Extending Problem Frames to Deal with Stakeholder Problems
An Agent- and Goal-Oriented Approach
Sam Supakkul and Lawrence Chung
The University of Texas at Dallas
ssupakkul@ieee.org, chung@utdallas.edu

ABSTRACT
The Problem Frames approach captures problems as seen by developers, but not by stakeholders. This paper presents a framework that extends the problem diagram of the Problem Frames approach to represent stakeholder problems using “soft-problem”, a notion referring to an undesirable situation that negatively affects stakeholder goals and may have less clear-cut resolution criteria. A soft-problem may be refined to more specific sub-problems and root causes, which are then traced to corresponding solutions in a diagram called Problem Interdependency Graph. The framework has been applied to the 1992 London ambulance case study, which shows that the framework helps to more precisely represent the stakeholder problems suffered by the existing system (“as-is”) together with the solutions to be provided by the new system (“to-be”) to such problems. The study also shows that the framework helps to determine whether the problems have been sufficiently addressed and, if not, why.

Keywords
problem frames, agent-oriented, goal-oriented, soft-problem

1. INTRODUCTION
The Problem Frames approach [10] represents problems as seen by developers, but not by stakeholders. It has been said that “a problem unstated is a problem unsolved” [19]. Yet these stakeholder problems are not explicitly represented in the Problem Frames approach. How can requirements engineers represent and ensure that stakeholder problems are sufficiently addressed and, if not, why.

In the real world, a problem often refers to an undesirable situation that makes it difficult to achieve a goal [25, 4], which is clearly illustrated by the famous phrase “Houston, we’ve had a problem” when the Apollo 13 crew alerted the mission control in Houston to a problem related to a component losing its power-supply that could threaten the crew’s safety [17]. Operational or system-level problems such as that of the Apollo 13 have clear-cut resolution criteria whether the component has recovered from the power outage based on its precise electrical specification. However, stakeholder-level problems may have less clear-cut or no resolution criteria, as suggested by “You can’t remove all the safety hazards from your life, but you can reduce them” [23]. How can requirements engineers deal with this kind of “soft” problems that have no hard resolution criteria?

In many cases, a situation could be both a boon and a bane. For example, an approaching storm would be welcomed by farmers suffering from a long drought as a desirable phenomenon, but the same phenomenon would be a problem for people wishing for a pleasant outdoor picnic. For this reason, a stakeholder problem should be defined in the context of the stakeholder goals being affected.

A number of approaches (Fishbone diagram [9], Fault Tree Analysis [12], the i* framework [15], Abuse Frames [13], Mis-use Cases [21], Anti-goal [24]) have been used to deal with problems. However, none provides a comprehensive framework for dealing with the above questions, and represents stakeholder problems in a manner that is consistent with the real world notion of problems as discussed above.

This paper presents a framework that extends the problem diagram of the Problem Frames approach, with the notion of “soft-problems” to represent stakeholder problems that jeopardize some stakeholder (agent) goals. It also deals with the traceability of the soft-problems to the solutions represented by requirements in the problem diagram. The agent and goal representation are adapted from the i*/Tropos framework [26, 22]. Soft-problems can be decomposed into more specific sub-problems, which are used to identify root causes and explore alternative solutions in a diagram called Problem Interdependency Graph using the qualitative analysis framework adopted from the NFR Framework [3].

This framework has been applied to the 1992 London ambulance system case study [18], as an experimental study, which shows that the framework helps to more precisely represent the stakeholder problems suffered by the existing manual dispatch system (“as-is”), and subsequently traced to the solutions provided by the new computer-aided dispatch (CAD) system (“to-be”) to such problems. This study also shows that the framework helps determine whether the problems have been sufficiently addressed and, if not, why.

The rest of the paper is organized as follows: Sec. 2 gives a brief review of the adopted frameworks; Sec. 3 presents the proposed framework; Sec. 4 discusses our empirical study, while Sec. 5 discusses related work and observations; finally, Sec. 6 summarizes the paper and directions for future work.
2. ADOPTED FRAMEWORKS

This framework adopts, and extends, the Problem Frames approach [10], the i*/Tropos framework [26, 22], and the NFR Framework [3].

2.1 The Problem Frames Approach

The Reference Model for requirements and specifications [7] defines requirements as customer needs in terms of the effects the system has on the environment, and specifications as effects from the system point of view. The Problem Frames approach [10] captures both these aspects in relation to relevant world and system domains, making it a desirable method for requirements engineering.

The Problem Frames approach captures the requirements and specification in a diagram called problem diagram. Figure 1 shows a problem diagram for the London ambulance manual dispatch (adapted from [8]) where DispatchAmbulance is a requirement; Calls, Resources, and Incidents are two given domains and one designed domain respectively; and ManualDispatch is a machine domain representing a system or a part of a system that implements the requirement. A given domain represents a part of world, while a designed domain is the physical representation of some information [10]. The environmental effects are represented by the referenced phenomena e, f, and g that are associated with the refer references (denoted by a directed dash-line) and the constrain references (denoted by a directed dash-line) relationships between the requirement and relevant given and designed domains. The corresponding system effects are represented by shared phenomena a, b, c, and d that are associated with the interfaces between the given/designed domains and the machine domain. A referenced phenomenon (e.g. e) is mapped to one or more implementable shared phenomena (e.g. a).

The Problem Frames approach regards the machine domain as a computer system (or a part of it) [10], but it has also been used to represent an organizational level process or system [2]. This paper adopts both notions to represent the 1992 CAD computer system and the manual dispatch (system/process).

2.2 Agent- and Goal-Oriented Approaches

A goal is a stakeholder need to be achieved. The statement “Citizen wants his/her safety” is represented using the i*/Tropos framework as shown in Fig. 2.a along with our adaptation for more explicit representation of the “want” and “topic” relationships. A goal may be either a hardgoal (e.g. ambulance dispatched) or a softgoal (e.g. safety and security) depending on whether it has clear-cut achievement criteria. In this paper, we are interested in softgoals as they are usually the ultimate goals of an agent as any hardgoal is generally traceable to a softgoal (e.g. ambulance dispatched is to achieve public safety).

A softgoal is defined in terms of Type and Topic, and therefore named using “Type/Topic” convention [3] (e.g. Safety /Citizen) where Type is a non-functional concern while Topic defines the context for the Type. Put together, Type and Topic represent a world phenomenon. There are three kinds of softgoals: NFR softgoals, operationalizing softgoals, and claim softgoals for representing objectives, solutions, and arguments in a Softgoal Interdependency Graph (SIG).

A softgoal may be refined using an And/Or-decomposition (denoted by a single/double arc) to more specific sub-goals as shown in Fig. 2.b where Safety/Citizen is And-decomposed on the Topic to Safety/Life and Safety/Property. Goal decomposition may be repeated on the Type or Topic until each leaf-level sub-goal is detailed enough to be identified with alternative solutions. For example, Safety/Life is identified with alternative solutions, including Testing [Products] and EmergencyCare [SeriousInjury/Sickness]. Each goal is related to a solution by a contribution to indicate how well the solution contributes to the goal achievement where Make (denoted by +++) indicates a “good enough” solution while Help (denoted by +) indicates a somewhat good solution. A solution may also have some positive or negative side-effects toward other softgoals, such as those contributions denoted by a dash-line in Fig. 2.b.

The solutions that maximize the positives and minimize the negatives are chosen and labeled with a check mark to denote being satisfied (i.e. sufficiently satisfied). For example, EmergencyRoomTreatment and InPatientTreatment are both solutions for EmergencyCare, but the former is more desirable and chosen for its positive side-effect toward the Timeliness softgoal. The labels are propagated upward the SIG based on the type of the labels and contributions. For example, the satisfied label (check) of the Ambulance-Service softgoal is propagated to the satisfied label of the PreHospitalCare softgoal over the Make (+++) contribution. The label evaluation procedure defines rules for propagating different types of labels and contributions.

Figure 1: A Problem Diagram for the London Ambulance Manual Dispatch

Figure 2: (a) Agent and Wanted Goal Representation, (b) A Softgoal Interdependency Graph for Citizen Safety
3. A FRAMEWORK FOR DEALING WITH STAKEHOLDER PROBLEMS

Using this framework, stakeholder problems are addressed in two steps. First, they are explicitly represented, and then are analyzed for root causes and solutions.

3.1 Representing Stakeholder Problems

A stakeholder problem is defined as an undesirable situation that negatively affects one or more stakeholder (soft)goals. There are two kinds of soft-problems: undesirable situations and undesirable mechanisms. An undesirable situation is an undesirable world phenomenon while an undesirable mechanism refers to an operation or technique that leads to the undesirable situation, or contributes to another undesirable mechanism. A more formal definition for soft-problem is:

\[
\text{SoftProblem} = \langle \text{Phenomenon}, \text{Impact} \rangle
\]

Phenomenon = \langle \text{SomeMinus}, \text{Softgoal}, \text{Stakeholder} \rangle

SomeMinus = \text{Break} \mid \text{Hurt}

Softgoal = \langle \text{Phenomenon} \rangle

As an example, the statement “Ineffective manual ambulance dispatch is an undesirable situation that breaks the timeliness of emergency care goal that is wanted by the patient” can be expressed as:

\[
\text{IneffManDisp} = \langle \text{Ineffectiveness,ManualDispatch}, \langle \text{Break,Timeliness,EmergencyCare}, \text{Patient} \rangle \rangle
\]

Alternatively, soft-problems can be depicted in a problem diagram as shown in Fig. 3, which shows that Ineffectiveness[ManualDispatch] undesirable situation (denoted by a thin octagon) breaks (-) the Timeliness[EmergencyCare] NFR softgoal that is wanted by the Patient stakeholder. The soft-problem is realized by the ManualDispatch machine domain that represents the “as-is” system in the London ambulance case study. The soft-problem is sufficiently addressed (Break/+) by ComputerAidedDispatch operationalizing softgoal, a goal representing the solution to be realized by the 1992CAD (“to-be”) system. For brevity, this diagram is an excerpt of a more complete problem diagram from the empirical study in Fig. 6.

The syntax of the extended problem diagram is defined in the meta-model shown in Fig. 4. This meta-model adopts, integrates, and extends the formal definitions and meta-models of the adopted frameworks [11, 22, 16]. The meta-model also supports the notion of adversary/hostile agents that may want to achieve some of the problems; e.g., a hacker wants a bank web site to be compromised.

Figure 3: Representing Stakeholder Problems in an Extended Problem Diagram

Figure 5: Analyzing Stakeholder Problems in a Problem Interdependency Graph

It is likely that the “to-be” and “as-is” systems would share some common or similar domains. For example, in Fig. 3, the 1992CAD is associated with CADResources, a specialized domain of the Resources domain. A specialized domain is a domain that inherits properties from its generalized domain, and may introduce additional or specialized properties. In Fig. 3, CADResources inherits resources such as ambulances used by the ManualDispatch, and adds additional resources (e.g., ambulance tracking devices) that are required by the 1992CAD system.

The semantics of the domain property inheritance is defined by a domain IS-A constraint (adapted from [6]) where \( \text{isa}(D_1, D_2) \) is true if \( D_1 \) is a specialized domain of \( D_2 \), and \( \text{propertyof}(P_1, D_1) \) is true if \( P_1 \) is a property of \( D_1 \).

\[
\text{isa}(D_1, D_2) \land \text{propertyof}(P_1, D_1) \leftrightarrow \text{propertyof}(P_1, D_2) \land \text{isa}(P_1, P_2)
\]

As an example, the semantics of the inheritance of ambulances by the CADResources domain is:

\[
\text{isa}(\text{CADResources,Resources}) \land \text{propertyof}(\text{Ambulance,Resources}) \Rightarrow \text{propertyof}(\text{Ambulance,CADResources}) \land \text{isa}(\text{Ambulance,Ambulance})
\]

3.2 Analyzing Stakeholder Problems

A soft-problem is analyzed in an iterative and interleaving four-step process of problem refinement, root cause analysis, solution exploration, and problem resolution analysis.

Step 1: Problem Refinement. A soft-problem may be refined to more specific sub-problems using an And-/Or-decomposition. For example, in Fig. 5, Ineffectiveness[ManualDispatch] undesirable situation is Or-decomposed to four sub-problems, including Ineffectiveness[Communication]. In turn, it is further Or-decomposed to Ineffectiveness[CommWithinAmbulance] and Ineffectiveness[CommWithinDispatchCenter]. Decomposition may be repeated until each leaf-level sub-problem is identifiable with a causal problem. In general, Or-decomposition is used when any sub-problem could by itself jeopardize the stakeholder goal. On the other hand, And-decomposition is used when all sub-problems must exist to have an impact on the goal.

Step 2: Root Cause Analysis. Each leaf-level sub-problem is identified with possible causal problems. The problem-cause relationship is either a Make(++) or Help(+) contribution to indicate the influence that the cause has on
causing the problem. In Fig. 5, Ineffectiveness [CommWithinDispatchCenter] undesirable situation is determined to be caused by PhysicalMovement [IncidentForm] undesirable mechanism (denoted by a thick octagon), which is attributed with a Make(++) contribution to suggest its strong influence. Each cause may be further decomposed and/or identified with its own causes to form a cause-and-effect chain until root causes are reached.

**Step 3: Solution Exploration.** For each root cause, alternative solutions are identified to address (e.g. to prevent, recover, or deter) the problem. How effective the solution is against the problem is attributed by a Break(–) or a Hurt(-) contribution. In Fig. 5, SoftwareBasedInformation is a solution that effectively addresses (suggested by a break(–)) the PhysicalMovement root cause problem [18]. Because the report [18] did not describe any discarded possible alternative solutions, for illustration purposes, we hypothetically include Faster ConveyorBelt as an example of an alternative solution that could have been considered. SoftwareBasedInformation is considered to be the most desirable solution. It is therefore labeled as satisficed (checked), while Faster ConveyorBelt is labeled as denied (crossed).

**Step 4: Problem Resolution Analysis.** To determine whether the original soft-problem is sufficiently denied is to determine whether its sub-problems are sufficiently denied. In the case of an Or-decomposition, a soft-problem is sufficient denied when all of its sub-problems are considered sufficiently denied. Or more formally (adapted from [16]),

$$\text{denied}(\text{and}(P, \{P_1, P_2, \ldots, P_n\})) =$$

$$\text{denied}(P_1) \land \text{denied}(P_2) \land \ldots \land \text{denied}(P_n) \land$$

$$\text{satisficed}(\text{and}(P, \{P_1, P_2, \ldots, P_n\}))$$

This rule states that problem $P$ that is Or-decomposed to sub-problem $P_1$, $P_2$, ..., $P_n$ is considered denied when $P$ is denied, and $P_2$ is denied, ..., and $P_n$ is denied, while the $P_1$ is denied, while the Or-decomposition is valid. For example, in Fig. 5, Ineffectiveness [Communication] is considered sufficiently denied as both sub-problems from the Or-decomposition are sufficiently denied (crossed).

For an And-decomposition, a problem is sufficiently denied when any of its sub-problems is sufficiently denied. Or more formally (adapted from [16]),

$$\text{denied}(\text{and}(P, \{P_1, P_2, \ldots, P_n\})) =$$

$$\text{denied}(P_1) \lor \text{denied}(P_2) \lor \ldots \lor \text{denied}(P_n) \land$$

$$\text{satisficed}(\text{and}(P, \{P_1, P_2, \ldots, P_n\}))$$

This rule states that problem $P$ that is And-decomposed to sub-problem $P_1$, $P_2$, ..., $P_n$ is considered denied when $P_1$ is denied, or $P_2$ is denied, ..., or $P_n$ is denied, while the And-decomposition is valid.

To determine whether a soft-problem is sufficiently denied by a solution, we adopt the label evaluation procedure from the NFR Framework [3] for evaluating the labels of the soft-goals. For example, in Fig. 5, the satisficed label (check) on SoftwareBasedInformation is reversed to the denied label (crossed) on PhysicalMovement over the Break(–) contribution. The evaluation procedure also provides rules for other types of labels and contributions. The label evaluation is repeated upward the SIG until the top soft-problem is denied. Otherwise, it means that some of its sub-problems may not be sufficiently denied. For example, Fig. 5 shows that the top problem, Ineffectiveness [ManualDispatch], is not sufficiently denied because one of its sub-problems, Ineffectiveness [SpecialIncidentIdentification], is not sufficiently denied (not crossed). This should prompt the requirements engineers to repeat step 1-4 until the top soft-problem is sufficiently denied.
4. AN EMPIRICAL STUDY

This framework has been applied to the 1992 London ambulance case study [18] as an experiment to evaluate this approach. We studied the public report [18] that describes the problems related to the previous manual dispatch system and the corresponding intended solutions to be provided by the 1992 CAD system. We derived and captured both implied and stated relevant stakeholders, goals, and problems in a problem diagram adapted from [8], which we then captured more specific cause-effect and problem-solution relationships in a Problem Interdependency Graph (PIG). The resulting problem diagram and PIG are shown in Fig. 6 and 7 respectively.

The resulting models appear to make the information more explicit and understandable than the textual description in the report, thanks to their visual and semantically richer representations, which should make it amenable to visual or semi-automated analysis and detection of requirement problems. For example, Fig. 7 shows that the top soft-problem affecting the Timeliness goal is not sufficiently addressed (not crossed) because one of its sub-problems, Human Judgement [Duplicate Call Handling], is not addressed. This may suggest that a solution, therefore a requirement, is missing.

Conversely, Fig. 7 also shows that a planned feature, Automated Resource Allocation, is orphaned without a corresponding problem mentioned in the report, which may suggest that either a problem is overlooked or the solution is unnecessary (gold-plating). It turned out the feature may have been unnecessary as a later successful CAD system [5] does not have this feature. In fact, the report [18] states that the automated allocation feature was too ambitious, and was a factor leading to the infamous failure of the 1992 CAD system.

5. RELATED WORK AND DISCUSSION

A number of approaches (Fishbone diagram [9], Fault Tree Analysis [12], i* [15], Abuse Frames [13], Misuse Cases [21], Anti-goal [24]) have been used to deal with stakeholder problems. However, our approach is a more comprehensive framework that provides similar features found in these approaches with additional benefits. For example, our approach provides: (1) an explicit representation for stakeholder problems that are implicit in [15, 13, 24] (e.g. reusing positive concepts for negative concepts such as using goals for problems); (2) an explicit representation for adversary agents that are not supported by [9, 12, 13, 24]; (3) the Problem Interdependency Graph for root cause analysis that is not available in [13, 21, 24] where we provide richer constructs than [9, 12, 15, 13] (e.g. And/Or-decompositions and Make/Help contributions); (4) a unique representation of stakeholder problems in the context of goals, agents, requirements, and specifications in the same diagram.

Attempts have been made to integrate agent-goal-oriented approaches with the Problem Frames approach [14, 2]. In [14], a problem diagram is represented as an i* Strategic Dependency diagram, representing domains as agents and the requirement as a task dependency. An i* Strategic Rationale diagram is used to perform goal-oriented analysis for achieving the requirement by the dependent agents. Similarly, [2] also uses the Strategic Rationale diagram for the same purpose, but without a Strategic Dependency diagram. Our approach treats the problem diagram as the requirement and specification artifacts [7, 20] for the late-requirement phase with extensions to include agents, goals, and problems, to capture the world artifacts for the early-requirement phase [26]. Therefore, our approach can be seen as providing a bridge between the two requirement phases. However, at this preliminary stage, our approach has only been integrated with the Problem Frames approach to provide the
traceability between goals and the machine domain, but not yet fully integrated at the phenomenon level.

Syntactically, the Problem Interdependency Graph (PIG) is a mirror image of the Softgoal Interdependency Graph (SIG) [3] where the undesirable situation and the undesirable mechanism are opposite concepts of the NFR softgoal and the operationalizing softgoal respectively. This allows seamless modeling and analysis of goals and problems using the same facilities. However, the resulting goal and problem models are not likely to be mirror images of each other. For example, if a PIG has the vulnerable WEP encryption [1] as an undesirable mechanism, the corresponding SIG would not simply contain “do not use WEP”, but would rather contain some of more secure encryption techniques.

6. CONCLUSION

This paper presents a framework for dealing with stakeholder problems. Specific contributions include: (1) explicit representation of stakeholder problems using a new first-class modeling concept called “soft-problem”, as well as “adversary” that may wish to achieve such soft-problem; (2) an extended problem diagram that captures related agents, goals, problems, requirements, and specifications that supports the capturing of “as-is” and “to-be” systems; and (3) a qualitative framework for problem analysis documented in a diagram called Problem Interdependency Graph. The empirical study shows that the framework could help ensure that stakeholder problems are sufficiently addressed and, if not, help identify missing or goal-plating requirements.

Future work beyond the current preliminary results includes a more closely integration with the Problem Frames approach at the phenomenon level and a potential integration with other negative modeling such as the Anti-goal and the Abuse Frames approaches.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable constructive critiques and insights. We also thank Ebenezer A. Oladimeji and Weimin Ma for their help with the manuscript and many helpful discussions.

7. REFERENCES