Environment Modeling for Complex, Dynamic and Distributed Systems

Fabian Kneer

Dortmund University of Applied Sciences and Arts,
Emil-Figge-Str. 42, 44227 Dortmund, Germany
fabian.kneer@fh-dortmund.de

Abstract. [**Context and Motivation:**] Complex, dynamic, and distributed systems confront requirements engineers with new challenges. The systems operate in dynamic environments with an increasing complexity. Modeling the context for understanding, developing and validate the requirements becomes a difficult task. [**Question/ problem:**] Current requirements engineering techniques for gathering information are not sufficient to capture the system behavior and the environment for this class of systems. [**Principal ideas/ results:**] A new approach is needed to support the development of an environment model, the idea is to use *runtime information* (e.g., from executing early prototypes of the system) to gather additional facts about the environment. [**Contribution:**] In this paper I present the motivation and challenges for the environment modeling for complex, dynamic, and distributed systems and an overview of the progress of my thesis. For this purpose I present my previous work on this field and discuss the proposed work for my thesis.

Keywords: Dynamic and distributed systems, environment models, runtime information

1 Introduction

The part of the real world that is relevant to understand a system is usually called its *(operational) environment* (we consider this term as synonymous to *context* in this paper). A proper understanding of the environment is a prerequisite to write requirements for a system.

There are characteristics of a system that complicate the gathering of information on the environment and its requirements. These characteristics are:

– *Complexity:* there is a constant trend that systems' complexity is increasing. One reason is the increasing amount of user requirements , while new legal regulations is one other. Besides the system, also the environment gets more complex, too. The trend towards more connectivity between systems is one reason.

- *Dynamic:* the behavior of a dynamic system is affected by changing users' needs and uncertainties in their operational environment. These systems often have a considerable lifetime and operate in different situations, which leads to evolving or changing requirements. A system needs feedback mechanisms to analyze the different environment situations and adapt to satisfy the requirements.
- *Distribution:* a distributed system consists of components located on networked computers, which communicate and coordinate their actions. They share different resources and capabilities to reach a common goal or collaborate to reach individual goals.

Examples for these characteristics can be found in the mobility area. A future car for instance will be connected with other cars or with parts of the traffic infrastructure, like intelligent traffic lights. Theses connected entities leads to a dynamic environment for the car with services and information that are not always available. If we look at autonomous driving, the complexity of the environment and the resulting requirements is immense. The mobility domain is just one example for systems with these characteristics. Other examples can be found in the Internet of Things (IoT) domain.

Problem. The classic RE techniques for gathering information are not fully sufficient to capture the environment and the requirements of such a system characterized above. One reason is that the information about environment and system behavior available from stakeholders is a priori incomplete. The missing information leads to incomplete specifications and incorrect software.

Due to the dynamic nature of such systems, requirements evolve during runtime and are a priori not completely available or not fully understood to elicit them with techniques like interviews.

Newer approaches to requirements engineering, e.g., from the field of self-adaptive systems [1], address the dynamic nature of the environment. However, the results of a systematical literature review that I performed, indicate that approaches mainly focus on *modeling* and not on gathering requirements or environment information (see Section 2 for details).

Goal. The goal of my thesis is to provide a better understanding of environment modeling for complex, dynamic and distributed systems. This includes (1) the identification of missing concepts of environment models and (2) the adoption of techniques to collect this information. The idea is to adopt techniques beyond the RE phase, more precisely from downstream software development: e.g., prototyping, simulation, and field test. An implementation of a RE tool to support the improved modeling process will be provided as a proof of concept.

The proposed work will address the following research questions:

[**RQ1**] What are the core concepts for dynamic and distributed systems to become represented in environment models and how to represent them?

[**RQ2**] What methods are useful to identify the relevant information for environment models beyond the RE phase?

[**RQ3**] What kind of information are needed by an requirements engineer to improve the development process and the adaption process? Sufficient information about the environment of a system are needed to develop, understand, and validate requirements. An adaptive system uses the environment information for a better decision space and to improve the overall adaption process.

Solution. Our first approach towards this goal was to weave runtime information about the requirements and variable parameters of a dynamic system back into the requirements specification, see [2]. The goal was to produce a better understanding of (1) the system behavior and (2) the evolution of the requirements and parameters. This information is useful to maintain and update a system and to increase the trust of the user in the adaptation, because he or she can understand the runtime decisions.

Based on this approach we consider an iterative, tool-supported process for gaining runtime information about the environment and the requirements, which is depicted in Figure 1. First, the requirements are gathered and an environment model will be modeled by a requirements engineer. Then, a prototyping and evaluation framework is used to support the development of a system. The frameworks can be used to generate probes for an application and the whole adaption mechanism (with integrated MAPE loop) out of a specification. It includes an instrumentation for simulation and field test, which collects runtime data on environmental entities.

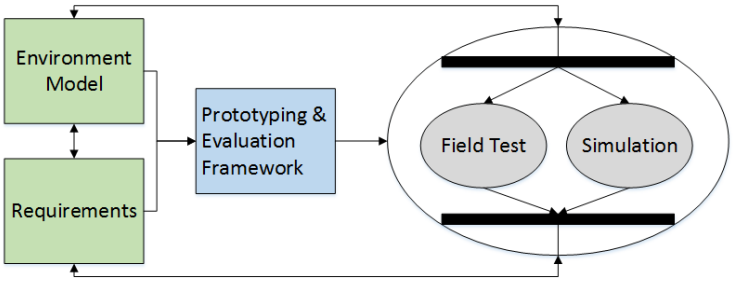

Fig. 1. Iterative Enhancement of Environment Model and Requirements

We are convinced that an early field test can help to find missing environment data and to improve the general knowledge about the system behavior. Simulations can be used as an alternative to the field test, if the environment model contains sufficient information.

Based on the information gathered during a field test or simulation - the requirements and the environment model enhanced in the next iteration. The before-mentioned information is gathered by tracing components inside of the prototype and by observations and feedback from the requirements engineer.

In my thesis I focus on the environment model as it is the most important artifact in the process. It is used for the generation of the prototype, to describe the environment for simulation, and to collect the environment information gathered during simulation and early field test.

The remainder of the paper is structured as follows. Sec. 2 represent a short summary of a systematic literature review on environment modeling. Sec. 3 describes the process of my PhD thesis with an overview on previous work, Sec. 3.1, and a discussion on the proposed work, Sec. 3.2.

2 Related Work

We performed a systematic literature review (SLR) on environment modeling for adaptive systems. The SLR provides first results for the research question [**RQ1**] und [**RQ2**]. The search strings are based on three main facets: research object - *environment model*, qualification - *adaptive* and restriction - *requirements engineering* and the alternative terms. We have chosen IEEE Xplore Digital Library, Springer Link and Google Scholar as databases for identifying the publications. At the start of the selection process we gathered 455 publications, which were tested by predefined selection and exclusion criteria. This process resulted in 58 relevant publications.

In the SLR, we considered elicitation and validation of the environment. Unexpected for us are the few results for these activities. Elicitation is addressed mostly with a few sentences that refer to interviews and meetings. Only one paper compares different elicitation methods like brainstorming, 6-3-5 method, six thinking hats, field observation, apprenticing and interviews. We expected more focus on field observation, because the information about the environment is a priori often not available as mentioned previously.

In conclusion, we argue that we need a systematic process for eliciting, modeling, and validating the environment: an environmental model provides a reference for requirements, it provides a basis to anticipate changes in the environment, it allows to derive test scenarios, and it helps to monitor the environment at runtime.

We identified four major modeling approaches: goal-based, agent-based, rule-based, and state machine modeling. As an example for an approach to model the environment we discuss the goal-oriented approach. A goal-oriented approach focuses on modeling the goals of an actor and how a goal can be achieved over alternative ways. The modeling process trying to find actors and models the relationship between them. There are no opportunities to model a system boundary. It is not relevant for the modeling approach if an actor is part of the system or not, it is important what parts belong to a specific actor and how the actors depend on each other. Additional elements or annotations are used to identify context elements.

Figure 2 shows two different ways to represent the environment (i* Example). First, it is a physical or virtual element in the environment of an *actor*, which is needed to achieve a goal of this actor. The element is represented as a *resource*

element [3]. To handle the complexity and dynamic of an environment, the approaches focusing on specification and specialization of the resource element by adding *attributes*, *annotations*, or *new siblings*. For example, an annotation as a clear label for an element that lies in the environment, or a specialization into *monitored resources*, which only provide information, or *controlled resources*, which can be changed from a system.

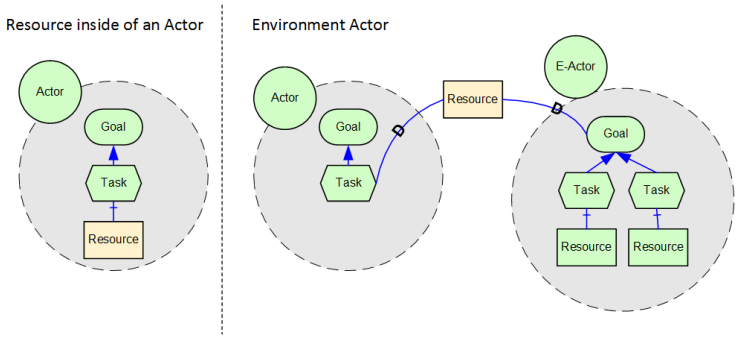

Fig. 2. Views on Environment (i* Example)

Second, the environment is represented as actor itself [4]. This leads to a more specific representation of the environment with detailed view of the goals in the environment and how it can achieve them. This also helps to work against the missing environment information, because the environment is not only seen as an interface which provides information or services - instead a more detailed representation of the environment element is presented. This indicates increasing need for more environment information.

3 Research Progress

The starting point of the thesis is the iterative enhancement process (shown in Figure 1) tailored for complex, dynamic, and distributed systems. Two empirical studies are planned to analyze and compare the identified approaches for environment modeling. Based on the results a new integrating environment model will be developed and integrated in a RE tool. The tool and the process will be analyze and validated with empirical studies.

In the remainder of this section the previous performed work will be presented and the plans for the rest of the thesis will be discussed.

3.1 Previous Work

As mentioned previously, our first approach in this area was a **feedback-aware requirements document** in [2]. The approach tries to bridge the gap between

development time and runtime representations of requirements in order to keep them consistent and to facilitate better understanding. We propose to weave the feedback from the runtime system into requirements documents using a domain-specific language that largely retains the informal nature of requirements. An annotated requirements document helps to get a better understanding of the system's actual behavior in a given environment. The approach is implemented using mbeddr, a novel set of domain-specific languages for developing embedded systems, and illustrated using a running example.

Next we suggested a **prototyping and evaluation framework for self-adaptive systems** in [5]. The goal of the framework is to ease the prototyping (and possibly development) of adaptive systems. For this purpose, the framework offers implementations of selected approaches to adaptive systems based on the MAPE loop. Another goal of the framework is to ease the evaluation of self-configuring systems, to allow for instance benchmarks between different approaches. The framework is able to collect data on a subset of the metrics at runtime about overall quality, effort, and cost.

For bench-marking we propose **a case from the IoT domain**, smart cities in particular, which comprises of hardware and software components, see [6]. Starting point is a smart street lighting system with communication between the lamps and passing cars. Our initial results of running a case study with a model-based prototyping framework on the smart street light are in [6]. The framework includes a software simulation of street lights and the events from the passing cars. The case can be used as a benchmark to compare several approaches in order to make a more informed decision which approach to choose.

A systematic literature review on environment modeling for adaptive systems was performed, in particular from a requirements perspective. We addressed the goals, the modeling concepts as well as the methodological aspects of environment modeling in our survey. As major results of our survey, we provide an ontology of typical goals of environment models in adaptive system research and a meta-model of existing environment modeling concepts. As a negative finding – and a research opportunity – we find that so far methodological aspects of environmental modeling for requirements engineering have received very little attention.

3.2 Proposed Work

Currently, I'm working on a more specific conception of the environment. This includes my view on relevant and irrelevant parts or information that should be included in an environment model - given the extended ways of using it.

I am planing to **enlarge the case study on IoT** as a base for empirical studies to evaluate the identified approaches for environment modeling from our survey. The advantages and disadvantages will be determined. Based on the gathered information a **general meta-model for environment modeling** will be developed.

As a result of my PhD thesis, I will develop a **prototype RE tool** based on the meta model that includes supporting mechanisms to create environmental models and to enrich this model by prototyping, simulation, and field test.

The resulting RE tool will be validated using the IoT case mentioned above and if possible at that time in cooperation with a local company.

4 Acknowledgment

This research was supported by graduate school of University of Applied Sciences and Arts, Dortmund.

References

[1] M. Morandini, L. Penserini, A. Perini, and A. Marchetto. "Engineering requirements for adaptive systems". In: *Requirements Engineering* 22.1 (2015), pp. 77–103.

[2] E. Kamsties, F. Kneer, M. Voelter, B. Igel, and B. Kolb. "Feedback-Aware Requirements Documents for Smart Devices". In: *Requirements Engineering: Foundation for Software Quality - 20th International Working Conference, REFSQ 2014, Essen, Germany, April 7-10, 2014. Proceedings*. Ed. by C. Salinesi and I. van de Weerd. Vol. 8396. Lecture Notes in Computer Science. Springer, 2014, pp. 119–134.

[3] N. A. Qureshi, I. J. Jureta, and A. Perini. "Towards a Requirements Modeling Language for Self-adaptive Systems". In: *Proceedings of the 18th International Conference on Requirements Engineering: Foundation for Software Quality*. REFSQ'12. Essen, Germany, 2012, pp. 263–279.

[4] J. P. Carvallo and X. Franch. "On the Use of i* for Architecting Hybrid Systems: A Method and an Evaluation Report". In: *The Practice of Enterprise Modeling: Second IFIP WG 8.1 Working Conference, PoEM 2009, Stockholm, Sweden, November 18-19, 2009. Proceedings*. Ed. by A. Persson and J. Stirna. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 38–53.

[5] F. Kneer and E. Kamsties. "A Framework for Prototyping and Evaluating Self-adaptive Systems - A Research Preview". In: *Joint Proceedings of REFSQ-2016 Workshops, Doctoral Symposium, Research Method Track, and Poster Track co-located with the 22nd International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2016), Gothenburg, Sweden, March 14, 2016*. Ed. by E. Bjarnason et al. Vol. 1564. CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[6] F. Kneer and E. Kamsties. "A Case Study on Self-configuring Systems in IoT Based on a Model-Driven Prototyping Approach". In: *Information and Software Technologies - 22nd International Conference, ICIST 2016, Druskininkai, Lithuania, October 13-15, 2016, Proceedings*. Ed. by G. Dregvaite and R. Damasevicius. Vol. 639. Communications in Computer and Information Science. 2016, pp. 732–741.