Identifying Ambiguity Problems in User Stories : A Proposed Framework

Anis Amna

Ghent University, Tweekerkenstraat 2, 9000 Ghent, Belgium

Abstract

The Requirements Engineering (RE) community has addressed ambiguity issues in natural language-based requirements as a critical yet unresolved problem. While some studies argue that ambiguity is an intrinsic problem in natural language-based requirements, our review study indicates that the ambiguity in user stories is linguistics and cognitive problems. However, the questions on why cognitive factors trigger ambiguity have not been extensively studied and how to envisage ambiguity in user stories is not yet standardized. Our study aims to answer those questions by proposing a framework to help users identify ambiguity in user stories. This paper broadly describes our research progress and the research plan to do an experiment involving advanced students in identifying ambiguity in user stories. Further directions of the research are also discussed.

Keywords [1]

Agile Requirements Engineering, User stories quality, Ambiguity

1. Introduction

The pattern of ambiguity in natural language-based requirements is a challenging issue that has attracted many scholars. Textual requirements are inherently ambiguous, making complete elimination of ambiguity difficult. Moreover, the involvement of people from different domains makes the requirements vulnerable to misinterpretation, replication, or incompleteness [4]. However, Chantree et al. [5] argue that some ambiguity could be left as it is as long as it does not lead to misunderstanding (i.e., *innocuous ambiguity)*.

The studies attempting to disambiguate natural-based requirements have presumably stood in this perception. Thus, those aiming to detect, reduce, and eliminate ambiguities have targeted textual requirements that lead to *nocuous ambiguity*. These studies have typically utilized various NLP techniques. Most of those solutions have been tested in controlled experiments, while fewer studies have explored how they might be implemented in practice [2].

For user stories as a type of textual requirements, a systematic literature review that we conducted indicates that 22% of the studies address ambiguity in user stories (36 of 165 studies in total) [1]. Among those numbers, a limited number of the studies (7 of 36 reviewed studies) have examined individual characteristics (e.g., education, experience, main occupation) to avoid misunderstanding between business people and developers [16], minimize the risk of missing requirements [6], and improve user stories consistency with system behavior [18]. A few studies have examined the correlation between human-related (i.e., cognitive) factors and user stories representation [9], while similar studies in detecting, removing, or reducing ambiguity in user stories have not been found.

In this context, user stories are used to describe expected system behavior (e.g., [6, 16]), with reader's knowledge (i.e., developers) implicitly positioned as an intuitive understanding of the intended meaning of user stories. Failure to correctly interpret the intended meaning of user stories may lead to

Proceedings of the Doctoral Consortium Papers Presented at the 34th International Conference on Advanced Information Systems Engineering (CAiSE 2022), June 06–10, 2022, Leuven, Belgium
EMAIL: AnisRahmawati.Amna@UGent.be (A. Amna)
ORCID: 0000-0002-6966-9234 (A. Amna)

inconsistencies in system architecture, insufficient (or incomplete) requirements, or feature duplication [1]. Consequently, embracing contextual knowledge while formulating user stories and/or interpreting user stories meaning is important to avoid arbitrary interpretations, especially when domain experts have different experiences or intentions [2, 11]. In addition, enhancing human-related abilities to select the appropriate terminology that would fit the reader's knowledge, allow a better understanding of the domain, and support learning and discovery process will be useful to avoid ambiguity in user stories.

Providing these situations, our study seeks to fill this research gap by scrutinizing ambiguity issues in user stories from a cognitive perspective. We are interested in examining the relationship between human-related factors and ambiguity in user stories and identifying the extent to which the factors have led to multiple interpretations, inconsistency in system architecture, insufficient (or incomplete) requirements, and feature duplication. We define criteria to help users detect ambiguity in user stories by pinpointing ambiguous user stories and the negative impacts on requirements quality (i.e., vagueness, inconsistency, insufficiency, duplication) [1]. In our future study, we will perform an experiment to observe whether the proposed framework can be useful in helping users identify potential problems that might lead to ambiguity in user stories.

1.1. Research Objectives

Our study aims to examine human-related factors having led to different ambiguity issues in user stories. The insight will be valuable to investigate the role of human-related factors in identifying nocuous ambiguity in user stories. In order to achieve that benefit, we define our objectives as follows: (1) explore the relationship between human-related factors and linguistics problems related to ambiguity in user stories, (2) identify how human-related factors have caused multiple interpretations, system inconsistency, requirements insufficiency, and feature duplication, (3) develop an artifact (e.g., framework, conceptual model, algorithm, prototype) to assist users in identifying different types of ambiguity (i.e., lexical, syntactic, semantic, pragmatic) in user stories, (4) evaluates the effectiveness of the cognitive artifact to help users in achieving a better understanding of user story meaning and improve learning and discovering processes.

1.2. Research Questions

According to the research objective, the research questions are composed as follows:

- RQ1: To what extent and why do human-related factors contribute to linguistics problems related to ambiguity in user stories?
- RQ2: To what extent and why do human-related factors contribute to multiple interpretations of user stories and cause requirements quality problems?
- RQ3: How to eliminate the human-related factors that have been identified as causing factors of ambiguity during user stories formulation and/or review?
- RQ4: How to evaluate the effectiveness of the proposed solution to avoid nocuous ambiguity in user stories?

2. Research Methodology

The research questions will be answered following this research method: First, we identify human-related factors that possibly influence reader’s interpretation while formulating and/or reviewing user stories. At this stage, we performed a literature review on the application of cognitive theories in Human-Computer Interaction (HCI) [8], software engineering (SE) [8], decision making [17], and requirements elicitation [15]. The result was then documented as cognitive variables that can be included as variables in our research.

Next, we describe what kind of potential problems frequently occur in user stories. To do so, we reflect on our previous study [1] to structurize the problems and classify those according to linguistic levels and requirements quality problems. We also review previously proposed frameworks for user story quality (i.e., the QUS framework [13]) and requirements quality in Agile software development

(i.e., Agile Requirements Verification framework [7]) to develop our framework for detecting problems with user stories that could lead to ambiguity.

Afterward, we perform an experiment to test whether the framework is effective in helping users identify potential problems in user stories that might trigger ambiguity. Apart from testing effectiveness, the experiment's data will be analyzed to identify which cognitive factors impact the identification of potential ambiguity problems in user stories. Based on this analysis, we consider adjusting the framework and developing a tool to help users improve their awareness while formulating and/or reviewing user stories.

3. Ambiguity Problems in User Stories

Being unambiguous has been cited as one of the numerous qualities that high-quality software requirements should possess [7, 13]. In user stories context, Heck and Zaidman [7] define unambiguity as a full sentence without spelling or grammatical error. Meanwhile, Lucassen et al. [13] confer unambiguity as terms or abstractions in user stories that can be interpreted in a single meaning. This latter definition is widely used in RE community to identify (nocuous) ambiguity due to negative impacts on requirements and software quality [1, 5, 10].

Among those definitions, linguistic aspects of user stories have been pointed to as the main focus. It is reasonable, as selecting appropriate terminology that fits the reader's knowledge is important to comprehend the domain for which software is developed better and improve learning and discovery processes [12]. However, other studies also recognized that *unacknowledged* ambiguity might negatively impact requirements quality [1, 5]. This oversight is presumably caused by a lack of reader's ability to comprehend the correct meaning of user stories [5, 11]. Unfortunately, this drawback has not been explicitly covered in the available standards.

We recently conducted a systematic literature review to observe the state-of-the-art studies addressing ambiguity in user stories [1]. The study reveals that linguistic ambiguities in user stories lead to some requirements quality problems. Further, ambiguity problems related to requirements quality can be attributed to linguistic aspects of user stories and cognitive processes for comprehending the intended meaning of user stories.

As a result of the identified requirements quality problems, we rearrange the criteria and categorize them in accordance with the linguistic levels at which ambiguity problems might arise. Our framework extends the QUS framework [13] and Agile Requirements Verification framework [7] by presenting criteria for ambiguous user stories by externalizing the user stories context. Using the framework, we expect developers to identify problems in user stories that potentially cause ambiguity.

3.1. Ambiguity Problems in User Stories Related To Requirements Quality: a Proposed Framework

In order to identify ambiguity in user stories, we used the definition published in our systematic literature review to identify ambiguity problems in user stories [1]. The study classified ambiguity problems according to linguistic aspects (i.e., lexical, syntactic, semantic, pragmatic) and the aftermath of ambiguity problems that affected requirements quality. Classification is used to establish a framework that specifies each definition and the potential violations while identifying potential problems that could lead to ambiguity problems in user stories.

The first problem that we mostly identified in user stories is vagueness [1]. Vagueness is defined as the use of inappropriate or non-standard terminology to formulate user stories and the failure of user stories to follow a particular template such as Connextra: *"As a <role>, I want to <goal>, so that <reason>"*. In Berry and Kamsties [3], the terminology problem was identified as *homonymy* and *polysemy* problems. Meanwhile, Lucassen et al. [13] identified the problem as *ambiguity* at the semantic level, *minimal issue* at the syntactic level, and *full sentence and uniformity issue* at the pragmatic level. Our framework adopted these concepts and put *no-homonymy and no-polysemy* as lexical criteria that user stories should meet. As for avoiding vagueness caused by grammatical problems, user stories should be written in an *atomic* structure (i.e., do not contain compound sentences and are free from

grammatical errors). To avoid misinterpretation from the readers (i.e., developers) to comprehend the intended meaning of user stories, they should be *explicitly* expressed using the terminology of the domain [7] and *uniformly* written in a similar format [13].

Inconsistency is another problem that often hides until the next stages of RE activities (e.g., requirements specification, requirements validation) [1]. In software development, this problem arises as an inaccurate system architecture that might cause software compliance problems. Inconsistency persists due to complexity in the user stories structure or a contradiction between the goal element of user stories and the requested feature. Lucassen et al. [13] address this problem by the *well-formed* criterion at the syntactic level of user stories and the *conflict-free* criterion at the semantic level. Compared to those, Heck and Zaidman [7] reckon consistency as no contradiction or conflicting feature requests. Our framework adopts those concepts and adds new criteria for the pragmatic level, namely *collective understanding*. This is inspired by the *navigable link* criterion from Heck and Zaidman [7], which is useful for checking the consistency of user stories and other conceptual models. However, this original definition might not suit Agile Software Development (ASD) as the methods encourage the use of minimal artifacts to document requirements. Therefore, despite *navigable link*, we construct *collective understanding* criteria to remind users that user stories should be comprehended as a set of requirements, and unclear words and/or phrases should be clarified through other user stories.

Our review study identified that insufficiency problems in user stories are typically caused by high turnover [6], lack of experience [19], and inability to comprehend the expected meaning of user stories [14] as the causes. The QUS framework [13] accommodates these problems in the *conceptually sound* criterion, while the Agile Requirements Verification framework [7] addresses those with *annotation availability*. However, despite *conceptually sound* and *annotation availability*, we construct *flexible* and *traceable* criteria to remind users that user stories should allows flexibility while implementing user stories into system features, and at the same time they should be able to traced back with other user stories or requirements artifacts (e.g., conceptual models).

Requirements duplication is the last identified problem related to ambiguity in user stories. While this problem is less investigated, our review study found several similarity analyses have been used to identify the potential duplication in user stories [1]. Hence, each user story should represent one required functionality. This is presented as the *unique* criterion, similar name to the QUS framework [13], while the definition follows the Agile Requirements Verification framework [7] of the *correct summary* criterion.

Table 1.
Example of the framework to identify user stories problems that are possibly lead to ambiguity

RE-related problem	Criteria (according to linguistic level)	Description	Violation
Vagueness	***Lexical***		
	No-polysemy	User stories should not use individual words or phrases having multiple related meanings	User stories use individual words or phrases having more than one related meaning
Inconsistency	***Syntactic***		
	Well-formed	User stories should be free from grammatical error	User stories have at least one grammatical error, such as a missing role or goal, or an unclear antecedent that is referred to by pronouns
Insufficiency	***Semantic***		
	Traceable	User stories context should correlate to others.	User stories context does not correlate to others.
Duplication	***Pragmatic***		
	Unique	User stories should describe the unique functionality of the systems	More than one user stories have the same functionality

The framework criteria will be used as a guide for (novice) users to identify problems in user stories and specify what requirements quality could be impacted if the ambiguity problems are not acknowledged until the late stage of software development. The example of user stories problems that

could potentially impact ambiguity and how to deal with the problems are presented in **Table 2** as follows.

Table 2.
Example of user stories compliance to the framework

RE-related problem	Criteria (according to linguistic level)	Description	Explanation
Vagueness	***Lexical***		
	No-polysemy	*"As a site member, I want to modify my profile[*], so that only my name will appear" (US1)*	The word *"profile"* in *US1* can be interpreted as *"user profile"* or *"company profile"*
		"As a site member, I want to modify my company profile[*], so that only my company name will appear" (US1c)	The word *"profile"* in *US1* can be changed to *"company profile"* to clarify the meaning
Inconsistency	***Syntactic***		
	Well-formed	*"As a site visitor, I want to get a notification of all[****] upcoming certification course[****,**] and[*] page through courses[**] if there is a lot, so that I can choose one[**] for me" (US2)*	[*]) compound sentence problem can be avoided by splitting *US2* into *US2a* and *US2b* [**]) The word *"certification course", " courses", and "one"* in *US2* have unclear antecedent
		"I want to page through them if there is a lot[***]" (US2c)	[***]) sentence fragment problem can be avoided by completing *US2c* into *US2b*
		"As a site visitor, I want to get a notification of all[****] upcoming certification courses[****], so that I can choose the best course[****] for me" (US2a)	[****]) pronoun disagreement and reference problem can be avoided by revising *US2* into *US2a*
		"As a site visitor, I want to page through them if there is a lot[***], so that I can choose the best course for me[*****]" (US2b)	[*****]) *US2b* is not considered a violation if we remove WHY_dimension
Insufficiency	***Semantic***		
	Traceable	*"As a site member, I want to inactivate my profile[*], so that the profile cannot be found" (US3)*	[*]) Profile inactivation in *US3* should be done with the approval of the site administrator in *US4*
		"As a site administrator, I want to approve company profile inactivation[*], so that I can approve profile inactivation request" (US4)	
Duplication	***Pragmatic***		
	Unique	*"As a site member, I want to modify[*] my profile, so that only my name will appear" (US1)*	[*]) only one user story is allowed; one should be removed
		"As a member, I want to edit[*] my profile, so that I can select my identity that will appear" (US1a)	

4. Research Progress

At the first stage of our study, we reviewed 36 of 165 relevant studies addressing ambiguity in user stories [1]. The result shows that ambiguity issues have been caused by the characteristics of user stories that are intrinsically ambiguous [2]. Our review study revealed that the contextual meaning of user stories could not be entirely grasped by automatic detection, and Natural Language Processing (NLP) techniques do not sufficiently fulfill this problem. It is because human-related (i.e., cognitive) factors have contributed to triggering user stories misinterpretation [9, 20].

Considering these, we are interested in exploring the extent to which the human-related factors have contributed to user stories problems that could lead to ambiguity. Despite the definition of requirements ambiguity proposed by Kamsties [11], there is no well-acknowledged standard to assess potential ambiguity problems in user stories. Therefore, we propose a framework to help (novice) developers identify different types of ambiguity problems in user stories and understand the implication of those problems to the contextual meaning of user stories.

4.1. Research Novelty

Despite the fact that human-related factors have significantly involved software performance [6, 19], a limited number of studies addressing ambiguity in user stories have limitedly explored the factors. This situation resulted in unconfirmed knowledge regarding human-related factors and ambiguity problems in user stories that emerged as requirements quality problems in the later stages of RE activities (e.g., requirements specification, requirements validation).

We aim to fulfill the gap by investigating human-related factors in causing ambiguity problems in user stories. In order to do so, we systematically reviewed the studies addressing ambiguity problems in user stories [1]. The study indicated that textual ambiguity that emerged at different linguistic levels had caused requirements quality problems, including vagueness, inconsistency, insufficiency, and duplication. Providing this fact, we use these consequences to define ambiguity problems in user stories and construct quality criteria for user stories to avoid potential problems (see **Table 1**).

Our study is distinct from others in proposing our new framework based on the QUS and Agile Requirements Verification framework to help users identify ambiguous problems in user stories that could trigger requirements quality problems [1]. We will set up an experiment relying on manual analysis to observe whether the framework is helpful in identifying problems in user stories that potentially lead to ambiguity.

5. Further Direction

We will perform an experiment involving advanced students to observe whether our proposed framework helps identify different types of ambiguity problems in user stories from different linguistic levels. The experiment consists of four main parts: short questionnaires about participant background, a survey regarding the conceptual understanding of ambiguity in user stories, an experimental task to identify different types of ambiguity problems in user stories, and a questionnaire related to user feedback on the usefulness of the framework to identify ambiguity problems in user stories.

In the first part, we will distribute questionnaires to understand participant profiles. The participants will be asked to describe their primary occupation, educational background, age, and gender. The second part will introduce our proposed framework as quality metrics to identify different types of ambiguity problems that potentially occur in user stories. We will carry out a survey to observe the participants' understanding of the concepts and accomplish an experimental task to identify different types of ambiguity problems in user stories. At the end of the experiment, a reflective questionnaire will be distributed to capture participant feedback regarding the usefulness of our proposed framework to help them identify different types of ambiguity in user stories. We will ask participants to reflect on their general attitude during the experiment to identify which cognitive factors have influenced them to identify ambiguity problems in user stories.

6. Acknowledgements

This research is supervised by Prof. Geert Poels, Ghent University, Belgium.

7. References

[1] Amna, A.R., Poels, G.: Ambiguity in user stories : A systematic literature review. Inf. Softw. Technol. 145, January, 106824 (2022).

[2] Bano, M.: Addressing the challenges of requirements ambiguity: A review of empirical literature. In: 2015 IEEE Fifth International Workshop on Empirical Requirements Engineering (EmpiRE). pp. 21–24 IEEE (2015).

[3] Berry, D.M., Kamsties, E.: Ambiguity in Requirements Specification. Perspect. Softw. Requir. 7–44 (2004).

[4] Buchan, J. et al.: Applying Distributed Cognition Theory to Agile Requirements Engineering. In: Requirements Engineering: Foundation for Software Quality. pp. 186–202 Springer International Publishing (2020).

[5] Chantree, F. et al.: Identifying nocuous ambiguities in natural language requirements. In: 14th IEEE International Requirements Engineering Conference (RE'06). pp. 59–68 (2006).

[6] Dilorenzo, E. et al.: Enabling the Reuse of Software Development Assets through a Taxonomy for User Stories. IEEE Access. 8, 107285–107300 (2020).

[7] Heck, P., Zaidman, A.: A Quality Framework for Agile Requirements: A Practitioner's Perspective. (2014).

[8] Hollan, J. et al.: Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research. ACM Trans. Comput. Interact. 7, 2, 174–196 (2000).

[9] Jia, J. et al.: Understanding software developers' cognition in agile requirements engineering. Sci. Comput. Program. 178, 1–19 (2019).

[10] Kamsties, E. et al.: Detecting Ambiguities in Requirements Documents Using Inspections. (2001).

[11] Kamsties, E.: Understanding Ambiguity in Requirements Engineering. In: Aurum, A. and Wohlin, C. (eds.) Engineering and Managing Software Requirements. pp. 245–266 Springer Berlin Heidelberg, Berlin, Heidelberg, Heidelberg (2005).

[12] Lindland, O.I. et al.: Understanding quality in conceptual modeling. IEEE Softw. 11, 2, 42–49 (1994).

[13] Lucassen, G. et al.: Improving agile requirements: the Quality User Story framework and tool. Requir. Eng. 21, 3, 383–403 (2016).

[14] Melegati, J., Wang, X.: QUESt: new practices to represent hypotheses in experiment-driven software development. In: Proceedings of the 2nd ACM SIGSOFT International Workshop on Software-Intensive Business: Start-ups, Platforms, and Ecosystems - IWSiB 2019. pp. 13–18 ACM Press, New York, New York, USA (2019).

[15] Nickles, K.R.: Judgment-based and reasoning-based stopping rules in decision-making under uncertainty. Diss. Abstr. Int. Sect. A Humanit. Soc. Sci. 56, 3-A, 1005 (1995).

[16] Ordóñez, H. et al.: An Impact Study of Business Process Models for Requirements Elicitation in XP. In: International Conference on Computational Science and Its Applications. pp. 298–312 Springer Verlag (2015).

[17] Poore, J.C. et al.: Personality, cognitive style, motivation, and aptitude predict systematic trends in analytic forecasting behavior. J. Cogn. Eng. Decis. Mak. 8, 4, 373–393 (2014).

[18] Rocha Silva, T. et al.: Evaluating the usage of predefined interactive behaviors for writing user stories: an empirical study with potential product owners. Cogn. Technol. Work. 1–21 (2019).

[19] Wautelet, Y. et al.: Evaluating the Impact of User Stories Quality on the Ability to Understand and Structure Requirements. In: IFIP Working Conference on The Practice of Enterprise Modeling. pp. 3–19 Springer International Publishing (2019).

[20] Wautelet, Y. et al.: On Modelers Ability to Build a Visual Diagram from a User Story Set: A Goal- Oriented Approach. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). pp. 209–226 (2018).