Utilizing Rule Deviations in IT Ecosystems for Implicit Requirements Elicitation

Leif Singer Olesia Brill Sebastian Meyer Kurt Schneider
Software Engineering Group
Leibniz Universität Hannover
{leif.singer, olesia.brill, sebastian.meyer, kurt.schneider}@inf.uni-hannover.de

Abstract

IT ecosystems are ultra-large-scale software systems that consist of various, constantly interacting and partly autonomous subsystems as well as the users of the overall system. Because of their strong integration with everyday life, these systems are often not even perceived as IT systems by its users. This is a problem for requirements engineering, as users might not know of or may not be interested in the capabilities of the system at all. This hinders the ongoing development of the system and might prevent new kinds of utilization and new business models from being realized.

By introducing rules into the infrastructure of IT ecosystems that are being monitored for adherence by agents interacting in the system, deviations from these rules can be harnessed for finding potential candidates for new or changed requirements. The deviations can be processed using techniques like data mining and pattern recognition and then forwarded to requirements engineers for review. They may then leverage these implicitly expressed requirements to identify actual changes in the needs of the users of the systems, enabling further advancements of the IT ecosystem.

1. Introduction

IT systems are constantly growing larger and more complex. They comprise agents that are adaptive and autonomous, creating interactions that are hard to anticipate. This paper uses the term *IT ecosystem* for these kinds of systems to discern and amplify these phenomena also found in biological systems.

The agents – human users, software services, and hardware devices alike – utilize an infrastructure provided by the IT ecosystem to communicate and interact with each other. To ensure certain minimum requirements and guide the overall goals of the IT ecosystem, this paper assumes an infrastructure which includes a rule system for monitoring and enforcing the system's agents' adherence to these requirements. Also, this paper assumes a base set of rules to already exist.

While some of these rules will be enforceable – such as requiring secure communications between all participants – others might rather resemble guidelines that would solely be beneficial, but not essential to adhere to, e.g., "cars should be optimally distributed on the parking site".

Human users have a special role in this setting, as they might not be aware of or interested in the system that surrounds them. Their interface to the IT ecosystem can be located somewhere in a spectrum that ranges from explicitly using a smartphone for communicating with other agents, to very subtle interaction like being recorded by surveillance cameras that try to detect the formation of crowds.

This lack of awareness, although justified, creates new challenges for requirements engineering. The stakeholders in IT ecosystems belong to groups ranging from service providers that can supply explicit requirements, to users who are not even aware of the system. The latter group cannot reasonably be expected to be interested in or invest effort into requirements elicitation activities. Especially in environments as complex as IT ecosystems though, requirements change all the time. Therefore, there is a strong need for alternative requirements elicitation methods suitable for this group.

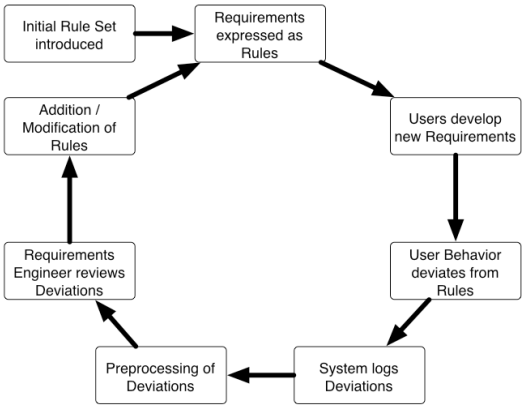

Figure 1: The iterative process as implied by the presented approach.

This position paper proposes a method that uses deviations from the aforementioned rules for finding new potential requirements. This approach, in its simplest form, requires no effort on the stakeholders' part and would be able to capture requirements knowledge the user isn't even aware of. It realizes an

iterative process for growing, refining and managing requirements knowledge, as illustrated in figure 1. Especially from the view of the stakeholders, this approach makes requirements elicitation a very light-weight activity.

The rest of this paper is organized as follows. The succeeding section will outline related work in the field of requirements engineering for ultra-large-scale systems. Section 3 gives an overview of common methods for requirements engineering and the challenges they face in the systems considered in this paper. Section 4 outlines a rule-based infrastructure for IT ecosystems, while section 5 uses this infrastructure to show how deviations from those rules might be used to derive new or changed requirements from the behavior of the users of the system. A scenario in which a system tries to propose good parking lots to the visitors of a SmartFair illustrates the approach. Section 6 details possible methods for finding good candidates from the rule deviations and proposes a process that requirements engineers would then follow to create actual changes to the rules of the system. The last section provides conclusions, shows the current state of the research of this approach and planned future work.

2. Related Work

The main idea of an IT ecosystem as an ultra-large-scale system is described by Herold et. al. [1]. The systems belonging to an IT ecosystem are able to adapt their behavior at runtime. For this adaptation, context information is especially important. This information is used in [1] to build an adaptive architecture for such a system. Nevertheless, the requirements have to be captured first, which is the focus of the approach described in this paper.

Usage context and user expectations have to be considered for requirements elicitation as well, for which Sitou and Spanfelner propose a set of possible models [2]. Context information not only includes location, but also participants, activities and environment [2]. The paper at hand considers the special challenges of a certain type of end users – those unaware of or not interested in the system – and proposes an approach that relies on the usage of data about the environment of the system, such as weather, time, location data, and others.

Georgas describes specifications for an architecture-based adaptation policy for self-adaptive systems in [3]. His approach utilizes rules and artificial intelligence to steer the actual adaption of systems, while the approach presented in this paper employs similar methods for requirements elicitation in systems comprised of autonomous, adaptive agents.

A solution to solve rule conflicts is described by Callele et. al. [4], who analyzed in-game justice systems of video-games. In some situations, they show, there is a need to override security requirements by emotional requirements to assure a game is still engaging its players. In IT ecosystems, there are also situations in which users have to deviate from guidelines to achieve security, e.g. in accidents. This strengthens this paper's assumption that the context in which rules are being evaluated is essential.

Huang and Mobasher propose an approach that leverages data mining methods to manage thousands of stakeholders in large projects [5]. While the order of magnitude of the number of stakeholders may be similar to the one of the approach presented in this paper, they assume a basic interest by the stakeholders, rendering their approach unsuitable for the situations considered in this paper, which assumes stakeholders to be completely disinterested or even unaware of the IT ecosystem surrounding them.

Mei and Easterbrook use goal models to derive new requirements [6]. The ideas presented in this paper could be used to refine those goal models. The new requirements found via the approach presented in this paper could be used to identify new soft goals and complete the structure of contribution in the goal model itself.

3. Requirements Elicitation in IT Ecosystems

In IT ecosystems, many different groups of stakeholders are present. Each of these groups has their own needs, behavioral peculiarities, and preferences. Examples for these groups are the system itself, service providers, or end users. The latter can still be subdivided into multiple groups – there might be end users with a strong affinity for technology, but there might also be those that do not perceive the IT ecosystem at all, merely unconsciously relying on the services it provides via interfaces like traffic lights or advertising boards. This group is the focus of the approach presented in this paper – the unconscious end users.

For each of these groups of users, different challenges might be encountered when eliciting requirements for the ecosystem. While the technophile might want to delve in technological detail during discussions, the unconscious end user might not only be unaware of the IT ecosystem, but also have no interest at all in helping in requirements elicitation. Also, end users might not even be aware of having any requirements. Nevertheless, to improve the ecosystem for its stakeholders, those requirements need to be found out.

Classical requirements engineering methods for elicitation, like customer interviews or workshops, do not work as well in the above situation. The lack of interest and awareness results in too little motivation

from end users for the mentioned classical methods to be effective. Material incentives, such as monetary rewards, would not truly improve the situation, as there is no intrinsic motivation that these extrinsic factors could support. Also, even if rewards could increase the end users' motivation, there still is a high chance they might not even be aware of their requirements, rendering direct questioning useless.

Requirements engineering methods in these environments therefore need to fulfill a set of different requirements themselves in order to be effective. Their cost to stakeholders, in terms of money, time, or anything else, should approach zero, as end users as those described beforehand do not consider themselves to be involved with the system at all. Also, as for the most part there will already be an IT ecosystem in place when facing the problems described, the elicitation methods should be able to find not only new, but also changes to previously existing requirements.

Maximizing unobtrusiveness also is an essential requirement for elicitation in the considered scenarios. The following paragraph illustrates a few methods pursuing that goal, before detailing the approach presented in this paper.

Questions could be integrated into the actual usage of the system's services and only ask the user whether their interaction with the system was satisfying. This could even be part of everyday devices like smartphones or navigation systems. Another approach to make the explicit questioning of users more light-weight could use short videos that show use cases and misuse cases acted out, bearing the potential for creating a lower barrier for interaction with stakeholders. These videos could be produced at low costs, as they would not be required to achieve a professional look – their sole purpose would be to enable the involvement of end users for elicitation and validation purposes.

This paper proposes a different approach that is even less obtrusive. It is based upon the assumption that there are rules in IT ecosystems that can be broken, and that these deviations from the rules could be used to derive new or changed requirements. Therefore, a rule system needs to be present in the system.

4. Rule-based Infrastructure

Due to the interactions occurring between autonomous, adaptive systems in an IT ecosystem, emergent effects not considered earlier might surface at some point. E.g., a system for guiding cars to parking lots could adapt to drivers who prefer places sheltered by trees. In winter, that adaption would lead to undesired system behavior, as now those lots near a building are more favorable, since low temperatures strengthen the desire for *short* walks from one's car.

In face of these possible conflicts, some goals should be assured in IT ecosystems – two of these are the reliability and the correctness of the overall system. A soft goal that may be opposed to these is the utility end users derive from using the system. The system thus needs to constantly strive after improving utility, as derived from the model the system has of the user, while still guaranteeing reliability and correctness. This conflict is not as present in traditional software development projects, as the adaptive capabilities of IT ecosystems are a constant challenge not found in those projects. The balance the ecosystem strives after must constantly be recreated.

To still be able to comply with these minimum requirements, this paper proposes a system of rules to constrain the impact of emergent effects. Using these rules, especially non-functional requirements could be expressed and consequently ensured by the IT ecosystem's infrastructure. Not all requirements would be expressed as rules, though – merely those appropriate for achieving the goals of reliability, correctness, and utility.

To weigh rules against each other, some semantic information must be associated with the rules. For example, an IT system should be able to differentiate negotiable soft goals and non-negotiable hard rules. Also, the temporal and spatial applicability of rules should be constrainable: Some rules might only be valid in winter, while some only apply in summer; also, rules might differ between parking lots and highways.

From these rule attributes, an order on the rules could then be derived. One possible order would assign the highest priority to those rules safeguarding life and limb of human participants in the system. Rules that are based on laws would follow, while those with lesser priorities would be rules that pursue softer goals like utility – e.g., finding the optimal parking lot, as illustrated in the following section. Such an order is a requirement for the automatic preprocessing of rule deviations for the subsequent handling by a requirements engineer, as shown in the following sections.

Hard rules should be enforceable system-wide, while soft goals should at least be monitored. Therefore, it is not an option to implement them in the system's agents themselves, as the agents' intentions cannot always be assumed to be favorable for the system. The rule system needs to be a part of the IT ecosystem's infrastructure, which acts as a communications channel between the different agents present in the system. This would prevent several rules from being circumvented – e.g., the infrastructure could just reject communications from known offending agents or ignore all messages not complying with the rules for secure messaging in the system. The infrastructure is a medium that all agents need to use for communication in the IT ecosystem and

therefore a suitable candidate for implementing rules in the absence of a central control instance.

5. Observing Rule Deviations

Since there are autonomous agents in an IT ecosystem, it may not be possible to enforce a rule. An autonomous agent, such as a human, a surveillance camera or a car, may at any time decide to ignore a specific rule to achieve a better outcome for itself. On the one hand, this may imply a new or changed requirement, which should then lead to a new rule or a variation of the old rule.

On the other hand, breaking a rule could simply mean some prohibited behavior occurred. For example, law does simply not allow breaking the speed limit in traffic.

In order to check which kind of rule deviation applies, it is necessary to log the rule, the action the autonomous agent took and the context in which the deviation took place. Additionally, successful appliances of the rules should be logged, so a relative measure for rule deviations can be obtained.

Rules that are based on law are usually not negotiable. Therefore, deviations from such rules will not lead to new or improved requirements and can be ignored. Generally speaking, rules that are less negotiable will have a lesser chance of their deviations leading to new requirements. Contrarily, rules that are soft goals are more likely to lead to new requirements when deviated from.

Rules that are not based on law can be negotiated most of the time. A deviation from such a rule should lead to a review that checks whether the underlying requirement has changed or whether a new requirement has formed.

As IT ecosystems are ultra-large-scale systems, deviations from rules happen too often to manually review each deviation. But as a rule-based environment relies on the correctness of its rules, it is still necessary to handle deviations somehow. Since reviews and refinements of rules and requirements have to be handled by human requirements engineers, the deviations have to be filtered and ordered in a useful way.

6. Extraction of Requirements

After rule deviations have been logged, it is possible to extract new requirements from them or refine those who already exist in the system.

We believe that autonomous agents that break a rule are not interested in actually specifying a new rule, leave alone new or changed requirements. Contrary to *engaged* users of ultra-large-scale software systems who are willing to write down their wishes in a forum [5], the approach presented in this paper tries to track down implicit requirements that have to be extracted from the logged rule deviations. This extraction has to be done manually by a requirements engineer. Therefore, it is necessary to preprocess the batch of logged deviations to make them accessible for manual processing.

We consider rules that get violated frequently to probably be good candidates for a reconsideration of the underlying requirements. Since the infrastructure would log rule deviations and rule adherences, the violation frequency may be calculated for obtaining a suitable order for reconsideration.

To extract new requirements or refine already existing ones, the deviations have to be checked for patterns. In this step, the context in which the deviation occurred is needed. A scenario in the following paragraphs illustrates this.

Take for example a rule that allocates a parking lot on the premises of a *SmartFair* for a visitor of the fair. The rule says the parking lot should be as near to the entrance as possible. The underlying requirement is that one would walk too long to the entrance when there are free parking lots in front. Inspecting the rule violations, we find out that most of the drivers did not choose the proposed parking lot, but a different one further away from the entrance. Ignoring the context in which the deviation happened would not lead to a sufficient explanation. But taking the logged context into account, we can find out that it was a sunny and warm day. So all drivers searched for a parking lot that was in the shadows, and those were not near the entrance.

This observation leads to the new requirement that on warm days, the parking lot should be in the shadows and therefore to a refined rule. Employing approaches like clustering, data mining, and pattern recognition, finding that correlation should be a reasonable task for a computer system. It could at least find likely candidates for possible correlations and present those to the human requirements engineer with added weight.

As shown in this short example, it is essential to find out and document the reasons *why* a certain rule is broken in order to create rules that conform to the individual wishes of autonomous agents.

For human engineers to be able to determine possible reasons and to refine the user model that the system uses, the context of the rule deviations needs to be recorded. To be able to decide what to record, the system should have a model of the environment which is based on an ontology that relates objects and data in the system to each other – e.g., the system could know the concept of a driver, who is related to a car, which in turn might have a weather sensor that the system might query for context data. This ontology would then need to be iteratively refined.

This context data could also be used for improving light-weight, unobtrusive feedback mechanisms. E.g.,

the system might, at the moment of the deviation, notice that apart from the deviation, just two other factors A and B differ from situations in which the rule was respected. Therefore, the user is presented with a feedback form on his smartphone asking whether factor A, factor B, or something else entirely made them change their behavior. This could be restricted to situations in which the system has already built a cluster of similar deviations into which the current deviation would fit. Otherwise, there still is a chance of it being just an outlier and therefore not a candidate for a new requirement.

As the requirements engineer creates or refines the rule, he documents the rationales leading to the altered rule system. This step can be used to create or extend an experience component such as an experience base. This not only leads to new or improved requirements and therefore better adapted rules, but also helps dealing with new deviations in the future by installing a system for quality improvement [7].

The reasons leading to the observed rule deviations will not necessarily be obvious even to human users. New rules or modifications of existing rules should therefore not be automatically created by a computer system. Nevertheless, an experience component can help the requirements authors by proposing possible correlations of rule deviations and by contrasting aspects of the stored contexts against each other. Each additional link between a rule deviation and its context could be used to clarify and expand the network of connections between the agents interacting in an IT ecosystem, making it easier to recognize correlations for both a computer system and a human requirements engineer – consequently making it easier to find new requirements and thereby advancing the system itself.

7. Conclusions and Outlook

In IT ecosystems, the main interest of end users is deriving actual utility from the system. But at the same time, the system needs to make sure everything is working reliably and correctly. For an IT ecosystem to deal with these conflicting priorities, this paper assumes a system of rules that are used to represent some of the requirements.

Since, on the one hand, requirements in such huge and complex systems change constantly, and on the other hand end users might have no interest in requirements engineering at all, a method is needed to harmonize these conflicting circumstances.

To achieve that goal, this paper proposes utilizing the behavior of end users by analyzing the deviations from rules for finding candidates for potential new or changed requirements. This method suits the considered scenario, as it requires no or almost no interaction with the end user while still being able to provide potential new or changed requirements.

The approach presented in this paper is still in its early stages and could not be evaluated yet. In a joint project with two other universities from Lower Saxony in Germany, the authors of this paper are currently creating the foundations for a prototype that will be used to evaluate the presented approach.

The current focus is building a suitable rule-system supporting the approach. Subsequently, data mining and pattern recognition techniques will be selected and evaluated to derive the potential changes in requirements from rule deviations.

References

[1] S. Herold, H.Klus, D. Niebuhr, A.Rausch, "Engineering of IT Ecosystems: Design of Ultra-Large-Scale Software-Intensive Systems", *Proceedings of the 2nd international workshop on Ultra-large-scale software-intensive systems(ICSE 2008),* ACM, 2008, pp. 49-52.

[2] W.Sitou, B.Spanfelner, "Towards Requirements Engineering for Context Adaptive Systems", in *Proceedings of the 31st Annual International Computer Software and Applications Conference (COMPSAC 2007),*IEEE Computer Society, 2007,pp. 593-600.

[3] J.C.Georgas, "Knowledge-Based Architectural Adaptation Management for Self-Adaptive Systems", in *Proceedings of the 27th International Conference on Software Engineering Doctoral Symposium (ICSE 2005)*, St. Louis, 2005, p. 658.

[4] D. Callele, E. Neufeld and K. Schneider, "Balancing Security Requirements and Emotional Requirements in Video Games", *16th IEEE International Requirements Engineering Conference*, Barcelona, Spain, 2008, pp. 319f.

[5] J.C.Huang, B.Mobasher, "Using data mining and recommender systems to scale up the requirements process",in *Proceedings of the 2nd international workshop on Ultra-large-scale software-intensive systems (ICSE 2008)*, 2008, pp.3-6.

[6] L. Mei, S. Easterbrook, "Evaluating User-centric Adaptation with Goal Models", in *Proceedings of the 1st International Workshop on Software Engineering for Pervasive Computing Applications, Systems, and Environments (ICSE 2007)*, IEEE Computer Society, 2007.

[7] V. R. Basili, "The Experience Factory: packaging softwareexperience", in *Proceedings of the 14th annual Software Engineering Workshop,* NASA Goddard Space Flight Center, GreenbeltMD 20771, 1989