to define a total order over analysis predicates, such that for $v(i) \in \mathcal{V}, v_1 \geq v_2 \Leftrightarrow v_1 \rightarrow v_2$, as there are no implication relationships between satisfaction/denial labels and unknown labels, and as we have chosen not to add implications producing conflict labels (Sect. 3.3). We are, however, able to define and utilize the following partial orders.

$$\forall i \in I: S(i) \geq PS(i) \quad \Leftrightarrow S(i) \rightarrow PS(i) \\ D(i) \geq PD(i) \quad \Leftrightarrow D(i) \rightarrow PD(i) \tag{2}$$

These partial orders have been used to reduce the number of axioms required to express propagation in Table 1. In addition, we can define a conceptually useful total order where $v_1 \geq v_2$ implies that v_1 is more desirable (or "higher") than v_2. This order is as follows:

$$S(i) \geq PS(i) \geq U(i) \geq C(i) \geq PD(i) \geq D(i). \tag{3}$$

Here we chose an optimistic ordering between $U(i)$ and $C(i)$, with the idea that no information (unknown) is better (closer to being satisfied) than conflicting information. From this ordering, we can define max and min labels.

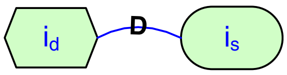
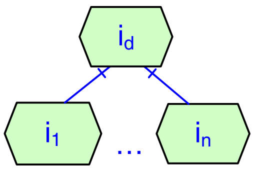
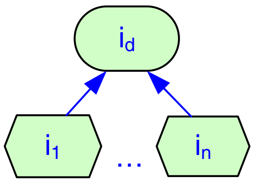
Definition 11 (max (min) label) Given a set of analysis labels, $v(i_1)\ldots v(i_n), v \in V$, over $i_1 \times \cdots \times i_n$, $i \in \mathcal{I}$, the maximum (minimum) label is the largest (smallest) label, v, given the ordering in Eq. 3.

From this, we can define propagation over decomposition links, listed in the middle of Table 1:

$$\text{Given}: rdec: i_1 \times \cdots \times i_n \rightarrow i_d, v(i_1)\ldots v(i_n) \in \mathcal{V}, \\ \text{minimum}(v(i_1)\ldots v(i_n)) \rightarrow v(i_d). \tag{4}$$

Means-ends links. Similarly, Means-Ends links depict the alternative tasks which are able to satisfy a goal, indicating an OR relationship, taking the maximum label of intentions

Table 1 Propagation axioms for dependency, decomposition, and means-ends

Dependency	$\mathbf{V(i_s)}$	$\mathbf{V(i_s) \rightarrow V(i_d)}$
	$v \in V$	$v(i_s) \rightarrow v(i_d)$
Decomposition	$\mathbf{V(i_d)}$	$\mathbf{V(i_1)\ldots V(i_n) \rightarrow V(i_d)}$
	S	$(\bigwedge_{j=1}^{n} S(i_j)) \rightarrow S(i_d)$
	PS	$(\bigwedge_{j=1}^{n} PS(i_j)) \rightarrow PS(i_d)$
	U	$(((\bigvee_{j=1}^{n} U(i_j)) \wedge (\bigwedge_{k=1}^{j} PS(i_k) \wedge \bigwedge_{p=j+1}^{n} PS(i_p))) \rightarrow U(i_d)$
	C	$(((\bigvee_{j=1}^{n} C(i_j)) \wedge (\bigwedge_{k=1}^{j} \neg PD(i_k) \wedge \bigwedge_{p=j+1}^{n} \neg PD(i_p))) \rightarrow C(i_d)$
	PD	$(\bigvee_{j=1}^{n} PD(i_j)) \wedge (\bigwedge_{k=1}^{j} \neg D(i_k) \wedge \bigwedge_{p=j+1}^{n} \neg D(i_p))) \rightarrow PD(i_d)$
	D	$(\bigvee_{j=1}^{n} D(i_j)) \rightarrow D(i_d)$
Means-ends	$\mathbf{V(i_d)}$	$\mathbf{V(i_1)\ldots V(i_n) \rightarrow V(i_d)}$
	S	$(\bigvee_{j=1}^{n} S(i_j)) \rightarrow S(i_d)$
	PS	$(((\bigvee_{j=1}^{n} PS(i_j)) \wedge (\bigwedge_{k=1}^{j} \neg S(i_k) \wedge \bigwedge_{p=j+1}^{n} \neg S(i_p))) \rightarrow PS(i_d)$
	U	$(((\bigvee_{j=1}^{n} U(i_j)) \wedge (\bigwedge_{k=1}^{j} \neg PS(i_k) \wedge \bigwedge_{p=j+1}^{n} \neg PS(i_p))) \rightarrow U(i_d)$
	C	$(((\bigvee_{j=1}^{n} C(i_j)) \wedge (\bigwedge_{k=1}^{j} PD(i_k) \wedge \bigwedge_{p=j+1}^{n} PD(i_p))) \rightarrow C(i_d)$
	PD	$(((\bigwedge_{j=1}^{n} PD(i_j)) \rightarrow PD(i_d)$
	D	$(\bigwedge_{j=1}^{n} D(i_j)) \rightarrow D(i_d)$

in the relation (bottom of Table 1). To increase flexibility, the OR is interpreted to be inclusive.

$$\text{Given}: r^{me}: i_1 \times \cdots \times i_n \rightarrow i_d, v(i_1)\ldots v(i_n) \in \mathcal{V}, \\ \text{maximum}(v(i_1)\ldots v(i_n)) \rightarrow v(i_d) \quad (5)$$

Contribution links. We adopt the Contribution link propagation rules from the NFR procedure, as shown in Table 2. These rules intuitively reflect the semantics of contribution links. For instance, the *Make* link represents a positive contribution which is sufficient to satisfy a softgoal. Therefore, this link propagates satisfied and partially satisfied labels as is. For negative evidence, links are treated as symmetric (evidence is also propagated in the inverse). In other words, if an intention *Makes* another intention when it is satisfied, it effectively *Breaks* this intention when it is denied. As a result, the *Make* link propagates denied and partially denied labels as is. Propagation rules for the *Help* link are similar, except that this link provides only a partial positive contribution. As a result, full evidence is weakened when passing through this link, although partial evidence remains partial (is not weakened enough to be non-existent).

The propagation rules for the *Break* and *Hurt* links are nearly symmetric to *Make* and *Help*; positive evidence becomes negative and negative evidence becomes positive. Asymmetry occurs when denied is propagated through break, with the idea that negative evidence through a negative link is positive, but not strong enough to produce full satisfaction [10]. The *Some+* and *Some-* links are evaluated pessimistically, treating them as *Help* and *Hurt* links, respectively. As such they are omitted from Table 2. *Conflict* and *Unknown* labels always propagate without modification, unless through an unknown link, where a *Conflict* becomes *Unknown*.

The rules in Table 2 can be expressed using propagation axioms, similar to the axioms described for dependency, decomposition, and means-ends links. Generally, given the

Table 2 Propagation rules showing resulting labels for contribution links adapted from [10]

Source Label		*Contribution* Link Type				
	Name	Make	Help	Break	Hurt	Unkn.
✓	Satisfied	✓	✓.	✗	✗.	?
✓.	Partially Satisfied	✓.	✓.	✗.	✗.	?
?	Unknown	?	?	?	?	?
⚡	Conflict	⚡	⚡	⚡	⚡	?
✗.	Partially Denied	✗.	✗.	✓.	✓.	?
✗	Denied	✗	✗.	✓.	✓.	?

type of contribution link, $r^c \mapsto R^m, R^{hlp}, R^u, R^{hrt}, R^b$, and the source label, $v(i_s)$, a rule for each row/column combination of Table 2 of the form $v(i_s) \rightarrow v(i_d)$, can be defined. For example, for a help contribution link (R^{hlp}) from and intention i_s to an intention i_d (row ✓, column Help), $S(i_s) \rightarrow PS(i_d)$.

3.5.2 *Resolving multiple contributions*

Softgoals are often the recipient of multiple incoming contribution links, each of which produces an evaluation label as per the rules in Table 2. In the forward direction, it is our desire to resolve (combine) multiple incoming labels into a single, resulting label. We collect incoming labels in a *label bag* and then resolve labels either by identifying cases where the label can be determined automatically, or by human judgment: presenting the incoming labels to the user and asking for a single resulting label.

Automatic resolution. We describe the cases where multiple incoming labels in forward analysis can be resolved automatically in Table 3. If there is only one incoming label (case 1), the result is that label. If there are multiple labels of the same polarity with one full label (case 2), the result is the full label. If the same human judgment has already occurred within the same analysis run, the previous answer will be used (case 3). Finally, if a previous human judgment produced a full label, and the set of labels has become more positive or more negative matching the polarity of the full label, the result is automatically the same full label (case 4).

For instance, in our running example, given our initial labels, the Immediacy [Service] softgoal in Kids and Youth receives both a partially denied and a fully denied label from incoming contribution links, resolved to a denied label using Case 2 in Table 3, reflecting the idea that evidence propagated to softgoals is roughly cumulative. We show the example including final analysis results for both forward and backward analysis in Fig. 3. A detailed explanation of the results is given in Sect. 3.7.

Human Judgment. Human judgment is used to decide on a label for softgoals in the cases not covered in Table 3. By representing incoming analysis labels in their predicate form, we can formally define what it means for an intention to require human judgment.

Definition 12 (*Need for human judgment*) An intention, $i \in I$, needs human judgment if:

- i is the recipient of more than one incoming contribution link, i.e., there exists an r_1 and $r_2 \in \mathcal{R}$ such that $r_1^c: i_1 \rightarrow i$ and $r_2^c: i_2 \rightarrow i$, AND:

Table 3 Cases where softgoal labels can be automatically determined (adapted from [30])

Label bag contents	Resulting label
1. The bag has is only one label, e.g., ✗ or ✓.	The label: ✗ or ✓.
2. All labels in the bag are of the same polarity, and the full label is present, e.g., ✓., ✓, ✓. or ✗, ✗	The full label: ✓ or ✗
3. The human judgment situation has already occurred for this intention and the answer is known	The known answer
4. A previous judgment situation for this intention has produced ✓ or ✗, and the new contributions are of the same polarity	The full label: ✓or ✗

- There is an analysis predicate conflict, as defined in Definition 8.
- Or, $PS(i)$ or $PD(i)$ holds and i has not received human judgment in the current algorithm iteration.

Human judgment may involve promoting partial labels to a full label, or combining many sources of conflicting evidence. When making judgments, domain knowledge related to the destination and source intentions should be used. For example, the resulting label for Comfortableness in Fig. 3 is determined by human judgment. According to the propagation rules in Table 2, and given our initial labels, this softgoal receives a partially denied label from Chat and a partially satisfied label from Text. Here, using our knowledge of the domain, we decide that kids would be mostly comfortable having a text service, with their level of comfort not significantly decreased by not being able to chat, labeling the softgoal as partially satisfied. Situations such as this would be good areas for potential discussions with stakeholders involved in the modeling process.

When recording a human judgment, the judgment can be stored as a new propagation axiom reflecting the decision of the user(s). In the example above, the following axiom would be added:

$$D(\text{Chat}) \wedge \text{S}(\text{Text}) \rightarrow \text{PS}(\text{Comfortableness}). \qquad (6)$$

The utility of interactive judgments is tested with various empirical studies described in Sect. 6.

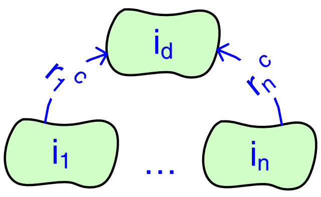
3.6 Backward analysis

In this section, we provide technical details concerning the backward analysis procedure. When asking an "Are the goals achievable?" question, we essentially wish to constrain the model using both our target analysis labels and the semantics of label propagation, as described by our propagation rules in Sect. 3.5.1. Although it is possible to use the forward propagation axioms as constraints for backward analysis, use of only these axioms makes it difficult to find derived label targets, i.e., labels which are indirectly required to achieve target labels. For this reason, and for ease of understanding, we explicitly encode backward axioms for all types of relationships. Such formalization and implementation choices are further discussed in Sect. 5.4.

3.6.1 Backward propagation rules

Dependency, decomposition, and means-ends links. Backward propagation rules for dependency, decomposition, and means-ends links are identical to the forward, but are written in the opposite implication direction. For example, in Fig. 3, for Help be acquired to be satisfied, in the forward direction, Chat and/or Text must be satisfied, $S(\text{Chat} \vee S(\text{Text}) \rightarrow S(\text{Help be acquired})$. The backward axiom expresses the other direction: $S(\text{Help be acquired}) \rightarrow S(\text{Chat} \vee S(\text{Text})$. We can express the general form for backward propagation of satisfaction for means-ends links with n sources and destination i_d as $S(i_d) \rightarrow (\bigvee_{j=1}^{n} S(i_j))$. Backward axioms for other evaluation labels and relationships can be derived by reversing the direction of the implication ($\rightarrow$ to $\leftarrow$) for each rule in Table 1.

Table 4 Backward contribution propagation axioms

Backward contribution	$V(i_d)$	$V(i_d) \rightarrow V(i_1)\ldots V(i_n)$
	S, PS	$PS(i_d) \rightarrow (\text{for } r_j^c \in \text{ Pos }, \bigvee_{j=1}^{n} PS(i_j) \vee \text{ for } r_j^c \in \text{ Neg }, \bigvee_{j=1}^{n} PD(i_j))$
	C	$C(i_d) \rightarrow \big(\bigvee_{j=1}^{n} C(i_j) \vee (\text{for } r_j^c \in \text{ Pos }, \bigvee_{j=1}^{n} PS(i_j) \wedge \text{ for } r_j^c \in \text{ Neg }, \bigvee_{j=1}^{n} PS(i_j))$
		$\vee(\text{for } r_j^c \in \text{ Pos }, \bigvee_{j=1}^{n} PD(i_j) \wedge \text{ for } r_j^c \in \text{ Neg }, \bigvee_{j=1}^{n} PD(i_j))$
		$\vee(\text{for } r_j^c \in \text{ Pos }, \bigvee_{j=1}^{n} PS(i_j) \wedge \text{ for } r_j^c \in \text{ Pos }, \bigvee_{j=1}^{n} PD(i_j))$
		$\vee(\text{for } r_j^c \in \text{ Neg }, \bigvee_{j=1}^{n} PS(i_j) \wedge \text{ for } r_j^c \in \text{ Neg }, \bigvee_{j=1}^{n} PD(i_j))\big)$
	D, PD	$PD(i_d) \rightarrow (\text{ for } r_j^c \in \text{ Pos }, \bigvee_{j=1}^{n} PD(i_j) \vee \text{ for } r_j^c \in \text{ Neg }, \bigvee_{j=1}^{n} PS(i_j))$
	U	if $r_j^c \mapsto \mathcal{R} \setminus R^u$, for $j = 1\ldots n$, $U(i_d) \rightarrow \bigvee_{j=1}^{n} U(i_j)$

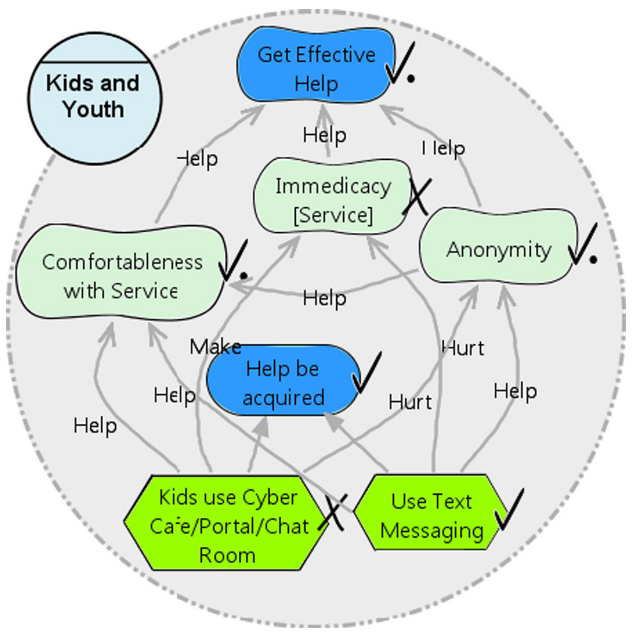


Fig. 3 Kids and Youth actor showing forward and backward evaluation results for using text messaging

Contribution links. In the backward direction, when an intention, i, is the recipient of multiple contribution links (there exists an $r_1 \ldots r_n \in R$ such that $r^c_1 : i_1 \to i \ldots r^c_n : i_n \to i$), the destination label for i, $v(i_d)$ is used to place constraints on the labels of one or more sources, $v_j(i_j) \in \mathcal{V}$, for j from $1 \ldots n$. For example, if $PS(i_d)$, we assume that at least one of the incoming labels is PS, meaning that one of the positive links propagates at least a PS label (i.e., $\exists j,\ r_j \in$ Pos, such that $v_j(i_j) \mapsto PS$) or one of the negative links propagates at least a PD label (i.e., $\exists k,\ r_k \in$ Neg, such that $v_k(i_k) \mapsto PD$).

Further backward axioms make similar assumptions. We list backward contribution propagation rules in Table 4, using our partial ordering (Eq. 2) to simplify axioms.

When analyzing the model in the backward direction, in addition to finding labels through backward propagation, we wish to consider the consequences of the analysis predicates which hold as part of the suggested solution. In other words, we want to consider the forward consequences of the labels in the solution. For example, given our backward constraint over Get Effective Help, the solver may pick a solution where Comfortableness with Service is partially satisfied and where Immediacy Service is denied. This satisfies our constraint that at least one of the contributing softgoals is partially satisfied. However, the denial of Immediacy should be factored back into the analysis results for Get Effective Help. In this case, this intention is both partially satisfied (assigned by the user as a target) and partially denied (via Immediacy). To account for such consequences, our backward analysis algorithm makes use of both forward and backward contribution axioms. Backward to find a possible set of analysis predicates which satisfies target labels, and forward to understand the consequences of such possible choices.

3.6.2 Human judgment in backward analysis

As a result of using both backward and forward propagation rules as part of backward analysis, just as in forward analysis, it is possible that a softgoal may be the recipient of more than one analysis label.

Backward analysis requires human judgment under the same conditions as forward analysis (Sect. 3.5.2), when a softgoal is the recipient of conflicting or partial, unresolved evidence. In the backward case, given a target label for a softgoal, $v(i)$, the user must provide a set of possible labels for softgoals which contribute to this softgoal. Specifically, the user is asked the following:

Results indicate that i must have a label of $v(i)$. Enter a combination of evaluation labels for intentions contributing to i which would result in $v(i)$ for i:

$(\forall j, j = 1 \ldots n, r_j : i_j \to i)$

I_j, r^c_j, (choose S, PS, U, C, PD, D, or Don't care)

…

For example, for a target of partially satisfied for Get Effective Help in Fig. 3, the user would be asked to provide a set of potential labels for incoming softgoals, specifically, users are asked:

Results indicate that Get Effective Help, must have a label of partially satisfied. Enter a combination of evaluation labels for intentions contributing to Get Effective Help which would result in partially satisfied for Get Effective Help:

Contributing intention	Link type	Select label
Comfortableness	Help	<selection>
Immediacy	Help	<selection>
Anonymous	Help	<selection>

In this case, the user would like for all three of these goals to be at least partially satisfied. The user also has the option to select "Don't care" instead of a specific analysis label, indicating that a softgoal may have any label, i.e., its contribution is insignificant in light of the other contributions.

When recording a human judgment, the judgment can be stored as a new propagation axiom reflecting the decision of the user(s). In the example above, the following axiom would be added:

$$PS(\text{GetEffectiveHelp}) \rightarrow \text{PS}(\text{Comfortableness}) \wedge PS(\text{Immediacy}) \wedge \text{PS}(\text{Anonymity}). \quad (7)$$

3.6.3 *Understanding contradictions*

Applying backward analysis described thus far allows users to ask "are certain goals achievable? if so, how?" In this section, we describe mechanisms for answering "If not, why not?" when a solution which achieves desired targets cannot be found.

When the backward procedure is unable to find a solution, there is a contradiction (e.g., PS(Chat) and $\neg PS$(Chat)), as described in Sect. 3.3. Our implementation uses existing tools to find intentions which are involved in a contradiction (see Sect. 5.3.4 for details). We can differentiate between intentions on the path involved in the contradiction, and intentions which are the "source" for the contradiction, in our example, Chat. When a contradiction occurs as part of backward analysis (no solution is found), we show intentions involved in the contradiction in orange (medium gray), and logical sources of the contradiction in red (dark gray). Additional text describes assigned analysis labels producing the contradiction. An example contradiction for our Kids model subset is shown in Fig. 4.

When a contradiction is found, the user has the opportunity to backtrack through previous judgments, entering more possibilities which are feasible in the domain, if any.

3.7 Analysis examples

In this section, we illustrate the semiautomated version of both forward, "what if?" and backward, "are certain goals achievable?" analysis using our motivating example. We illustrate both approaches to analysis over the contents of the Kids and Youth actor in Fig. 3, using a subset of the original example in order to reduce details.

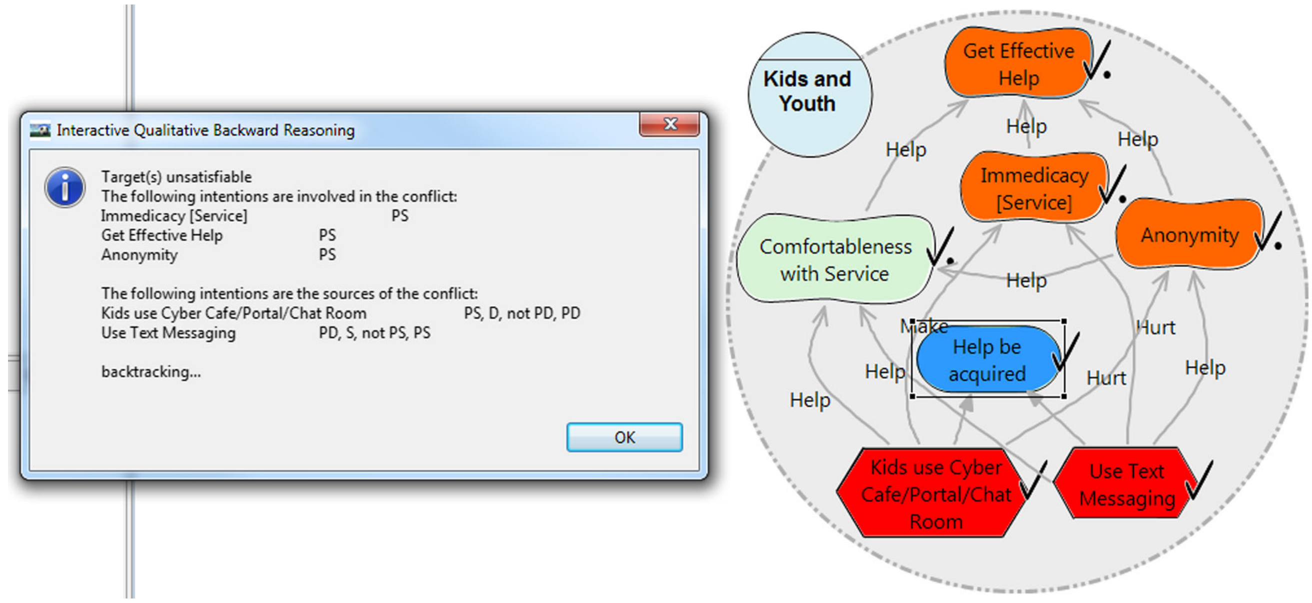
3.7.1 *Forward analysis example*

From a "what if?" perspective, we would like to explore the effects of choosing different combinations of the two task alternatives: Chat and Text. For example, if we were to start with exploring the effects of implementing Text and not Chat, we would place initial labels of satisfied and denied, respectively. When initiating the algorithm for such analysis questions, initial labels would be propagated, iteratively, through links, stopping to collect human judgment when necessary. We illustrate the iterative steps as follows.

Iteration 1. Initial labels are propagated through the first set of links, with Text and Chat directly as sources. Comfortableness (with Service) receives incoming labels of partially denied, partially satisfied, and requires human judgment, Immediacy receives labels of denied and partially denied, resulting in an automatic label of denied, Help be acquired is satisfied via the satisfaction of one means-ends alternative, and Anonymity receives labels of partially satisfied and partially satisfied, also requiring human judgment.

The judgment questions are posed to the user, for example:

Comfortable in Kids and Youth has received the following labels. Please select a resulting label.



```
Target(s) unsatisfiable
The following intentions are involved in the conflict:
Immediacy [Service]                    PS
Get Effective Help             PS
Anonymity                      PS

The following intentions are the sources of the conflict:
Kids use Cyber Cafe/Portal/Chat Room          PS, D, not PD, PD
Use Text Messaging             PD, S, not PS, PS

backtracking...
```

Fig. 4 Example contradiction in the backward analysis run for the kids subset model

Partially Denied from Use Chat Room
Partially Satisfied from Use Text Messaging
< selection from list of possible labels >

In which case, the user decides that Comfortableness has a conflict label. A similar question is posed for Anonymity, receiving partially satisfied from both Chat and Text. In this case, the user judges the softgoal to be partially satisfied, deciding not to promote multiple partial satisfaction labels to the full label.

Iteration 2. The algorithm propagates through the next set of links, using provided judgments as part of the set of labels to be propagated. Judgment is again required for Comfortableness, now having an additional input of partially satisfied from Anonymity. In this case, the softgoal is judged to be partially satisfied. Get Effective Help has incoming labels of partially denied from Immediacy, partially satisfied from Anonymity and Conflict from Comfortableness (the label being propagated is still the label from the previous iteration). The user decides this softgoal has a conflicting evaluation label.

Iteration 3. The algorithm propagates the label of the first judgment collected in the last round, partially satisfied for Comfortableness, and re-asks judgment for Get Effective Help. This time there are incoming labels of partially satisfied and partially denied (as before) and now partially satisfied from Comfortableness. In this case, with the new partial positive evidence, the user decides that Get Effective Help is partially satisfied. All labels have now been propagated and the procedure ends.

In this run of the analysis procedure, we have asked "What if Text is implemented and Chat is not?" Result show us that Immediacy would not be satisfied, while Comfortableness and Anonymity would be partially satisfied, resulting in a judgment of partial satisfaction for Get Effective Help. Although this selection requires some trade-offs among identified goals, it may be a viable alternative. Final results over our model subset are shown in Fig. 3.

We can perform forward analysis over the entire counseling model in a similar manner. Example results evaluating the Chat alternative over the entire model are shown in Fig. 5.

3.7.2 Backward analysis example

In the forward direction, we have found a solution which achieves key goals in our model. However, we would like to know if there are others. Looking at Kids and Youth, we would like to find a solution, if possible, which achieves the actor root goals: Get Effective Help and Help be Acquired. As Get Effective Help is a softgoal as part of a highly interconnected model, we set its target to partially satisfied, while the target of Help be Acquired is set to satisfied. The backward algorithm makes several interactive iterations over this analysis question, as described in the following.

Iteration 1. The algorithm tries to find a satisfying assignment of analysis labels given our targets. One is found; however, there are intentions which require human judgment. Judgment is gathered for the intention(s) closest to the root(s), the user is prompted for judgment for Get Effective Help, asking the question as specified in Sect. 3.6.2. As before, the user would like for all three of these goals to be at least partially satisfied. The judgment is encoded and the algorithm again tries to find a satisfying assignment of analysis labels, considering the new judgment.

Iteration 2. The algorithm again finds a satisfying assignment. This time there are three intentions equidistant to the root which require human judgment: Immediacy, Anonymous, and Comfortableness. Analysis questions are posed to the user using the wording and structure as presented. Immediacy has a target of partially satisfied, with contributing make and hurt contributions for Chat and Text, respectively. As such, the user chooses satisfied for Chat and denied for Text.

Anonymity has a target of partially satisfied, with contributing hurt and help contributions for Chat and Text, respectively. Given this local information, the user chooses labels of denied for Chat and satisfied for Text. As these two questions are posed simultaneously, the user may be aware of the contradiction in his/her choices and chose to force a backtrack to previous judgments. However, as such conflicts may not be obvious, or as users may not have experience in goal models or analysis, we continue the example selecting labels from a local perspective. In the third judgment, Comfortableness has a target of partially satisfied, with three intentions contributing via help links. The user selects partially satisfied for Anonymity and satisfied for both Chat and Text. The procedure again tries to find a solution, given the latest round of human judgments.

Iteration 3. A conflict is found, specifically, Text and Chat have labels of both satisfied and denied (see Fig. 4). The procedure backtracks through the last set of human judgments. Given the target labels for Immediacy and Anonymity, the user sticks with her judgments, indicating that she has no more possible combinations for these targets. For Comfortableness, there may be more possible combinations of labels; however, entering such labels will not solve the current conflict, so the user skips this judgment. The algorithm backtracks up to the last set of judgments, specifically the judgment for Get Effective Help.

Returning to this judgment, it is now clear that Immediacy and Anonymity cannot be achieved simultaneously.

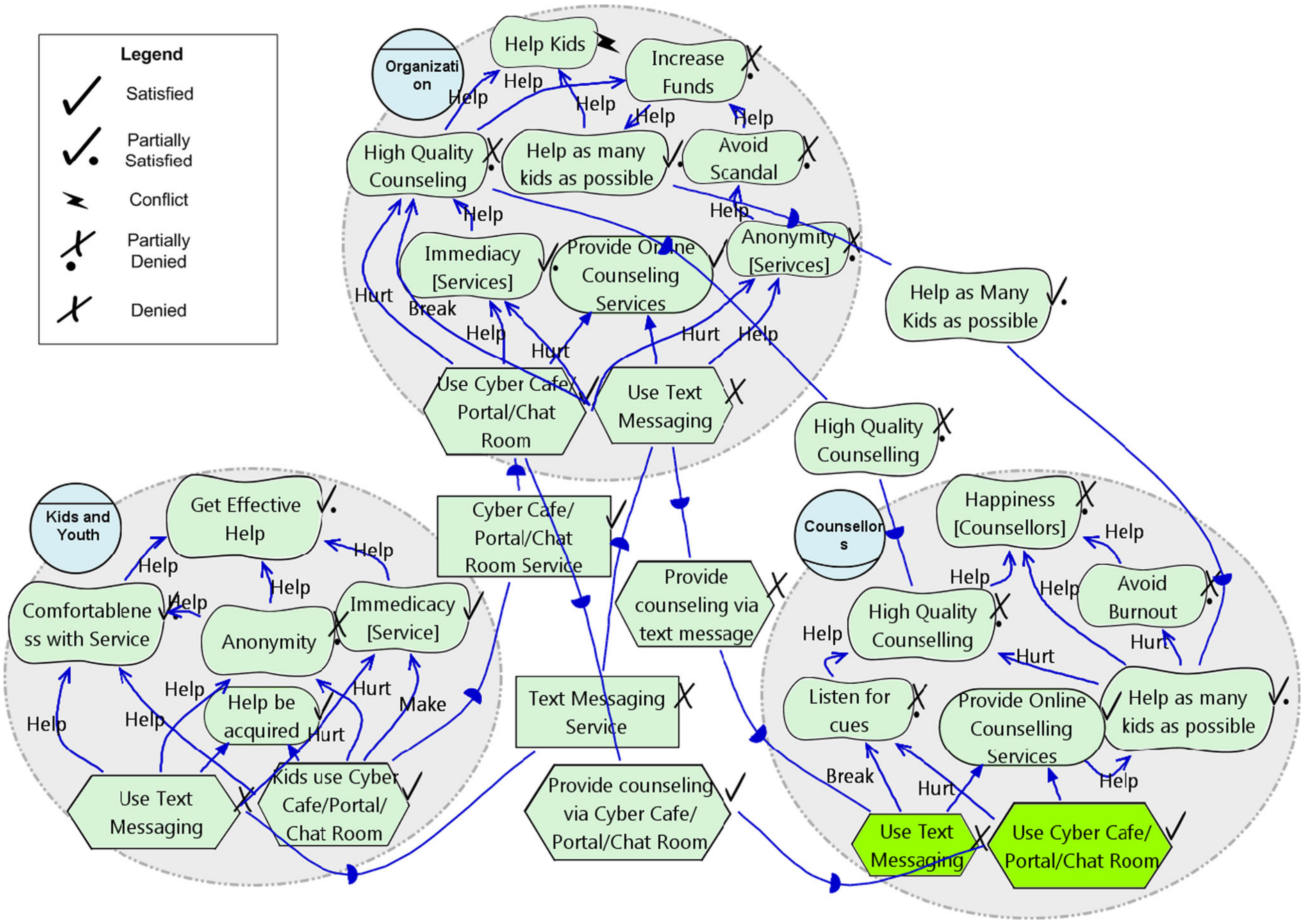


Fig. 5 Model for youth counseling showing evaluation results for using a Cyber Café/Portal/Chat Room (initial forward labels in *green/darker gray*) (adapted from [28, 30])

The user will have to either make a trade-off between these softgoals or look for further alternatives. In this case, the user judges that Anonymity is more important for Kids and Youth than Immediacy, as kids would be reluctant to use a service that reveals their identity even in urgent cases. Thus, the new judgment for Get Effective Help asks for partially satisfied for both Anonymity and Comfortableness, but provides no constraints over Immediacy, selecting “don’t care.”

Iteration 4. The algorithm finds a satisfying assignment, with Anonymity and Comfortableness requiring human judgment (as the algorithm has backtracked, previous judgments over these nodes are discarded). The user enters the same judgment as previous for Anonymity (Text satisfied, Chat denied). For Comfortableness, given a target label of partially satisfied and three incoming help links, the user may be able to live without one or the other of Text and Chat. In this case, the user enters a combination of partially satisfied for Anonymous, denied for Chat, and satisfied for Text.

Iteration 5. The new judgments are encoded, this time the procedure finds a satisfying assignment of labels which do not require human judgment. Final results (see Fig. 3) show it is possible to partially achieve Get Effective Help and provide help using the Text option and not the Chat, making a trade-off between Anonymity and Immediacy, and lowering requirements for Comfortableness. Results are shown in Fig. 3.

As with forward analysis, we can pose backward analysis questions over the entire counseling model. However, as this model is highly interconnected with many trade-offs, we are unable to find an assignment of leaf labels (solution) which sufficiently satisfies key goals without a conflict. Specifically, the hurt link from Help as many kids as possible to High Quality Counseling in Counselors makes it impossible for either Happiness [Counselors] in Counselors and Help Kids in Organization to be at least partially satisfied.

4 Modeling and analysis usage methodology

In order to facilitate the use of agent-goal models for early RE analysis, we provide a set of guidelines for elicitation and scoping, model creation, iteration, and analysis. Case study experience has led us to believe that a highly specific methodology for creating and analyzing agent-goal models may be too restrictive, due to a high variance in application domains and available modelers. We advocate this methodology as only a general guide, or a series of suggestions. Although the suggested methodology is described in many steps in sequence, the method is meant to be iterative and flexible. If the methodology is followed without the direct participation of stakeholders, each stage may result in questions which should be answered by domain experts. This knowledge should be incorporated back into the model at every stage.

The methodology is divided into three parts: purpose and elicitation, model creation, and analysis. Ideally, this approach would be applied in cooperation with domain representatives. This allows representatives to have a sense of ownership over the model and the decisions made as a result of the modeling process, as described in [47]. However, it may be difficult to acquire stakeholder buy-in to the modeling process, and in these cases, analysts can undertake the modeling process using other sources, including interviews, documents and observations.

Earlier versions of the model creation section of the methodology were presented in [28, 30], while an initial version of the suggested analysis steps appeared in [35]. Here, the description is combined and summarized, adding illustrations using the counseling service example. We summarize our methodology in Fig. 6.

4.1 Stage 1: Purpose and elicitation

Identify scope and purpose of the modeling process. In the social service example, the purpose of the first phase of the study was to identify and evaluate the effectiveness of various technical alternatives for online youth counseling. As such, the models focused on the organization's use of technology interfacing with the internet, and on those individuals in the organization who used or directed such systems.

Identify modeling participants and/or model sources. In the example, stakeholders were generally unfamiliar with modeling as a tool for analysis and had difficulty committing significant amounts of time. As a result, models were developed by the analysts using stakeholder interviews and information gained through site visits. Snippets of the models, or tabular information derived from the models, were presented back to the stakeholders for verification and discussion.

4.2 Stage 2: Model creation

We can divide model creation into five iterative steps as shown in the middle of Fig. 6. A subset of the actors, dependencies, intentions, and relationships identified in the case study have been shown in Fig. 1.

4.3 Stage 3: Analysis

Apply the evaluation procedures introduced in Sect. 3 to the model. The first two sections of each analysis type are meant to act as "sanity checks" in the model, checking that it produced sensible answers for a variety of questions, while the last section is intended to support analysis of questions from the domain.

4.3.1 Alternative effects (forward analysis)

Forward analysis begins by *identifying leaf intentions in the model.*

Implement as much as possible. As a baseline analysis alternative, analyze the effects of choosing (satisfying) as many leaf intentions as possible. Such a baseline helps to provide comparable results for additional analysis alternatives. In the counseling service model, this would equate to satisfying both alternatives, Text and Chat. In this case, many of the model elements are at least partially denied.

Implement as little as possible. As an additional baseline analysis alternative, analyze the effects of not choosing (denying) as many intentions as possible. In the counseling service model, this would equate to denying both alternatives, Text and Chat. In the model, this baseline is more positive than implementing both alternatives, giving an indication that the modeled alternatives are not viable.

Reasonable analysis alternatives. Select analysis alternatives which seem likely or promising in the domain. In the counseling service model, reasonable alternatives are to implement one or the other or both of Text and Chat. We

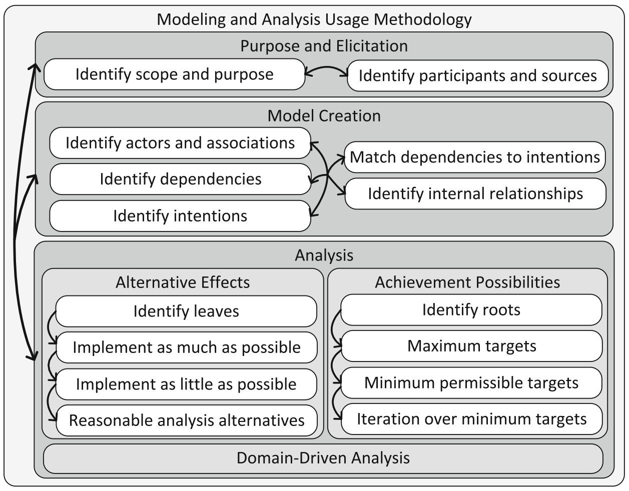

Modeling and Analysis Usage Methodology
Purpose and Elicitation
Identify scope and purpose
Identify participants and sources
Model Creation
Identify actors and associations
Identify dependencies
Identify intentions
Match dependencies to intentions
Identify internal relationships
Analysis
Alternative Effects
Identify leaves
Implement as much as possible
Implement as little as possible
Reasonable analysis alternatives
Achievement Possibilities
Identify roots
Maximum targets
Minimum permissible targets
Iteration over minimum targets
Domain-Driven Analysis

Fig. 6 Visual summary of suggested modeling and analysis usage methodology

have explored one of these scenarios in Sect. 3, we may continue by exploring the other two.

4.3.2 Achievement possibilities (backward analysis)

Backward analysis begins by *identifying root intentions in the model*.

Maximum targets. Assign target levels of satisfaction to the top intentions in the model which reflect the maximum desired level of satisfaction. Typically, this will involve all top intentions being fully satisfied; however, this can be relaxed if it is already known that full satisfaction is not possible for all top goals. In Fig. 1, the modeler may start by assigning each of the four root intentions a label of fully satisfied. Currently, this set of targets is not achievable in the model; thus, targets must be gradually relaxed.

Minimum permissible targets. Assign target levels of satisfaction or denial to root intentions in the model which reflect the minimum level of satisfaction/denial that may be permissible. What is the modeler willing to give up? What must be (at least partially) satisfied? If an intention does not have to be at least partially satisfied, no target label should be placed. Note that there may be more than one combination of minimum targets, i.e., if the modeler gives up one intention, we must have another intention instead. Minimal targets for Fig. 1 are partially satisfied for the root softgoals and fully satisfied for the two hard goals. As this particular model is strongly connected with many softgoals, even this minimum target is not achievable via backward analysis. Minimal targets were achievable over the model subset in our Sect. 3.7 example.

Iteration over minimum targets. The previous step has identified a minimum level of satisfaction for target intentions. If an alternative which achieved this minimum target was found, try gradually increasing the satisfaction level of top goals, each time checking feasibility within the model. As the previous minimum target was not achievable, we skip this step in our example model.

4.3.3 Domain-driven analysis

Once initial baseline analysis questions have been asked over the model, we can use the model to answer other relevant domain questions. For example,

- Which design options are the most viable?
- Will a particular option work? For whom?
- Will the goals of a certain stakeholder be satisfied?
- Will a particular goal be satisfied?
- Can a set of particular goals be satisfied at the same time?

In the example model, questions could include the following:

- Which of the two alternatives (Text and Chat) is more viable? Why?
- Why is it not possible to achieve minimum target labels in backward analysis?
- As the model does not contain viable alternatives, ask:
  - Is the model missing an important concept or relationship? Can this be added?
  - What further alternatives can be considered?

5 Implementation

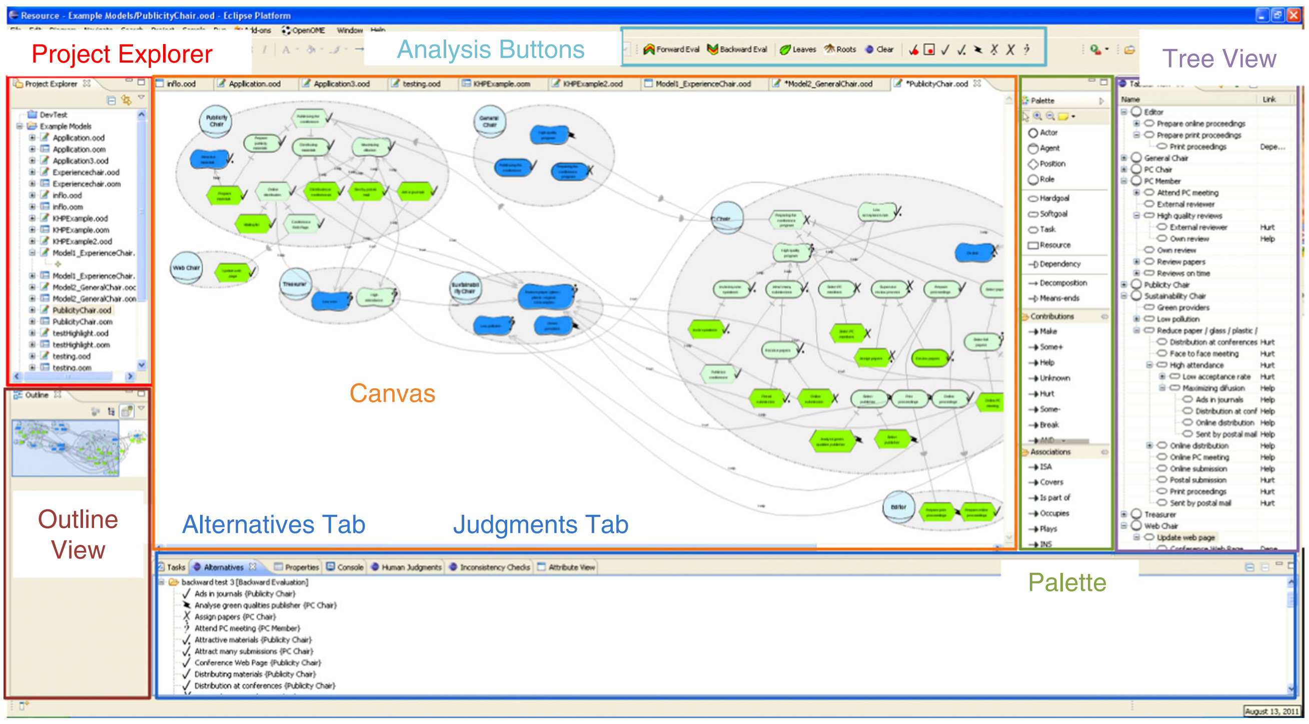
In this section, we describe the implementation of our framework in the OpenOME tool. We provide an overview of the tool, show a summary of the metamodel used for implementation, describe the implementation of forward then backward analysis, including algorithm complexity, provide details on available analysis visualization techniques included with the tool, discuss implementation choices, and report scalability results.

5.1 OpenOME Tool

The analysis framework has been implemented in OpenOME, an open-source requirements modeling tool. The tool supports modeling of the social and intentional viewpoint of a system, allowing users to capture the motivations behind system development in a graphical form. OpenOME is an eclipse-based application, making use of the eclipse and graphical modeling frameworks (EMF and GMF). OpenOME has been developed by various researchers and students, with support for forward and backward interactive analysis added after initial tool development. OpenOME supports several other analysis-related features such as the storage of analysis results and human judgments and preliminary consistency checks over human judgment, as outlined in [33]. The layout of OpenOME features can be seen in Fig. 7 screenshot. Windows, Linux, and Mac releases of OpenOME can be downloaded from Sourceforge, while documentation and tutorials are available on the OpenOME Trac Wiki page [2].

5.1.1 Framework metamodel

The metamodel used in the OpenOME tool contains the concepts and relationships needed to draw *i** models as described in Sect. 2. Additional concepts are added to support interactive analysis. A simplified view of the OpenOME metamodel is shown in Fig. 8. We examine the concepts and relationships in the metamodel by dividing it into categories: "core" *i** concepts (white), specialized *i** types (gray border), and concepts needed for interactive analysis (green/gray). Core *i** types, include the Model itself, concepts which are Dependable, i.e., can be part of a dependency link, Actors which are Containers (can contain other objects), Intentions, and Associations which are Links. We can decompose the core concepts into more specific types, for example, Intentions are specialized into Goal, Resource, Softgoal, and Task. We include classes which implement interactive analysis in green (gray), including EvaluationLabel, Alternatives, HumanJudgments, LabelBag (holding labels waiting for judgment).



backward test 3 [Backward Evaluation]
✓ Ads in journals {Publicity Chair}
Analyse green qualities publisher {PC Chair}
X Assign papers {PC Chair}
? Attend PC meeting {PC Member}
✓ Attractive materials {Publicity Chair}
✓ Attract many submissions {PC Chair}
✓ Conference Web Page {Publicity Chair}
✓ Distributing materials {Publicity Chair}
✓ Distribution at conferences {Publicity Chair}

Fig. 7 Screenshot of the OpenOME tool identifying feature layout

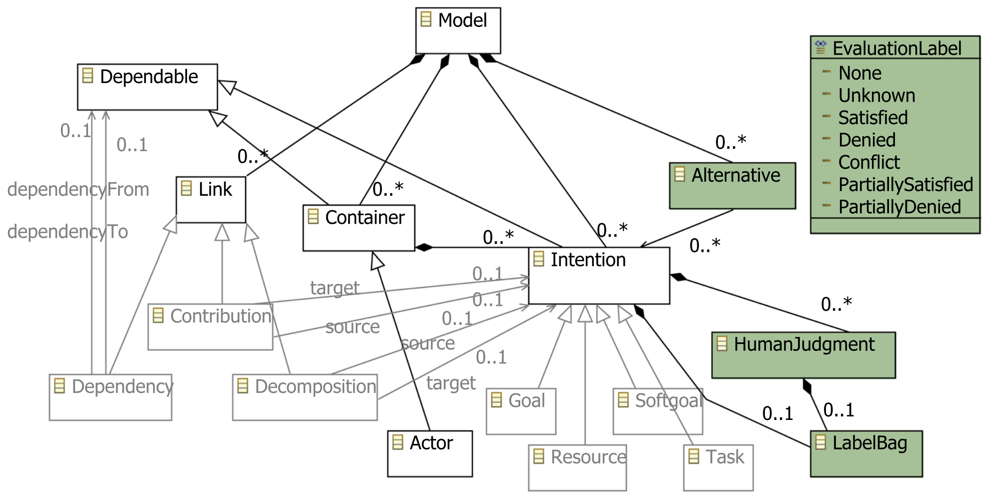


Fig. 8 Subset of the OpenOME metamodel with "Core" *i** concepts (white, black border), specializations (gray border) and concepts needed for interactive analysis (*green/gray fill)*

```
1  ForwardEvaluation($I$, $R$, $IL$) {
2    init($LQ$, $IL$);
3    while !$LQ$.empty() {
4      step1($LQ$);
5      step2($LQ$); }
6
7  init($LQ$, $IL$) {
8    queue $LQ$;
9    for (Intention $i \in IL$) {
10     $LQ$.push($i$); } } //add to queue
11
12 step1($LQ$) {
13   queue $LQ2$;
14   while !$LQ$.empty() {
15     $i_s$ = $LQ$.pop();
16     for (Relation $r \in r : i_s \rightarrow i_d$) {
17       Label $v$ = findResultLabel($r$, $i_s$, $i_d$);
18       if $i_d$.type = Softgoal {
19         //store for later judgment
20         $i_d$.storeLabel($v$); }
21       else {
22         $i_d.v$ = $v$; //set label
23         if ($i_d \notin LQ2$) {
24           $LQ2$.push($i_d$); } } } }
25   $LQ$ = $LQ2$; }
26
27 step2($LQ$) {
28   for (Intention $i \in i$.type == Softgoal) {
29     Label $v$ = automaticCases($i$);
30     if ($v$ is null) {
31       //automatic cases don't apply
32       if (<$i$, $v$> $\in HJ$)
33         //judgment already exists
34         $i$.setLabel($v$);
35       else {
36         //get new judgment
37         $i$.setLabel(promptUser($i$));
38         $HJ$.add($i$, $v$); } }
39     else {$i$.setLabel($v$);}
40     if ($i_d \notin LQ$) {
41       $LQ$.push($i_d$); } } }
42
43 findResultLabel($r$, $i_s$, $i_d$) {
44   Label $v$; //result of propagation rules
45   if $r \mapsto r^c$ { $v$ = ContRules($r$, $i_s$); }
46   else {
47     for (Relation $r \in r : (i_1, \ldots, i_n) \rightarrow i_d$) {
48       if ($r \mapsto r^{me}$)
49         { $v$ = MERule($i_1.v, \ldots, i_n.v$) where $r^{me}$: (
             $i_1, \ldots, i_n$) $\rightarrow i_d$; }
50       if ($r \mapsto r^{dec}$)
51         { $v$ = DecompRule($i_1.v, \ldots, i_n.v$) where $r^{dec}$
             : ($i_1, \ldots, i_n$) $\rightarrow i_d$; }
52       if ($r \mapsto r^{dep}$)
53         { $v$ = $i_s.v$; } } }
54   return $v$; } }
```

Algorithm 1: Forward Analysis Algorithm

5.2 Forward analysis implementation

The linear-time forward algorithm is implemented in the OpenOME tool, using Java. The forward algorithm adopts the structure outlined in the NFR procedure [10], by including iteration over two steps: propagation and label resolution. In the first step, all present labels are propagated through all outgoing links using the rules described in Sect. 3.5. In the second step, the resulting evaluation labels for softgoals are determined, using either the automatic cases in Table 3, or human judgment. Once the labels for all intentions have been determined in the second step of the algorithm, the cycle starts again. The labels to be propagated are kept track of using a queue of intentions to which the labels are assigned, `LQ`, starting with the initial labels, and adding each final label produced in step 1 and 2. The algorithm terminates when all labels have been propagated and this queue is empty.

Simplified pseudocode describing the forward analysis algorithm is shown in Algorithm 1. As our implementation is object-oriented, we use a system of objects and attributes to describe the intentions, relations, and analysis labels in the pseudocode. For example, we use $i.v$ to indicate the analysis label for an intention, i, indicating that the label is stored as an attribute of an intention (v is used to avoid $i.l$, which may be difficult to read). The type of each intention in the set intention type is referenced by an attribute, i.type. The algorithm stores a list of all the human judgments made in the HJ list.

The algorithm starts with the set of all intentions, I, relations, R, and the set of initial labels, IL (line 0). It iterates over steps 1 and 2 until the label queue is empty (lines 2-5). An init function initializes the label queue with analysis labels already in the model (initial labels) (lines 7-10). In step one, each label to propagate is removed from the label queue and the resulting propagated label is calculated (`findResultLabel`) (12-17). The algorithm uses methods `ContRules`, `MERule`, and `DecompRule`, referring to the propagation rules described in Sect. 3 (43-54). If the label to propagate has as softgoal destination, the resulting label is stored in that intention (20). Otherwise the label is added directly to the model and the label queue (21-24). In step 2, each unresolved label bag is resolved, either using automatic cases or human judgment (`promptUser`) (27-41). The results are added to the label queue (41).

As the procedure allows the placement of initial labels, $v(i_1)\ldots v(i_n) \in V$, on non-leaf nodes, it is necessary to define how these labels are affected by subsequent propagation. In the case of hard intentions (non-softgoals), subsequent propagation overrides the initial label, as it is important for users to see whether the model contradicts initial assumptions. In the case of softgoals, initial labels are placed in the bag of labels, leaving conflicts between initial and propagated labels to human judgment. Similarly, the forward procedure assigns specific semantics to a mixture of link types; for example, an intention which is a depender intention and is the parent of a decomposition link. In this case, the min label propagated from each type of relation would be assigned. For simplicity, we omit the treatment of non-leaf initial labels and mixture of link types from Algorithm 1. More details on each can be found in [24, 30].

5.2.1 *Model cycles, termination, and computational complexity*

Goal models often contain cycles, labels which indirectly contribute to themselves. Often these situations will converge to a particular label, but in some situations they may fluctuate between labels indefinitely. To avoid this, we implement the relatively shallow solution of storing a count of each of the combinations intentions and labels that have been placed in the label queue. Once the count as reached a fixed number, r, the same combination cannot be placed in the label queue again. This solution allows for a certain number of label fluctuations for non-looping situations, but will put a cap on the number of iterations which can occur.

In our current implementation, if there are n intentions in the model, supporting a total of 6 analysis labels and a cap of r times in the label queue, the label queue has a maximum lifetime size of $6rn$, and the algorithm must terminate. The running time of the algorithm is linear, $O(n)$, where n is the size of the model.

5.3 Backward analysis implementation

The backward implementation uses a SAT solver to find satisfying assignments of labels given propagation rules as constraints. The approach encodes the model in CNF, and then iteratively runs the SAT solver, prompting the user for input regarding intentions which required human judgment after each run. When human judgment is no longer needed and a satisfying assignment is found, the procedure ends, providing an answer. If a satisfying assignment is not found, the procedure tries to backtrack over human judgments. If a satisfying assignment is not found and no further human input can be given, the procedure ends, informing the user that the target is not possible.

5.3.1 *Background: SAT*

SAT solvers are algorithms which accept a Boolean formula in CNF, composed of a conjunction of clauses. The algorithm searches for a truth assignment of the formula's clauses to make the formula true. It does so by making a series of decisions concerning the labels of variables, backtracking if a decision proves to be not viable. If a solver can find a satisfying assignment, it returns only one such assignment, saying nothing about the presence of other permissible answers. Although the SAT problem is NP-Complete, algorithms and tools that can solve many SAT problems in a reasonable amount of time have been developed, for example, the zChaff tool [43], used in this work.

5.3.2 *Expressing qualitative, interactive propagation in CNF*

To express the problem of assigning evaluation labels to an agent-goal model in terms of a CNF SAT formula, we follow the formalization in [20], adopting their classification of the components of the formula as follows:

- The target labels for the procedure, $\phi Target$
- Axioms describing forward propagation, $\phi Forward$
- Axioms describing backward propagation, $\phi Backward$
- Axioms describing invariant properties of evaluation labels, $\phi Invariant$
- Any additional constraints on propagation, $\phi Constraints$

The SAT formula is constructed as follows:

$$\begin{aligned}\phi = \phi Target \wedge \phi Forward \wedge \phi Backward \\ \wedge \phi Invariant \wedge \phi Constraints.\end{aligned} \quad (8)$$

Target. The *target* for an evaluation is simply a conjunction of the desired labels for each target intention. We could constrain the target further by saying that the target should only have that label; for example, if our target is $PS(i)$, we add $\neg C(i)$ and $\neg U(i)$ and $\neg PD(i)$, but we want to allow for targets to have conflicting labels, making them candidates for human intervention.

Invariant. As invariant axioms, we include the partial order in Eq. 2, more specifically we include the following axioms:

$$\begin{aligned}\forall i \in \mathcal{I} : S(i) &\rightarrow PS(i) \\ D(i) &\rightarrow PD(i)\end{aligned} \quad (9)$$

Constraints. When using the analysis procedure, the user could add any additional constraints into the SAT formula, following the approach of [20]. In our example, we constrain leaf intentions which are hard (non-softgoals) such that they must be assigned at least one of the six evaluation labels, and their assignment must not be conflicting (Definition 8). Restricting the model formalization in this way ensures that the answer provided by the SAT solver applies a single analysis label to all connected hard intentions. In our example, we would add these constraints for our two leaf intentions, Chat and Text.

5.3.3 *Restrictions on agent-goal model*

In order to produce an agent-goal model which can be more easily translated into CNF form and to ensure the convergence and termination of the algorithm, we place the following restrictions on an $i*$ model:

- *Each intention has at most one Decomposition, Dependency or Means-Ends relation which determines its level of satisfaction or denial, i.e.,* $\forall i \in I$*, only one of* $R^{dep} : I \rightarrow i$*,* $R^{dec} : I \times \cdots \times I \rightarrow i$*, or* $R^{me} : I \times \cdots \times I \rightarrow i$ *holds for i.*
- *The model must have no cycles, i.e., for every path in the model,* $r_1, \ldots, r_n \in R$*,* $r_1 : i_1(\times \cdots \times I) \rightarrow i_2$*,* $r_2 : i_2(\times \cdots \times I) \rightarrow i_3, \ldots, r_{n-1} : i_{n-1}(\times \cdots \times I) \rightarrow i_n$*,* i_k *must not equal* i_j, for $1 < i, j < n$.

The first rule means that models must avoid a mixture of hard links, i.e., the backward procedure is limited in that it does not explicitly account for a mixture of dependency links with Means-Ends or Decomposition links. In this case, the analysis predicates which are propagated through each type of link would apply simultaneously, possibly resulting in an analysis predicate conflict for non-softgoal intentions. Such cases may prevent the solver from finding a solution. We plan to expand our backward procedure with additional rules to handle these cases.

The second rules force modelers to resolve cycles. The reader may note that these restrictions apply only to backward and not forward analysis; future work should expand the backward implementation to remove these restrictions.

5.3.4 Analysis visualization techniques

It can be challenging to follow analysis through complex paths in the model. We have implemented visualization mechanisms to alleviate such difficulties [31]. Specifically, we highlight model leaves and roots as potential starting points of analysis (e.g., Fig. 4), highlight intentions involved in human judgments, and provide conflict visualization as described in Sect. 3.6.3.

To implement conflict visualization as part of backward analysis, we use a SAT solver which provides an *unsatisfiable (UNSAT) core*, a list of clauses in the CNF which result in a contradiction. These clauses can be used to form a resolution proof, showing how the clauses work together to produce a contradiction, i.e., $(a \vee \neg a)$. Finding a minimal unsat core is a computationally difficult problem, but many approaches exist for finding a small but not minimum core (for example [7]). Presenting this information to the user in a form which is understandable to users presents a challenge.

Our implementation of conflict highlighting parses intentions and analysis predicate assignment in the UNSAT core using a recursive procedure starting at the root clauses (including analysis targets) of the core, traversing toward the sources of the contradiction $(a \wedge \neg a)$. Intentions involved in the contradiction are collected along the recursion. When a contradiction occurs during backward analysis, our implementation highlights intentions involved in the contradiction (orange) and sources of the contradiction (red). Users are also presented with a list of the intentions and analysis labels that would produce the contradiction. An example is shown in Fig. 4.

5.3.5 Backward analysis algorithm

Simplified pseudocode describing the backward analysis algorithm is shown in Algorithm 2. The algorithm converts the model to CNF form (line 8), using the axioms described in Sects. 3.5 and 3.6. The algorithm loops, terminating when a solution is found and no judgments are needed (line 18), when a solution is found but judgments cannot be made (line 33), or when no solution is found and there are no judgments to backtrack over (line 41).

The algorithm calls zChaff to find a solution for the `cnf` (line 12). If a solution is found (line 11), the algorithm finds intentions needing human judgment (line 14). If none exists, the procedure ends successfully. If judgments must be resolved, the procedure finds the top (closest to a root) intentions which need human judgment (line 19). The target for each of these intentions is found by running the solver using only backward rules (line 22) and taking the maximum label result for each intention, using the ordering in Eq. 2.

The user is prompted for each top intention requiring judgment (line 26), and the judgment is added to the cnf as described in Sect. 3.3 (line 29). If the user provided judgments, the list of top intentions is added to a stack (line 35). If, in the main loop, zChaff cannot find a solution (line 36), zMinimal is used to find the UNSAT core and display conflict information (line 38-39). In this case, or when the user has no more judgments to add (line 32, 40), the algorithm backtracks, popping the last set of intentions needing human judgment from the stack (line 46) and backtracking over the `cnf` (removing the judgment axioms and adding back in the default forward and backward propagation axioms) (line 47–49). If there are no judgments in the stack for backtracking, the algorithm terminates with a negative result (line 54). Otherwise, control is returned to the main loop (line 9) where the process starts again.

As with forward analysis, the procedure allows the placement of target labels even on non-root intentions. Analysis may cause other labels for such intentions to become true, making them a potential source of conflict (for hard elements) or an area requiring human judgment (for softgoals).

```
Dimacs cnf; bcnf;
zChaffSolver solver;
zMinimalSolver minSolver;
ModeltoAxiomsConverter converter;
Stack<Intentions> hjStack;

BackwardEvaluation(I, R, IL) {
  cnf = converter.convert(I, R, IL);
  while( true ) {
    int result = solver.solve(cnf);
    if (result == 1) {
      Results rslt = solver.getResults();
      //find intentions needing judgment
      Intentions needHJ = findHJ(rslt);
      if (needHJ.size() == 0) {
        //solution, no judgments needed
        showMessage("Success!");
        return; }
      Intentions topJudgments = findtop(
          needHJ);
      bcnf = converter.convertb(I, R, IL);
      solver.solve(bcnf);
      Results bRslt = solver.getResults();
          //solve using backward rules
      int hjCount = 0;
      for (Intention i: topJudgments) {
        //get new judgment
        if (prompt(i, bRslt.get(i))) {
            hjCount++;  //count judgments
            //add new judgment to encoding
            cnf = converter.add(cnf,i); }}
      if (hjCount == 0) {
        //no more user judgments to add
        if (backtrack() == -1) {
          return; } }
      else {
        hjStack.push(topJudgments); } }
    else if (result == 0) {
      //no solution, find unsat core
      minSolver.solve(cnf);
      showMessage ("Backtracking: " +
          minSolver.getResults());
      if (backtrack() == -1) {
        return; } } } }

int backtrack() {
  if (hjStack.size() > 0) {
    //there are judgments for backtracking
    Intentions needHJ = hjStack.pop();
    for (Intention i: needHJ) {
      //backtrack over the last judgments
      cnf = converter.backtrack(cnf,i); }
    return 1; }
  else {
    //no judgments for backtracking
    showMessage(Target(s) unsatisfiable.);
    return -1; } }
```

Algorithm 2: Backward Analysis Algorithm

5.3.6 *Computational complexity and termination*

Computational Complexity. In practice, the running time of SAT approaches would be affected by the number of Means-Ends (OR) decompositions and multiple incoming contribution links, as each of these structures provide further labeling alternatives, increasing the search space. We exclude a detailed exploration of the runtime complexity of zChaff or zMinimal, marking these labels as (zChaff) and (zMinimal). The main loop in `BackwardEvaluation()` in Algorithm 2 will loop until `hjCount == 0`. In the worst case, each iteration involves a single new judgment for every intention. If a model has n intentions and the maximum number of incoming relations for each intention is q, there is a maximum of $6^q \times n$ possible judgments, where $q < n$. Although the worst case is 6^q, in practice only a small subset of these judgments will be made in each analysis run.

The complexity of the initial axiom conversion is $6r$, where r is number of relations in the model ($|R|$). The cost of adding or backtracking human judgment on the converter is also r (finding the right axiom by relation). In addition, the worst case runtime of findtop is $2n$, and backtrack is $2rn$. If zChaff returns a satisfying result, the worst case runtime is either $2rn + 3n+$ (zChaff) or $2rn$, else, when the problem is not satisfiable, it is $2rn+$ (zMinimal). Assuming (zMiminal) $\approx$ (zChaff), the worst case runtime for `BackwardEvaluation()` is $6^q n(rn^2 + 3n + (\text{zChaff})) + 6\text{r}$, or $O(6^q(rn^2 + n(\text{zChaff})))$. Although this is an exponential label, q is usually a small number, less than 5 or 6.

Termination. If the user continues to make the same judgments, the procedure will not terminate. However, the current implementation provides a list of previous judgments attempted which did not produce a solution. As there are a finite number of intentions each with a finite number of sources, there are a finite number of possible human judgments (6^q). If the user does not continually reuse judgments, the procedure terminates.

5.4 Analysis implementation choices

Unified versus separate procedures. Although the forward and backward procedures involve similar concepts and mechanisms, we have chosen to implement them using separate procedures. Backward analysis can be thought of as a type of constraint satisfaction problem, as such we express the automated part of the procedure as a Satisfiability (SAT) problem. As SAT is a well-studied problem, we use externally implemented solvers in our implementation, taking advantage of the efficient algorithms and optimizations available in solvers such as zChaff [43]. We enable human judgment by wrapping SAT calls in iterative Java code. We could use the same implementation to implement forward analysis, as the forward axioms are encoded as part of the backward procedure; however, constraint satisfiability problems are theoretically NP complete, whereas forward analysis can be implemented in a linear algorithm. Thus, we encode the forward algorithm, without using SAT, in Java.

Alternatives to SAT. In the early stages of this work, we considered encoding agent-goal model propagation as a constraint satisfaction problem (CSP) or satisfiability modulo theories (SMT) problem. However, in order to capture the presence of analysis predicate conflicts (Definition 8) and the subsequent need for human judgment, each intention would have to be assigned multiple variables, one for each analysis label, making the encoding roughly as complex as our SAT encoding. Consideration was also given to the use of an incremental SAT solver, reusing the state-space when clauses are added to the encoding. However, as our algorithm not only adds, but removes and re-adds clauses, these types of algorithms were not easily applicable. See [5] for a more detailed discussion of choices in formalizations and use of existing solvers when implementing goal model analysis.

Explicit backward axioms. When developing a procedure for backward propagation, we have several choices concerning the encoding. We could use the forward axioms in Sect. 3.5.1 as constraints, passed to the solver. These constraints, along with the target values, and the constraint that non-softgoal leaves must be assigned a non-conflicting label (Sect. 5.3.2), could be used to find a solution, if one exists – a set of analysis predicates which satisfies these constraints. Such a solution may still require human judgment, if particular intentions have conflicting labels (Definition 8). However, using only the forward propagation axioms, it would be difficult to determine derived or indirect targets, as required for human judgment. For example, in Fig. 3, all backward targets are entered directly by the user, either as an initial value or as a result of human judgment. Imagine the situation where the Anonymity softgoal connected to Get Effective Help indirectly, via an intermediate softgoal X. The target value for X would be acquired from the backward judgment for Get Effective Help, but the target value for Anonymity must be inferred automatically. If only forward axioms are used, one or more analysis predicates would hold for this intention, but it would be difficult to tell which predicate is desired as an indirect target. Thus, to find such indirect targets, we explicitly encode backward propagation, as is done in [20, 21]. Although this approach allows an explicit and more intuitive propagation in the backward direction, the additional axioms may affect performance. Future work should evaluate how this and other alternative implementation choices affect efficiency.

5.5 Performance

In this section, we analyze the computational performance of the forward and backward algorithm implementations. We test their operation on models of a variety of sizes and argue for a maximum practical model size for interactive early RE modeling.

Model size in practice. As we have argued in the Introduction, early RE models are highly qualitative, social models, and as such are difficult or impossible to generate automatically. This means that early RE models must be created by hand. Manual creation of early RE models places cognitive constraints on their size and complexity. Beyond a certain level, the models are too complicated to understand, modify, or analyze effectively.

We believe that we have hit this level of complexity manually creating large *i** models in our past case studies. The largest model created for the counseling service case study contained approximately 525 relations and 350 intentions, 230 of which represent quality criteria and system goals, the rest of which represent specific tasks in the current system [23]. Working with such a large model was cognitively difficult and impractical in practice. Only the model author was able to (with difficulty) navigate or analyze the model.

Considering this model, and other similar examples, we argue that the optimal model size for domain understanding and analysis is much smaller than the size of this model (<200 elements). The exact "optimal" size is difficult to measure precisely and depends on factors within the domain and the experience of the modelers. In fact, the bottleneck in interactive analysis is not so much the computational complexity of the procedure, but the number of human judgments asked over a model.

Scalability tests. We test the speed of the analysis implementations over several realistically sized models created as part of case studies. We run two forward and two backward analysis questions over each model, capturing running times. We differentiate between the actual computation time, and the time taken for users to read and act on various input windows, including human judgment windows and messages about conflicts in backward analysis. Tests are run on a PC with a 1.8GHz Intel(r) CoreTM Duo Processor T2400 CPU and 2.5 GB of RAM.

We select three models which we judge to be of small, medium, and large size, relative to our experiences in case studies and examples. The first model is a small model of an application, of a similar size to the counseling subset in Fig. 3. The second model captures conference greening and is partially shown in Fig. 7. The third model is the result of a group case study for the inflo modeling tool described in Sect. 6.2. We summarize model sizes in Table 5. Although the last model is smaller than our estimated "max" model size, this model is the largest we have encoded in our OpenOME tool (previous, larger models were created in Microsoft Visio before OpenOME was available).

When selecting analysis alternatives for each model, we selected a mix of initial labels describing both sanity checks and interesting domain questions. As alternatives are evaluated by the first author, timings for human judgments are not necessarily realistic or reflective of the interactive and collaborative aims of the procedure.

Tables 6 and 7 provide the timing results from the analysis runs from the forward and backward tests, respectively. Some of the backward analysis alternatives did not find viable alternatives, either the implementation reported that there was no solution (alternative 2 in Model 1) or the user gave up after several rounds of judgment (alternative 1 in Model 2 and alternative 2 in Model 3). In the latter cases, the implementation always reported an answer, but after several rounds of relaxing constraints for the required target, it became clear the targets could not be reasonably attained.

Examining the running times, we see that the computation time (the total time the user is waiting for an answer) for forward analysis is small (<4 s), even for larger models. As expected, the bottleneck in forward analysis is human judgments. In the backward analysis, computational time is longer but still manageable. Over the larger models, it can take up to 30 seconds for the tool to produce an answer. In some cases, the computation time for backward analysis exceeds the judgment time, making implementation efficiency a point of future work.

6 Evaluation

In this section, we summarize studies applying our analysis framework. As these studies were conducted as the framework was under various stages of development, they test evolving components of the framework, both using and not using systematic analysis and the OpenOME tool. We summarize the studies conducted, the framework components applied, the study designs, tool support used, hypotheses, major conclusions, and reference to further detail in Table 8 (note that studies 3, 4 and 5 share the same hypotheses). In this section, we provide a brief summary of each study, assessing the findings and threats to validity as a whole.

6.1 Studies using manual forward analysis

Initial examples. Earlier work has applied an initial version of the forward analysis described in Sect. 4 to a variety of settings, including a study of trusted computing technology ([23, 36]).

Counseling service. Manual forward analysis was applied to the counseling service study used as an example in earlier sections. This multi-year strategic analysis project underwent several stages with different areas of focus [13]. The first stage focused on modeling, analyzing, and understanding the organization as a whole, with an emphasis on the role of online counseling. The second stage of the project focused on increasing the efficiency of the existing online counseling system, while the final stage focused on analyzing the knowledge management needs of the organization.

In the first two stages, models were created based on transcripts of interviews with several roles in the organization. In the third stage, models were created on-the-fly? during stakeholder interviews. Forward analysis was applied to explore the effectiveness of options for online counseling and knowledge management. The results of the models and analysis were presented to the organization, using reports, tables, and presentation slides containing small excerpts of the model. The analysis was well-received by the organization, bringing to light several issues and provoking interesting discussion. Final outcomes included a requirements specification document and a knowledge management report. Resulting *i** models were used in several studies, exploring viewpoints [12], applying patterns [49], and modeling knowledge transfer effectiveness [48].

These studies have provided experiential evidence that such analysis increases model iteration, prompts further elicitation, and improves domain knowledge. Unfortunately, our experience concerning model iteration resulting from interactive analysis is only anecdotal for the first two stages of the study (the effects were observed, but not carefully recorded). In the third stage, we began to collect measures of such iteration. One model focusing on communication contained 181 links and 166 elements before evaluation, while after evaluation the same model had 222

Table 5 Sample agent-goal model sizes

Model	Content	Concept	Count
Model 1	Simple application	Actors	1
		Intentions	6
		Relations	7
Model 2	Conference greening	Actors	8
		Intentions	56
		Relations	74
Model 3	Inflo tool	Actors	12
		Intentions	103
		Relations	145

Table 6 Running time (seconds) and statistics for forward analysis runs

Measurements	Model 1		Model 2		Model 3	
	Alt 1	Alt 2	Alt 1	Alt 2	Alt 1	Alt 2
Num judgments in analysis	2	2	15	15	23	22
Num intentions receiving judgments	2	2	9	9	16	16
Max judgment time	4.109	4.875	5.813	6.390	19.734	15.078
Min judgment time	2.750	4.297	2.531	2.141	2.718	2.969
Average judgment time	3.429	4.586	4.328	3.930	8.048	6.296
Total judgment time	6.859	9.172	64.922	58.954	185.106	138.517
Total computation time	0.25	0.156	1.547	3.499	3.347	3.436
Total analysis time	7.109	9.328	66.469	62.453	188.453	141.953

Table 7 Running time (seconds) and statistics for backward analysis runs

Measurements	Model 1		Model 2		Model 3	
	Alt 1	Alt 2	Alt 1	Alt 2	Alt 1	Alt 2
Num judgments in analysis	5	3	4	2	1	5
Num intentions receiving judgments	2	2	1	2	1	2
Max judgment time	9.594	13.078	145.453	36.219	9.766	40.547
Min judgment time	3.047	2.062	2.032	12.813	9.766	4.438
Average judgment time	7.187	25.906	55.523	24.516	9.766	18.162
Total judgment time	35.937	8.635	222.094	49.032	9.766	90.814
Num non-judgment messages	2	2	4	1	1	4
Total time for non-judgment messages	4.796	9.077	72.220	2.265	3.437	49.984
Total computation time	0.579	17.616	30.905	1.047	2.391	150.765
Total analysis time	41.312	35.328	325.219	52.344	15.594	291.563

links and 178 elements, a difference of 41 and 12, respectively. In another model, the link count rose from 59 to 96 and the element count rose from 59 to 76. These numbers do not take into account changes such as moving links or changing element names. Models in this stage of the study were created by three individuals, with evaluation performed by two individuals, indicating that this effect is not specific to a particular modeler or evaluator.

Exploratory experiment. Based on experience applying forward analysis in practice, a small exploratory experiment was conducted in order to more precisely test the perceived benefits of the forward procedure, summarized by hypotheses H1-H4 in column 6 (row 3) of Table 8 (more details available in [24, 30]). Results did not provide strong evidence to support claimed benefits, showing that benefits, when they occur, can occur both with systematic and ad hoc model analysis. The last two hypothesis, concerning elicitation and domain knowledge proved to be difficult to test empirically. Although we believe that the interactive, iterative procedures designed in this work will have a positive effect on prompting elicitation and increasing domain knowledge; future studies focused on more measurable effects, such as increasing model completeness and accuracy.

6.2 Studies using forward and backward analysis in OpenOME

Motivated by our practical and experimental experiences, individual and group case studies were designed and administered to further test the hypothesized benefits of interactive analysis (H1-H4). Study design aimed to find a balance between the rich (but difficult to measure) experiences of our industrial study and the controlled (but somewhat artificial) environment of our experiment.

Individual case studies. The studies were administered in two rounds, using at total of 10 participants (students with some *i** experience). In both rounds, half of the subjects used the systematic analysis procedure in OpenOME while the other half answered questions using ad hoc analysis (over models in OpenOME). The subjects using systematic *i** analysis received an additional round of training for the forward and backward procedures (15 minutes). The study involved a think-aloud? protocol, with the first author present to observe the progress and answer questions. Participants were encouraged to ask questions about the model whether they had them. Every participant was asked a series of follow-up questions concerning their

Table 8 Summary of studies evaluating components of the analysis framework

Study name	Study domain	Type of analysis used	Study design	Tool support	Hypotheses	Major conclusions	References
Initial examples	Several: e.g., trusted computing	Forward manual analysis	Exploratory studies with a few modelers, no industrial feedback	OME	Forward analysis may be useful to explore alternatives	Forward analysis helped decision making, exploration, improved model	[23, 24, 36]
Counseling service	Real world not-for-profit counseling service	Forward manual analysis	Action research/exploratory case study, deliverables and feedback from counseling service	Microsoft Visio, no built-in analysis	Forward analysis may be useful to explore alternatives	Forward analysis helped decision making, exploration, improved model	[12, 48, 49]
Exploratory experiment	Conference greening	Forward manual analysis	Small experiment with 5 academic participants, analysis with and without systematic procedure	Microsoft Visio, no built-in analysis	H1: Analysis: aids in finding non-obvious answers to domain analysis questions	Benefits of analysis occur both with ad hoc and systematic analysis	[30, 31] models described in [9]
Individual case studies	Conference greening and university experiences	Semiautomated forward and backward analysis	Exploratory case study with 10 participants, performing ad hoc or systematic forward/backward analysis	OpenOME, forward and backward interactive analysis	H2: Model Iteration: prompts improvements in the model H3: Elicitation: leads to further elicitation of information in the domain	Some analysis success and difficulties, little model iteration or elicitation, some increase in domain knowledge. Little differences between ad hoc and systematic analysis. Several additional useful findings beyond hypotheses	[24, 35]
Group case study	Requirements for Inflo modeling tool	Semiautomated forward and backward analysis	Group case study with four academic participants cooperatively building and analyzing a model	OpenOME, forward and backward interactive analysis	H4: Domain Knowledge: leads to a better understanding of the domain	Lacking driving domain questions, revealed model issues, slight model improvements, improved domain knowledge	[24, 35]
Follow-up visualization studies	Conference greening, university, and inflo	Semiautomated analysis with visualizations	Repeat studies with 5 individuals from individual and group case studies, focus on visualization tasks	OpenOME with analysis visualizations	Do visualizations aid analysis comprehensibility?	Helped select initial labels, prompt some model changes, improved conflict understanding, but still some difficulties	[31]

experience. The total time for each study in both rounds was two hours or less.

The first round, involving 6 participants, used models from the conference greening domain, reducing the environmental footprint of the conference. The three models contained between 36 and 79 intentions, 50 and 130 links, and 5 and 15 actors. Analysis questions were aimed to represent interesting questions over the domain. For example "If every task of the Sustainability Chair and Local Chair is performed, will goals related to sustainability be sufficiently satisfied?"

The results of the first round of the study performed with six participants showed minimal model changes or elicitation questions, as well as participant difficulties in understanding the models, due to their large size and the participants unfamiliarity with the domain. The decision was made to revise the study and instead allow participants to make their own models over a domain they were familiar with—student life. In the second round, the four participants were provided with some leading questions (e.g., Who is involved? What do the actors want to achieve?), then spent 25 min creating smaller models describing their student experiences. Initial results motivated the development of the suggested modeling and analysis methodology described in Sect. 4. As such, Round 2 participants were asked to use this methodology to analyze their student life model.

Results. Quantitative and qualitative data (audio, video, models, observer notes) were collected and coded for both rounds of the study. Results for hypothesis H1 (Analysis Results) were mixed, some participants gave explicit answers to the questions, some referred to analysis labels in the model as answers to the question, while yet others had difficultly producing answers to the questions. Only some participants were able to interpret question results in the context of the domain. Similarly, participants often had difficulty in translating questions into initial labels in the model. Difficulties were experienced both with and without systematic analysis.

Regarding H2 (Model Iteration), participants made only a few changes to the models when conducting analysis. There were slightly more changes made with ad hoc than systematic analysis, and there is no notable difference between participants analyzing their own or other's models. We also see no significant differences between results given and not given the suggested modeling and analysis methodology. Results for H3 (Elicitation) showed that participants asked very few domain-related questions, with no interesting differences between groups. Seven out of ten participants indicated they had a better understanding of the domain after the study (H3). In this case, analysis was helpful using both systematic and ad hoc approaches.

In addition to findings relating to our initial hypotheses, our qualitative analysis produced other findings revealing potential benefits of interactive analysis. Specifically, results showed that systematic analysis increased the consistency of model interpretation by providing a precise semantics, increased the coverage of analysis across the model, and helped to reveal model incompleteness. Study results provide evidence that modelers made inconsistent human judgments, e.g., giving an intention a fully satisfied label when the incoming evidence was one partially satisfied label and one partially denied label. We outline future work which may warn users against such inconsistencies in Sect. 8.3.

Group case study. A second study was conducted involving a group of four graduate students and a professor who were in the process of designing and implementing a tool (Inflo) to support modeling and discussion of "back of the envelope" calculations. Three two-hour modeling and analysis sessions were devoted to constructing and discussing a large $i*$ model representing the tool, its users, and their goals. During each session, time was devoted to applying both the forward and backward analysis procedures, letting the participants make decisions over the human judgments posed by the procedures. The first author/facilitator played a participatory role, drawing the model and administering the analysis with constant feedback and input from the participants. The final model is used in the procedure scalability tests in Sect. 5.5.

Results. In the Inflo case, the modelers did not have any driving domain questions, as the purpose of their participation was to better understand the system under development, not to solve problems which were not yet apparent; therefore, the analysis questions asked were somewhat artificial (H1). Some analysis alternatives did help to find sanity issues in the model; for example, if the inflo system was built, the trolls (malicious users) win, according to the model. Analysis did prompt some changes in the inflo case, for example, removing links, but the changes were not extensive (H2). In this case, the modelers and the stakeholders were the same, so any questions raised by the modeling or analysis process were discussed and resolved immediately (H3). Feedback through surveys for the inflo group revealed that analysis helped clarify tradeoffs, and the meanings of intentions (H4), although several usability issues with the procedure were found, several of which were addressed by further rounds of implementation.

As with the individual studies, analysis of results for the group study reveals analysis benefits beyond the initial hypotheses. Application of systematic evaluation in a group setting did produce several situations where human judgment caused discussion among participants. For example, the participants discussed whether getting feedback was really necessary in order to make models trustworthy after this contribution appeared in a backward judgment situation. In other examples, the group had discussions about the exact meaning of goals appearing in judgments situations, for example "what is meant by flexibility?" In the study, the participants felt that analysis was not useful until the model reached a sufficient level of completeness. This was echoed by one participant in the individual studies. Future work should investigate the qualities of a model that make it sufficiently complete for analysis.

Follow-up visualization studies. In order to test the practical utility of the visualizations described in Sect. 5.3.4, we performed five follow-up studies using participants from the initial eleven studies described in the previous section. Each session lasted 30 minutes to an hour. Participants were specifically asked to comment on the new interventions: Do the leaves/roots highlighted in the model make sense? Can you understand why there is a conflict?

Reaction to root and leaf highlighting was positive, with participants understanding the results of the automatic highlighting. Once leaves and roots were identified by the application, participants had an easier time selecting initial labels for analysis when compared to the previous study rounds. In the Inflo case, when leaves or roots were identified, this prompted changes, adding more incoming contributions to some sparsely connected roots, producing richer, more complete results over the model.

Results concerning conflict highlighting show that this intervention is helpful in understanding model conflicts; however, a considerable knowledge of *i** modeling and analysis is needed to completely understand the causes of the conflict. Despite the need for *i** knowledge, highlighting of conflict intentions made it much easier for the facilitator to understand and explain conflicts in the model, and all participants indicated that conflict highlighting was helpful.

6.3 Threats to validity

We summarize several threats to the validity of our studies. In our individual and group studies, we collected several measures to test our hypotheses (analysis results, model changes, questions raised). It is difficult to know whether these are effective measures of our respective hypotheses, for example, is increased understanding due to analysis or only modeling? Would participants be able to use analysis results to draw conclusions in the domain? Although we have measured model changes in several studies, it is hard to know whether these changes are always beneficial, improving model quality.

Participants in the group and individual studies were students (and one Professor), threatening external validity. However, participants had a wide variety of backgrounds and education levels, increasing confidence in the generalizability of our results.

Case studies applying analysis to realistic domains were facilitated by analysts who had some knowledge of *i** and interactive analysis; thus, we may have introduced bias to the results. However, several analysts were new to *i** and analysis and still noted benefits of analysis. Likewise, the individual and group studies were facilitated by an *i** and analysis expert.

The nature of the domains may have some effect on results. In the individual studies, participants found the domains to be either too unfamiliar or too familiar. The counseling service study is a very social-oriented domain; some of the benefits of interactive analysis may not be as applicable in a more technical domain with less human interaction.

7 Related work

In this section, we summarize existing techniques for goal model analysis, evaluating them in light of the contributions of the proposed framework. In previous work, we have presented a literature review of goal model analysis techniques, including an analysis of the objectives of goal model analysis and guidelines for selecting between existing procedures [32]. Here, we include a summarized and updated version of this review. We focus on procedures which provide satisfaction analysis, answering questions similar to the analysis procedures introduced in this framework. We then briefly summarize other procedures which answer different types of analysis questions over goal models.

Satisfaction analysis. We identified a number of procedures which analyze the satisfaction or denial of goals in a model, similar to the procedures introduced in this framework. These procedures use model links to propagate initial labels in either the forward [4, 10, 20, 40, 41, 44, 50] or backward [20, 21, 41] direction, answering "what if?" and "are certain goals achievable?"

Some satisfaction analysis procedures present results in terms of qualitative labels representing satisfaction or denial, similar to the labels used in this work [4, 10, 20, 21]. Several procedures offer quantitative analysis, using numbers to represent the probability of a goal being satisfied or denied [21, 41, 50] or to represent the degree of satisfaction/denial [4, 40].

Other procedures produce binary results, where goals have only one of two labels, typically satisfied or not [14, 38, 44]. For example, the Techne approach uses quality constraints to approximate all softgoals, as such, model analysis does not consider partial labels, and all elements are either satisfied or not [38].

Recent work has applied goal modeling and quantitative satisfaction analysis to facilitate business intelligence, taking input labels from data via atomic and composite data indicators and mapping them to quantitative or qualitative analysis results [25, 45].

One of the primary features distinguishing between these approaches is their means of resolving multiple contribution incoming labels. Some procedures separate negative and positive evidence, making it unnecessary to resolve conflicts in order to find solutions over the model [20, 21]. Other procedures make use of predefined qualitative or quantitative rules to combine multiple sources of evidence [4, 40]. Further procedures, including the ones in this framework and analysis in the NFR framework [10], are interactive, using human intervention to resolve partial or conflicting evidence.

Our previous work has aimed to compare approaches for goal model satisfaction analysis in order to determine whether these differences between procedures make a significant difference in the analysis results [24, 34]. Seven forward analysis procedures (described in [4, 10, 20]), including the procedure described in this work, were applied to three example goal models (taken from [4, 21, 28]. The results were compared using a mapping between qualitative and quantitative scales. The analysis showed that results differed between procedures, especially for "softer" models with many softgoals or dependencies, leading to the conclusion that goal model satisfaction procedures are better used as heuristics, emphasizing the benefits of these procedures beyond the provision of analysis results, e.g., prompting model iteration, and facilitating communication.

We have adapted several of the concepts used in our forward analysis procedure from the pre-existing, interactive NFR analysis procedure [10]. Our approach goes beyond this work in several ways, e.g., by providing formal semantics to analysis, adding the capability for backward analysis, and providing visualizations. Several of the formal aspects presented in our framework were inspired by existing procedures for backward reasoning with goal models ([20, 21]). However, our approach is novel in that it axiomatizes propagation in the $i*$ framework (including dependency and unknown links), combines evidence for each intention into a single analysis label (including conflict and unknown), includes iterative human intervention (resolving conflicting or partial evidence), and provides information on model conflicts when a solution cannot be found.

By focusing on the contributions of our framework, as listed in the introduction, we can identify further points which distinguish our framework from existing satisfaction approaches, making it more appropriate for Early RE. Although existing analysis approaches support "what if?" and "are these goals achievable?" analysis questions, to our knowledge, we are the only approach which supports analysis over sources of contradictions ("if is not possible, why not?"). Existing work has not taken into account the coverage of model analysis results, while our validation studies have shown that root and leaf visualizations help to make model analysis more complete.

Our work provides a suggested methodology for model creation and analysis, while several techniques for goal model analysis do not provide an explicit methodology beyond the analysis algorithm (e.g., [37]). Others focus on technical aspects concerning how to apply the analysis procedure, but do not describe iteration over the model and analysis results (e.g., [10, 41]).

Our framework aims to increase the completeness and correctness of the model. Most available analysis procedures proceed with the assumption that the model is complete and correct. Although some procedures include interaction as part of the analysis process, e.g., [8, 10, 15, 18, 44], these approaches aim less at encouraging iteration and more on using stakeholder expertise to initiate analysis or judge analysis output. Other analysis procedures mention iteration over analysis inputs in order to find the most satisfactory solution (e.g., [1, 19]). Some approaches consider the possibility of iteration over the model, e.g., [17], but treat such changes as a side effect of errors or inadequacies and not as a desired outcome of the analysis process. Work by Liaskos et al. addresses model iteration as a positive benefit of iteratively applying planning and analysis, but focuses on iteration over model preferences [42].

We have aimed to create analysis procedures which are simple from the user's perspective, validating usability through cases studies, while existing goal model procedures do not explicitly aim for simplicity. Although some approaches use realistic case studies to validate the usability of their work, the focus of such studies is not on usability from the point of view of stakeholders, with model analysis usually performed by researchers. Such approaches do not explicitly consider or evaluate the ability of stakeholders to comprehend analysis results over either simple or complex models.

Other goal-oriented analysis approaches. Several approaches aim to measure qualities over the domain, such as security, vulnerability, and efficiency, using metrics over constructs in the model. These procedures can answer questions like "how secure is the system represented by the model?" or "how risky is a particular alternative for a particular stakeholder?" (e.g., [15]). Methods have applied AI-type planning to find satisfactory sequences of actions or design alternatives in goal models. These procedures can be used to answer questions such as "what actions must be taken to satisfy goals?" or "what is the best plan of action according to certain criteria?" (e.g.[8]). Several approaches have added temporal information to goal models to allow for simulation over the network represented by model constructs. In these approaches, a particular scenario is simulated, and the results are checked for interesting or unexpected properties. These procedures can answer questions like "what happens when a particular alternative is selected?" (e.g., [18]). Several approaches provide ways to perform checks over the models supplemented with additional information, allowing users to ask questions like "is it possible to achieve a particular goal?" or "is the model consistent?" (e.g., [17]).

Non-goal approaches. We could also examine related approaches outside of goal modeling, such as approaches for trade-off analysis in RE, or approaches for modeling and decision making in business. Although, these approaches may offer useful ideas, they do not allow for the high-level modeling and analysis facilitated by goal models, well-suited for early RE. Thus, we focus our review of related approaches to those using goal orientation.

8 Conclusions

8.1 Contributions

Our framework has made several contributions. We have provided *analysis power*, supporting "what if?"-type questions, including "what are the effects of a particular analysis alternative?", "are goals sufficiently satisfied?", and "whose goals are satisfied?" In addition, we allow users to ask "is it possible to achieve certain goal(s)?", "if so how?", "who must do what?", and "if is not possible, why not?" Our validation studies showed that for forward analysis in realistic studies such as the counseling service study, analysis was very helpful in comparing and assessing technical alternatives and knowledge transfer mechanisms, including allowing for "as-is" to "to-be" comparisons. The inflo study revealed that backward analysis was useful in answering basic analysis questions which tested the sanity of the model.

We have provided a *methodology* for the creation and analysis of agent-goal models, with an emphasis on interaction and iteration. Our framework allows the user to resolve partial or conflicting evidence via human judgments, supplementing high-level models with their domain knowledge, involving stakeholders in the analysis process, and encouraging beneficial model changes.

Experience in realistic case studies indicates that interactive analysis reveals unknown information and causes beneficial model iteration. However, when using the procedure in more artificial environments, without the presence of driving domain questions, far fewer discoveries and changes are made. Similarly, experimental results show that both interactive and ad hoc analysis raise questions and provoke model changes. Overall, we claim that in the appropriate situation—knowledgeable modelers motivated by driving questions in a real domain—interactive analysis can reveal gaps in knowledge and provoke beneficial iteration.

Our framework supports *high-level analysis* by deliberately avoided requiring additional information beyond what is typically required by *i** models, with a focus on high-level, early analysis. Our *formal definition* of *i** considered common deviations in order to effectively balance the need to provide a precise model interpretation with the need for inexpressiveness to represent imprecise early RE concepts. Case study experience has demonstrated the ability of the analysis to reason over concepts such as security, confidentiality, and quality of counseling, drawing conclusions over intentions which are hard to define formally. Validation study results show that systematic analysis increases the consistency of model interpretations, e.g., propagation through contribution links. These factors would make analysis results more consistent or reliable when comparing results over the same model, potentially with different evaluators.

Our framework addresses usability by providing a guiding methodology and providing a semiautomated implementation in OpenOME. The tool hides formal details from the user, using analysis labels, lists of analysis results, and color-based visualizations. In validation studies, participants were able to use the tool to apply both the forward and backward analysEs with minimal training. Deficiencies were noted more in their ability to understand the meaning behind *i** syntax than their ability to apply analysis. Several of usability issues noted in our studies (e.g., applying initial labels, understanding results) were addressed in subsequent rounds of implementation and iterations over the suggested methodology.

We have considered both the computational and interactive *scalability* of our framework, showing that analysis is scalable to models of a reasonable size. Models larger than this would be no longer cognitively scalable for manual creation and analysis comprehension.

8.2 Limitations

Our framework has made significant progress toward effective analysis of early RE agent-goal models, but still has several limitations.

Goal modeling limitations. By using agent-goal models for early RE analysis, we inherit all of the challenges and limitations inherent to this type of modeling, including the complexity and scalability of models, as demonstrated by several of our examples. Although analysis can help to make sense of models, analysis can only do so much to ease the cognitive load of complex goal models. Future work in agent-goal model scalability, for example, [16], could be promising as a point of integration with our approach.

Alternative selection. The procedures in this framework focus on the evaluation of individual analysis alternatives; although multiple results are stored in implementation, this work does not provide specific guidance in how to compare the results of multiple analysis alternatives. Future work should investigate techniques which help to guide people in comparing and selecting between the results of multiple analysis alternatives.

Generalizability. The procedures introduced in this work have been designed for and applied to the *i** framework. We argue that these procedures can generalize relatively easily to similar frameworks (e.g., GRL [3], NFR [10], Tropos [6]). Applying our procedures to less similar goal modeling frameworks (e.g., KAOS [11], AGORA [39]) would prove more challenging. Our interactive analysis is especially applicable to models containing softgoals and contribution links, creating areas of model contention requiring human intervention. If other goal modeling frameworks do not contain such areas, concepts and algorithms introduced in this work are not easily applicable.

Validation results. The results of our validation studies are mixed. Although we have found evidence to support iteration over models and elicitation in the domain as a result of interactive analysis, we have also found cases where this iteration and elicitation does not present itself prominently. Future studies should include a comparison with fully automated analysis.

8.3 Future work

We have identified several areas of potential framework expansions. We summarize several of these areas here.

Implementation optimizations. Future work should aim to optimize the backward analysis algorithm described in Sect. 5.3; for example, zChaff solver results could be stored in a stack, popping results when backtracking. Explicit backward axioms for non-contribution links could be removed from the encoding. The number of human judgment situations could be reduced in both procedures by reusing judgments across analysis alternatives. However, automatic reuse of judgments may discourage users from reconsidering and revising their judgments. Currently our implementation displays all existing judgments in a separate tree view (see Fig. 7).

Judgment consistency checks. Case study experiences show that when the judgment made by the user differs from what is suggested by the model, the modeler may be motivated to revise the model. However, in our studies we found several occasions where novice modelers made judgments that were inconsistent with the structure of the model, and did not use these opportunities to make changes or additions to the model. Preliminary work has outlined several consistency checks between the judgment and the model, and between old and new judgments [33]. Such checks allow us to embed modeling expertise within the tool, encouraging the user to resolve inconsistencies when possible.

Multiple solutions. Currently, backward analysis uses a solver which provides a single solution, if such a solution exists. Future improvements to the framework implementation could make use of a solver which finds multiple solutions, if they exist, (e.g., [22]) allowing a user to select a particular solution to pursue. Alternatively, one could allow the users a "find next" option, asking the solver to find another solution matching targets and judgment constraints, if one exists. In either case, further algorithms and guidance for selecting between available solutions may be needed.

Model evolution. As our analysis framework aims to encourage model iteration, expansions to the framework should handle continuously evolving models. A change in a model could prompt an automatic re-evaluation of the model, propagating as far as possible, and then prompting the user if new judgments are needed. Or, in an effort to promote model comprehension, the user could be shown what parts of the analysis results were affected by their changes, if any.

Analysis of uncertain models. Recent work has described the application of a formal framework representing modeling uncertainty to goal models in an RE context [46]. Further work has integrated this approach with an automated version of the forward goal model analysis described in our framework [27]. Such analysis allows one to ask questions such as "given model uncertainties, what analysis results are possible?" and "what uncertainties must be resolved to achieve target values?" The first author is currently working with collaborators to extend this work, integrating analysis over uncertain models approach with backward analysis. Future work will investigate the changes necessary to make analysis of uncertain models interactive, allowing for human judgment over conflicting or partial evidence.

From early to late RE. Future work should guide users in moving from early RE models, and the type of analysis introduced in this work, into more detailed RE models. Such are the models introduced and used in many of the existing goal model analysis approaches, requiring detailed information such as probability, priority, or temporal ordering. Recent work aimed at business intelligence models simultaneously uses early qualitative and later quantitative analysis [25]. Here, analysis can be qualitative over less specified areas of the model and quantitative, using domain-specific equations, in more specified areas. Analysis results are mapped together, facilitating complete model propagation. Our qualitative analysis could fit well into this approach.

Confidence in analysis results. Future work can aim to measure the perceived confidence in analysis results based on several factors such as confidence in the sources of the model, the structure of the model (e.g., how many softgoals), the length of propagation paths, the sources of initial evaluation labels, and the means of propagation (e.g., qualitative through propagation links or quantitative using domain-specific formula). Such confidence measures can help to guide users in whether or not the analysis results should be used as a heuristic only, or can be more trusted, using concrete domain measures.

Varying levels of automation. It would be useful to allow users to modify the level of automation. Depending on their confidence in the model (accuracy, completeness), they could select a level of automation along a sliding scale, ranging from judgment in all potentially contentious areas to full automation using set rules to combine evidence, such as in [4]. Future work should investigate situations where users choose to increase or decrease the level of automation, and how well this facilitates effective RE analysis.

Further validation. Further validation should be conducted, testing the methodology and implementation, including new interventions such as human judgment checks. Such studies could try to test a variety of types of analysis (ad hoc, interactive, fully automatic) in realistic settings; however, challenges in designing effective studies (realistic vs. easily measurable) must be addressed.

Acknowledgments Financial support for various stages of this work has been provided by the Bell University Labs, the Natural Sciences and Engineering Research Council of Canada, the Ontario Graduate Scholarship Program, the Canada, the Ontario Graduate Scholarship Program, the NSERC Business Intelligence Network, and ERC advanced grant 267856 "Lucretius: Foundations for Software Evolution". The OpenOME tool was ported from OME and expanded by Yijun Yu, Neil Ernst, and many undergraduate students. The presented work has benefited from feedback from many colleagues at the University of Toronto and Trento, including detailed comments on the current paper from Daniel Gross, and extensive input from the PhD Committee members of the first author: Steve Easterbrook and John Mylopoulos. We are grateful for the detailed comments from the anonymous reviewers, which helped to greatly improve the quality of the paper.

References

1. Datamonitor. http://www.datamonitor.com/
2. OpenOME, an open-source requirements engineering tool (2010). https://se.cs.toronto.edu/trac/ome/wiki
3. Amyot D (2003) Introduction to the user requirements notation: learning by example. Comput Netw 42(3):285–301
4. Amyot D, Ghanavati S, Horkoff J, Mussbacher G, Peyton L, Yu E (2010) Evaluating goal models within the goal-oriented requirement language. Int J Intell Syst 25(8):841–877
5. Borgida A, Horkoff J, Mylopoulos J (2014) Applying knowledge representation and reasoning to (simple) goal models. In: First international workshop on artificial intelligence for requirements engineering to appear, Newyork
6. Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J (2004) Tropos: an agent-oriented software development methodology. Auton Agents Multiagent Syst 8(3):203–236
7. Bruni R, Sassano A (2001) Restoring satisfiability or maintaining unsatisfiability by finding small unsatisfiable subformulae. Electron Notes Discret Math 9:162–173
8. Bryl V, Giorgini P, Mylopoulos J (2009) Supporting requirements analysis in tropos: a planning-based approach. In: Ghose A, Governatori G, Sadananda R (eds) Agent computing and multi-Agent systems, LNCS, vol 5044. Springer, Berlin, Heidelberg, pp 243–254
9. Cabot J, Easterbrook S, Horkoff J, Lessard L, Liaskos S, Mazon JN (2009) Integrating sustainability in decision-making processes: a modelling strategy. 31st international conference on software engineering companion volume, pp 207–210
10. Chung L, Nixon BA, Yu E, Mylopoulos J (2000) Non-functional requirements in software engineering, international series in software engineering, vol 5. Kluwer Academic Publishers,New York
11. Dardenne A, Lamsweerde AV, Fickas S (1993) Goal-directed requirements acquisition. Sci Comput Program 20(1–2):3–50
12. Easterbrook S, Yu E, Aranda J, Fan Y, Horkoff J, Leica M, Qadir RA (2005) Do viewpoints lead to better conceptual models? An exploratory case study. In: Proceedings of the 13th IEEE international requirements engineering conference, IEEE, pp 199–208
13. Easterbrook S, Yu E, Aranda J, Horkoff J, Strohmaier M, Fan Y, Leica M, Qadir RA (2011) Analyzing requirements for online presence kids help phone. In: Horkoff J (ed) Proceedings of the iStar Showcase '11. Exploring the goals of your systems and Businesses. Practical experiences with i* modeling. (unpublished). http://www.cs.toronto.edu/km/istar/iStarShowcase_Proceedings.pdf
14. Ernst NA, Mylopoulos J, Borgida A, Jureta IJ (2010) Reasoning with optional and preferred requirements. In: Proceedings of the 29th international conference on conceptual modeling (ER'10), LNCS, vol 6412. Springer, pp 118–131
15. Franch X (2006) On the quantitative analysis of agent-oriented models. In: Proceedings of the 18th international conference on advanced information systems engineering (CAiSE'06), LNCS, vol 4001. Springer, pp 495–509
16. Franch X (2010) Incorporating modules into the i* framework. In: Proceedings of the 22nd international conference on advanced information systems engineering (CAiSE'10), LNCS, vol 6051. Springer, pp 439–454
17. Fuxman A, Liu L, Pistore M, Roveri M, Mylopoulos J (2003) Specifying and analyzing early requirements: some experimental results. In: Proceedings of the 11th IEEE international requirements engineering conference, IEEE, pp 105–114
18. Gans G, Jarke M, Lakemeyer G, Vits T (2002) SNet: A modeling and simulation environment for agent networks based on i* and ConGolog. In: Proceedings of the 14th international conference on advanced information system engineering (CAiSE'02), LNCS, vol 2348. pp 328–343
19. Giorgini P, Mylopoulos J, Nicchiarelli E, Sebastiani R (2004) Formal reasoning techniques for goal models. J Data Semant 1:1–20
20. Giorgini P., Mylopoulos J., Sebastiani R (2004) Simple and minimum-cost satisfiability for goal models. In: Proceedings of the 16th conference on advanced information systems engineering (CAiSE'04), LNCS, vol 3084. Springer, pp 20–35

21. Giorgini P, Mylopoulos J, Sebastiani R (2005) Goal-oriented requirements analysis and reasoning in the Tropos methodology. Eng Appl Artif Intell 18(2):159–171
22. Griggio A (2012) A practical approach to satisfiability modulo linear integer arithmetic. JSAT 8:1–27
23. Horkoff J (2006) Using i* Models for evaluation (2006). Master's thesis, University of Toronto
24. Horkoff J (2012) Iterative, interactive analysis of agent-goal models for early requirements engineering. Ph.D. thesis, Department of Computer Science, University of Toronto
25. Horkoff J, Barone D, Jiang L, Yu E, Amyot D, Borgida A, Mylopoulos J (2014) Strategic business modeling: representation and reasoning. Softw Syst Model 13(3):1015–1041
26. Horkoff J, Elahi G, Abdulhadi S, Yu E(2008) Reflective Analysis of the Syntax and Semantics of the i* Framework. In: Proceedings of 2nd international workshop on requirements, intentions, and goals in conceptual modeling (RIGiM'08), LNCS, vol 5232. pp 249–260
27. Horkoff J, Salay R, Chechik M, Di Sandro A (2014) Supporting early decision-making in the presence of uncertainty. In: Proceedings of the 22nd IEEE international requirements engineering conference. To appear
28. Horkoff J, Yu E (2009) Evaluating goal achievement in enterprise modeling an interactive procedure and experiences. In: Proceedings of the 2nd IFIP WG8.1 on the practice of enterprise modeling (PoEM'09), Springer, pp. 145–160
29. Horkoff J, Yu E (2010) Finding solutions in goal models: an interactive backward reasoning approach. In: Proceedings of the 29th international conference on conceptual modeling (ER'10), LNCS, vol 6412. Springer, p 59
30. Horkoff J, Yu E (2010) Interactive analysis of agent-goal models in enterprise modeling. Int J Inf Syst Model Des IJISMD 1(4):1–23
31. Horkoff J, Yu E (2010) Visualizations to support interactive goal model analysis. In: Proceedings of the 5th international workshop on requirements engineering visualization (REV'10), pp 1–10, IEEE
32. Horkoff J, Yu E (2011) Analyzing goal models: different approaches and how to choose among them. In: Proceedings of the ACM symposium on applied computing., SAC '11ACM, New York, pp 675–682
33. Horkoff J, Yu E (2011) Detecting judgment inconsistencies to encourage model iteration in interactive i* analysis. In: Proceedings of the 5th international i* workshop, pp 2–7
34. Horkoff J, Yu E (2013) Comparison and evaluation of goal-oriented satisfaction analysis techniques. Requir Eng 18(3):199–222
35. Horkoff J, Yu E, Ghose A (2010) Interactive goal model analysis applied–systematic procedures versus Ad hoc analysis. In: Proceedings of the 3rd IFIP WG8.1 on the practice of enterprise modeling (PoEM'10), LNBIP, vol 68. Springer, pp 130–144
36. Horkoff J, Yu E, Liu L (2006) Analyzing trust in technology strategies. In: Proceedings of the international conference on privacy security and trust (PST'06), p 1, Press
37. ITU-T (2008) Intern. Telecom. Union: Recommendation Z.151 (11/08): recommendation Z.151 (11/08) User requirements notation (URN) language definition. Tech. rep., ITU-T International Telecommunications Union, Geneva, Switzerland
38. Jureta IJ, Borgida A, Ernst NA, Mylopoulos J (2010) Techne: towards a new generation of requirements modeling languages with goals, preferences, and inconsistency handling. In: Proceedings of the 18th IEEE international requirements engineering conference, pp 115–124, IEEE
39. Kaiya H, Horai H, Saeki M (2002) AGORA: attributed goal-oriented requirements analysis method. In: Proceedings IEEE joint international conference on requirements engineering, pp 13–22, IEEE
40. van Lamsweerde A (2009) Reasoning about alternative requirements options. In: Borgida AT, Chaudhri VK, Giorgini P, Yu ES (eds) Conceptual modeling: foundations and applications. Springer, Berlin, Heidelberg
41. Letier E, Lamsweerde AV (2004) Reasoning about partial goal satisfaction for requirements and design engineering. ACM SIGSOFT Softw Eng Notes 29(6):53–62
42. Liaskos S, McIlraith S, Sohrabi S, Mylopoulos J (2011) Representing and reasoning about preferences in requirements engineering. Requir Eng 16:227–249
43. Mahajan YS, Fu Z, Malik S (2004) Zchaff 2004: an efficient SAT solver. In: Proceedings of the 7th international conference on theory and applications of satisfiability testing (SAT04), pp 360–375
44. Maiden N, Lockerbie J, Randall D, Jones S, Bush D (2007) Using satisfaction arguments to enhance i* modelling of an air traffic management system. In: Proceedings of RE'07, pp 49–52
45. Pourshahid A, Richards G, Amyot D (2011) Toward a goal-oriented, business intelligence decision-making framework. In: E-technologies: transformation in a connected world, LNBIP, vol 78. Springer, pp 100–115
46. Salay R, Chechik M, Horkoff J (2012) Managing requirements uncertainty with partial models. In: Proceedings of the 20th IEEE international requirements engineering conference, pp 1–10
47. Stirna J, Persson A, Sandkuhl K (2007) Participative enterprise modeling: experiences and recommendations. In: Proceedings of the 19th conference on advanced information systems engineering (CAiSE'07), vol 4495. Springer, pp 546–560
48. Strohmaier M, Horkoff J, Yu E, Aranda J, Easterbrook S (2008) Can patterns improve i* modeling? Two exploratory studies. In: Proceedings of the 14th international working conference on requirements engineering foundation for software quality, Springer, pp 153–167
49. Strohmaier M, Yu E, Horkoff J, Aranda J, Easterbrook S (2007) Analyzing knowledge transfer effectiveness an agent-oriented modeling approach. In: Proceedings of the 40th annual Hawaii international conference on system sciences, p 188b, IEEE Computer Society
50. Tanabe D, Uno K, Akemine K, Yoshikawa T, Kaiya H, Saeki M (2008) Supporting requirements change management in goal oriented analysis. Proceedings of the 16th IEEE international requirements engineering conference, pp 3–12
51. Yu E (1997) Towards modelling and reasoning support for early-phase requirements engineering. In: Proceedings of 3rd IEEE international symposium on requirements engineering, pp 226–235, IEEE
52. Yu E, Giorgini P, Maiden N, Mylopoulos J (2011) Social modeling for requirements engineering. MIT Press, Cambridge