Information and Software Technology 58 (2015) 110–122

Contents lists available at ScienceDirect

Information and Software Technology

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

# Automated events identification in use cases

J. Jurkiewicz *, J. Nawrocki

*Poznan University of Technology, Institute of Computing Science, ul. Piotrowo 2, 60-965 Poznań, Poland*

ARTICLE INFO

*Article history:*
Received 22 January 2014
Received in revised form 18 September 2014
Accepted 22 September 2014
Available online 6 October 2014

*Keywords:*
Requirements engineering
Use cases
Functional requirements
Natural language processing

ABSTRACT

*Context:* Use cases are a popular method of expressing functional requirements. One contains a main scenario and a set of extensions, each consisting of an event and an alternative sequence of activities. Events omitted in requirements specification can lead to rework. Unfortunately, as it follows from the previous research, manual identification of events is rather ineffective (less than 1/3 of events are identified) and it is slow.

*Objective:* The goal of this paper is to propose an automatic method of identification of events in use cases and evaluate its quality.

*Method:* Each step of a main scenario is analyzed by a sequence of NLP tools to identify its performer, activity type and information object. It has been observed that performer, activity type and some attributes of information objects determine types of events that can occur when that activity is performed. That empirical knowledge is represented as a set of axioms and two inference rules have been proposed which allow to identify types of possible events. For each event type an NLG pattern is proposed which allows to generate description of the event type in natural language. The proposed method was compared with two manual approaches to identification of events: *ad hoc* and HAZOP-based. Also a kind of Turing test was performed to evaluate linguistic quality of generated descriptions.

*Results:* Accuracy of the proposed method is about 80% (for manual approaches it is less than 1/3) and its speed is about 11 steps/minute (ad hoc approach is 4 times slower, and HAZOP-based approach is 20 times slower). Understandability of the generated event descriptions was not worse than understandability of the descriptions written by humans.

*Conclusions:* The proposed method could be used to enhance contemporary tools for managing use cases.



## 1. Introduction

Requirements highly influence different aspects of software development [23,24,29,31]. One of the approaches to specifying functional requirements is the concept of use case. It was proposed by Jacobson [20] and further developed by Cockburn [5,12] and other authors [19]. Roughly speaking, a use case is a description of user-system interaction expressed in natural language. The focal element of a use case is the main scenario consisting of a sequence of steps that describe the mentioned user-system interaction. While a step is being executed, some events might happen. Descriptions of those events along with alternative scenarios are also included in a use case.

The question of how to identify a (more or less) complete list of possible events arises. Research shows that non-identified events (some call them "missing logic and condition tests for error handling" [37]) are the most common cause of changes in requirements which result in requirements volatility and creeping scope. The importance of event-completeness has been recognized by many authors (although they do not use this specific term), including Carson [11], Cox et al. [13], Mar [28] and Wiegers [41]. The approach they recommend is based on identification of events performed by experts using brain storming or other "manual" techniques. There is also a method called H4U [22]. It is a form of a HAZOP-based review, which aims at identifying events in use cases. The main disadvantage of manual identification of events is the time and effort required by such a process. Thus, a question arises whether it is possible to identify events automatically and obtain good quality event descriptions in the sense of event-completeness and their readability.

The above question is also interesting from the artificial intelligence point of view. The following saying is attributed to Pablo Picasso: *Computers are useless. They can only give you answers* [2]. Identifying of events that might happen during execution of a

* Corresponding author.
*E-mail addresses:* jjurkiewicz@cs.put.poznan.pl (J. Jurkiewicz), jnawrocki@cs.put.poznan.pl (J. Nawrocki).

http://dx.doi.org/10.1016/j.infsof.2014.09.011
0950-5849/

use-case step resembles a dialog in which a human is saying a simple sentence (e.g. *This evening I am going to the opera house.*—that corresponds to a use-case step), and a computer is asking a question (e.g. *What if all the tickets are sold?*—that question includes the event *all the tickets are sold*). To ask such event-related questions requires the understanding of a sentence, i.e. some knowledge is necessary (in this example it is knowledge about *opera* and *tickets*), and the ability to infer about events with use of that knowledge.

In this paper a method of automatic identification of events in use cases is presented. First, in Section 2, the model of use-case based requirements specification is outlined. The central part of the method is the inference engine that was used for the generation of event names (event descriptions). It is discussed in Section 3. The mechanism for analysing a use-case step with a Natural Language Processing (NLP) tool is presented in Section 4. The events are generated using Natural Language Generation (NLG) techniques—see Section 5. Empirical evaluation of the proposed approach is discussed in Section 6. Related work is discussed in Section 7.

## 2. Use-case based requirements specification

In this paper it is assumed that a functional requirements specification document contains the following elements:

- Actors – descriptions of actors interacting with the system being built.
- Information Objects – presentation of domain objects the actors and the system operate on. Every information object can have a set of different properties assigned to it (the possible properties are described in Section 3).
- Use Cases – descriptions of user-system interactions in the form of scenarios performed in order to achieve user's goals. Every use case consists of a name describing a user's goal and a main scenario consisting of a sequence of steps. Steps in the main scenario may be interrupted by events described within extensions. Additionally, alternative scenarios describe how given events should be handled. An exemplary use case is presented in Fig. 1.

Use cases have been used in both research and industry since their introduction in 1985 [20]. Research shows that scenario-based notations (including use cases) are the most popular way of describing customer requirements [31]. Two main factors are responsible for their popularity: firstly, the clear focus on describing the goals of the system; secondly, natural language which the requirements are expressed in. The latter makes it easier for the customer and prospective end-users to understand the requirements document, however, it makes it hard to process the specification in an automatic way.

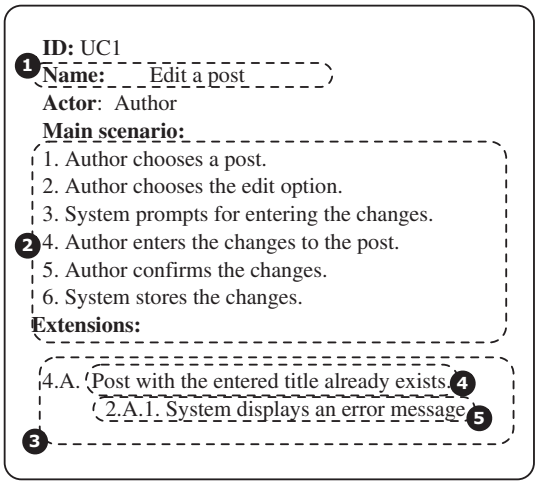


**Fig. 1.** Example of a textual use case. Main elements: 1 – name, 2 – main scenario section, 3 – extensions section, 4 – an event, and 5 – an alternative scenario.

## 3. Inference engine for identification of events

### 3.1. The problem

It is assumed that a functional requirements specification on the input contains descriptions of actors, descriptions of information objects, and use case stubs, i.e. use cases without their extension parts (see Fig. 1). To complete use case stubs with event descriptions the following problem is discussed in this paper:

**Events identification problem**: Given a step from the main scenario of a use case identify all the events that can happen when the step is executed. Only events which can be detected by the system are of interest (that viewpoint is supported by Adolph et al. [5] in their *Detectable conditions* pattern).

### 3.2. Abstract model of a use-case step

The process of event identification consists of four phases:

1. Preprocessing of the description of actors and information objects that creates a dictionary of these elements (it resembles the creation of a symbol table by a compiler during parsing).
2. Analysing a step using NLP techniques (a main scenario is analysed step by step).
3. Inferring the possible events for a given step (that results in abstract descriptions of events which are independent of natural language).
4. Generating event descriptions in natural language.

The overview of this process is presented in Fig. 2. The most important part of the presented method is inferring about the possible events.

Our approach to infer about events is based on a model of human–computer interaction, which evolved from our study of available use cases and events associated with them. Our aim was to provide the simplest possible model which would allow to infer about all the events we have found in the considered use cases. The model is twofold and is illustrated in Fig. 3.

**Model of the real world** (Fig. 3A). Some events are a consequence of a state in which a real object exists (e.g. a store is empty). The model represents real world objects (e.g. a book in a library, an article in a newspaper). What is interesting, for the purpose of inferring events, quite a simple tool proved to be powerful enough: it is the set theory. A fragment of the real world we are interested in can be viewed as individual objects (i.e. items) or collections of objects (i.e. sets). Furthermore, items might be simple or they might be compound and consist of other items. It is important to notice that some activities do not change the real world – they just provide us with information about the world (state) *as-is*. On the other hand, there are some activities which change the world (i.e. its state) and they lead to a new state (*as-to-be*). In our view, without awareness of the last category of activities, identifying some events would be impossible. Moreover, the state of a real world objects can change over time, e.g. a credit card can expire on some day.

**Model of IT system** (Fig. 3B). Some events are associated with human–computer communication or computer-computer interaction (e.g. an external service is not available). This