Conflict Identification and Resolution for Trust-Related Requirements Elicitation A Goal Modeling Approach

Angela Borchert* and Maritta Heisel
University of Duisburg-Essen, Duisburg, Germany
{angela.borchert, maritta.heisel}@uni-due.de

Received: January 3, 2021; Accepted: February 18, 2021; Published: March 31, 2021

Abstract

In requirements engineering, goal modeling is a popular approach for requirements elicitation and improvement. It can not only support in specifying software goals and requirements but also in identifying and resolving conflicts between and among them. In this work, we extend the method for eliciting trust-related software features, called *TrustSoFt*, by the *i* goal modeling framework*. We extend the i* notation adapted to TrustSoFt and further provide a guideline for goal model creation, conflict identification, and conflict resolution in order to ensure that user concerns are addressed in the best possible way by the software to be developed. Examples are provided for an online dating application regarding the user concerns "fake profiles" and "data misuse". As a result, goal modeling impels requirements engineers to properly annotate trustworthiness goals and requirements. It facilitates the identification of relevant elements for the issue being modeled and results in fine-graded trustworthiness requirements. We consider goal models as useful input for the development of trust-related software features.

Keywords: goal modeling, requirements elicitation, conflicting requirements, trustworthiness, social media

1 Introduction

User-centred software engineering has many benefits. Resulting systems are more likely to satisfy users, not fail in their purpose and can have a positive impact on user health and safety [26]. Taking over the user perspective facilitates the development of systems that respond to the user's needs and, thus, is likely to be perceived as trustworthy by the users [10].

The method for eliciting trust-related software features (TrustSoFt) focuses on user-centred software engineering. It is a requirements elicitation method that addresses trust concerns of users with the objective to enhance their safety [5]. However, in our previous work, we recognized that although all trustworthiness goals and requirements resulting from TrustSoFt are valuable for addressing trust concerns, some are problematic in a concurrent implementation [6]. According to Hunter and Nuseibeh [21], such inconsistencies arise due to mistakes during requirements elicitation or because of conflicts between requirements. In order to deal with such conflicts among goals and/or requirements, in our previous work [6], we included risk regarding user concern, trustworthiness goals and requirements as determinants to TrustSoFt. Based on risks, requirements engineers can assess which of the conflicting goals and requirements contribute more to the user benefit. Risk assessment is used for a prioritization

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), 12(1):111-131, Mar. 2021
DOI:*10.22667/JOWUA.2021.03.31.111*

*Corresponding author: University of Duisburg-Essen, Faculty of Engineering, Department of Software Engineering, BB 918, Oststrasse 99, 47057 Duisburg, Germany, Tel: +49-203-379-4503

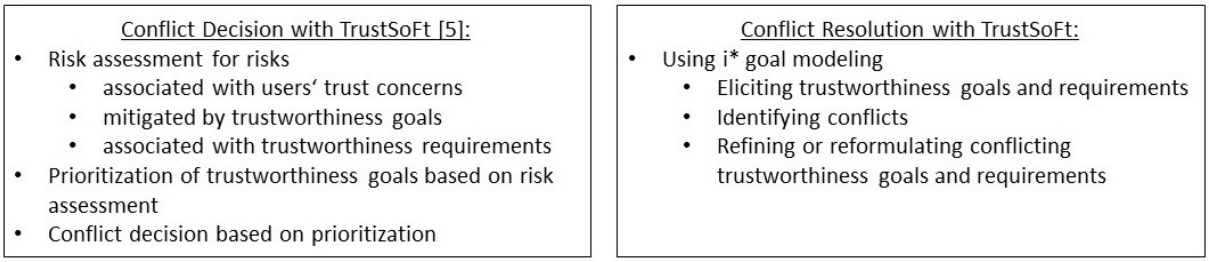


Figure 1: Comparison of the previous approach [6] with the one in this paper.

of trustworthiness goals, which in turn serves as a decision basis for either one or the other conflicting goal or requirement to be implemented [6].

However, such conflict decisions lead to the dilemma that by omitting one option, the requirements engineer drops the user benefit associated with it. Instead, we now aim to follow the proposal of Nuseibeh and Easterbrook [31] to identify the cause of conflicts and resolve them. Goals and requirements shall be specified in a way that no conflict arises so that the need of conflict decisions shall be minimized. On these grounds, we want to make use of goal modeling as a supportive technique for TrustSoFt. By goal modeling, requirements engineers can increase their understanding of the application field and the user [30]. This facilitates user-centred requirements elicitation in a structured and correct way [39]. In addition, we use goal modeling for conducting conflict identification and resolution within TrustSoFt. Figure 1 depicts a brief overview of the different approaches of our previous work and this work. For more insights regarding conflict decisions, please check on our previous work [6].

The remainder of this work is as follows. First, we provide a theoretical background about TrustSoFt and goal modeling (Section 2). There, we briefly present the i* goal modeling framework as we make use of its notation here. As a next step, we describe how goal models can be created in the context of TrustSoFt (Section 3). Afterwards, we explain how the goal models can be used to identify and resolve conflicting goals and requirements that have been elicited (Section 4). This is followed by demonstrating goal model creation, conflict identification and conflict resolution in the context of an online dating application (Section 5). Finally, we refer to related work (Section 6), discuss our contribution and point out future research (Section 7) to end with a conclusion (Section 8).

2 Theoretical Background

This work builds upon the method for eliciting trust-related software features (TrustSoFt). TrustSoFt is extended to a model-based approach to support its execution, but also to identify and resolve conflicts between trustworthiness goals and requirements. To this end, goal modeling is a suitable modeling approach. Therefore, we introduce TrustSoFt, goal modeling and the i* framework for goal modeling in this section.

2.1 The TrustSoFt Method

The method for eliciting trust-related software features (TrustSoFt) aims to mitigate trust concerns of users [5, 6]. It is a method that is especially relevant during the requirements elicitation phase of software engineering. A special aspect of the method is that it considers trustworthiness facets of a user's interaction partners in its process. Trustworthiness facets represent traits that are evaluated by an individual user in order to assess the trustworthiness of interaction partners. These can be for example i) the application, ii) the service provider and iii) other end users. TrustSoFt is based on the assumption that

risks associated with the software use are reduced when users can better assess the trustworthiness of interaction partners. Depending on the trustworthiness assessment of these parties, users decide whether or not to interact with them and thereby accept or reject associated risks.

Therefore, TrustSoFt is especially relevant for developing applications for so-called computer-mediated introduction (CMI). CMI can be assigned to social media, where sympathy and trust are major drivers to meet people in the physical world, who users got to know online on these platforms [33]. Examples for CMI applications are online dating and sharing economy. There, for instance, risks involve data misuse or privacy violations caused by the application or the service provider, and fake profiles, online fraud, or physical violence by other users [11, 18].

TrustSoFt involves 13 steps that are depicted in Figure 2. The boxes highlighted in green represent method components relevant for this work, which will be elaborated in more depth later in Section 3. The first step of TrustSoFt is about the identification of users' *trust concerns* (1). Trust concerns can be defined as a specific issue the user is uncertain about in a particular application field [24]. These concerns arise, because the user has a lack of trust in involved parties, which is not properly addressed by the software. Therefore, the user sometimes has behavioral strategies to reduce the concern on her own. Such strategies are called *workarounds* [1]. They are interesting to know about, because they might give hints for potential software features. Trust concerns and workarounds can be elicited by reviewing previous research or conducting quantitative or qualitative research. They seem to be subjectively particularly relevant to the user and can lead to emotional turmoil [25]. Therefore, it is important to check on the actual risk of the issue for a more objective perspective (2). Thereby, the requirements engineer understands how frequent and precarious the concerns are, which is a basis for conflict decisions later in the process (12). For conducting a risk assessment, we refer to the ISO 31000 standard [15].

The next step is to specify *trustworthiness goals* for the software. (3). Trustworthiness goals describe objectives that the system should address and which correspond to the objectives users intend to achieve during software use [28]. They are trust-related, because they shall mitigate users' trust concerns. Once the goals have been defined, the risks identified in Step 2 shall be assigned to the goals that will reduce them if they are achieved (4). This step is relevant for prioritizing the goals in terms of their impact on re-

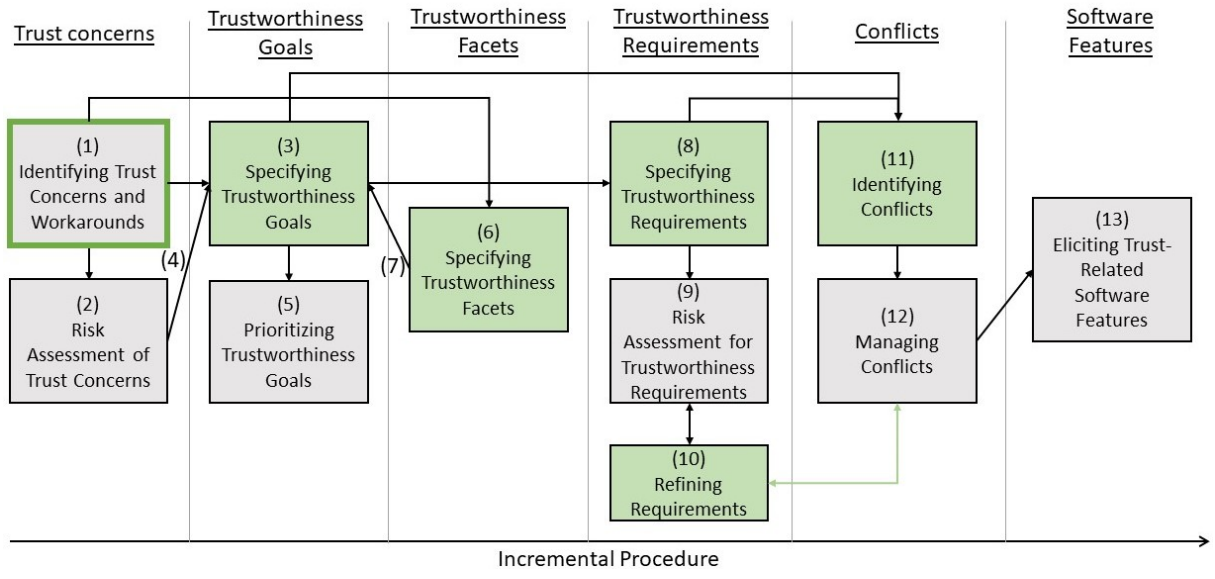


Figure 2: TrustSoFt Method Flow. The green boxes highlight the steps relevant for the model-based approach introduced in this work. The green frame shows the starting point for the goal modeling.

ducing the associated risks (5). We propose to use estimated risk values from Step 2 for the prioritization process. Goal prioritization is relevant for subsequent conflict decisions (12).

Afterwards, trustworthiness facets are specified (6). Facets can be either inferred by considering the input of Step 1 or from trust research. If the input of Step 1 is user interviews or surveys, people might have pointed out what they pay attention to when dealing with interaction partners in the context of trust concerns. Regarding the consideration of trust research, most often similar concerns have been examined or defined before to the ones identified in Step 1. As an example, research has identified the information privacy concern "Collection" as significant for for Internet users. The concern collection expresses that users are worried about the amount of personal data possessed by others relative to the received benefits for sharing the data [27]. Culnan and Bries related collection to distributive justice and perceived fairness of the outcome [12]. Fairness was again identified to influence trustworthiness [19]. Therefore, fairness is a relevant facet for the concern collection. If such conclusions cannot be drawn, it is up to the requirements engineer to assume what facets might be important to the user's trustworthiness assessment. Specified facets are then assigned to trustworthiness goals (7). Thereby, they are expected to be considered during goal realization and implemented by the system. The system should represent or present facets in order to aid users in their trustworthiness assessment of involved stakeholders. Therefore, each goal shall at least address one facet, if possible.

Subsequently, *trustworthiness requirements* are specified for the system (8). They determine what the system needs to do in order to achieve the trustworthiness goals and facets. For the defined requirements, a risk assessment should be carried out (9). Unlike the previous one, this risk assessment focuses on risks that are not mitigated but accompany requirements when implemented. If the risk assessment identifies unacceptable risks, the requirements shall be reformulated in a way that the risks become acceptable (10). Then, the risk assessment needs to be repeated to reevaluate the risk level. If a risk still remains unacceptable, the concerned requirement should not be implemented in the system.

Now that the goals and requirements for the application to be developed have been specified, it is up to the requirements engineer's expertise to compare them in order to identify conflicts (11). We distinguish between soft and hard conflicts [6]. While soft conflicts allow simultaneous implementation of conflicting requirements even though they limit each other's effectiveness, hard conflicts describe an impossibility that both can be realized at the same time. In both cases, a conflict decision must be made (12). In case of hard conflicts, the goal prioritization from Step 5 needs to be consulted. The goals from which conflicting requirements originate, are compared regarding their priority. For conflict decision, the conflicting requirement that originates from the goal with the highest priority shall be implemented. If the respective goals have equal priority, stakeholder preferences should be taken into account to make a decision (e.g. privacy preference). For deciding on soft conflicts, it must be investigated whether a goal or requirement refinement can resolve the conflict (green arrow, 10). Reformulated requirements and goals must be reassessed concerning their risks (4, 5, 9). If the conflict can be resolved by refinement, all respective requirements and goals can be realized. Otherwise, the goal prioritization has to be consulted again, as it is described before for the hard conflicts.

As a last step of the method, software features shall be elicited (13). They represent frontend or backend approaches how the corresponding requirements can be realized.

2.2 i* Goal Modeling

Goals can be defined as "an objective the system under consideration should achieve [by referring] to intended properties to be ensured" [39]. Based on that, they provide the rationale for requirements and are a criterion for requirement pertinence and completeness of a requirements specification [42]. In addition to their high usefulness for requirements elicitation, goals also indicate intentions and desires of stakeholders leading to an improved domain understanding. Furthermore, they point the way to developing

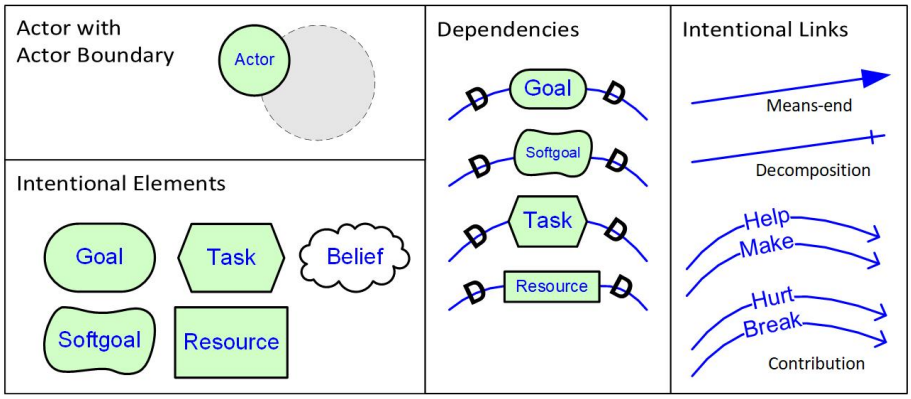


Figure 3: Basic Elements of the i* Language. Graphic inspired by Franch et al.[16]

software that fulfills its purpose [23]. For these reasons, goal modeling is a useful method for the early phase of requirements engineering. In this stage, there are a multitude of options how the system could be developed. Goal modeling does not only capture the options but also the accompanying consequences so that the requirements engineers is able to explore the solution space.

The i* goal modeling framework is an agent-oriented goal modeling notation proposed by Yu [41]. It is especially conceived to illustrate socio-technical systems in an organizational context. Thereby, it considers agents' intentions and their dependencies among each other. In order to model those, the i* notation can be used to define two different model views. In the *Strategic dependency* (SD) model, only involved stakeholders, called actors, and dependencies between them are depicted. The *Strategic rationale* (SR) model focuses on the actors' intentionalities in the given context.

Over the past three decades, the i* framework has been subject to new research influences and has evolved into several variations [16]. Therefore, we briefly introduce the main language elements relevant for this work. An overview of the language elements is depicted in Figure 3. For detailed information, we refer to the original work of Yu [41], but also Franch et al. [16] and the iStarWiki of the RWTH Aachen [38].

2.2.1 Actors and Actor Association Links

Actors are active entities. They apply their know-how in order to achieve their goals. Actors can be human-beings, organizations or technologies. In the i* models, actors can be used in a general way or in classified forms. Classified forms are especially relevant for modeling an organizational structure. Then, actor association links are used among the actors to depict their relationships. As we do not deal with the organization structure but focus on the application, we refrain from a detailed description of the various actors and association links due to space constraints. For our purpose, we only refer to the *INS-relationship*, which represents the instantiation of a more general entity in an instance. For example, a CMI application can be an instance of a CMI service provider.

2.2.2 Actor Boundary

As mentioned before, intentions of actors are considered within i* models. These are illustrated by intentional elements that are depicted within the boundary of an actor. Everything within an actor's boundary is in her control, while everything outside boundaries is in dependence with other actors. Usually, actor boundaries are depicted as circles (see Figure 3). However, if there are space constraints, other shapes such as rectangles can also be used [38].

2.2.3 Intentional Elements

Intentional elements depict different types of an actor's intentionality. They are described below.

Goal. Goals describe an actor's desire for a state of the world to be achieved. It specifies what and not how it shall be achieved. Based on objective criteria, it can be evaluated whether a goal is achieved or not.

Softgoal. Similar to the goal, a softgoal also represents an actor's wish about a certain state. However, the criteria when a softgoal is satisfied is not clear-cut. Softgoal satisfaction is rather dependent on the actor's subjective evaluation.

Task. A task represents an activity that describes a behavioral procedure in a particular way. If there are multiple actions necessary to perform a task, it can be further decomposed in sub-tasks (see Section 2.2.5)

Resource. Resources are physical or informational entities. Included in an i* model, it is taken as granted in its existence.

Belief. Different to the other elements, a belief does not represent an intentionality, but a condition that the actor holds to be true. Therefore, it cannot be classified as an intentional element in a strict sense. A belief affects the intentionality of an actor and influences it in its effect.

2.2.4 Dependencies

In addition to actor association links, dependency links also connect two actors with each other. In such a dependency relationship, an actor can either be a dependee or a depender. While the dependee is dependent on someone else, the depender is the one another actors depends on. Dependencies always involve a dependum, which is in fact an intentional element representing the dependence subject between a dependee and depender. The dependum can only be achieved in collaboration of the two. The various types of dependencies are briefly described in the following:

Goal Dependency. Here, the depender depends on the dependee to realize a certain desired state. For the depender, it does not matter how the dependee realizes the goal.

Softgoal Dependency. Softgoal dependencies are nearly the same to goal dependencies. The depender needs the dependee and her know-how for attaining the softgoal. However, the dependee must meet the depender's subjective expectations in order to satisfy the softgoal.

Task Dependency. In task dependencies, the depender needs the dependee to execute a specific activity. The reason for the task is not relevant for the dependee. The dependee has the freedom of action to do everything in order to perform the task within the formulated constraints.

Resource Dependency. Resource dependency symbolizes the need of the depender for a physical or informational entity that the dependee has control about. By the resource dependency, the resource is made available by the dependee for the depender.

2.2.5 Intentional Links

The links described in this section may connect the various intentional elements from Section 2.2.3 with each other. This allows complex intentionalities to be unraveled in a structured form, leading to greater insights.

Means-End Link. This link demonstrates the means with which an end can be achieved. It is used to associate a goal with a task that, by its execution, contributes to achieving the goal. The arrow points from the means to the end.

Decomposition Link. Decomposition links allow to decompose complex tasks into sub-elements like subgoals, softgoals, subtasks or resources. There can be multiple decompositions originating from one task. In addition, one distinguishes between AND-, OR- or XOR-decompositions describing the logical need of sub-elements for accomplishing a task. An AND-decomposition means that all elements originating from the outgoing task must be realized, while the OR-composition allows to freely decide which and how many elements are accomplished. In contrast, a XOR-decomposition forces to decide for one of the decomposed elements.

Contribution Links. Contribution links connect softgoals with other intentional elements. They depict how elements contribute to the satisfaction of a softgoal. These contributions can either be positive or negative. Positive contributions are expressed by *make* and *help* links, while negative contributions are *break* and *hurt* links. An element connected to a softgoal with a make or break link is sufficient enough to completely satisfy or deny a softgoal. An element connected to a softgoal with a help or hurt link has a positive or negative impact on the softgoal, but is not sufficient for satisfying or denying it.

3 Goal Model Creation for TrustSoFt

Extending TrustSoFt to a model-based method by goal modeling has several advantages. The main objective is the identification and subsequent resolution of conflicts between specified requirements or goals. In addition, it guides a stepwise, accurate execution of the method that may provide further insights for the development process. Moreover, the development of user-centred software can be ensured, because goal models shall be created from a user perspective. This means that the intentional elements of actors are specified in the sense of the user. Finally, goal modeling assists the TrustSoFt documentation. Thereby, requirements engineers can retrace and discuss every step of TrustSoFt.

In the following, we introduce how the i* goal modeling framework can be used for TrustSoFt. For this purpose, we extend the i* notation so that it is adapted to the TrustSoFt steps. Then we explain the procedure to apply i* goal modeling within TrustSoFt. In doing so, we refer to the TrustSoFt method flow in Figure 2 and refer to the TrustSoFt steps highlighted in green.

3.1 Extended i* Notation for TrustSoFt

The i* notation is a useful instrument to properly specify trustworthiness goals (3), elicit trustworthiness requirements (8), illustrate trustworthiness facets (6), and, thus, relate the facets to the requirements and goals (7). Thereby, the TrustSoFt components are mapped to the i*notation and the intentional elements. An overview of the modified i* elements is shown in Figure 4. The elements and links to relate them with each other are explained in the following.

Actors are drawn in the same way as proposed by the i* notation. Since TrustSoFt focuses on CMI, it might be useful to include multiple end-users as actors to model interpersonal issues. Since end-users have the same issues regarding the other actors in the model, modeling everything multiple times would be redundant and space consuming. Therefore, we take over the perspective of a "main" user, who is depicted with a black, bold frame (see Figure 4).

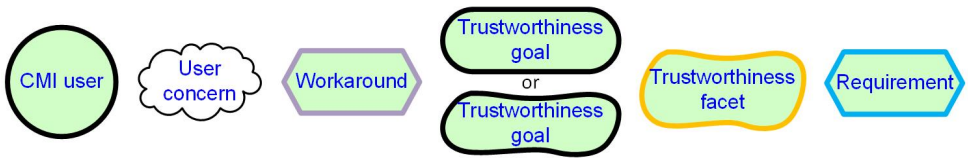


Figure 4: TrustSoFt components mapped on i* notation

User concern and workaround. The concern is depicted as a belief of the i* notation (see Figure 4) within the actor boundary of the user. Concern and belief are similar in that they cannot be proven as true for a specific case. Each goal model includes only one user concern, which shall be countered in the model. A user concern yields in a softgoal and workaround, if known. The softgoal describes what the user wants to achieve in order to resolve the concern. It is connected with the concern by a contribution link. The workaround is illustrated in form of a task with a lilac, bold frame (see Figure 4) and is connected with the concern by a decomposition link. It specifies what the user does in order to reduce the concern without the support of the application.

Trustworthiness goals are depicted within the actor boundary of the software to be developed. They are either represented by a goal or a softgoal from the i* notation. If the evaluation criteria for achieving a trustworthiness goal can be clearly determined, it should be presented as a goal. If the criteria is dependent on the subjective impression of an actor, it should be represented by a softgoal. Trustworthiness goals are highlighted by a black, bold frame (see Figure 4). Each model should include exactly one trustworthiness goal. If it is represented by a goal, it is always connected via a means-end-link to a task/requirement (see next paragraph) that describes how to achieve it. Via the requirement, a trustworthiness goal can be refined into sub-goals (in form of goals or softgoals of the i* notation). If the trustworthiness goal is presented as a softgoal, it can be directly refined into sub-goals by contribution links. Sub-goals always contribute to a trustworthiness goal in order to realize it. If this is not the case, it is another trustworthiness goal that should be depicted in another goal model.

Modeling only one trustworthiness goal in a model has the advantage that the model does not become too large and remains clear. Moreover, different issues are delimited from each other. Thereby, conflicting requirements that originate from different trustworthiness goals can be easier identified.

Trustworthiness requirements are depicted as tasks with a blue, bold frame (see Figure 4) within the actor boundary of the software to be developed. Every task in the software boundary is a requirement, while outside the software boundary, they are "simply" tasks.

A requirement relates to a software goal via a means-end-link, if the software goal is depicted as a goal, or by a contribution link, if the trustworthiness goal is presented as a softgoal. Requirements can also be decomposed into sub-goals, resources or trustworthiness facets that are relevant to realization. The requirements engineer should always consider what trustworthiness facet a requirement can present or represent in the software (see next paragraph). Thereby, she can model the assignment of trustworthiness goals and facets (7, Figure 2), because the facet is indirectly connected to the trustworthiness goal via the requirement by the used links.

Trustworthiness facets are represented by softgoals with a yellow, bold frame (see Figure 4). Both are similar in that softgoals represent a *state to achieve*, while facets describe a *desire to be*. This "desire to be" is also perceived subjectively differently by each actor. In addition, facets differ from other softgoals in that they have an impact on perceived trustworthiness, according to research.

Trustworthiness facets can be illustrated in goal models as dependencies or as intentional elements within actor boundaries. As dependencies, trustworthiness facets express how an actor wishes another actor to be. Within an actor boundary, they describe what the actor should be like or what the actor should represent in order to be perceived as trustworthy. This is partly true for the actor "application", because it does not only present itself, but also the service provider and, in case of social media applications, other end-users [7]. When a facet is represented within the boundary of the application, the requirements engineer must clearly indicate in the annotation of the facet when it is associated with another actor.

After the requirements engineer has selected a trust concern as model subject, relevant trustworthiness facets should be identified and noted outside the goal model (see arrow from (1) to (6)). The identification process is the same as described in Section 2.1. Once the trustworthiness goal has been specified in the goal model, the requirements engineer should consider at a semantic level which of the previously noted trustworthiness facets can be related to and addressed by the trustworthiness goal. The assignment of trustworthiness goal and trustworthiness facets takes place via the trustworthiness requirements in the model (7). Therefore, a facet is connected with a decomposition link to a requirement that in turn is connected with the trustworthiness goal (see paragraph above). The challenge of Step 7 of TrustSoFt is to simultaneously specify requirements that are suitable for achieving the trustworthiness goal and also fit for addressing at least one of the before noted trustworthiness facets. If this is not possible, then the intended facets might not be relevant for the trustworthiness goal. In some cases, the requirements engineer can even identify further relevant facets when reflecting what facet a requirement can present or represent.

The more facets the requirements engineer can include into a goal model, the better is the understanding of the users' trustworthiness assessment. The higher the number of facets within the software boundary, the more facets are presented or represented by the software to be developed. This impacts the degree of how good users can assess the trustworthiness of involved parties via the software.

3.2 Procedure to Use i* Goal Modeling for TrustSoFt

In order to create a goal model in the context of TrustSoFt, the starting point is an identified trust concern (1, green frame). During TrustSoFt execution, multiple trustworthiness goals can be derived from one concern. However, only one trustworthiness goal should be addressed per model at a time for reasons of clarity. Consequently, multiple goal models must be created in order to illustrate one trust concern for the software development process. The number of models for one concern depends on the number of specified trustworthiness goals during TrustSoFt execution.

In order to develop user-centred software, we mainly use the SR model to model and examine the intentional elements of the user and the application. Still, some elements of the SD model are included to represent the context of the trust concern in terms of involved parties and dependencies. It must be noted that including the actor boundary for the service provider as well as its intentional elements might also yield in solution approaches how to handle user concerns. The guideline introduced in this section can be applied for that, as well. However, resulting insights would be at an organizational level. Instead, TrustSoFt draws on a software engineering perspective. For applying goal modeling in the context of TrustSoFt, requirements engineers shall proceed as follows:

1. Draw the actor "user", her boundary, the trust concern that shall be addressed in the model and accompanying intentional elements. Note relevant facets for the trust concern outside the model.

2. Add the SD model components with regard to the user concern. Include the actor "application" as an instantiation of the actor "service provider" and add further relevant actors like other users or

third parties. Illustrate the dependencies among the actors.

3. Continue with the SR model components by adding the actor boundary of the application. Add intentional elements that are necessary to address the user's concern and intentions by the application.

During the specification of intentional elements, words in the annotation of already modeled elements (in natural language) can point to relevant aspects that are not yet addressed by elements in the model. For example, it is not unusual that tasks involve resources that are not included in the model at that time but are important for the context of the trust concern. These relevant keywords should be highlighted in bold within the element's annotation and then added to the model as a new element. This not only supports the requirements engineer in identifying further elements but is also useful in comprehending the creation process of the model after its finalization.

4 Conflict Identification and Resolution by Using Goal Models

After the goal models have been created, they can be used to identify conflicting elements (11, Figure 2). Conflicts always arise between different goal models and not within one. While intentional elements from one model fit together to accomplish a goal, elements from different models do not need to be in line to meet the specific user concern addressed by the respective models. Leaving such conflicting elements unresolved, so that the requirements engineer has to decide on one or the conflicting element for implementation in the software, would mean high opportunity costs. Then, a concern is not addressed in the best possible way leading to a lower risk reduction (4). Therefore, we are interested in resolving conflicts so that every model can be completely realized. In our previous work, we experienced that conflicts often arise from requirements that are too vaguely formulated [6]. Lack of precision can lead to some freedom of action for the service provider, potentially resulting in a less beneficial outcome for the user than originally intended by TrustSoFt. Therefore, we suggest a refinement and precise reformulation of trustworthiness requirements (10) to resolve conflicts. This can be perfectly realized with i* goal modeling, since refinement is supported by the notation using decomposition links, for example.

4.1 General procedure

In the process of conflict identification and resolution, the requirements engineer works with the before developed goal models. The models are evaluated pairwise by checking on the intentional elements each. By considering only individual elements instead of entire element bundles, such as complete task decompositions, the complexity of the model can be broken down. Thus, the evaluation of an element is free from any bias that may be caused by the influence of other elements. In this way, problematic elements can be detected more easily.

During the identification and resolution process, the frame colours of the elements are adjusted. Figure 5 presents an overview of the possible frame colours representing the element status. The chosen intentional elements within the figure are only exemplary. A red frame represents that an element is in conflict with elements of the other goal model. A green frame shows that an element resolves a conflict. These elements must be implemented in the software to be developed. Transparent reddish frames highlight elements as "potentially problematic". This means that a conflict can arise depending on how such an element is realized. An element can be either directly or subsequently identified as potentially problematic. This may happen when conflicts are identified or resolved.

For conflict identification and resolution, it is also important to consider that intentional elements can refer to others. This can occur either through an element's annotation or at a semantic level. If

Figure 5: Examples for conflicting, resolved and potentially problematic intentional elements

an element refers to another one by a mention in its annotation, the words have usually already been highlighted in bold when the goal model was created. In the case of a semantic-level reference, the reference is contained in the context of the issue. Then, it is about the expertise of the requirements engineer to identify the reference. Most often, such a reference is evident in the model by the fact that the referencing element is directly or indirectly linked to the referred element. An element reference is important in so far as a change in the frame color of the referencing element may also imply a change in the frame color of the referred element. An example for that is described in Section 5.1.

4.1.1 Conflict identification

To identify conflicts, the requirements engineer should evaluate one intentional element in terms of others from the other model at a semantic level. As an expert, she knows the effects of an element and can decide, for example, whether a requirement compromises another or interferes with a trustworthiness goal. If this is the case, these elements must be colored bold red to document the conflict. The conflict should be noted and named outside the goal model in a list, because there can be multiple conflicts between two goal models. If the conflicting element references another element or is connected with other elements by an AND, OR, or XOR-decomposition, the requirements engineer must check whether the related element should also be highlighted as conflicting or as potentially problematic.

4.1.2 Conflict resolution

For conflict resolution, the elements marked in red must be refined or reformulated in such a way that the conflict between the models no longer exists. In the case of a conflicting goal - whether a trustworthiness goal or subgoal - the requirements engineer should trace it back to the requirement that realizes the goal. The way a goal is realized often reflects the nature of the conflict. Conflict resolution thus occurs on the requirement level. There are three different possibilities how to deal with a conflicting requirement - a) new requirement, b) new decomposition or c) sub-requirement check.

New requirement. The requirements engineer should investigate, whether it is possible to specify a new, unproblematic requirement that replaces the conflicting one at a semantic level. The new requirement must be marked green. Involved conflicting goals must be changed from red into transparent reddish. If the conflicting requirement cannot be replaced, the requirements engineer should review the decomposition of the conflicting requirement.

New decomposition. If the conflicting requirement is not decomposed to sub-requirements, it should be checked whether a decomposition is possible in such a way that the conflict does not longer exist. This means that a decomposition should include at least one requirement, which refines the conflicting one and is not in conflict with other elements. The decomposed elements that thereby contribute to the resolution must be marked green. As a consequence, the red elements that are involved with the resolved conflict receive a transparent reddish frame. Sometimes it is the case that multiple requirement decompositions are necessary in order to resolve a conflict, leading to decomposition chains. Here, it is up to the judgement and know-how of the requirements engineer how often she tries to perform a decomposition or whether she performs a conflict decision.

Sub-requirement check. In the case that the conflicting requirement is already decomposed into a sub-requirement, the requirements engineer needs to check its frame colour and consider related elements by an OR or XOR-decomposition. Uncoloured sub-requirements could already represent a solution approach that refines the conflicting requirement without being in conflict with the elements of the other goal model. In this case, the requirement has to be highlighted in green. Formerly conflicting elements involved in the conflict must be colored from red to transparent reddish. If a sub-requirement is highlighted red, it may have an uncoloured sub-requirement as an OR- or XOR-alternative. Otherwise, it must be checked whether further decompositions are possible or a new sub-requirement can be specified, as explained in the previous paragraphs.

If eliciting alternative requirements or refining conflicting requirements is not possible, the conflict cannot be resolved. Then, the goal model does not contain any green frames but red ones. This signals the necessity of a conflict decision, which is described in our previous work [6].

5 Application Example: Online Dating

We demonstrate the usefulness of goal modeling for TrustSoFt in the online dating application domain. To do so, we pick up the same example from our previous work [6]. It must be noted that this is only a small example to illustrate a soft conflict. Usually, there is a variety of goals and requirements than can be specified for a concern.

In the example, online dating users have two major trust concerns. One concern is about fake profiles (Concern A) [34]. By fake profiles, other users pretend to be someone else, which is also known as catfishing [35]. The other concern relates to data misuse, which means that other parties, such as the service provider, other end users, or third parties, have access to personal data that they use for their own purposes without any authorization (Concern B) [11, 37].

The previous work identified "Checking user authenticity" as a goal for Concern A (Goal A), which led to Requirement A: "Asking users to disclose personal information". For Concern A, the trustworthiness facets *data-related quality*, *honesty* and *performance* have been identified as relevant for the user [6]. Data-related quality means the quality of data representation [29]. Honesty describes the sharing of truthful information [4]. Performance relates to the actual behaviour, way of social interaction and the use of one's resources [9]. For Concern B, "only using data that are relevant for providing the service" was identified as a goal (Goal B). Therefore, Requirement B states to "notify users about the necessity of the data for the service". This requirement addresses the trustworthiness facet "transparency" [6]. Transparency describes the degree to which information is shared with involved actors [14]. In the previous work, we identified a conflict between Requirement A and Goal B.

For this example, we now use goal modeling as introduced in Section 3 to model the two concerns. The model creation, model meaning, conflict identification and conflict resolution are explained below using Figures 6 and 7. Goal models are dynamic. They evolve during the whole process and have frame colour adjustments, which is why the figures do not completely fit to any time of the explanation. In order to give an example of an initial and final state of a goal model, Figure 6 depicts the goal model after conflict resolution and Figure 7 after conflict identification. For easy explanation of the models, we introduce names for each intentional element, which are also shown in the figures – underlined in the annotation of the elements.

5.1 Example 1: Fake Profiles in Online Dating

As a first step, we draw the *online dating user* as an actor and her boundary. Then, we add *Fake profiles* as a concern (FP-C) which may harm the user's softgoal to interact with "real" people (FP-U-G), but

without denying it (hurt-contribution). In interviews, online dating users reported that they check other social media sites to see if the user they are interacting with exists there, as well, and if the data matches what they know about the person [32]. Therefore, we add this as Workaround FP-W that supports the user in realizing Softgoal FP-U-G (help-contribution) and that is motivated by her Concern FP-C (decomposition). Subsequently, we consider relevant trustworthiness goals for Concern FP-C referring to trust research. As in our previous work, we identify honesty [4], data-related quality [29] and performance [9].

As a next step, we set up the SD model. As relevant actors we additionally identify *other online dating user*, *online dating service provider* and *online dating application* as an instantiation of the service provider (INS-relationship). As this model includes two users, we highlight the one for who we have created the actor boundary with a bold black frame.

Focusing on the dependencies, we realize that the online dating user is dependent on the online dating service provider only having "real" users in its portfolio for matching users with each other (FP-U/SP-1). This is represented by a softgoal dependency, as "real users" lie in eye of the online dating user. Since the online dating user is part of the service provider's portfolio, the service provider is dependent on the user to be honest about her identity. We include the trustworthiness facet "Honesty" the Softgoal Dependency FP-SP/U-2 with a yellow frame. The same dependency is valid between the two online dating users when they interact with each other (see Softgoal Dependency FP-U/U).

After that, we continue with the SR model creation by adding the actor boundary for the online dating application. The softgoal dependency of the user to interact with "real" people (FP-U/A-1) is now connected to the trustworthiness goal specified as *"User authentication"* (FP-TWG, black bold frame). We keep the remaining two trustworthiness facets "data-related quality" and "performance" in mind. The trustworthiness goal can be achieved by the requirement *"Checking on user authenticity"* (FP-Req1, means-end link). Then, Requirement FP-Req1 is decomposed into Requirement FP-Req2 *"Asking users*

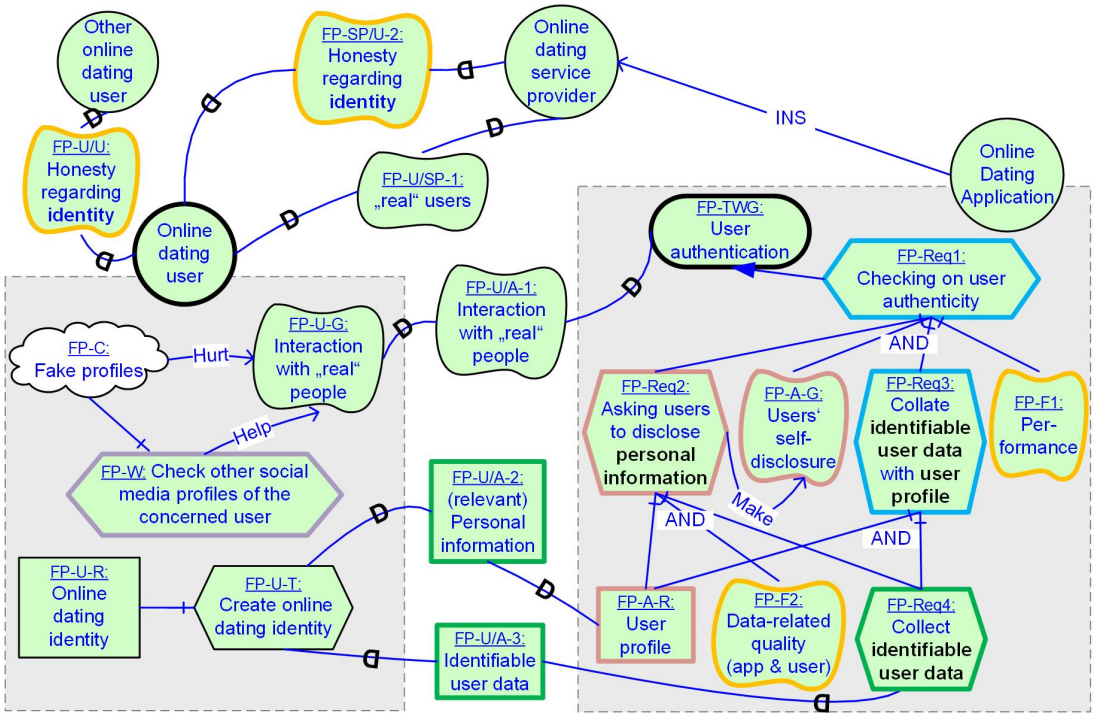


Figure 6: i* goal model for user concern "fake profiles" after conflict resolution

to disclose personal information", the Softgoal FP-A-G *"Users' self-disclosure"* and Requirement FP-Req3 *"Collate identifiable data with user profile"*. Moreover, Requirement FP-Req1 can be addressed to the Trustworthiness Facet FP-F1 *"Performance"* for the online dating application, because the user can evaluate the application's performed effort. In addition, by implementing Requirement FP-Req1, the Softgoal FP-A-G can be satisfied (make-contribution). All decomposed elements must be considered in order to check on user authenticity (FP-TWG, AND-decompositions). While formulating Requirement FP-Req2, we recognize *"personal information"*, *"identifiable user data"* and *"user profile"* as important key words that should be included within the model as separate elements. The keywords are highlighted in bold.

Based on these findings, we decompose Requirement FP-Req2 into the Resource FP-A-R *"user profile"*, the Requirement FP-Req4 *"Collect identifiable user data"* and the Trustworthiness Facet FP-F2 *"data-related quality"*. Data-related quality can be attributed to the application and the user, because the application demands a certain data quality for the authenticity check, while the user provides the relevant data. Resource FP-A-R and Requirement FP-Req4 are also important for Requirement FP-Req3 (decompositions). In order to establish FP-A-R and FP-Req4, an interaction with the online dating user is necessary. Therefore, we create the Resource Dependency FP-U/A-3 from the application to the user with the dependum *"Identifiable user data"*. This dependency connects the task of the online dating user FP-U-T *"Create online dating identity"* with the Requirement FP-Req4. Former research found that social media users present themselves different online compared to their "real" self [3]. Therefore, we decompose Task FP-U-T into Resource FP-U-R *"Online dating identity"*. In order to present herself online, the user passes on *personal information* to the online dating application that is depicted in a user profile (FP-A-R). At the same time, the application depends on the personal information to create a user profile. This is presented by Resource Dependency FP-U/A-2.

5.2 Example 2: Data Misuse in Online Dating

After drawing the online dating user as an actor and including her boundary in the model, *"Data misuse"* is included as Concern DM-C. Data misuse hurts the user's Softgoal DM-U-G in *"being private during app usage"*, which is contrasting Concern DM-U-G (hurt-contribution). As in our previous work, we identify the trustworthiness facet "transparency" as relevant for the user to evaluate the probability of data misuse [14].

Afterwards, the SD model is constructed. For data misuse, relevant actors are *online dating service provider*, *online dating application*, which can be instantiated from the provider (INS relationship), and *third parties*. The service provider and third parties have a data exchange, which is illustrated by the two resource dependencies DM-SP/TP1 and DM-TP/SP-2. The service provider receives a payment for making user data available to third parties. This exchange is a known business model for social media [36] like CMI. However, the online dating user demands a *"privacy-respecting data processing"* from the service provider, which is symbolized by Softgoal Dependency DM-U/SP. Moreover, the user depends on the online dating application concerning a *"private usage"* (Softgoal Dependency DM-U/A) in order to satisfy her Goal DM-U-G.

As a next step, we create the boundary of the online dating application. Softgoal Dependency DM-U/A is connected to Trustworthiness Goal DM-TWG *"User privacy"*. DM-TWG is refined by Softgoal DM-A-G *"Data minimization"* that helps in satisfying DM-TWG (help-contribution). This in turn can be guaranteed by Requirement DM-Req1 *"Only using user data relevant for providing the service"* (make-contribution). Here, the words *"user data relevant"* and *"service"* seem to be critical keywords, because they are indeterminate in their meaning. Therefore, we mark them bold to be included as separate elements within the model. On this basis, we decompose Requirement DM-Req1 into Requirement DM-Req2 *"Specify provided services"* and Resource DM-R *"Relevant user data"*. As it should be tried to

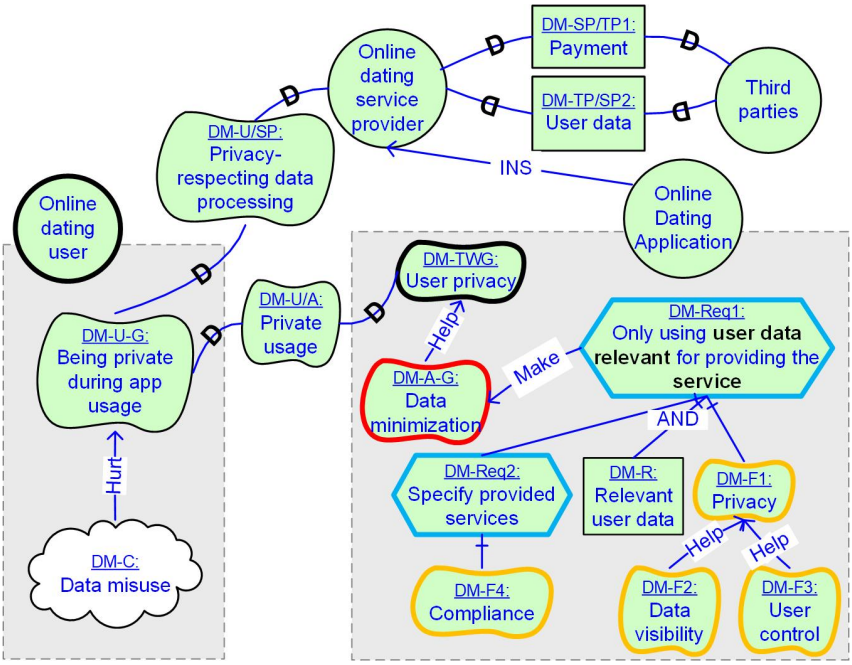


Figure 7: i* goal model for user concern "data misuse" after conflict resolution

relate a requirement with at least one trustworthiness facet (see Section 3.1), we think of the facets that we identified as relevant in the beginning of the goal model creation. The facet "transparency" does not fit to Requirement DM-Req1. Therefore, we consider new facets that can be assessed by the user when Requirement DM-Req1 is implemented in the system. We identify Trustworthiness Facet DM-F1 *"Privacy"* as relevant [29]. Regarding Requirement DM-Req2 and Resource DM-R, the provided services and relevant user data must be defined by the requirements engineer outside of the goal model.

As a next step, we try to relate a trustworthiness facet to Requirement DM-Req2. Again, "transparency" is not suitable. Therefore, we come to the conclusion that it is not appropriate for the subgoal "data minimization" (DM-A-G). Instead, we identify the trustworthiness facet *"Compliance"* (DM-F4) as relevant due to the following reason. Specifying provided services results in a clear understanding what the application offers. Thereby, the application can comply with the expectations of involved actors and with standards, regulations or laws that might be available for such services or accompanying techniques. This can result in "Compliance" [29]. Moreover, Trustworthiness Facet DM-F1 can be refined. Privacy can be related to "systems that provide users with the means to have visibility and control on how the users' private information is used" [29]. Therefore, we create *"Data visibility"* and *"User control"* as Trustworthiness Facets DM-F3 and DM-F4 that support the realization of DM-F1 (help-contributions) and should be considered during the development of the online dating application.

5.3 Identifying and Resolving the Example Conflict

We now check the two goal models for conflicts by collating the individual intentional elements at a semantic level. We identify Softgoals FP-A-G "User's self-disclosure" and DM-A-G "Data minimization" as contradictory depending on how they are realized (Conflict 1). Both softgoals are marked red. Moreover, Requirement FP-Req2 interferes with Requirement DM-Req1 and Resource DM-R, because it is not clear whether the personal information requested is relevant to the provision of services. Therefore, Requirement FP-Req2 receives a red frame (Conflict 2). As Requirement FP-Req2 refers to Resource

Dependency FP-U/A-2 by the bold marked keywords "personal information", we check whether FP-U/A-2 interferes with Requirement DM-Req1 and Resource DM-R, too. Evaluating the situation, we validate certain personal information as relevant user data (DM-R) for providing the service (DM-Req2) to which we also count user authentication (FP-TWG). Therefore, we change the wording of FP-U/A-2 to *"(relevant) personal information"*. As personal information is contained in a user profile, we mark Resource FP-A-R as potentially problematic, depending on what personal information is used for the profile.

For Resource Dependency FP-U/A-3, we do the same evaluation. As identifiable user data (FP-U/A-3) can be counted as relevant user data (DM-R) for providing the service (DM-Req2) user authentication (FP-TWG), it does not conflict DM-Req2 and DM-R. Thus, the dependee Requirement FP-Req4 is also not conflicting.

After we identified the conflicting elements, we now want to resolve Conflict 1 and 2. For Conflict 1, we take a look at Goal FP-A-G and DM-A-G. As explained in Section 4.1.2, we trace the goals back to the requirement they are linked to. Goal FP-A-G is connected to Requirement FP-Req2. FP-Req2 is marked red and is decomposed. Thus, we check its Sub-Requirement FP-Req4. FP-R4 is not marked red. As explained in the previous paragraph, FP-Req4 is conflict-free, because identifiable user data is classified as relevant user data (DM-R) and user authentication as a provided service by the application. Therefore, we assess Conflict 1 as resolved and highlight FP-U/A-3 and FP-Req4 in green. Based on that and complying with Section 4.1.2, Goal FP-A-G is changed into transparent reddish. In addition, by resolving Conflit 1, we also resolved Conflict 2, as the conflicting Requirements FP-Req2 and DM-Req1 were included in Conflict 1.

Nonetheless, we check on the goal model for data misuse in order to update it concerning the conflict resolution. The former conflicting Goal DM-A-G is turned from red to transparent reddish. Then, it is traced back to its requirement. Requirement DM-Req1 is not marked red. Due to the previous conflict analysis and resolution, we are sure that DM-Req1 is free of conflicts. Therefore, we highlight it in green.

5.4 Example Deviations Concerning Previous and this Work

The application of i* goal modelling to the two examples yields the conclusion that we specified and formulated goals and requirements incorrectly in our previous work [6] in which we did not make use of i* goal modeling. In the following, we compare the trustworthiness goals and requirements from our example of the previous (see Section 5, Goal A and B, Requirement B) to the ones of the goal models from this work (see Sections 5.1 and 5.2, Goal FP-TWG and DM-TWG, Requirement FP-Req1, DM,Req1 and DM-Req2).

As we compared the examples of this work with the ones of the former work [6], we recognized that Goal A and B are formulated like requirements. Instead of stating what the system should achieve, they express what the system should do. In this work, we corrected the trustworthiness goals by specifying Goal FP-TWG and DM-TWG. Goal A and B are instead included as Requirement FP-Req1 and DM-Req1 in the goal models. Moreover, Requirement B is not significant to realize Goal B and Requirement DM-Req1. It covers different important aspects, but it is not tailored to the issue. Therefore, we specified Requirement DM-Req2 as a new, fine-tuned requirement in this work. Based on these findings, we conclude that i* goal modeling impels requirements engineer to be correct and more precise in the annotation of trustworthiness goals and requirements.

6 Related Work

TrustSoFt is a method that aims for requirements elicitation and improvement. While requirements elicitation is about finding new requirements, requirements improvement denotes a check of the initial requirements for errors, inconsistencies or the consideration of critical properties [20]. In the following, we evaluate other goal modeling approaches that Horkoff and Yu [20] identified as significant for these purposes and that are also similar to TrustSoFt.

For requirements elicitation, Horkoff and Yu refer to Asnar and Giogini [20]. They extended Tropos goal modeling [8] for modeling risks in order to elicit countermeasures [2]. Thereby, they aim to satisfy user goals at an organizational level by realizing only goals with acceptable risks and costs. The work of Asnar and Giogini is of interest to us in that it is similar to our previous work on conflict decisions considering risk. However, TrustSoFt does not focus on risk modeling at the organizational level as it is agent-oriented and focuses on trust issues from a software development perspective. Moreover, it resolves conflicts between different goals and their requirements, rather than finding the one requirement that can achieve a goal with the lowest risk and cost. Nevertheless, the work of Asnar and Giorgini could inspire us on how to combine our previous work [6] with this one. This would allow us to use goal modeling to additionally consider risks and perform conflict decisions.

Horkoff and Yu identified the Annotated Goal-Oriented Requirements Analysis (AGORA) as an appropriate approach for requirements improvement [20]. AGORA supports requirements engineers in identifying conflicting goals, choosing or adopting goals from a set of goals to resolve a conflict, and analyzing the impact of requirements changes [22]. For that reason, Kaiya et al. included contribution values and preference matrices within goal models. Contribution values represent the degree of the contribution of a sub-goal to achieving the main goal. Concerning preference matrices, they represent how much stakeholders relevant to the issue being modeled prefer a goal. The main differences between AGORA and TrustSoFt are that AGORA is software-centred and that it illustrates a “state of the world” by modeling software states, actions and conditions as goals. In contrast, TrustSoFt is user-centred and agent-oriented by providing actor insights. TrustSoFt distinguishes between different intentional elements in the goal models. Consequently, AGORA and TrustSoFt represent different perspectives in requirements engineering, what makes them hardly comparable. Yet, AGORA inspires us for new useful goal modeling elements that might be picked up for TrustSoFt in future work. Modified contribution values could be used as an evaluation criterion for requirements when they are implemented and tested after the requirements elicitation phase. Preference matrices are not relevant for TrustSoFt as we only consider goals and requirements beneficial for the user. However, including risk matrices of requirements (cf. Step 9 of Figure 2) in goal models could be a first step to combine our previous work [6] with this one.

Another interesting approach is the one of Gans et al. [17]. They introduce a multi-perspective modeling approach that focuses on team-oriented business process analysis in the context of social media. Thereby, they modeled different stakeholder viewpoints under the perspective of trust in individuals, confidence in the network and distrust. In their multi-perspective modeling approach, i* goal modeling is one of the used modeling techniques for their analysis. There, they examine the vulnerability of stakeholders by the SD model. Furthermore, they consider the temporal sequence of actions in the SR model, which is expressed in the i* goal models by new elements displaying the pre- and postconditions of tasks. These are related to the viewpoints and expectations of stakeholders. Overall, Gans et al. relate trust issues of social media stakeholders to their monitoring of social network rules and requirements. They analyze how they impact each other. This is in contrast to our work. We elicit requirements for software that supports users in their trustworthiness assessment of involved parties (e.g. in the context of social media).

7 Discussion

This work extends the requirements elicitation method TrustSoFt to a model-based approach for requirements improvement. Thereby this work contrasts with the previous work [6], in which risk is used as a determinant in TrustSoFt for conflict decisions. Instead, this work incorporates i* goal modeling in TrustSoFt for identifying and resolving conflicts to avert conflict decisions. In addition, i* goal modeling seeks to improve the feasibility of TrustSoFt.

Overall, we can confirm that by i* goal modeling, requirements engineers can learn more about the user and her trust concerns, which facilitates the development of a user-centred application. It is particularly helpful to first take the user's perspective, then model the dependencies of the actors involved, and finally determine how the system can serve the user. Especially regarding the last point, i* goal modeling supports a precise execution of TrustSoFt. Moreover, it supports a correct annotation of trustworthiness goals and requirements.

Another notable support provided by the modified i* framework is the instruction to highlight keywords in bold within the annotation of goals and requirements that refer to relevant aspects which have not been modeled at that point. The bold keywords prevent missing relevant elements and thus facilitate the identification of further elements. In addition, bold keywords within the annotation of an element could imply that the element is vaguely formulated. As a consequence, the element requires higher precision, which we put into practice by element refinement (e.g. decomposing requirements). In doing so, solution approaches can added to the goal models even before conflicts are identified. This leads to fast and effortless conflict resolution.

Another advantage is that i* goal modeling supports the refinement of requirements, which enables a better fine-tuning of them. This allows to address the trustworthiness facets more precisely. Sometimes it is even possible to address additional facets that are not considered in the beginning. Thereby, end-users can be better supported in their trustworthiness assessment of the application, the service provider and other end-users. For this reason, we plan to take up i* goal models as a distinct basis for the final step of TrustSoFt – the elicitation of trust-related software features. In future work, we aim to evaluate a structured elicitation of software features that realizes trustworthiness requirements and facets at the same time. As a brief outlook, we see the goal models as an input for two elicitation strategies: First, for reviewing pattern collections (cf. [13]) of existing software features that can be related to requirements and facets at the same time. Second, for brainstorming sessions by software engineers (cf. [40] as a creativity approach to develop new features that fit specific cases.

Brainstorming sessions are also a solution approach for a major limitation of i* goal modeling. We have recognized that the creation of i* goal models is significantly influenced by the requirements engineer who performs it. This is especially true for modeling dependencies or annotating goals and requirements. Such subjective influences shape the entire model leading to different models despite the same input. Therefore, we recommend creating goal models in a team of requirements engineers to reduce this subjective bias. In addition, it is useful to create multiple goal models for the same user concern and trustworthiness goal to attain a variety of solution approaches based on different reasoning.

8 Conclusion

This work introduces the method for eliciting trust-related software features (TrustSoFt) as a model-based approach using i* goal modeling. For this reason, the i* notation is adapted to TrustSoFt. Thereby, requirements engineers are supported in precisely executing TrustSoFt step by step and correctly annotating trustworthiness goals as well as requirements for addressing trust concerns of users. As a result, user-centred software can be developed. Moreover, the model-based extension enables requirements en-

gineers in identifying and resolving conflicting trustworthiness goals and requirements. Instead, resolved conflicts allow requirements engineers to implement all formerly conflicting elements in the software being developed. In this way, the trust concerns of users are addressed in the best possible way. For future work, we consider goal models resulting from TrustSoFt as a good basis to perform the last step of the method, which is the elicitation of trust-related software features.

References

[1] S. Alter. Theory of workarounds. *Communications of the Association for Information Systems*, 34(55):1041–1066, March 2014.

[2] Y. Asnar and P. Giorgini. Modelling risk and identifying countermeasure in organizations. In *Proc. of the First International Workshop on Critical Information Infrastructures Security (CRITIS'06), Samos, Greece*, volume 4347 of *Lecture Notes in Computer Science*, pages 55–66. Springer, Berlin, Heidelberg, September 2006.

[3] A. Audrezet, G. De Kerviler, and J. G. Moulard. Authenticity under threat: When social media influencers need to go beyond self-presentation. *Journal of Business Research*, 117, July 2018.

[4] G. Bellucci and S. Q. Park. Honesty biases trustworthiness impressions. *Journal of Experimental Psychology: General*, 149(8):1567—-1586, August 2020.

[5] A. Borchert, N. E. Díaz Ferreyra, and M. Heisel. A conceptual method for eliciting trust-related software features for computer-mediated introduction. In *Proc. of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE'20), Online Streaming*, pages 269–280. Springer, May 2020.

[6] A. Borchert, N. E. Díaz Ferreyra, and M. Heisel. Balancing trust and privacy in computer-mediated introduction: featuring risk as a determinant for trustworthiness requirements elicitation. In *Proc. of the 15th International Conference on Availability, Reliability and Security (ARES'20),Virtual Event, Ireland*, pages 1–10. ACM, August 2020.

[7] A. Borchert, N. E. Díaz Ferreyra, and M. Heisel. Building trustworthiness in computer-mediated introduction: A facet-oriented framework. In *Proc. of the 2020 International Conference on Social Media and Society (SMSociety'20), Toronto, Ontario, Canada*, pages 39–46. ACM, July 2020.

[8] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, May 2004.

[9] J. A. Colquitt, B. A. Scott, and J. A. LePine. Trust, trustworthiness, and trust propensity: A meta-analytic test of their unique relationships with risk taking and job performance. *Journal of applied psychology*, 92(4):909–927, July 2007.

[10] L. L. Constantine. Trusted interaction: User control and system responsibilities in interaction design for information systems. In *Proc. of the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06), Luxembourg, Luxembourg*, volume 4001 of *Lecture Notes in Computer Science*, pages 20–30. Springer, Berlin, Heidelberg, June 2006.

[11] D. Couch, P. Liamputtong, and M. Pitts. What are the real and perceived risks and dangers of online dating? perspectives from online daters: Health risks in the media. *Health, Risk & Society*, 14(7-8):697–714, October 2012.

[12] M. J. Culnan and R. J. Bies. Consumer privacy: Balancing economic and justice considerations. *Journal of social issues*, 59(2):323–342, July 2003.

[13] J. Deng, E. Kemp, and E. G. Todd. Managing ui pattern collections. In *Proc. of the 6th ACM SIGCHI New Zealand Chapter's International Conference on Computer-Human Interaction: Making CHI Natural (CHINZ'05), Auckland, New Zealand*, pages 31–38. ACM, July 2005.

[14] M. W. DiStaso and D. S. Bortree. Multi-method analysis of transparency in social media practices: Survey, interviews and content analysis. *Public Relations Review*, 38(3):511–514, September 2012.

[15] I. O. for Standardization. Risk management - Principles and guidelines. Standard., 2018. `https://www.iso.org/obp/ui/#iso:std:iso:31000:ed-2:v1:en` [Online; accessed on March 22, 2021].

[16] X. Franch, L. López, C. Cares, and D. Colomer. The i* framework for goal-oriented modeling. In *Domain-specific conceptual modeling*, pages 485–506. Springer, Cham, July 2016.

[17] G. Gans, M. Jarke, S. Kethers, and G. Lakemeyer. Continuous requirements management for organisation networks: a (dis) trust-based approach. *Requirements Engineering*, 8(1):4–22, February 2003.

[18] L. Hallam, M. Walrave, and C. J. De Backer. Information disclosure, trust and health risks in online dating. In *Sexting*, Palgrave Studies in Cyberpsychology, pages 19–38. Palgrave Macmillan, Cham, March 2018.

[19] B. C. Holtz. From first impression to fairness perception: Investigating the impact of initial trustworthiness beliefs. *Personnel Psychology*, 68(3):499–546, September 2015.

[20] J. Horkoff and E. Yu. Analyzing goal models: Different approaches and how to choose among them. In *Proc. of the 2011 ACM Symposium on Applied Computing (SAC'11), TaiChung, Taiwan*, pages 675–682. ACM, March 2011.

[21] A. Hunter and B. Nuseibeh. Managing inconsistent specifications: Reasoning, analysis, and action. *ACM Transactions on Software Engineering and Methodology*, 7(4):335–367, October 1998.

[22] H. Kaiya, H. Horai, and M. Saeki. Agora: Attributed goal-oriented requirements analysis method. In *Proc.of the 2002 IEEE Joint International Conference on Requirements Engineering (ICRE'02) , Essen, Germany*, pages 13–22. IEEE, September 2002.

[23] E. Kavakli and P. Loucopoulos. Goal modeling in requirements engineering: Analysis and critique of current methods. In *Information modeling methods and methodologies: Advanced topics in database research*, pages 102–124. IGI Global, 2005.

[24] D. Kipnis. Trust and technology. *Trust in organizations: Frontiers of theory and research*, 39:50, 1996.

[25] H. Krasnova, O. Günther, S. Spiekermann, and K. Koroleva. Privacy concerns and identity in online social networks. *Identity in the Information Society*, 2(1):39–63, December 2009.

[26] M. Maguire, J. Kirakowski, and N. Vereker. Respect: User centred requirements handbook, July 1998. `https://repository.lboro.ac.uk/articles/report/RESPECT_User_centred_requirements_handbook/9354023` [Online; accessed on March 22, 2021].

[27] N. K. Malhotra, S. S. Kim, and J. Agarwal. Internet users' information privacy concerns (iuipc): The construct, the scale, and a causal model. *Information systems research*, 15(4):336–355, December 2004.

[28] N. G. Mohammadi, T. Bandyszak, S. Paulus, P. H. Meland, T. Weyer, and K. Pohl. Extending software development methodologies to support trustworthiness-by-design. In *Proc. of the CAiSE 2015 Forum co-located with 27th International Conference on Advanced Information Systems Engineering (CAiSE'15), Stockholm, Sweden*, pages 213–220, June 2015.

[29] N. G. Mohammadi, S. Paulus, M. Bishr, A. Metzger, H. Koennecke, S. Hartenstein, and K. Pohl. An analysis of software quality attributes and their contribution to trustworthiness. In *Proc. of the 3rd International Conference on Cloud Computing and Services Science (CloudSecGov'13), Aachen, Germany*, pages 542–552. SciTePress, May 2013.

[30] J. Mylopoulos, L. Chung, and E. Yu. From object-oriented to goal-oriented requirements analysis. *Communications of the ACM*, 42(1):31–37, January 1999.

[31] B. Nuseibeh and S. Easterbrook. Requirements engineering: a roadmap. In *Proc. of the 2000 Conference on the Future of Software Engineering (ICS'00), Limerick, Ireland*, pages 35–46. ACM, May 2000.

[32] B. Obada-Obieh, S. Chiasson, and A. Somayaji. "don't break my heart!": User security strategies for online dating. In *Proc. of the Usable Security Mini Conference (USEC'17), San Diego, California, USA*. NDSS, February 2017.

[33] B. Obada-Obieh and A. Somayaji. Can i believe you? establishing trust in computer mediated introductions. In *Proc. of the 2017 New Security Paradigms Workshop (NSPW'17), Santa Cruz, California, USA*, pages 94–106. ACM, October 2017.

[34] A. Rege. What's love got to do with it? exploring online dating scams and identity fraud. *International Journal of Cyber Criminology*, 3(2), July 2009.

[35] M. Simmons and J. S. Lee. Catfishing: A look into online dating and impersonation. In *Proc. of the International Conference on Human-Computer Interaction (HCII'20), Copenhagen, Denmark*, Lecture Notes in Computer Science, pages 349–358. Springer, Cham., July 2020.

[36] A. Sorescu. Data-driven business model innovation. *Journal of Product Innovation Management*, 34(5):691–696, September 2017.

[37] M.-S. Stoian et al. 'friends, dates and everything in between': Tinder as a mediating technology. *Journal of Comparative Research in Anthropology and Sociology*, 10(01):49–57, October 2019.

[38] R. A. University. i Star Wiki, 2011. `http://istar.rwth-aachen.de/tiki-view_articles.php` [Online; accessed on March 22, 2021].

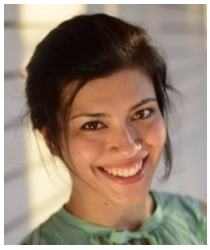
[39] A. VanLamsweerde. Goal-oriented requirements engineering: A guided tour. In *Proc. of the 5th IEEE International Symposium on Requirements Engineering (ISRE'01), Toronto, Ontario, Canada*, pages 249–262. IEEE, August 2001.

[40] P. Weichbroth. Facing the brainstorming theory. a case of requirements elicitation. *Studia Ekonomiczne*, 296:151–162, 2016.

[41] E. Yu. Modeling strategic relationships for process reengineering. *Social Modeling for Requirements Engineering*, 11(2011):66–87, May 2011.

[42] K. Yue. What does it mean to say that a specification is complete? In *Proc. of the 4th International Workshop on Software Specification and Design (IWSSD-4'87), Monterey, California, USA*. IEEE, April 1987.

Author Biography



Angela Borchert Angela Borchert received the B.Sc. and M.Sc. in Applied Cognitive and Media Studies from University Duisburg-Essen. There, she is currently a doctoral candidate at the department of software engineering. Moreover, she is part of the research training group "User-Centred Social Media". Her research interests have an interdisciplinary focus based on the disciplines of computer science and psychology. She is particularly interested in trust research, human-human interaction online and user-centred software development. The application field of her research is especially online dating and sharing economy.

Maritta Heisel is a full professor for software engineering at the University of Duisburg-Essen, Germany. Her research interests include software development methodologies and pattern-based software engineering, with a focus on requirements engineering and software design. She is particularly interested in methods for establishing different quality properties of software, including safety, security, and privacy. She is a principal investigator in the research training group on user-centered social media of the University Duisburg-Essen. As a member of the board of the "Essener Kolleg für Geschlechterforschung" (Essen College for Gender Studies), she is also interested in investigating gender differences in computer science.