REF: a Practical Agent-Based Requirement Engineering Framework

Paolo Bresciani[1] and Paolo Donzelli[2]

[1] ITC-irst
Via Sommarive 18, I-38050 Trento-Povo (Italy)
bresciani@itc.it
[2] Department of Computer Science — University of Maryland
College Park - MD (USA)
donzelli@cs.umd.edu

Abstract. Requirements Engineering techniques, based on the fundamental notions of agency, i.e., *Agent*, *Goal*, and *Intentional Dependency*, have been recognized as having the potential to lead towards a more homogeneous and natural software engineering process, ranging from high-level organization needs to system deployment. However, the availability of simple representational tools for Requirements Engineering still remains a key factor to guarantee stakeholders involvement, facilitating their understanding and participation.
This paper introduces *REF*, an agent-based Requirements Engineering Framework designed around the adoption of a simple, but effective, representational graphical notation. Nevertheless, a limited expressiveness of the graphical language may constrain the analysis process, reducing its flexibility and effectiveness. Some extensions are proposed to enhance REF capability to support requirements engineers in planning and implementing their analysis strategies, without affecting however REF clarity and intuitiveness.

1 Introduction

Agent- and goal-based Requirements Engineering (RE) approaches have the potential to fill the gap between RE and Software Engineering [5, 4]. The concepts of *Agent*, *Goal*, and *Intentional Dependency*, in fact, applied to describe the social setting in which the system has to operate, lead towards a smooth and natural system development process, spanning from high-level organizational needs to system deployment [4]. Goals are valuable in identifying, organizing and justifying system requirements [14, 2], whereas the notion of agent provides a quite flexible mechanism to model the stakeholders.

However, the concrete application of such approaches has been until now limited only to few case studies. Several causes of this still immature adoption of agent- and goal-based paradigms for RE may be identified. Below we consider only two of them.

First, although the notion of goal is central in some RE consolidated approaches like ***i**** [15], GBRAM [1, 2], and KAOS [8], an integrated and comprehensive requirements analysis methodology, clearly linked, or link-able, to the subsequent phases of software development, still is an open issue. At best of our knowledge, only the Tropos methodology [5, 4] fully addresses this issue. Yet, not full consideration has been given

by Tropos itself to the design of a precise process for the RE phases (early requirements and late requirements), due to the wide set of aspects that have to be captured.

Second, concerning the RE component of Tropos (or ***i****, to which Tropos RE is largely inspired), it is worth noticing that its considerably rich modeling framework, although promises to be capable of capturing several aspects relevant for the following phases, shows a certain level of complexity, so resulting understandable to only a strict group of practitioners. When the use of an ***i****-like modeling language has to be extended to non-technical stakeholders, it may be appropriate to give up with the full language expressiveness and modeling flexibility, in favor of a more straightforward and simple way to communicate with the stakeholders.

In such a perspective, the paper introduces an agent- and goal-based RE Framework (called *REF*) previously applied to an extensive project for the definition of the requirements of a simulation environment [11]. Simple, yet reasonably expressive, REF allows non technical stakeholders to elicitate requirements, in collaboration with a requirements engineer which, at the same time, is provided with an effective methodology and process for requirements acquisition, analysis and refinement, and for communicating, in an easily intelligible way, the results of her analysis to the stakeholders.

In the following, after a brief introduction to REF (Sections 2), a case study, is adopted (Sections 3) to critically revise the analysis process underlying the current methodology, to point out some of its current limits, and, finally, to suggest some notational and methodological extensions (Section 4). The tradeoff between the REF simplicity (and usability) and its expressiveness is carefully analyzed. Finally, observed advantages are discussed in the conclusive Section.

2 REF

REF is designed to provide the analysts and the stakeholders with a powerful tool to capture high-level organizational needs and to transform them into system requirements, while redesigning the organizational structure to better exploit the new system.

The framework tackles the modeling effort by breaking the activity down into more intellectually manageable components, and by adopting a combination of different approaches, on the basis of a common conceptual notation.

Agents are used to model the organization [9, 11, 16]. The organizational context is modeled as a network of interacting agents (any kind of active entity, e.g., teams, humans and machines, one of which is the target system), collaborating or conflicting in order to achieve both individual and organizational goals. *Goals* [9, 11, 8] are used to model agents relationships, and, eventually, to link organizational needs to system requirements. According to the nature of a goal, a distinction is made between *hard-goals* and *soft-goals*. A goal is classified as *hard* when its achievement criterion is sharply defined. For example the goal *"document be available"* is a hard-goal, being easy to check whether or not it has been achieved (i.e., is the document available, or not?). For a *soft-goal*, instead, it is up to the goal originator, or to an agreement between the involved agents, to decide when the goal is considered to have been achieved. For example, the goal *"document easily and promptly available"* is a soft-goal, given that

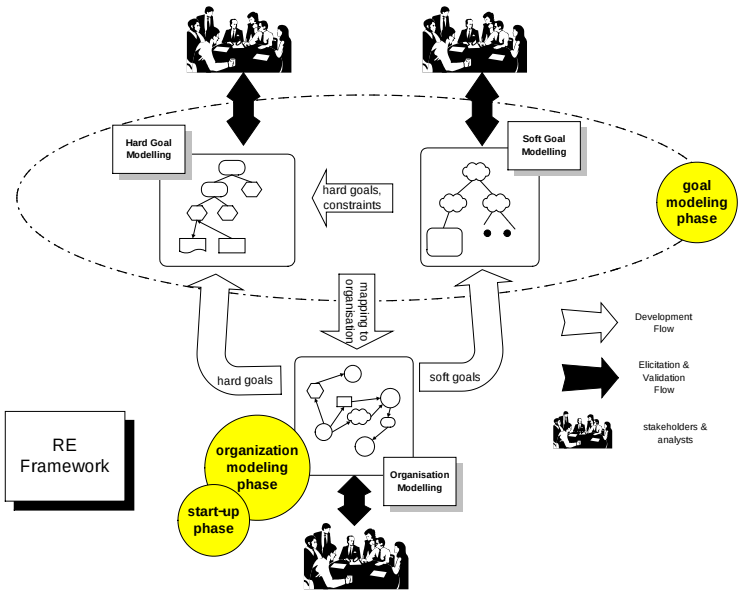

Fig. 1. The Requirements Engineering Framework (REF)

when we introduce concepts such as easy and prompt , different persons usually have different opinions.

REF, tackles the modeling effort by supporting three inter-related activities as listed below (see also Figure 1). The three modeling activities do not exist in isolation, rather they are different views of the same modeling effort, linked by a continuous flow of information, schematized as Development, and Elicitation & Validation flows.

Organization Modeling, during which the organizational context is analyzed and the agents and their goals identified. Any agent may generate its own goals, may operate to achieve goals on the behalf of some other agents, may decide to collaborate with or delegate to other agents for a specific goal, and might clash on some other ones. The resulting goals will then be refined, through interaction with the involved agents, by hard- and soft-goal modeling.

Hard-Goal Modeling seeks to determine how an agent can achieve a received hard-goal, by decomposing it into more elementary subordinate hard-goals, *tasks*[3], and *resources*[4]. Supported by the REF graphical notation, the analyst and the agent will work together to understand and formalize how the agent thinks to achieve the goal, in terms of subordinate hard-goals and tasks that he or she will have to achieve and perform directly, or indirectly, by passing them to other agents.

Soft-Goal Modeling aims at producing the operational definitions of the soft-goals, sufficient to capture and make explicit the semantics that are usually assigned im-

[3] A *task* is a well-specified prescriptive activity.

[4] A *resource* is any concrete or information item necessary to perform tasks or achieve goals.

plicitly by the involved agents [3, 6, 7]. Unlike for an hard-goal, for a soft-goal the achievement criterium is not, by definition, sharply defined, but implicit in the originator intentions. The analyst's objective during soft-goal modeling is to make explicit such intentions, in collaboration with the goal originator. However, depending on the issue at hand, and the corresponding role played by the two agents (i.e., the originator and the recipient) within the organization, also the recipient may be involved in the process, to reach a sharply defined achievement criterium upon which both of them can agree. Again, the analyst and the agents will cooperate through the support of the REF graphical notation.

In the three modeling activities, REF uses a diagrammatic notations which immediately convey the dependencies among different agents and allow for a detailed analysis of the goals, upon which the agents depend. The adopted graphical notation is widely inspired by the ***i**** framework [15] for RE [16] and business analysis and re-engineering [17], and thus open to be integrated in or extended by the Tropos methodology.

An important aspect of REF is to adopt the ***i**** notational ingredients at a basic and essential level, in favor of a higher usability and acceptability by the stakeholders. In the next Sections the notation and the methodology is briefly introduced by means of a case study. Mainly, Soft-Goal Modeling will be considered. The main aim during Soft-Goal Modeling is to iteratively refine each soft-goal in terms of subordinate elements, until only hard-goals, tasks, resources, and *constraints* are obtained (that is, until all the soft aspects have been dealt with) or each not refined soft-goal is passed on another agent, in the context of which will then be refined. *Constraints* may be associated with hard-goals, tasks, and resources to specify the corresponding quality attributes. Thus, the resulting set of constraints represents the final and operationalized views of the involved quality attributes, i.e., the quality models that formalize the attributes for the specific context [3, 6].

3 The case study

We refer to an on-going project aiming at introducing an Electronic Record Management System (ERMS) within a government unit. The impact of such a system on the common practices of the communities and sub-communities of knowledge workers is quite relevant.

A ERMS is a complex Information and Communication Technology (ICT) system which allows for efficient storage and retrieval of document-based unstructured information, by combining classical filing strategies (e.g., classification of documents on a multi-level directory, cross-reference between documents, etc.) with modern information retrieval techniques. Moreover, it provides mechanisms for facilitating routing and notification of information/document among the users, and supporting interoperability with similar (typically remote) systems, through e-mail and XML.

Several factors (international benchmarking studies, citizens demand, shrink budgets, etc.) called for the decision of leveraging new technologies to transform the organization into a more creative, and knowledgeable environment. The initial organization model is shown in Figure 2. Circles represent agents, and dotted lines are used to

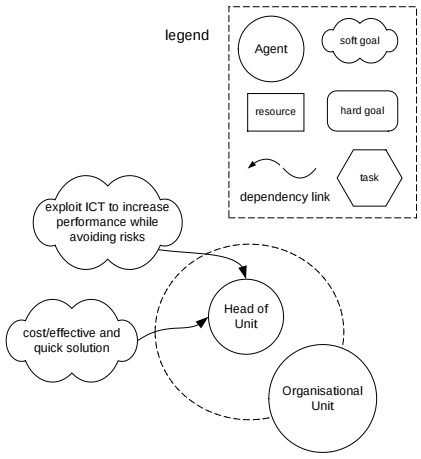

Fig. 2. Introducing the ERMS: the initial organization model

bound the internal structure of complex agents; that is, agents containing other agents. In Figure 2, the complex agent Organization Unit corresponds to the organizational fragment into which it is planned to introduce the new ERMS, whereas the *Head of Unit* is the agent, acting within the Organizational Unit, responsible for achieving the required organizational improvement (modeled by the soft-goals exploit ICT to increase performance while avoiding risks, and cost/effective and quick solution).

Goals, tasks, resources and agents (see also next Figures) are connected by dependency-links, represented by arrowhead lines. An agent is linked to a goal when it needs or wants that goal to be achieved; a goal is linked to an agent when it depends on that agent to be achieved. Similarly, an agent is linked to a task when it wants the task to be performed; a task is linked to an agent when the agent is committed at performing the task. Again, an agent is linked to a resource when it needs that resource; a resource is linked to an agent when the agent has to provide it. By combining dependency-links, we can establish dependencies among two or more agents. As mentioned, the soft-goals modeling process allow the analysts and the stakeholders to operationalize all the soft aspects implicitly included in the meaning of the soft-goal.

Thus, for example, Figure 3 describes how the soft-goal exploit ICT to increase performance while avoiding risks is iteratively top-down decomposed to finally produce a set of tasks, hard-goals, and constraints that precisely defines the meaning of the soft-goal, i.e., the way to achieve it. Figure 3, in other terms, represents the strategy that the *Head of Unit* (as result of a personal choice or of a negotiation with the upper organizational level) will apply to achieve the assigned goal. Again, the arrowhead lines indicate dependency links. A soft-goal depends on a sub-ordinate soft-goal, hard-goal, task, resource or constraint, when it requires that goal, task, resource or constraint to be achieved, performed, provided, or implemented in order to be achieved itself. These dependency links may be seen as a kind of top-down decomposition of the soft-goal. Soft-goals decompositions may be conjunctive (all the sub-components must be satis-

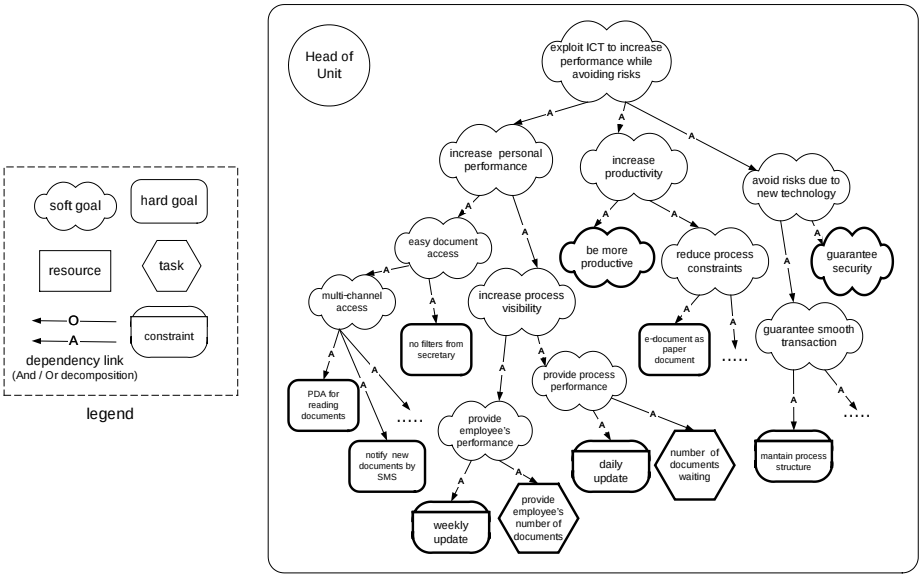

Fig. 3. The "exploit ICT to increase performance while avoiding risks" Soft-Goal Model

fied, to satisfy the original soft-goal), indicated by the label A on the dependency link, or disjunctive (it is sufficient that only one of the components is satisfied), indicated by the label O on the dependency link (see Figure 5).

According to Figure 3, the Head of Unit has to increase personal performance, to increase productivity of the whole unit, and also to avoid risks due to new technology. Let's consider in details only the first sub-soft-goal, i.e., increase personal performance. It spawns two subordinate soft-goals, easy document access, for which the Head of Unit will require a multi-channel access system in order to be able to check and transfer the documents to the employees also when away from the office, and increase process visibility, to take better informed decisions. In particular, the soft-goal increase process visibility will eventually lead to the identification of some tasks (functionalities) the system will have to implement in order to collect and make available some data about the process (e.g., number of documents waiting) and about the employees (provide employee's number of documents that have been assigned), and of some associated constraints, represented by a rounded-rectangle with a horizontal line, characterizing such data. In Figure 3, for example, they specify the frequency of update: daily for the process data and weekly for the employee's ones.

4 Adding special links to support the analysis process

As described in Section 2, REF aims at providing a representational framework for requirements discovery and analysis, characterized by a sufficiently expressive graphical notation that, at the same time, be simple enough to be easily and quickly grasped by

the stakeholders, even by those unfamiliar with RE. Indeed, these are very important aspects that, as demonstrated by several case studies [13, 12, 11], make REF applicable to real projects, and ensure a concrete involvement of the stakeholders, allowing for a quicker and more effective knowledge acquisition and requirements elicitation process.

REF simplicity and effectiveness is mainly based on two key points: 1) the use of only one type of link (the dependency link); 2) the focus, during both hard- and soft-goal analysis, on only one goal (and the involved agents) at time; this leads to the drawing of very simple goal analysis diagrams, strictly generated by a top-down process. These two aspects make REF different from other approaches, in particular from ***i****. Indeed, we believe that these two characteristics allow for a very easy reading of the goal models; the second feature, in particular, allows the stakeholders (and the analysts, as well) to concentrate the attention on one problem at time, and not being worried about the order in which different analyses of different sub-diagrams should be interleaved in order to obtain different possible diagrams: the goal diagram is always a tree, and it is always generate in the same shape, whatever node expansion sequence is followed.

In the following, we analyze whether or not these two very important REF characteristics may represent a limit to some relevant aspects of the process of domain description. In particular, we tackle two possible cases in which the present version of REF tends to show some limits, describe them by means of our case study ERMS, and propose simple extensions to REF, to allow for a finer control during the process of model description and requirements elicitation.

4.1 Sharing goals (tasks, constraints . . .)

Let us here analyze the fact that REF produces only trees, as goal diagrams. Thus, there is not the explicit possibility to deal with sub-goals (or constraints, or tasks, or resources) that may be shared by different upper-level goals. This situation may be further distinguished in at least three different sub-cases.

First case: top-down tree expansion and analysis induces at introducing different sub-goals (or constraints, or tasks, or resources) for any different goal that is found during the goal analysis activity, even if different goals could be satisfied by the same sub-goal (or constraint, or task, or resource). For example, in Figure 3, two distinct constraints have been introduced for satisfying the two soft-goals provide process performance and provide employee's performance, namely the constraints daily update and weekly update. Instead, for example, the Head of Unit could have accepted that the two soft-goals, rather than requiring two different specialized constraints (as in Figure 3), would have shared the same constraint, e.g., a twice a week update (as in Figure 4). After all, according to REF, any sequence may have been followed in analyzing the two soft-goals, and the two constraints may have been introduced in two very different moments, making it very difficult to spot that a common (although slightly different) constraint could have been adopted. This compromise, instead, could have been identified and judged as acceptable if considered by the analyst together with the Head of Unit at the proper moment during the design activity. The differences between Figure 3 and Figure 4 are minimal, regarding only leaf nodes, as highlighted by the dotted circle. So,

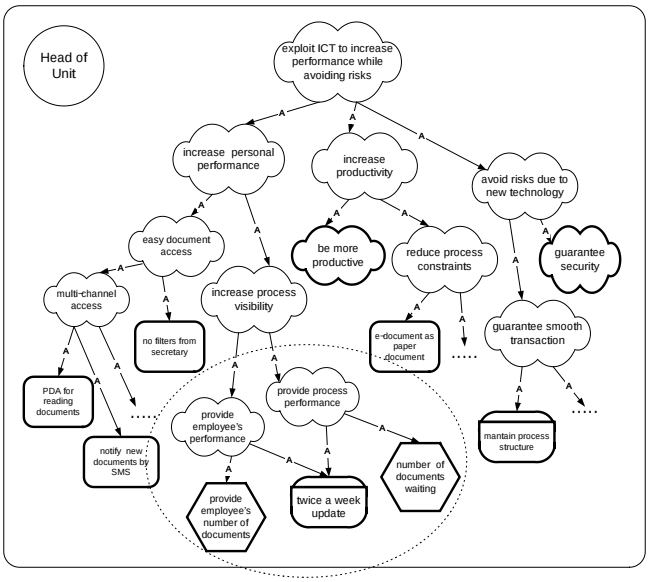

Fig. 4. The "exploit ICT to increase performance while avoiding risks" Soft-Goal Model revised

Figure 4 can be obtained as a simple transformation of Figure 3. But in more complex hypothetical cases, in which the two nodes collapsing in one were non-leaf nodes with possibly deep trees expanding from them, relevant parts of the two sub-trees, rooted in the two nodes, would have to be revised, in order to consider alternative non-tree-based analyses. In these cases, it would be strategic to be able to introduce a common child for the two different nodes before proceeding with the analysis of the nodes sub-trees. It is clear that, now, different diagram evolution strategies, and thus development sequences, may lead to quite different results or, even when producing the same result, this may be obtained with different degrees of efficiency. For example, a top-down bread-first diagram expansion could be probably preferred to a top-down depth-first strategy. In this way, it may appear appropriate to develop a shared sub-tree only once, with two advantages: 1) at the design level, the analysis has not to be carried out twice; 2) at the implementation level, the complexity of the system to be implemented will be reduced, being two potentially different requirements, and all the derived artifacts —from architectural design down to implemented code— collapsed in one.

Second case: as a specialization of the first one, we can consider the case in which a similar sub-goal sharing happens among goals attached to the same agent since its introduction in the organizational model, and not as a result of goal modeling. In this case, the REF methodology would lead the analyst to duplicate the sub-goal in two different diagrams, possibly with slightly different labels, although with the same semantics. Catching these cases as early as possible is very important in order to avoid duplicated analysis and assign higher priority and relevance to the analysis of the shared items.

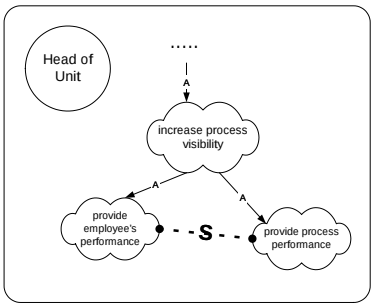

Fig. 5. A sharing between goals of the same agent

Third case: a more intriguing situation may arise when the very same sub-goal can be shared among two different agents, as a consequence of two different and autonomous analyses of two different goals of the two agents (there is no room here to present the figures to illustrate an example in this case, but see [10]). Again, the analysis of such a shared soft-goal immediately assume a higher relevance and priority over the analysis of other goals. Its satisfaction is desired by two agents! For example, leading to the selection, among all the possible available tools on the market, of only one kind of mobile access channel able to satisfy both the agents.

From the analysis of the previous three cases, clearly emerges the need of introducing in REF some mechanism to support the analysts during their refinement and navigation through the goals and sub-goals models. In particular, we propose to provide the analysts with a specific notation to be used to highlight situations where they believe that some commonalities could be hidden, i.e., that shared-goals could arise during the analysis. In other terms, to introduce in REF a notation suitable to act as a high-level reasoning support tool to enable the analysts to record their intuitions while building the goals models, by making notes to drive their strategies, e.g., to highlight where a top-down breath-first diagram expansion may be preferable to a top-down depth-first one. As such a notation, we introduce what we call the *S-connection* ("S" for Sharing), a link that does not have arrows, being the relationship perfectly symmetric.

Figure 5 shows a fragment of Figure 3 where the S-connection has been adopted. In particular, it shows how the S-connection could have been used during the analysis of the soft-goal to highlight in advance the possibility of sharing between the soft-goals provide employee performance and provide process performance (the first example case previously analyzed). In the same way, in Figure 6 is depicted the use of the S-notation to highlight, within the soft-goal analyzed in Figure 3, a possible sharing between the soft-goal increase personal performance, that the Head of Unit wants to achieve, and the soft-goal be more productive, that the Head of Unit imposes, transfers to, the Employee (the third example case previously analyzed). It is worth noting how the S-notation is only a reasoning support mechanism that tend to disappear once the analysis proceeds. In other terms, the S-notation purpose is to mark possible sharing situations to drive the analysis (e.g., bread-first, multi-agents analysis, repeated back to back comparisons,

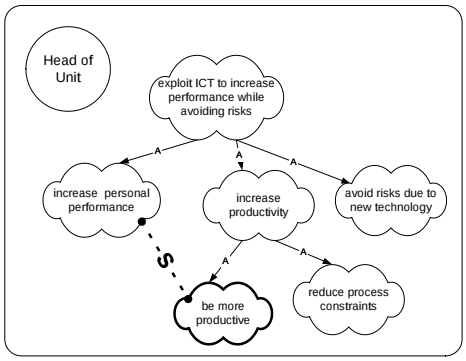

Fig. 6. A sharing between goals of different agents

and so on), but does not have any reason to exist any more once the goals have been exploded: the initial REF notation, with its simplicity, is sufficient for that regard.

4.2 Clashing goals (tasks, constraints . . .)

Another common situation regards the possibility of efficiently dealing with clashing needs (goals, or constraints, tasks, and resources). As well as during a top-down analysis a top-down introduced sub-goal may be recognized as helpful for another goal (possibly of another agent), similarly, the introduced sub-goal may be recognized as (possibly) harmful for another goal. In addition, during the analysis, new agents may have to be introduced into the context (e.g., the Head of Unit requires the Employee to be more productive), and such new agents may express their own needs by introducing new goals that may very easily clash with other goals already in the context.

Indeed, REF already provide the tool for the detailed recognition of such situations. In fact, when fully operationalized in terms of tasks and constraints, goals models can be adopted to detect clashing situations and to resolve them. Nevertheless, it is critical to foresee such possibly clashing situations as early as possible, even only at a very qualitative level. To enable the analysts to mark possible conflicting situations (and build their refinement strategy to deal with them), we introduce the *H-connection* ("H" for hurting). This is a powerful tool to detect possible conflicts and try to reconcile different stakeholders points of view, allowing to evolve the analyses only along the most promising alternatives. An example of application is given in Figure 7, where a H-connection is used to highlight a possible conflict between two goals before proceeding in their analysis (i.e., the soft-goal provide employee's performance is not broken down into tasks before taking into account the protect my privacy one —see also [10]).

5 Conclusions

The paper introduced an agent-oriented Requirements Engineering Framework (REF), explicitly designed to support the analysts in transforming high-level organizational

Fig. 7. A conflict between goals of different agents

needs into system requirements, while redesigning the organizational structure itself. The underlying concepts and the adopted notations make of REF a very effective and easy to deal with (usable) tool, able to tackle complex real case situations, while remaining simple enough to allow a concrete and effective stakeholders involvement.

REF is strongly based upon ***i****, the modeling framework suggested by Eric Yu [15, 17, 16]. However, it introduces some simplifications and tends to adopt a more pragmatic approach in order to obtain a greater and more active involvement of the stakeholders during the requirements discovery, elicitation and formalization process.

However, we felt that REF could be improved with the regard to the support it provided to the analysts in dealing with more complex, and system/organizational design related issues, such as shared and clashing stakeholders needs. In both cases, an early detection of such a situation could lead to better analysis results: shared needs could be objects of a more intensive analysis effort to exploit commonalities to reduce complexity and increase re-usability; clashing needs could be solved at a very early stage, to focus then the analysis only towards the most promising alternatives. Two graphical notations (i.e., the S-connection and the H-connection) have therefore been introduced to allow the analysts to mark such situations and better reason about how to build their strategy, while performing them. They are pure analyst-oriented tools, that do not affect REF usability in terms of stakeholders perspective, but greatly improve the analysts capabilities of building the strategy to deal with shared and clashing interests.

References

1. A. I. Antón. Goal-based requirements analysis. In *Proceedings of the IEEE International Conference on Requirements Engineering (ICRE '96)*, Colorado Springs, USA, Apr. 1996.
2. A. I. Antón and C. Potts. Requirements for evolving systems. In *Proceedings of the International Conference on Software Engineering (ICSE '98)*, Kyoto, Japan, Apr. 1998.
3. V. R. Basili, G. Caldiera, and H. D. Rombach. *Encyclopedia of Software Engineering*, chapter The Goal Question Metric Approach. Wiley&Sons Inc, 1994.
4. P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. TROPOS: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 2003. in Press.
5. P. Bresciani, A. Perini, , F. Giunchiglia, P. Giorgini, and J. Mylopoulos. A Knowledge Level Software Engineering Methodology for Agent Oriented Programming. In *Proceedings of the Fifth International Conference on Autonomous Agents*, Montreal, Canada, May 2001.
6. G. Cantone and P. Donzelli. Production and maintenance of goal-oriented software measurement models. *International Journal of Knowledge Engineering and Software Engineering*, 10(5):605–626, 2000.
7. L. K. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Kluwer Publishing, 2000.
8. A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50, 1993.
9. M. D'Inverno and M. Luck. Development and application of an agent based framework. In *Proceedings of the First IEEE International Conference on Formal Engineering Methods*, Hiroshima, Japan, 1997.
10. P. Donzelli and P. Bresciani. Goal-oriented requirements engineering: a case study in e-government. In J. Eder and M. Missikoff, editors, *Advanced Information Systems Engineering (CAiSE'03)*, number 2681 in LNCS, pages 605–620, Klagenfurt/Velden, Austria, June 2003. Springer-Verlag.
11. P. Donzelli and M. Moulding. Developments in application domain modelling for the verification and validation of synthetic environments: A formal requirements engineering framework. In *Proceedings of the Spring 99 Simulation Interoperability Workshop*, LNCS, Orlando, FL, 2000. Springer-Verlag.
12. P. Donzelli and R. Setola. Putting the customer at the center of the IT system — a case study. In *Proceedings of the Euro-Web 2001 Conference — The Web in the Public Administration*, Pisa, Italy, Dec. 2001.
13. P. Donzelli and R. Setola. Handling the knowledge acquired during the requirements engineering process. In *Proceedings of the Fourteenth International Conference on Knowledge Engineering and Software Engineering (SEKE)*, 2002.
14. A. van Lamsweerde. Goal-oriented requirements engineering: A guided tour. In *Proceedings of RE'01 - International Joint Conference on Requirements Engineering*, pages 249–263, Toronto, aug 2001. IEEE.
15. E. Yu. *Modeling Strategic Relationships for Process Reengineering*. PhD thesis, University of Toronto, Department of Computer Science, University of Toronto, 1995.
16. E. Yu. Why agent-oriented requirements engineering. In *Proceedings of 3rd Workshop on Requirements Engineering For Software Quality*, Barcelona, Catalonia, June 1997.
17. E. Yu and J. Mylopoulos. Using goals, rules, and methods to support reasoning in business process reengineering. *International Journal of Intelligent Systems in Accounting, Finance and Management*, 1(5):1–13, Jan. 1996.