UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

APPLYING TROPOS REQUIREMENTS ANALYSIS FOR DEFINING A TROPOS TOOL

Paolo Bresciani and Fabrizio Sannicolò

May 2002

Technical Report # DIT-02-106

Applying Tropos Early Requirements Analysis for defining a Tropos tool

Paolo Bresciani and Fabrizio Sannicolò

ITC-irst
Via Sommarive, 18, I-38050 Trento-Povo, Italy
{bresciani,sannico}@irst.itc.it

1 Introduction

Tropos [8, 6] is a novel agent-oriented software engineering methodology characterized by three key aspects [7]. First, it pays attention to the activities that precede the specification of the prescriptive requirements, like understanding how the intended system would meet the organizational goals. Second, it deals with all the phases of system requirement analysis and all the phases of system design and implementation in a uniform and homogeneous way, based on common mentalistic notions as those of *actors*, *goals*, *softgoals*, *plans*, *resources* and *intentional dependencies*. Third, the methodology rests on the idea of building a model of the system-to-be that is incrementally refined and extended from a conceptual level to executable artifacts, by means of a sequence of transformational steps [1].

One of the main advantages of the Tropos methodology is that it allows to capture not only the *what* or the *how*, but also the *why* a piece of software is developed. This, in turn, allows for a more refined analysis of the system dependencies and, in particular, for a much better and uniform treatment not only of the system functional requirements, but also of the non-functional requirements. Tropos, although not exclusively, addresses particularly well the Agent Oriented Programming. In fact, the decision of using mentalistic notions in all the analysis phases has important consequences. Agent oriented specifications and programs use the same notions and abstractions used to describe the behavior of the human agents and the processes involving them; thus, the conceptual gap between users' specification (in terms of *why* and *what*) and system realization (in terms of *what* and *how*), is reduced to a minimum.

Tropos supports five phases of software development: *early requirements*, *late requirements*, *architectural design*, *detailed design* and *implementation* [5, 6].

The present short paper concentrates on the analysis of the early requirement phase for the definition of a support tool for the Tropos methodology itself, called, since now on, the *Tropos tool*. In particular, the task of encoding initial informal requirement into the diagrammatic format used in the methodology, as well as the incremental transformational process that is at the basis of the construction of the complete model, will be addressed.

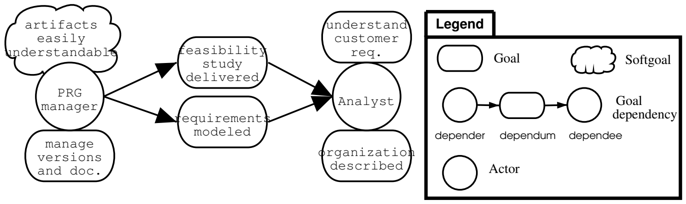

Fig. 1. An actor diagram specifying the stakeholders of the Tropos tool project.

2 Early Requirement Analysis

The early requirements analysis concerns with the understanding of a problem by studying an existing organizational setting. The intentions of the stakeholders are modeled as *goals* and *goal dependencies* among *actors*, and analyzed by means some form of goal analysis [3, 4]. The output of this phase is an organizational model including the relevant actors and their respective dependencies for the achievement of the goals and the softgoals, and for performing or obtaining resources. The model is visualized by means of *actor diagrams*, describing the network of social dependencies among actors, and *goal diagrams*, analyzing the goal fulfillments.

An awful lot of high level requirements has been identified, listed and justified in [2]; due to lack of space, only a short subset of these requirements are introduced here. For example, the actor diagram of Figure 1 shows the actor `PRG manager` who depends on the `Analyst` for two goals: `feasibility study delivered` and `requirements modeled`.

A further step concerns the decomposition of each goal from the point of view of the actor who committed for its fulfillment. Goals are analyzed by using three basic analysis techniques: *means-ends analysis*, *contribution analysis* and *AND-OR decomposition* [3, 9]. Means-ends analysis proceeds by refining a goal into subgoals in order to identify goals, plans, resources and softgoals that provide means for achieving the goal (the end). Contribution analysis allows the designer to point out goals, softgoals and plans that can contribute positively or negatively at reaching the goal. AND-OR decomposition allows for a combination of AND and OR decompositions of a root goal into subgoals.

In Figure 2, the goal `requirements modeled` is the "end" in a means-ends analysis where the "mean" is the goal `build a requirements Tropos model`. This last is AND-decomposed into the four subgoals: `learn Tropos language`, `model managed`, `model documented` and `model reasoned`.

An instance of contribution analysis is provided, in the figure, between the softgoal `exhaustive guide` and the goal `learn Tropos language`. Of course, subgoals deriving from an AND-OR analysis can be further AND-OR analyzed. For example, `model managed` is decomposed into the `retrotraceability managed`, `multiple views visualized`, `diagrams edited` and `syntax of model checked`. The subgoals `automatic documentation generated` and `version managed` contribute positively at the achievement of the goal `model documented`. The first subgoal concerns with the automatic generation of the documentation about the conceptual model and/or

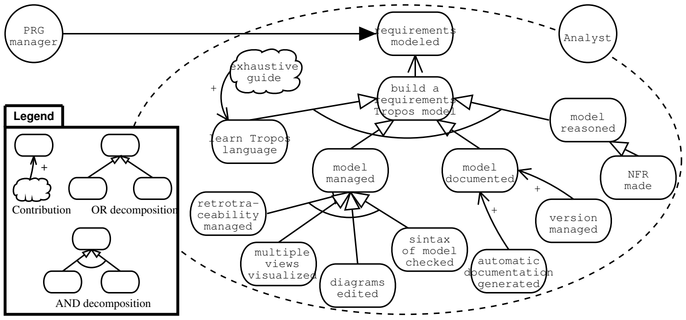

Fig. 2. Goal analysis from the perspective of the `Analyst`.

some views on it, while the second regards the management of different versions for each analyst.

The final setting of each Tropos phase may be reached after possibly several refinements, in each one of which not only new details may be added, but, also, already present elements and dependencies can be revised or even deleted [1].

It is obvious that the diagram of Figure 2 (and other developed during a first loop [2]) may be not sufficiently detailed. For this reason, iterative steps of incremental refinement of the model have to be performed. As already mentioned in the introduction, this way of proceeding is a typical feature of the Tropos methodology. The iterative process not only may require intra-phase refinements, but, possibly, also revisions of artifacts produced during early phases (inter-phases refinements). The importance of retrotraceability is, here, evident.

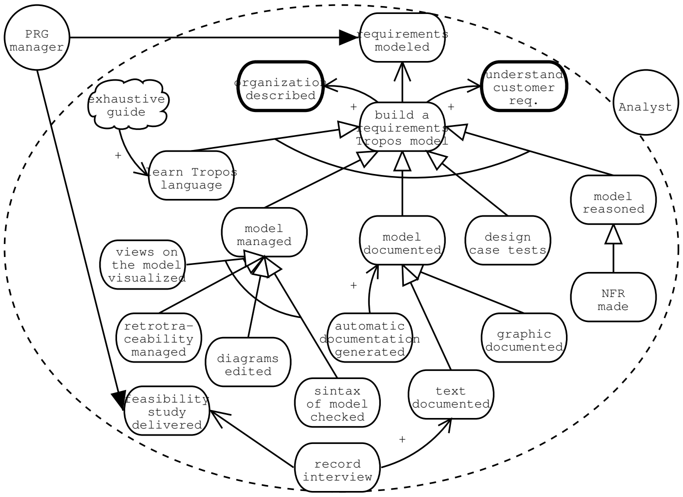

Fig. 3. Revising Goal analysis from the point of view of the `Analyst`.

Just as an example of the intra-phase refinement activity, the revision of the goal diagram in Figure 2 is presented in Figure 3. The contributions from `build a requirements Tropos model` to `organization described` and `understand customer requirements` are here introduced. These were initially considered as original `Analyst`'s goal (see Figure 1), but then not further analyzed in Figure 2. The revision proposed in Figure 3 is necessary to complete the analysis of the requirements of Figure 2, according to what was initially introduced in Figure 1. Among others elements introduced in the revision, let's list: `design case tests`, `text documented`, `graphic documented` and `feasibility study delivered` (see Figure 1), that became the end in a means-ends analysis where the "mean" is the goal `record interview`. For more details, see [2].

3 Conclusion

In the present paper the early analysis for a Tropos tool has been used as a case study for presenting some features of the Tropos methodology itself.

One point that deserves a note here is that the management of traceability has been raised as a crucial point for correctly dealing with the revision (specially the inter-phase revision) process. In future works, we aim at further developing this issue with other specifically focused case studies and examples.

References

1. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Modelling early requirements in Tropos: a transformation based approach. In P.Ciancarini M.J. Wooldridge, G. Weiß, editor, *Agent-Oriented Software Engineering II*, LNCS 2222, pages 151–168. Springer-Verlag, Montreal, Second International Workshop, AOSE2001 edition, May 2002.
2. P. Bresciani and F. Sannicolò. Applying Tropos to requirement analysis for a Tropos tool. Technical Report 0204-01, ITC-IRST, via Sommarive, Trento, April 2002.
3. L. Chung, S. Liao, W. Huaiqing, E. Yu, and J. Mylopoulos. Exploring alternatives during requirements analysis. *IEEE Software*, 18(1), jan, feb 2001.
4. A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50, 1993.
5. P. Giorgini, A. Perini, J. Mylopoulos, F. Giunchiglia, and P. Bresciani. Agent-oriented software development: A case study. In S. Sen J.P. Müller, E. Andre and C. Frassen, editors, *Proceedings of the Thirteenth International Conference on Software Engineering - Knowledge Engineering (SEKE01)*, Buenos Aires, June 2001.
6. F. Giunchiglia, A. Perini, and F. Sannicolò. Knowledge level software engineering. In Springer Verlag, editor, *In Proceedings of ATAL 2001, Seattle*, December 2001. Also IRST TR 0112-22, Istituto Trentino di Cultura, Trento, Italy.
7. A. Perini, P. Bresciani, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Towards an Agent Oriented approach to Software Engineering. In *Proceedings of the Workshop, dagli oggetti agli agenti: tendenze evolutive dei sistemi software*, Modena, Sep 2001.
8. A. Perini, P. Bresciani, 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 CA, 28 May - 1 June 2001.
9. F. Sannicolò, A. Perini, and F. Giunchiglia. The Tropos modeling language. A User Guide. Technical Report 0202-12, ITC-IRST, Jan 2002.