Empirical Evaluation of Tropos4AS Modelling

Mirko Morandini, Anna Perini, and Alessandro Marchetto

FBK-CIT, Trento, Italy,
{morandini,perini,marchetto}@fbk.eu,

Abstract. Our work addresses the challenges arising in the development of self-adaptive software, which has to work autonomously in an unpredictable environment, fulfilling the objectives of its stakeholders, while avoiding failure. In this context we developed the Tropos4AS framework, which extends the AOSE methodology Tropos to capture and detail at design time the specific decision criteria needed for a system to guide self-adaptation at run-time, and to preserve the concepts of agent and goal model explicitly along the whole development process until run-time.
In this paper, we present the design of an empirical study for the evaluation of Tropos4AS, with the aim of assessing the modeling effort, expressiveness and comprehensibility of Tropos4AS models. This experiment design can be reused for the evaluation of other modeling languages extensions.

Key words: Agent-oriented software engineering, Empirical studies, Self-adaptive systems.

1 Introduction

Today's software is expected to be able to work autonomously in an open, dynamic and distributed environment. *Self-adaptive software systems* were proposed as a solution to cope with the uncertainty and partial knowledge in such environments. The development of such software, which should automatically take the correct actions based on knowledge of what is happening, guided by the *objectives* assigned by the stakeholders, gives rise to various challenges: the software needs multiple ways of accomplishing its purpose, enough knowledge of its construction and the capability to make effective changes at runtime, to be able to autonomously adapt its behaviour to satisfy the requirements, shifting decisions which traditionally have been made at design-time, to run-time.

In our recent work we try to address these challenges, proposing the Tropos4AS (Tropos for Adaptive Systems) methodology [1]. Tropos4AS aids the software engineer in capturing and detailing at design time the specific knowledge and decision criteria that will guide self-adaptation at run-time. Moreover, it brings the high-level requirements, in form of goal-models, to run-time, to enable the system to monitor their satisfaction, to reflect upon them and to guide its behaviour according to them.

Like Tropos4AS, various extensions of the Tropos modelling language and methodology were proposed in the last years, specific for different purposes and various domains. However, only few attempts were made to assess such extensions by means of empirical studies. We present the design of two experiments with subjects, which have the scope to assess the novel extensions of Tropos4AS by comparison with the general methodology Tropos. Applying proper statistical tests, we are able to collect evidence on the effectiveness (modelling effort, model correctness and model comprehensibility) of Tropos4AS models, evaluating the results of modelling tasks, comprehension tasks and supporting questionnaires. The design is general and thus reusable for the evaluation of other specific extensions to general modeling languages.

2 Background

The Tropos4AS methodology extends Tropos to provide a process and modelling language that captures at design time the knowledge necessary for a system to deliberate on its goals in a dynamic environment, thus enabling a basic feature of self-adaptation. It integrates the goals of the system with the environment, and gives a development process for the engineering of such systems, that takes into account the modelling of the environment and an explicit modelling of failures. Tropos goal modelling is extended along different lines:

i) Capturing the influence of artifacts in the surroundings of an actor in the system to the actor's goals and their achievement process. This is achieved by modelling an actor's environment and defining *conditions* on the environment artifacts, to guide or guard state transitions in the goal satisfaction process, e.g. achievement conditions, goal creation conditions or failure conditions.

ii) The definition of goal types (maintain, achieve,...) and additional inter-goal relationships (inhibition, sequence), to detail the goal achievement and alternatives selection dynamics.

iii) Modelling possible failures, errors and proper recovery activities, to elicit missing functionalities to make the system more robust, to separate the exceptional from the nominal behaviour of the system, and to create an interface for domain-specific diagnosis techniques.

For the aim of providing an explicit representation of high-level requirements at run-time and lowering the conceptual gaps between the software development phases, we perserve the concepts of *agent* and *goal model* explicitly along the whole development process. The detailed Tropos4AS goal models represent the *"knowledge level"*, that is, the rationale behind the execution of specific tasks (i.e. plans). Adopting an implementation architecture which supports goal models, the software can navigate and reason on them and exploit available alternatives satisfy its requirements. A complete translation of requirements concepts to traditional software level notions, such as classes and methods of object-oriented programming, is avoided. This contributes to a smoother transition between the development phases, reducing loss and conceptual mismatch and simplifying the

tracing of decisions made in requirements analysis and design to the implementation and vice-versa.

A direct mapping from goal models to implementation concepts is defined, relying on agents with a BDI (Belief-Desire-Intention) architecture and a native support for the concepts of agent and goal. With a supporting middleware, an explicit, navigable and monitorable representation of goal models at run-time is realised. Tropos4AS (with the graphical modelling language presented in Figure 1) is detailed in [1], and [2]. Details for the operational semantics attributed to condition evaluation and to the satisfaction of goals in goal models, can be found in [3]. Note that the optimisation of a system's behaviour, by the use of run-time goal model reasoning, learning or knowledge acquisition strategies, is not part of, but would be complementary to Tropos4AS.

Tool support. The *Taom4E* Tropos modelling tool (selab.fbk.eu/taom) supports modelling of extended Tropos4AS models and includes a plug-in for an automated code generation (***t2x*** tool), base on the *Taom4E* Tropos modelling tool which uses the *Jadex* agent framework as implementation platform.

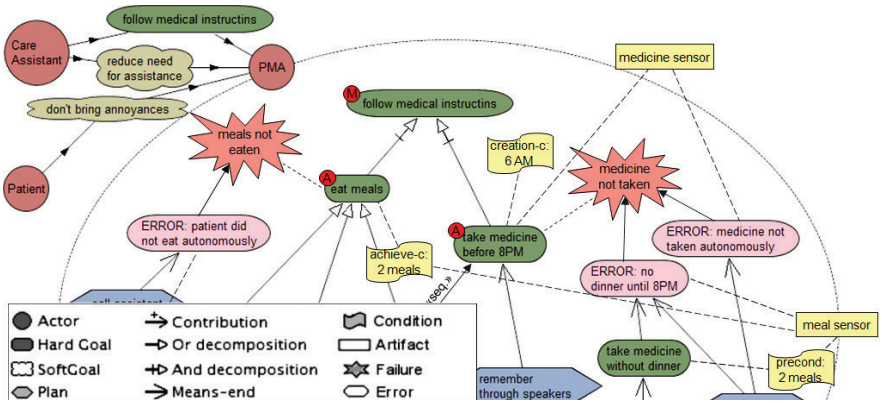

Fig. 1. Fragment of the Tropos4AS model for a patient monitoring system, one of the objects used in the the comprehension experiment.

3 Empirical Evaluation

The evaluation of novel extensions to a development methodology poses various challenges, since they are usually not yet extensively used in practice. Moreover, it is challenging to set up a fair and meaningful comparison for the evaluation of the introduced extensions: An empirical study which consists of a comparison of Tropos4AS with a methodology with a similar scope but with different roots, would inevitably also assess the performance of the whole Tropos language. Therefore, an evaluation limited to the novel extensions, which is our

scope, would be impossible. Conversely, an empirical study which involves implementation, would require participants that are experienced in the use of the implementation language, and demand a very high time effort. Also, a comparison of the whole *modelling process* would not be feasible within the given time constraints. Thus, to assess the novelties introduced with Tropos4AS, we propose to perform a controlled experiment with subjects, comparing the Tropos4AS modelling language to the underlying Tropos modelling language[1], which showed its effectiveness in various studies, e.g. [5].

The evaluation of a modelling language can be characterised by three main aspects: (1) the effort for modelling, (2) the effectiveness for capturing the requirements and (3) the comprehensibility of the obtained models. The study we propose consists of modelling and comprehension tasks performed by a group of subjects, and is divided into two experiments:

Modelling: we evaluate if Tropos4AS is effective in modelling self-adaptive systems, with an acceptable modelling effort, in comparison to Tropos.

Comprehension: we evaluate if the Tropos4AS modelling extensions increase the comprehensibility of the requirements of a system.

The design of the experiments follows the guidelines by Wohlin et al. [6] and allows to have a high degree of control over the study, to achieve results with statistical significance. Tropos and Tropos4AS represent the control *treatment* and the treatment to evaluate. The quality focus of the experiment concerns the capability of the treatments in supporting the analysts in requirements modelling and comprehension. The target *subjects* are researchers and Ph.D. students, while the *objects* of study are requirements specifications (textual and graphical) of two software systems with adaptivity features. It is however important that these systems can be modelled in a satisfactory way with both the general and the domain-specific methodology. We define three research questions (together with the relative null- and alternative hypotheses):

*RQ*1**:** Is the effort of modelling requirements with Tropos4AS significantly higher than the effort of modelling them with Tropos?

*RQ*2**:** Is the effectiveness of Tropos4AS models significantly higher than the effectiveness of Tropos models, for representing requirements of an adaptive system?

*RQ*3**:** Do Tropos4AS models significantly improve the comprehension of the requirements of a self-adaptive system, in comparison to Tropos models?

To investigate on these questions (with the aim of showing if the relative null-hypothesis can be rejected or not), we run a modelling and a comprehension experiment, which both adopt a paired, counterbalanced experiment design based on two laboratory sessions, such that the subject perform the experimental task twice, once with each object and treatment, exploiting all possible combinations. This lets us evaluate the performance of the subjects with both treatments, avoiding learning effects.

[1] We refer to the Tropos modelling language, as defined in [4]. In particular, we focus on Tropos *goal diagrams*, which are mainly affected by the novel extensions.

Design of the modelling experiment. The modelling experiment covers the research questions $RQ1$ and $RQ2$ and consists of:

1. a presentation and training session for the subjects, to introduce or refresh notions related to both treatments, and to explain the experiment tasks;
2. a pre-questionnaire to capture information about the experience of the subjects and about the clearness of the notations and of the experimental task;
3. two supervised laboratory sessions concerning a modelling task to be performed in an open time frame, asking the subjects to **model with as much details as possible the textual requirements specification**s handed out, with the assigned modelling language;
4. post-questionnaires asking about the perceived effort for each treatment and about personal opinions, comparing both treatments.

The research questions include the abstract terms *effectiveness* and *effort*, which have to be detailed in order to characterise these two terms for the scope of the study and to associate them to variables which can be evaluated. $RQ1$ is decomposed to aspects considering time, perceived effort, and the difficulties encountered while modelling, while the aspects for $RQ2$ consider the expressiveness of the modelling language as perceived by the subjects and the correctness of the models built. These aspects are evaluated collecting the questionnaire results (on an ordinal $1 \ldots 5$ *Likert* scale, from *strongly agree* to *strongly disagree*) and measuring the time spent. Moreover, model correctness is evaluated against an expert-made *gold standard* model, evaluating the coverage of three predefined software execution scenarios.

Design of the comprehension experiment. The comprehension experiment, covering $RQ3$ and conducted with the same subjects, consists of:

1. two laboratory sessions concerning a comprehension task to be performed in a fixed time, asking the subjects to **answer to five comprehension questions on the object assigned, by looking at the Tropos or Tropos4AS models handed out** (built by modelling experts) and the respective textual requirements specifications;
2. a final questionnaire with questions on the subjective perception of subjects with respect to the experiment and the treatments.

The main dependent variable is the correctness of the subjects' answers to the comprehension questions, measured by comparing the answers given to gold standard answers, in terms of precision and recall.

Statistical evaluation. To determine if the null-hypotheses can be rejected and thus an affirmative answer can be given to the research questions, considering the nature of the variables and the experiment design, we apply a non-parametric, paired *Wilcoxon* test, adopting a 5% significance level for the obtained *p-values* (refer to [6] for details). Medians, averages and *Cohen.d* effect size are applied to analyze trends and to estimate the magnitude of the obtained results. Similar tests are used to evaluate the adequateness of the experimental settings by an analysis of the pre-questionnaires.

4 Conclusion

We described the design of an empirical study consisting of two controlled experiments, which aims to evaluate the extensions introduced by the Tropos4AS framework to the Tropos modelling language. The structured experimental set-up would be suitable in general to contribute to the evaluation of modeling language extensions.

We run the experiment with 12 researchers and PhD-students, with two small systems as objects and proper questionnaires. The analysis of the obtained data with the abovementioned statistical tests gave positive, mostly statistically significant results for both the expressiveness and the comprehensibility of Tropos4AS [7], while the modelling effort (except for looking up in the language specifications) seems not to be significantly higher than for Tropos. Analyzing possible threats to validity, a statistical evaluation (ANOVA test) of various co-factors (object, subject experience, subject position, laboratory) has shown that there was no significant impact on the obtained results. Conversely, some subjects reported difficulties in traditional Tropos modelling because of missing concepts in the language. A complete analysis of the results and the replication packages are available in [2].

We plan to complete the assessment, repeating the study with a higher number of subjects and evaluating the complete modelling process, e.g. by an off-line (observational) case study on the development of a system in a dynamic domain.

References

1. Morandini, M., Penserini, L., Perini, A.: Towards goal-oriented development of self-adaptive systems. In: SEAMS '08: Workshop on Software engineering for adaptive and self-managing systems, ACM (2008) 9–16
2. Morandini, M.: Goal-Oriented Development of Self-Adaptive Systems. PhD thesis, DISI, Università di Trento, Italy (March 2011) Available at http://eprints-phd.biblio.unitn.it/511.
3. Morandini, M., Penserini, L., Perini, A.: Operational Semantics of Goal Models in Adaptive Agents. In: 8th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'09), IFAAMAS (May 2009)
4. Penserini, L., Perini, A., Susi, A., Mylopoulos, J.: High variability design for software agents: Extending Tropos. ACM Transactions on Autonomous and Adaptive Systems (TAAS) **2**(4) (2007)
5. Hadar, I., Kuflik, T., Perini, A., Reinhartz-Berger, I., Ricca, F., Susi, A.: An empirical study of requirements model understanding: Use Case vs. Tropos models. In: SAC. (2010) 2324–2329
6. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in software engineering: an introduction. Kluwer Academic Publishers, Norwell, MA, USA (2000)
7. Morandini, M., Marchetto, A., Perini, A.: Requirements Comprehension: A Controlled Experiment on Conceptual Modeling Methods. In: Proceedings of the first Workshop on Empirical Requirements Engineering (EmpiRE11). (August 2011)