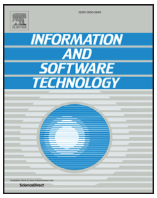
Information and Software Technology 000 (2017) 1–16

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



A semi-automated approach for generating natural language requirements documents based on business process models

Banu Aysolmaz[a,b,*], Henrik Leopold[a], Hajo A. Reijers[a], Onur Demirörs[c,d]

[a] Vrije Universiteit Amsterdam, Department of Computer Science, De Boelelaan 1105, 1081HV Amsterdam, The Netherlands
[b] Maastricht University, School of Business and Economics, PO Box 616, 6200 MD, Maastricht, The Netherlands
[c] Izmir Institute of Technology, Department of Computer Engineering, 35430, Urla, Turkey
[d] University of New South Wales, School of Computer Science and Engineering, Barker St, Kensington NSW 2052, Australia

ARTICLE INFO

Article history:
Received 9 March 2017
Revised 6 July 2017
Accepted 18 August 2017
Available online xxx

Keywords:
Requirements elicitation
Business process model
Natural language generation

ABSTRACT

Context: The analysis of requirements for business-related software systems is often supported by using business process models. However, the final requirements are typically still specified in natural language. This means that the knowledge captured in process models must be consistently transferred to the specified requirements. Possible inconsistencies between process models and requirements represent a serious threat for the successful development of the software system and may require the repetition of process analysis activities.

Objective: The objective of this paper is to address the problem of inconsistency between process models and natural language requirements in the context of software development.

Method: We define a semi-automated approach that consists of a process model-based procedure for capturing execution-related data in requirements models and an algorithm that takes these models as input for generating natural language requirements. We evaluated our approach in the context of a multiple case study with three organizations and a total of 13 software development projects.

Results: We found that our approach can successfully generate well-readable requirements, which do not only positively contribute to consistency, but also to the completeness and maintainability of requirements. The practical use of our approach to identify a suitable subcontractor on the market in 11 of the 13 projects further highlights the practical value of our approach.

Conclusion: Our approach provides a structured way to obtain high-quality requirements documents from process models and to maintain textual and visual representations of requirements in a consistent way.

1. Introduction

Business process modeling is an established method for documenting, analyzing, and improving organizational operations. What is more, it has become a widely accepted practice in software engineering [1–3]. In particular for analyzing requirements of business-related software systems business process modeling has proven to be an effective means [4]. Process models do not only provide an overview of the operations that must be supported by the to-be developed software systems, but also show how these operations are related to the different organizational roles and systems.

Despite this prominent role of business process modeling for requirements analysis, the actual specification of requirements is commonly conducted using natural language [5–8]. This means that the knowledge captured in process models must be consistently transferred to natural language requirements. On the one hand, this is a complex and time-consuming task [9,10]. On the other hand, updates at later stages in either the textual or the model-based requirements come with the risk of inconsistencies [11–13]. Such inconsistencies between the process model and the resulting requirements represent a serious threat for the successful development of the respective software system throughout the Software Development Lifecycle (SDLC). More specifically, they may result in a system that does not fully reflect the functionality defined in the process models.

To address this problem, we propose a semi-automated approach whose final output are generated requirements documents

[*] Corresponding author:
E-mail addresses: b.aysolmaz@maastrichtuniversity.nl, banuays@gmail.com (B. Aysolmaz), h.leopold@vu.nl (H. Leopold), h.a.reijers@vu.nl (H.A. Reijers), onurdemirors@iyte.edu.tr (O. Demirörs).

http://dx.doi.org/10.1016/j.infsof.2017.08.009
0950-5849/© 2017 Elsevier B.V. All rights reserved.

that integrate process model and execution-related data in an understandable fashion. As a result, organizations can systematically transfer the knowledge captured in their process models to other SDLC activities and create consistent and maintainable artifacts. Our proposed approach consists of three main steps. In the first step, we analyze the process models that are relevant for the system to be developed and identify the set of automatable activities. In the second step, we capture execution-related data, such as responsibilities, application systems, data needs, and additional constraints in a requirements model. In the third step, we automatically generate requirements documents from the created models via a template-based natural language generation algorithm. The consistency between the processes and the requirements is by definition guaranteed by the generation feature of the approach. To evaluate the impact of our approach on other key characteristics of high-quality requirements–readability, completeness and maintainability–, we conducted a multiple case study that involved 3 different organizations and a total of 13 software development projects. We found that the requirements documents generated by our approach were considered to be well-readable, almost perfectly complete, and beneficial for improving consistency as well as maintainability. Meeting these key requirement characteristics was found to be essential to enhance the usability of the requirements by domain experts, analysts, project managers, and software developers. In 11 of the projects, the generated artifacts were used for identifying a suitable subcontractor on the market for developing the respective systems, which confirmed the usability of the approach in practical settings.

The remainder of this paper is structured as follows. In Section 2 we elaborate on the background of our research and identify the research gap that we will address. In Section 3, we introduce our semi-automatic approach for generating requirements documents. In Section 4, we present and discuss the findings of our multiple case study. In Section 5, we elaborate on the steps required for adapting the presented approach to languages other than English. In Section 6 we discuss the implications of our work before concluding the paper in Section 7.

2. Background

In this section, we discuss the background of our paper. In Section 2.1, we first clarify the relevance and the value of generating natural language requirements. In Section 2.2, we then elaborate on the use of process models in requirements engineering. We close the section by pointing out what is still missing to define an approach for automatically generating high quality requirements from process models.

2.1. The value of requirements generation

While many would argue that models are the preferred means to foster communication, others favor requirements in textual format. At its heart, the question about the value of generating natural language requirements relates to the debate whether textual or visual representations are superior in terms of communication effectiveness. Interestingly, this debate is neither new nor limited to the field of requirements engineering. The first studies addressing this controversy date back to the seventies. At this time, psychologists empirically compared the expressive power of natural language texts with matrices, spatial maps, and tree representations [14–17]. Later, many studies from the field of computer science contributed to the debate. Among others, authors compared the comprehension performance of code-based representations and flow diagrams [18–20]. The conclusions of these and other works remain, however, contradictory. Some argue in favor of text-based other argue in favor of visual representations.

A satisfying explanation for these opposing views is provided by the Cognitive Theory of Multimedia Learning (CTML) [21], which has been developed through more than a decade of empirical research. Among others, it discusses the concept of *learning preference*, which suggests that both textual and visual representations should be presented at the same time. The rationale behind this concept is that people with different backgrounds may simply have different preferences and cognitive abilities. By providing both representations, they are provided with a choice.

Transferred to the field of requirements engineering, the CTML suggests that both models and natural language requirements should be used for capturing and discussing requirements. In fact, this view is supported by many researchers. For instance, Weber and Weisbrod discuss the importance of natural language requirements for communication, but also highlight that the sole use of natural language is hardly feasible for complex projects [22]. They propose the additional use of so-called requirements management information models (RMIs). In a similar way, Schatz et al. [23] and Davis [24] propose to combine text-based and model-based requirements. Nicolás and Toval even explicitly discuss the value of generation in this context [25]. They argue that generation reduces the effort and, at the same time, increases the quality and traceability of the requirements.

Recognizing the potential of automatically generating natural language requirements, we define a respective approach for process models in this paper. To highlight what is specifically missing to define such an approach, the next section reviews related work on process models in the context of requirements engineering.

2.2. Process models and requirements engineering

Many authors have emphasized the important role of process models in the context of specifying requirements of software systems [26–28]. Some authors even go so far as considering their use as mandatory [1,3]. However, the specific role of process models differs considerably among available approaches. Table 1 gives an overview of the most relevant works using process models in the context of requirements engineering. As Table 1 illustrates, we differentiate between works that use process models in a manual and in an automated way.

The related work that discusses *the manual use of process models* in the context of requirements engineering can be further categorized into works that elicit *textual* and that elicit *model-based* requirements from process models.

The main insight of the works from the first subcategory that elicit *textual* requirements from process models is that process models represent an effective way of steering the activity of requirements elicitation and enhance the completeness, correctness, and traceability of the final requirement statements [4]. Cardoso et al. analyze the level of automation for each activity in the process models and then define a set of textual requirements for the activities to be automated [4]. In a similar manner, Ma and Jiang define a set of textual requirements for each activity of a process [7]. Mayr et al. discuss that detailed notions for requirements should be specified based on process models and they also map requirements in sentence form to the process models [28]. Li et al. propose a method to link textual requirements to activities in the process model [8]. Such links help to identify dependencies between requirements consecutively being used for discovering missing and ambiguous text-based requirements. Demirörs et al. analyze and define not only functional requirements, but also non-functional, security, and hardware requirements based on process models. Lastly, Monsalve et al. elaborate on the usage of process modeling notations for eliciting and expressing user requirements on a strategic level. They find Qualigram more helpful in this respect than BPMN [30]. What all these works have in common is