IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 1, 2021, accepted March 28, 2021, date of publication April 2, 2021, date of current version April 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3070606

User Stories and Natural Language Processing: A Systematic Literature Review

INDRA KHARISMA RAHARJANA[1,2], (Graduate Student Member, IEEE),
DANIEL SIAHAAN[1], (Member, IEEE), AND CHASTINE FATICHAH[1], (Member, IEEE)

[1]Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya 60111, Indonesia
[2]Information Systems, Universitas Airlangga, Surabaya 60115, Indonesia

Corresponding author: Daniel Siahaan (daniel@if.its.ac.id)

This work was supported by the Ministry of Research and Technology/National Research and Innovation Agency through Penelitian Disertasi Doktor under Grant 27/E1/KPT/2020.

ABSTRACT *Context:* User stories have been widely accepted as artifacts to capture the user requirements in agile software development. They are short pieces of texts in a semi-structured format that express requirements. Natural language processing (NLP) techniques offer a potential advantage in user story applications. *Objective*: Conduct a systematic literature review to capture the current state-of-the-art of NLP research on user stories. *Method:* The search strategy is used to obtain relevant papers from SCOPUS, ScienceDirect, IEEE Xplore, ACM Digital Library, SpringerLink, and Google Scholar. Inclusion and exclusion criteria are applied to filter the search results. We also use the forward and backward snowballing techniques to obtain more comprehensive results. *Results:* The search results identified 718 papers published between January 2009 to December 2020. After applying the inclusion/exclusion criteria and the snowballing technique, we identified 38 primary studies that discuss NLP techniques in user stories. Most studies used NLP techniques to extract aspects of who, what, and why from user stories. The purpose of NLP studies in user stories is broad, ranging from discovering defects, generating software artifacts, identifying the key abstraction of user stories, and tracing links between model and user stories. *Conclusion:* NLP can help system analysts manage user stories. Implementing NLP in user stories has many opportunities and challenges. Considering the exploration of NLP techniques and rigorous evaluation methods is required to obtain quality research. As with NLP research in general, the ability to understand a sentence's context continues to be a challenge.

INDEX TERMS Agile software development, natural language processing, systematic review, user story.

I. INTRODUCTION

User stories are increasingly gaining a place in the software development process, especially in agile software development. User stories are the most widely used artifact in agile software development [1], [2] that express requirements from the user's point of view.

A user story is a semi-structured specification of requirements written in natural language. A user story template may take the following form [3]: *as* [WHO], *I want/want to/need/can/would like* [WHAT], *so that* [WHY]. It contains important elements of requirements: *WHO* wants it, *WHAT* is expected from the system, and optionally, and *WHY* it is important [3], [4].

The rise of agile software development has attracted researchers and practitioners into this research field [1], [5], [6]. User stories, as the most widely used artifact in agile software development, are challenging to explore. The fact that they are written in natural language makes them easily understandable to stakeholders. However, requirements written in natural language have drawbacks, such as ambiguity, inconsistency, and incompleteness [7]–[9].

Natural language processing (NLP) techniques offer potential advantages to improve the quality of user stories. NLP can be used to parse, extract, or analyze user story data. It has been widely used to help in the software engineering domain (e.g., managing software requirements [10], extraction of actors

The associate editor coordinating the review of this manuscript and approving it for publication was Hui Liu.

and actions in requirement document [11], software feature extraction [12], software testing [13], etc.).

Some studies have used the NLP approach applied to user stories to accelerate the software requirements process. As a new research field, it is interesting to obtain a clear understanding of NLP research on the user story direction.

This study aims to provide insight for researchers and practitioners about the state-of-the-art research related to the role of natural language processing on user story specification. This study also provides a future research direction related to user stories. Align with agile manifesto, i.e. uncovering better ways of developing software, this systematic literature review is conducted to achieve these objectives. Our specific objectives are to understand what research topics of the user stories have been explored, including the methods and tools used. The challenges of NLP research in user stories would also be identified.

The remainder of this paper is organized as follows: Section 2 provides an overview of the user story concept, followed by a brief overview of NLP; Section 3 provides a review of existing survey (review) papers on NLP and user story research; Section 4 presents the objectives, research questions, and review methods; Section 5 outlines the key findings of our study; Section 6 provides a discussion on the findings and identifies the study limitations; and finally, Section 7 draws the study conclusions.

II. USER STORY AND NLP

A user story is a short, semi-structured sentence that illustrates requirements from the user's perspective. A user story can be used to explain user desire or product description [14]. It consists of three aspects, namely aspects of who, what, and why. The aspect of "who" refers to the system user or actor, "what" refers to the actor's desire, and "why" refers to the reason (optional in the user story). These aspects are arranged into one sentence with a certain structure. Several formats/templates are usually used, including

As a <aspect of who>, I want <aspect of what>, so that <aspect of why>

As a <aspect of who> I need < aspect of what >, so that <aspect of why>

As a <aspect of who> I can < aspect of what >, so that <aspect of why>

In order to < aspect of why > as a <aspect of who>, I can <aspect of why>

The user story components consist of the following elements [15]:

Role: abstract behavior of actors in the system context; the aspect of who representation

Goal: a condition or a circumstance desired by stakeholders or actors

Task: specific things that must be done to achieve goals

Capability: the ability of actors to achieve goals based on certain conditions and events

NLP is a computational method for the automated analysis and representation of human language [16]. The use of NLP for software engineering tasks has become popular with the increasing volume of data from software artifacts. Examples of applications include requirement reuse [17], requirement ambiguity detection [18], requirement classification [19], [20], and sentiment analysis [21].

NLP techniques are usually used for text preprocessing (e.g., tokenization, Part-of-Speech (POS) tagging, and dependency parsing). Several NLP approaches can be used (e.g., syntactic representation of text and computational models based on semantic features). Syntactic methods focus on word-level approaches, while the semantic focus on multiword expressions [16].

III. RELATED SECONDARY STUDIES

No conducted secondary studies have focused on the user story's specification to the best of our knowledge. Several secondary studies related to this area focus on several issues/aspect/area (i.e., agile requirements engineering [1], [5], quality requirement management in agile software development [22], the evolution of use cases [23], and requirements engineering in model-driven development [24]). Table 1 summarizes these works.

Schön *et al.* and Inayat *et al.* [1], [5] conducted a literature study related to agile requirements engineering. Schön *et al.* [1] focused on stakeholder and user involvement, while Inayat *et al.* [5] focused on adapting agile requirements engineering practices. These studies differed from ours because they focused on the general part of agile requirements engineering, while we focused on user stories as one of the agile requirement artifacts.

Behutiye *et al.* [22] conducted a review that covered quality requirement management in agile software development. User stories, which are artifact requirements widely used in agile software development, were not specifically discussed. The quality elements in the user story were discussed by [25], who focused on the quality criteria for evaluating the correctness of written agile requirements.

Tiwari and Gupta [23] reviewed studies related to the evolution of use cases. Use cases are artifacts with almost the same functions as user stories. They stated that use cases increasingly utilize formal structures to facilitate software development life cycle (SDLC) activities. It is interesting to compare the development of use cases and user stories to obtain an appropriate comparison. Loniewski *et al.* [24] conducted a review study related to the use of requirements engineering techniques for model-driven development. The natural language (NL) requirements are usually used for the automation of the SDLC process.

Bakar *et al.* and Nazir *et al.* [26], [27] conducted a review study related to NLP application in engineering requirements. Bakar *et al.* [26] focused on extracting NL requirements for reuse in software product line engineering. Nazir *et al.* [27] focused on NL application in software requirements.

Although the related literature studies written in this section provided good information regarding requirements engineering, no studies focused on the NLP application in

TABLE 1. Related secondary studies.

Reference	Goal	Concerns in research questions
[21]	Systematically review several study cases on the use of requirements engineering methods within the model-driven development (MDD) practices and their level of automation	- Requirement approaches, methods, and best practices applied within MDD practices - Automation level of MDD in each practice
[20]	Review state-of-the-art studies on the evolution of the use cases, their implementations, their quality, open problems, and potential future directions	- Evolution of use cases - Most suitable document template - Variations of use case applications in several software development life cycle activities - Assessment methods of use case specification quality - Unsolved issues in use case specification - Future research direction in use case specification
[23]	Review cutting-edge methods, approaches, techniques, and best practices used in feature extractions from requirements represented in natural language	- Methods, approaches, techniques, and best practices to extract features from the requirement specification written in natural language
[4]	Understand how agile requirements engineering can be used as solutions to solve problems introduced during software development by traditional requirement techniques and methods Generate mapping between the adopted practices in requirement specifications and issues sealed by software development teams	- Best practices of requirements engineering in agile methodology - Challenges of traditional requirements engineering methodology - Open issues in agile requirements engineering
[1]	Identify cutting-edge approaches and methodologies of agile requirement specification with an emphasis on the stakeholder and user involvement	- Cutting-edge requirements engineering approaches and methodology that focus on stakeholder involvement and agile software development compliance - User perspective-based agile methodologies - Commonologies among requirement management approaches, methods, and best practices
[24]	Review the role of NLP in various aspects of requirement specification	- Popular software domains where NLP methods and techniques are applied - NLP activities applied in requirement specification - Popular CASE or tools recommended by requirements engineers and researchers
[22]	Identify the requirement quality attributes for assessing the suitability of agile requirement specification	- Requirement quality attributes for agile requirement specifications
[19]	Incorporate cutting-edge approaches, methods, and applications on quality requirement (QR) management in agile software development (ASD) and rapid software development (RSD)	- QR management in ASD and RSD - QR management approaches and methods in ASD and RSD - RQR management challenges and issues in ASD and RSD

user stories. Understanding current studies in this field can be beneficial for researchers when identifying future studies.

IV. REVIEW METHOD

We adopted procedures from [28] and [29] in preparing the SLR comprising three stages: review planning, conducting, and reporting. The 2009 PRISMA Checklist was adopted as a guide in writing this SLR report [30].

A. REVIEW PLANNING

We planned a review by identifying the research questions relevant to the objectives. We determined the search strategy and defined the detailed inclusion and exclusion criteria.

1) OBJECTIVES AND RESEARCH QUESTIONS

The rise of agile software development (ASD) research has led to the increase of research related to user stories, which are the most widely used artifacts in ASD. The user story format that uses natural language makes the NLP application an effective approach in user story research. As a new research area, it is interesting to know the direction of user story research that applies to NLP methods and techniques. This study mainly aims to survey the state-of-the-art use of NLP in user stories. We formulated the following research questions to fulfill these objectives:

RQ1: What are the uses of NLP for user stories?

RQ2: What are the approaches available in research related to NLP in user stories?

RQ3: What are the challenges of using NLPs in user story research?

2) SEARCH STRATEGY

We obtained relevant studies by identifying keywords, creating a search string, and defining a database and search parameters.

The set of keywords was determined based on the objectives and research questions, specifically the uses, approaches, and challenges of using NLPs in user story research. We identified two main categories to determine keywords based on objectives and research questions: ' natural language processing' and 'user story.' We pinpointed alternative spelling and synonyms to acquire comprehensive results. Table 2 lists the final set of keywords.

TABLE 2. Keyword used for search.

Category	Keywords
Natural language processing	natural language processing, natural language, NLP
User story	user story, user stories

We then connected the set of keywords using Boolean operators, such that the complete search string derived is

("natural language processing" OR "natural language" OR "NLP") AND ("user stories" OR "user story")

We made minor adjustments to the search string based on the electronic database characteristics. These adjustments were done without changing the determined set of keywords (e.g., making the search string lowercase, applying the search items only in the form of research articles if possible, and limiting the publication period from January 2009 to December 2020). We limited the publication period to only the last ten years in hopes of obtaining the latest state-of-the-art researches. Table 3 presents the details of the adaptation of the search string application in the electronic database.

TABLE 3. Search sources.

Electronic databases	SCOPUS ScienceDirect SpringerLink IEEE Xplore ACM Digital Library Google Scholar
Searched items	Research articles
Language	English
Publication period	January 2009 to December 2020
Search applied on	Abstract, title and keywords (SCOPUS, SpringerLink, IEEE Xplore) Full text (ScienceDirect, ACM Digital Library, Google Scholar)

3) INCLUSION AND EXCLUSION CRITERIA

We used the inclusion and exclusion criteria to select relevant studies.

Inclusion criteria: the study (I1) is a peer-reviewed publication, (I2) in English, (I3) published between January 2009 and December 2020, and (I4) related to the search terms specified (describing user stories using NLP).

Exclusion criteria: (E1) short papers, doctoral symposium papers, summary of conference keynotes, proposals, lecture notes, editorials, comments, tutorials, and review papers, and (E2) published in a predatory journal or conference.

We used abstracts, titles, and keywords to evaluate papers based on the inclusion and exclusion criteria for initial screening. When necessary, we also opened the full text of the paper to evaluate the inclusion and exclusion criteria.

We then downloaded the full text of relevant studies to re-assess the inclusion and exclusion criteria. We filtered out studies not in compliance with the criteria. Studies that fit our criteria were marked as primary studies. We eliminated redundant studies. With this approach, we can be more effective in choosing papers for primary studies.

4) BACKWARD AND FORWARD SNOWBALLING

We used the snowballing technique to acquire more comprehensive results and reduce the risk of missing relevant studies [31]. We applied backward and forward snowballing for each identified primary study. Backward snowballing was done by examining the reference list from the primary studies to pinpoint additional papers. Forward snowballing was accomplished by examining other papers citing primary studies. Each primary study identified is a subject of further backward and forward snowballing process.

B. CONDUCTING THE REVIEW

This section presents the results of the study search and selection process. We also present the quality assessment results herein.

1) STUDY SEARCH AND SELECTION

We searched the following online libraries based on the predefined search strings: SCOPUS, Elsevier ScienceDirect, SpringerLink Online Library, IEEE Xplore, ACM Digital Library, and Google Scholar.

We ran the search on electronic databases sequentially to make the search effective. First, we searched SCOPUS and recorded the results in a spreadsheet and Mendeley. Chronologically, the search was followed by that on ScienceDirect, SpringerLink, IEEE Xplore, ACM Digital Library, and Google Scholar. Some databases provide CSV file download features that simplify this task. We ran the screening process by checking the titles, abstracts, and keywords and applying the rules of the inclusion and exclusion criteria. Relevant papers were marked on a spreadsheet, downloaded, and included in Mendeley software. We also ensured that no redundant studies used this approach.

Searches on SCOPUS and Google Scholar were performed at the beginning and the last because both search engines are abstract indexing, collecting data from many sources. SCOPUS was used as the starting point because its data are curated. Google Scholar was used last because the search results had the most results [32]. The other databases included in the digital library category (e.g., ScienceDirect, SpringerLink, IEEE Xplore, and ACM Digital Library) were searches between SCOPUS and Google Scholar; hence, the paper that appears can be easily identified in case of redundancy, reducing efforts to manage redundant papers. Papers related to RQ also have a high likelihood of being discovered in this SLR.

A total of 64 relevant studies were found using this method. The full text of studies was assessed for eligibility. This assessment was done by reviewing the inclusion and exclusion criteria once again and confirming whether the article was eligible for the SLR topic. Thirty primary studies were identified.

The backward and forward snowballing techniques were applied after discovering the primary studies. For the backward snowballing, we used a reference list to obtain the relevant studies. Simultaneously, for the forward snowballing, we checked to see the citations of the selected studies in Google Scholar. For the initial screening, we read the title of the reference or citation to decide whether the studies were relevant. We downloaded the full text of the relevant study candidates to assess them using the inclusion and exclusion criteria. Fifty-two candidates were identified for the relevant studies. Three studies were added to the primary studies after applying the inclusion and exclusion criteria. Fig. 1 presents the study search and selection process.

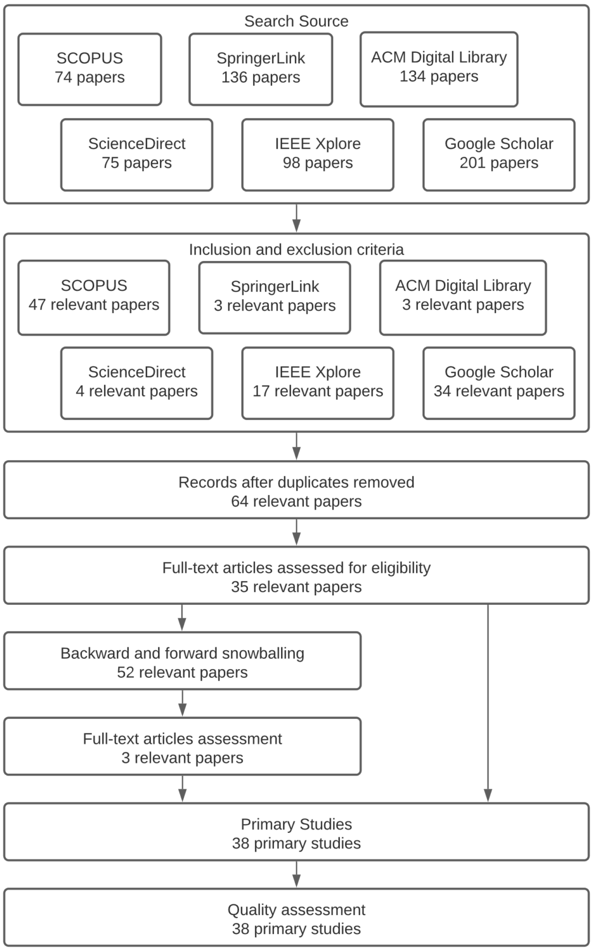

FIGURE 1. Study search and selection process.

2) QUALITY ASSESSMENT

We used quality assessment to evaluate the methodological quality of the primary studies. We adopted the quality assessment applied by [1]. Table 4 presents the checklist used to evaluate the quality of the included studies.

All primary studies (38 papers) were assessed based on the quality assessments (Table 4). The first item (QA1) assesses the purpose of each study. This question was answered positively in 92% of the studies. The second item (QA2) assessed if the study presents a detailed description of the approach. This question was responded to positively in 87% of the studies. The third item (QA3) asks about a validation method of the result. Only 26% of the studies employed appropriate validation methods. The fourth item (QA4) assesses if studies are based on research rather than opinion or viewpoint. Only 28% of the studies responded positively. The final item (QA5) searches for the number of citations obtained by studies. Consequently, 46% of studies were cited more than five times by other studies. Fig. 2 shows the quality assessment scores of the primary studies.

TABLE 4. Quality criteria for the study selection.

Item	Assessment criteria	Score	Description
QA1	Was there a clear statement of the objectives of the research?	−1	No, the objectives were not described.
		0	The objectives were partially but unclearly described.
		1	Yes, the objectives were well described and clear.
QA2	Does the research introduce a detailed description of the proposed solution or approach?	−1	No, details were missing.
		0	Partially, if you wish to use the approach or solution, you must read the references.
		1	Yes, the approach can be used with the presented details.
QA3	Is the proposed solution or approach validated?	−1	No, it was not validated.
		0	It was partially validated in a laboratory, or only portions of the proposal were validated.
		1	Yes, by a case study.
QA4	Does the research present an opinion or viewpoint?	−1	Yes, it does.
		0	Partially because the corresponding work was explained, and the paper was set into a specific context.
		1	No, the paper was based on research.
QA5	Has the study been cited in other scientific publications?	−1	No, no one cited the study.
		0	Partially. Between one and five scientific papers cited the study.
		1	Yes, more than five scientific papers cited the study.

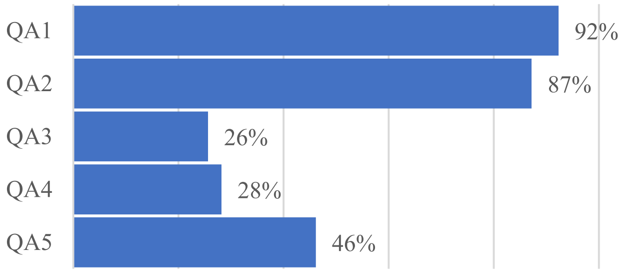

FIGURE 2. Percentage scores for the quality assessments of the studies.

3) DATA EXTRACTION AND SYNTHESIS

The data extraction was performed to obtain information relevant to the research question. The data were extracted following a predefined extraction form (Table 5). Using this form enabled us to record the full details of primary studies to address our research question.

TABLE 5. Data extraction form.

#	Study data	Description	Relevant RQ
1	Identifier	Unique ID for the study	Study overview
2	Title		Study overview
3	Authors		Study overview
4	Year		Study overview
5	Article source		Study overview
6	Type of article	Journal, conference, workshop, book chapter	Study overview
7	1st author country		Study overview
8	Application context	Industrial, academic	Study overview
9	Date of data extraction		Study overview
10	Research goal	What is the contribution of the study?	RQ1
11	Research goal category	Conceptual model extraction, software artifacts from user stories, user story similarity, priority, and size estimation, user story quality, user stories extraction	RQ1
12	Research methods	What research methods did the study employ?	RQ2
13	Data	What data did the study use?	RQ2
14	Validation	What validation technique did the study apply?	RQ2
15	NLP technique	What NLP technique did the study use?	RQ2
16	NLP tools	What NLP tool did the study use?	RQ2
17	Challenge and limitation	What challenges and limitations did the study acknowledge?	RQ3
18	Future work	What future work did the authors suggest?	RQ1

C. REPORTING THE REVIEW

The review results were reported by describing the summary of the studies and answering each RQ. The description of each RQ was based on the data extraction results. The 2009 PRISMA Checklist [27] was adopted as a checklist for issues that must be reported in the SLR.

V. REVIEW FINDINGS

This section describes the review findings. We included 38 primary studies in this SLR. For a full list of primary studies in this SLR, visit the web page at https://github.com/indrakharisma/NLPUserStory.

A. SUMMARY OF STUDIES

We identified 38 primary studies based on the review method. Six (15.8%) studies were published in journals; 22 (57.9%) were published in conferences, and ten (26.3%) were published in book chapters. The studies were evenly distributed in many publication venues, indicating that no single source was preferred by the authors.

Almost half of the primary study settings were preliminary studies. Eighteen studies (47.4%) expressed ideas and presented, at the very least, experimentation or case studies as proof of concept. Twenty studies (52.6%) used an in-lab academic setting for research. No studies used industry settings. However, several used real datasets from the industry in their research.

Related to the number of publications per year, Fig. 3 shows that the number of publications is continuously increasing. An increasing number of publications has been observed since 2014. The 2020 publications were recorded until December 2020.

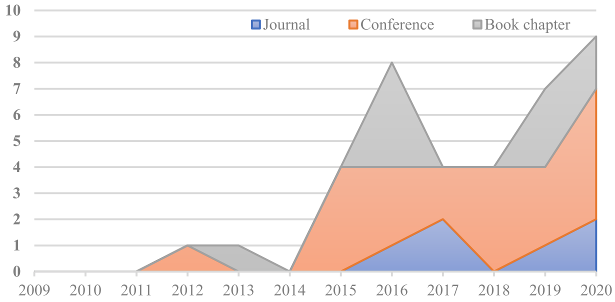

FIGURE 3. Distribution of the selected studies.

The correspondent/first author diversity of publications had an even distribution, spreading from Europe, Asia, America, Africa, and Australia. Other countries, including Italy, Turkey, Sweden, India, Indonesia, Iran, Thailand, Sri Lanka, USA, Mexico, Egypt, and New Zealand, also contributed papers (i.e., one primary study at the very least). We considered the location of the first author affiliation country to determine the authorship per geographical distribution (Fig. 4). Netherlands, Belgium, Germany, Brazil, and Morocco were the most productive country with four to six publications per country. Some studies were authored/co-authored by the same person, indicating the existence of an active research group in this field.

B. (RQ1) WHAT ARE THE USES OF NLP FOR USER STORIES?

The results of the primary studies illustrated several NL applications in user stories. We used the category of NLP RE tools [70] to classify the goal of the primary studies as follows: (a) discovering defects; (b) generating a model/artifact; (c) tracing links between model/NL requirements; and (c) identifying the key abstractions. Table 6 presents a summary of the primary studies based on these categories.

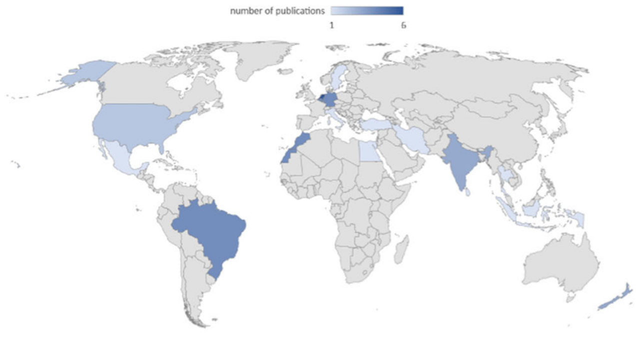

FIGURE 4. Authorship distribution per country.

TABLE 6. NLP in the user story goal.

Category	Goal	Studies that reported the goal
Discovering defects	Provide recommendations on incomplete requirements based on the knowledge gap	[33]
	Identify ambiguous user stories	[34]
	Define and measure quality factors from user stories	[4], [35]
	Obtain a security defect reporting form from user stories	[36]
	Indicate duplication between user stories	[37]
Generate model/artifact	Generate a test case from user stories	[38]–[43]
	Generate a class diagram from user stories	[44], [45]
	Generate a sequence diagram from user stories	[46]
	Generate a use case diagram from user stories	[47]–[49]
	Generate a use case scenario from user stories	[50]
	Generate a multi-agent system from user stories	[51]
	Generate a source code from user stories	[40]
	Generate a BPMN diagram from user stories	[40]
Identify the key abstractions	To understand the semantic connection in user stories	[52]–[54]
	Identify topics and summarizing user stories	[55], [56]
	Construct a goal model from a set of user stories.	[57]
	Define ontology for user stories	[58]
	Extract the conceptual model of user stories	[59], [60]
	To find the linguistic structure of user stories	[61]
	Prioritizing and estimation of user story complexity	[62], [63]
	Extracting user stories from text	[64]–[66]
Trace links between model/NL requirements	Tracking the development status of user stories from software artifacts	[67]
	Identify the type of dependency of user stories	[68]
	Traceability user stories and software artifact	[69]

Fig. 5 illustrates the year-wise distribution of the categorized primary study goals. Two topics are the major concerns that took most of the researchers' attention: identifying the key abstractions and generating models/artifacts. Both topics continue to be studied on an ongoing basis since 2015. The topic of key abstraction identification became the primary choice in the early phases because researchers are still trying to gain an understanding of a new and different characteristic of user stories. The topic of generating models/artifacts is always a challenge in software engineering research because it can accelerate the software development time.

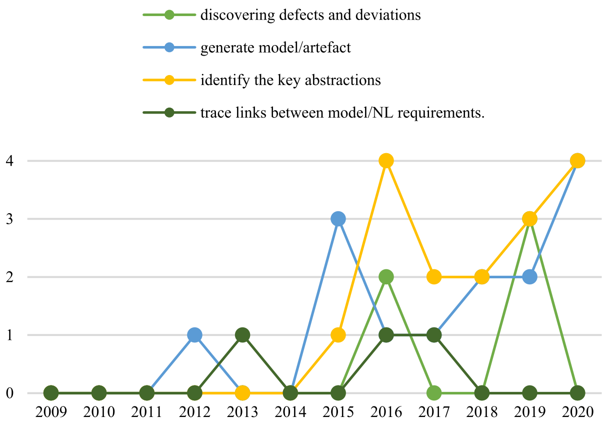

FIGURE 5. Year-wise distribution of the categorized primary study goal.

The following sub-sections present the direction of research conducted by primary studies for each category.

1) DISCOVERING DEFECTS

This category has the primary purpose of finding defects and deviations in user stories using natural language processing. We also included a primary study that aims to improve the requirements quality to this category. Five studies reported methods for finding defects or improving the quality of user stories. The category is meant to serve four purposes: (a) providing recommendations on incomplete requirements based on the knowledge gap [33]; (b) identifying ambiguous user stories [34]; (c) defining and measuring quality factors from user stories [4], [35]; (d) obtaining a security defect reporting form from the user stories [36] and (e) indicating duplications between user stories [37].

Bäumer and Geierhos [33] identified incomplete requirements with preprocessing, lemmatization, and POS tagging. Semantic role labeling was then performed to assign roles and actions. The software description was collected as a semantic data comparison for intuitive user guidance. Information retrieval was used for the similarity search components.

Dalpiaz *et al.* [34] identified ambiguous user stories by defining ambiguous meanings in user stories and calculating the ambiguity based on the semantic distance.

Galster *et al.* [35] identified quality attributes in user stories categorized according to their quality attributes (i.e., compatibility, maintainability, performance, portability, reliability, and security). Lucassen *et al.* [4] defined the user story quality. Quality is categorized as unique and conflict-free, uniform, independent, and complete. A tool called The

Automatic Quality User Story Artisan was built to perform the NLP process by identifying each quality criterion. The tool generates reports related to the quality of user stories.

Villamizar *et al.* [36] obtained a security defect reporting form from user stories with extracted user story key phrases (verb + nouns), linking them with security properties and high-level security requirements. The semantic similarity was also used to identify duplications between user stories [37]. The WuP similarity was utilized to determine the semantic similarity based on the aspects of what.

2) GENERATING THE MODEL/ARTIFACT

This category has the objective of generating software artifacts from natural language. The user story can be either input or output of the generated artifact. Fourteen studies reported methods for generating software model/artifacts from user stories, that is, generating a test case from user stories [35]–[39], [64], generating class diagrams from user stories [40], [41], generating sequence diagrams from user stories [46], generating a use case diagram from user stories [43]–[45], generating a use case scenario from user stories [50], generating a multi-agent system from user stories [51], generating a source code from user stories [40], and generating BPMN diagrams from user stories [40]. The software artifact generation aims to cut time and cost in software development and avoid inconsistencies, incompleteness, and incorrect requirements and artifact/software models.

Test case generation from user stories is a popular approach. One method used to generate a test case is capturing information related to ontology [36], [38], machine learning [39], dependency parsing [43], and transformation rules [35], [39].

Athiththan *et al.* [40] and Landhäußer *et al.* [38] extracted information on user stories and converted them into user story ontology. Domain knowledge and ontology are made according to the model to be created. Artifact generation is performed by combining information among the user story and domain ontologies. In this manner, Athiththan *et al.* [40] generated test cases, source codes, and BPMN diagrams from user stories. Meanwhile, [39], [41], [42] attempted to understand and analyze the pattern of user stories and perform a preprocessing for finding keywords. Nouns and verbs were analyzed to formulate the test cases.

Several researchers proposed generating UML diagrams from user stories. The main approach commonly used was to employ part-of-speech tagging to identify verbs and nouns as elements in UML diagrams. This technique was used to generate use case diagrams [48], sequence diagrams [46], and use case scenarios [50] from user stories. The same approach was taken by [44], [45] to generate class diagrams. In generating use case diagrams, [47] categorized the aspects of what in user stories into three categories, namely task, capability, and goal, to produce more detailed use case diagrams involving <include> and <extend> as dependency relationships. The same concept was used by [15] to generate multi-agent system development artifacts.

3) IDENTIFYING THE KEY ABSTRACTIONS

This category aims to identify the key abstractions from NL documents that help analysts understand unknown domains. The key abstraction identification was performed by 16 studies to understand the semantic connection in user stories [48]–[50], identify topics and summarizing user stories [55], [56], construct a goal model from a set of user stories [57], define the ontology for user stories [58], extract the conceptual model of user stories [53], [54], prioritize and estimate the user story complexity [56], [57], find the linguistic structure of user stories [61], and extract user stories from text [64]–[66].

Several methods can be used to obtain and understand the semantic connections in user stories [48]–[50] using the semantic similarity from user stories. Barbosa *et al.* [52] utilized the cosine similarity function and clustering using the K-medoids algorithm. Lucassen *et al.* [53] used the skip-gram implementation of word2vec to calculate the semantic similarity scores. Sharma and Kumar [54] employed the RV coefficient algorithm to measure the similarity.

Gunes *et al.* [57] proposed to generate a goal model from user stories automatically using NLP. This was achieved by parsing each user story with NLP techniques. Gulle *et al.* [55] identified topics inside crowd-generated user stories using Latent Dirichlet Allocation (LDA), Word Vectors, Word Embeddings, and Word Mover's Distance. In contrast, Resketi *et al.* [56] tried to summarize a set of user stories based on their frequencies.

Thamrongchote and Vatanawood [58] proposed to assist user story writing by gathering knowledge concepts utilizing the ontology concept. Classes, a hierarchy of ontology, schema graph, and synonym were defined from the user story data. This property was used to help write better user stories.

Lucassen *et al.* [53] introduced an automated approach tool called Visual Narrator, which extracts conceptual models from the user story requirements using heuristics rules. Wautelet *et al.* [59] determined the meta-model of user stories by identifying the unified model of user stories' descriptive concepts (role, task, capability, soft goal, and hard goal). Müter *et al.* [61] explored linguistic structures and action verbs in task user stories. The task of the user stories was analyzed to determine the word patterns widely used, especially verbs.

Several attempts have been made to produce effort size and priority based on user stories. Ecar *et al.* [63] proposed a functional size measurement method based on user stories and COSMIC methods. Meanwhile, Castillo-Barrera *et al.* [62] used bloom's taxonomy to classify the complexity of user stories.

Raharjana *et al.* [64], Rodeghero *et al.* [65], and Henriksson *et al.* [66] extracted user stories from free text. Rodeghero *et al.* used interview data to extract user story information, Raharjana *et al.* [64] used data to obtain user stories from online news to assist the elicitation software process, while Henriksson *et al.* [66] used heterogeneous digital sources.

4) TRACING LINKS BETWEEN MODEL/NL REQUIREMENTS

This category aims to trace the relationship between the NL description requirements or with other artifacts. Tracing the relationship between these models and NL requirements can assist during the software development process, particularly in inconsistency checking and change management [72]. Three studies focused on tracing the relationship between models and user stories: Plank *et al.* [67] tracked the development status of user stories from software artifacts; Soni and Gaur [68] identified the dependency type of user stories, and Lucassen *et al.* [69] tracked the traceability of user stories and software artifacts.

Plank *et al.* [67] illustrated the relationship between user stories and software development artifacts. The software development artifacts used included code comments, commit messages, bug reports, and the development of wiki information. Bag-of-word, similarity, and NER were also used to extract information in user stories. The status of user stories (to be implemented/in progress/completed) can be classified when the relationship mapping between user stories and software development artifacts is obtained.

Soni and Gaur [68] applied lexical analysis to user stories to obtain index terms. Lexical analysis, fuzzy set theory, and vector model are used to identify the type of dependency from user requirements. Lucassen *et al.* [69] tracked the software test artifact traceability with user stories by proposing the behavior-driven traceability method metrics. These metrics were generated based on user stories and source code using the behavior-driven development tests to track the correlation.

C. (RQ2) WHAT APPROACHES WERE AVAILABLE IN RESEARCH RELATED TO NLP IN USER STORIES?

To answer RQ2 about the approaches available in research related to NLP in user stories, we divided them into several pieces: NLP techniques, validation methods, and tools used.

1) NLP TECHNIQUES

We note the various NLP techniques reported in the primary studies. Table 7 presents detailed information. The terms used for the NLP techniques may be general. Some are very specific under the context used by the primary studies. Several studies reported utilizing more than one technique in conducting scientific research.

The technique widely used by studies is POS tagging. Thirteen studies confirmed using this technique. The other NLP techniques used are vector space model (six studies), named-entity recognizer (four studies), dependency (three studies), syntactic parse tree (three studies), preprocessing, bag-of-words, term frequency–inverse document frequency, WuP similarity, lemmatization, semantic role labeling, skip-gram, similarity matrix, fuzzy set theory, and open information extraction.

Part-of-Speech (POS) is a lexical category of a sentence, such as nouns, verbs, adjectives, and adverbs. The advantage of POS tags is that they can identify verb and noun phrases accurately; this helps researchers identify key elements in the user story, namely aspects of who, what, and why. The aspect of who usually consists of noun phrases, while the aspect of what and why consists of a verb followed by noun phrases. The POS tags technique makes it easy to identify the items needed to generate a model/artifact from a user story, such as classes, activities, and use cases for UML Diagrams. The disadvantage of using POS tags is that the performance of identifying unfamiliar words, for instance, words that not seen previously or slang, is low.

TABLE 7. NLP techniques in user story studies.

NLP methods	Freq	Studies
Preprocessing	4	[52] [33][56][55]
POS tag	13	[38] [33] [4][53] [60][40] [41][49][61][64][36] [45][57]
Named-entity recognizer	4	[38] [67] [40][64]
Bag-of-words	2	[67][56]
Vector space model	6	[67] [52][37][68] [65][34][55]
Machine learning	2	[67][35]
Clustering	2	[52] [53]
Term frequency–inverse document frequency	1	[37]
WuP similarity	1	[37]
Lemmatization	1	[33]
Semantic role labeling	1	[33]
Skip-gram	1	[53]
Similarity matrix	1	[53]
Fuzzy set theory	1	[68]
Dependency	3	[60] [50] [45]
Logistic regression	1	[65]
Open information extraction	1	[40]
Syntactic parse tree	3	[63][48] [50]
Latent Dirichlet Allocation	1	[55]

The following step after POS tagging may include implementing the dependency parsing or syntactic parse tree. Dependency parsing is the activity of extracting dependencies from a sentence that representing a grammatical structure and defining the relationships between words. The tree representation of a lexical category of a sentence may come in the syntactic parse tree. The advantage of using dependency parsing is knowing grammatical relationships in the sentence, such as identify the stakeholders and what they want within the explicit sentences.

Another NLP technique for identifying words and phrases chunks is to make use of a bag of words. Bag-of-words is a technique of grouping words and calculating their term frequency to measure their level of importance. Skip-gram is a variant of bag-of-words that collects n-grams but allows words to be skipped. The most common implementation of the bag-of word is used to classify text.

It leads to machine learning implementation on user story research. Several machine learning approaches are being utilized as NLP techniques in user story studies, such as clustering, logistic regression, vector space model, similarity, and fuzzy set theory. Machine learning approaches are divided into supervised learning, unsupervised learning, and reinforcement learning. The vector space model represents a text document as a vector so that the document relevance ranking can be calculated based on the document similarity theory. WuP similarity and similarity matrix are some of the other techniques to calculate the similarity between documents. The fuzzy set theory allows a gradual assessment of the membership of elements in a set, usually used in domains where the information is incomplete or imprecise.

The NLP techniques used to identify the aspect of who include Named Entity Recognition (NER) and Semantic role labeling. NER is a technique for finding and classifying named entities in unstructured text. They were usually used to identify people, organizations, or other entities written in the text. Semantic role labeling is the process of assigning a label to a word or phrase in a sentence indicating its semantic role. The advantage of NER and semantic role modeling is that it has great accuracy for text in a trained domain, but it may need to be improved when implemented in a new domain.

Although most of the studies do not explain the preprocessing technique in detail, however, this step is an essential step for preparing the data. Preprocessing is a stage for treating data into the desired form; the process usually includes tokenization, filtering, and stop-word removal. Lemmatization is the process of grouping a word's forms to be analyzed as one item dictionary form; another similar approach is stemming, which changes to its raw form.

2) VALIDATION METHODS

We examined four types of validation conducted by researchers to assess the results: precision and recall, case study/example, average time and effort comparison, and prototype demonstration.

Many primary studies employ case studies for evaluation methods. This evaluation method reports experiences based on best examples, which usually provide lessons learned. Besides, several studies used prototype demonstration as proof of their concept. Several other studies conducted evaluations by comparing the tool's performance with control elements, such as the average time and effort required by tools compared to groups of experts.

The evaluations of studies in the NLP field usually employed precision, recall, and F-measure as the quality indicators. Precision is how many of the items selected are relevant, as shown in (1). A recall is how many relevant items are selected, as shown in (2). F-measure unites precision and recall, as shown in (3).

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (1)$$

$$Recall = \frac{TruePositive}{True\ Positive + False\ Negative} \quad (2)$$

$$F - measure = 2x\frac{Precision * Recall}{Precision + Recall} \quad (3)$$

with
True Positive = the correctly labeled instances.
False Positive = incorrectly labeled instances.
False Negative = the missed-out instances by the system.

Unexpectedly, the evaluations using precision and recall are not the main evaluations conducted by the primary studies. Only ten studies used precision and recall, while 16 used case study example methods as validation methods. The average time and effort comparison and prototype demonstration were performed by two primary studies. Table 8 illustrates the validation methods of the user stories used in the primary studies.

TABLE 8. User stories study validation methods.

Validation	Freq	Studies
Case study/example	17	[52][44][46] [33] [68][58][47][15][62][63] [50] [49][61][64][54] [36][66]
Precision and recall	11	[38][37] [4] [60][65][40][48][34][35][45][56]
Average time and effort comparison	2	[39][41]
Prototype demonstration	2	[53][69]

The evaluation was done by comparing the results with the predictions made by human annotators and usually using a group of software developers or university students. What was evaluated was depending on the study purpose. Most of the datasets used by researchers were independently collected and privately stored for internal needs.

3) NLP TOOLS

Most studies used SpaCy or Stanford CoreNLP to conduct NLP. Some stated using word2vec, WordNet, LingPipe Toolkit, PropBank, TreeTagger, and Stanford POS tagger, while some did not report what tools they utilized. More than one tool was used in some studies (e.g., SpaCy and NLTK). Table 9 lists the NLP toolkits used in the studies.

The feature in the widely used tool is the POS tag, which is available in almost all tools. This feature is very useful in user story study because it can be used to chunk phrases into verb and nouns to quickly determine the aspects of who, what, and why in the user story. Also, most tools support preprocessing natural language as basic functionality, making it easier for researchers to carry out their research. Another useful feature is to calculate similarity. Word2vec is the most widely used similarity calculation implementation; besides SpaCy and WordNet also provide similar functionality with different implementation techniques.

D. (RQ3) WHAT ARE THE CHALLENGES OF USING NLP IN USER STORY RESEARCH?

The primary studies reported several challenges. Some were related to the improvement of recall and precision, dataset, understanding the correct interpretation of a sentence, and

TABLE 9. NLP tools in the user story studies.

Tools	Features	Freq	Studies
SpaCy	Tokenization, Part-of-speech (POS) Tagging, Dependency Parsing, Lemmatization, Similarity.	4	[53] [50][35] [57]
Stanford CoreNLP	Tokenization, Part-of-speech (POS) Tagging, Lemmatization.	4	[4][40][41][36]
Natural Language ToolKit (NLTK)	Part-of-speech (POS) Tagging	2	[4] [49]
word2vec	semantic arithmetic	3	[53][34][55]
WordNet	Semantic similarity	1	[37]
LingPipe Toolkit	end-of-sentence detection	1	[33]
PropBank	semantic propositions	1	[33]
TreeTagger	Part-of-speech (POS) Tagging	1	[48]
Stanford POS tagger	Part-of-speech (POS) Tagging	1	[61]

TABLE 10. Challenges.

Challenges	Description
Improving the recall and precision	Low precision [38] [4]
Dataset	Heterogeneity and low amount of data [52]
	Manual data tagging [47] [59][34]
Context/domain dependent	Cannot be generally used in all contexts of the problem[33]
	Not yet able to handle complex systems [40], [48]
Understanding the correct interpretation of a sentence	Compounds are difficult to identify correctly [60]
	Verbs may be difficult to link to the proper object [60]
	Conjunctions are also a challenge [60]
	The same verb but can be classified in different categories [62]
Human intervention	Complement rather than replace human decision making [35]

human intervention. Table 10 summarizes the challenges reported in the primary studies.

Throughout the recall and precision evaluation, the researchers reported that the precision results were still not as expected, even though the recall results achieved were in line with the expectations. Lucassen *et al.* [4] obtained consistent recall results above 90%, but the average precision value was still approximately 72–77% [38]. They even obtained very low precision values. However, we must understand that the different objectives, data, and research methods are not necessarily comparable to an apple-to-apple data comparison. The researchers agree that achieving a high-precision value is still challenging.

Datasets have several challenges, including heterogeneity, low amount of data, and manual tagging of data. The limited number of user story datasets openly available makes it difficult to obtain large amounts of user story data. The limited heterogeneity of the data faces an issue. Researchers usually independently collect user story datasets for their purposes. The problem of heterogeneity and low amount of data has become an issue when analyzing user stories using machine learning algorithms (e.g., clustering or semantic similarity). Another challenge is obtaining reliable ground truth data, which is usually done by manually tagging the data. Primary studies usually use groups of software developers or university students to conduct manual data tagging. University students are usually preferred because of ease of access to do manual tagging, especially for large data. Studies found that experience influences the outcome of manual tagging, but with special handling, the result does not bring up major issues. Special handling may include providing a clear explanation of what to do in the manual tagging process and the Kappa analysis to know the agreement level between respondents.

The results of user story studies using NLP generate results that are context/domain dependent [31], indicating that it cannot be generally used in all problem contexts. This is not a new problem in machine learning. The results would become more accurate if the data used are homogeneous. However, this does not apply to domains/problems that differ from the data used as the training data. A very large dataset is required to obtain generic results. In addition, NLP in user stories cannot yet handle complex systems, especially in the process of turning user stories into software artifact software [36], [43]. Most studies are still researching specific data and have not tried doing it in complex systems or real applications.

The automation process in NLP research on user stories still requires human intervention. For example, detecting the ambiguity of user stories can be time-consuming, even though it has been done using tools [34]. In broad outline, the results obtained cannot yet match human results [35]. The NLP implementation on software requirements usually cannot fully implement automation, but this can be accomplished in software development.

As in general research, understanding the proper sentence interpretation remains a challenge. Some challenges involve compounds that are difficult to correctly identify [60], verbs that can be difficult to link up to the appropriate object [60], and conjunctions [60]. The same verb can be classified into different categories [62].

VI. DISCUSSION

Several findings can be presented from the result of the literature review. We described the meaning of the findings related to our RQs and identified the study limitations.

A. GENERAL FINDING

We found that the geographic location of the authors varied across five continents. The contributions also spread from many countries, for example, from Europe (Netherlands, Belgium, Germany, Italy, Turkey, and Sweden), Asia (India, Indonesia, Iran, Thailand, and Sri Lanka), America (Brazil, USA, and Mexico), Africa (Morroco, and Egypt), and Australia represented by New Zealand. We observe that Europe is still the center of research in this area. Many primary studies from Europe have become references to other primary studies. The geographic location distribution is a good signal for the research area development. Studies on the NLP and user stories are already the concern of researchers from different countries.

More than half of the studies were preliminary studies, indicating that the research area is not mature and still at the early stage. This is normal because ASD as a research field is also newly developed [5].

The number of publications in this area increases every year. The conference and book chapters still dominate the publication area. This is natural for new and emerging fields of science because the conference and the book chapter offer a relatively fast process in a publication compared to journals. The year 2016 has also begun publication in journals that mark the improvement in research quality.

B. FINDINGS RELATED TO RQ1

The purposes of NLP and user story research still majorly focus on identifying abstraction and generating models. The abstraction identification was reasonably made in the early stages of this research because the researchers were still studying the characteristics of user stories. The semi-structured user story format was systematic and relatively easier to analyze. Some researchers tried to identify abstraction by defining the ontology and understanding the semantic relationship between the user stories to group them according to specific goals.

What has not been much discussed was how the user story extraction from free text is performed. The current research still concentrates on user story processing. The generation of user stories from free text has not yet been much explored. What makes it is complex is usually a free text characteristic that is difficult to understand and a language structure that needs to be analyzed deeper. Challenges like identifying the aspects of who, what, and why from free text and how to compose these three aspects in a user story must be addressed to achieve these goals.

If free text data are derived from software-related documents, such as app review, user comment, app description, and identification aspect of what, it might be possible to adopt the feature extraction software widely used by researchers. If the data comes from non-software-related documents, such as news or social media, extra effort would be required to distinguish between the aspects of what related to software requirements or not. The named-entity recognition technique can be implemented to obtain the aspect of who. To find the aspect of why, the causal relationship between the aspects of what must be recognized.

Generating models/artifacts from a user story is widely performed by researchers. Most take the noun phrase and the verb from a user story to be converted into software artifacts, such as a class diagram, a sequence diagram, a use case diagram, and a BPMN. Researchers also use supporting data from the source code to obtain a model pattern. Most researchers use predefined rules to generate user stories into artifact software.

In recent years, the research focus began to shift to the discovery of the defects and trace links between models/artifacts. From this, researchers should have learned a pretty good picture related to the abstraction of user stories. They have explored machine learning techniques and semantic similarity to do such research.

Like the use case, the user story format also has several functions, such as documentation, software artifact generation, and validation/testing [23]. All these functions were covered by the primary studies, but substantial improvement is necessary. The requirements written in the natural language have some issues, including inconsistency, incompleteness, and incorrectness. Some studies on user stories are concerned with these issues, especially on user story quality research.

C. FINDINGS RELATED TO RQ2

The availability of NLP tools that support thorough features can help researchers conduct their research according to research objectives. The majority of NLP research on user stories still focuses on using NLP for preprocessing and POS tagging. The identification of verbs and nouns is the basis for processing the user story—all the objectives of NLP in user story research using this technique. Specifically, the purpose of identifying the key abstractions and generating a model/artifact is usually enough to identify the aspects of who, what, and why and then map the appropriate artifacts accordingly. Meanwhile, to discover defects and trace links between models / NL requirements, most often need machine learning processing in achieving its goals, such as to calculate similarity value between artifacts.

Most of the NLP studies on user stories are based on syntax or word-level approaches, while the semantics approach has not been much explored. This is an opportunity to be able to maximize the NLP benefits in user story research. Cambria and White Cambria and White envisioned the evolution of NLP research through three eras of curves, namely syntactic curve (bag-of-words), semantics curve (bag-of-concepts), and pragmatics curve (bag-of-narratives). This can be adopted by user story research to be able to shift into the semantics curve. Research using deep learning in this field is still open for exploration, with the main obstacle being the availability of a large-size user story dataset.

The limited user story dataset that can be accessed indicates the need for open datasets. The majority of researchers own or collect data themselves. Quality datasets are important because poor raw data would produce poor results. Available user story datasets are limited (e.g., [73] and [74]). The challenge of providing this dataset is that these data are usually owned by software companies, which are reluctant to share due to privacy concerns. An open dataset is important for comparing results with previous studies. In addition, it facilitates access for researchers to conduct user story research.

The format of user stories with a broad scope can be strengths and weaknesses. This is problematic, especially in epic user stories with other sub-user stories. The user story scope may consist of goals, tasks, and capabilities that must be clearly defined when performing further processes. This is important if you want to use user stories to generate other software artifacts or see the traceability between user stories and software artifacts.

Even though the focus of the study's contribution emphasized the use of NLP in user stories, most studies did not include detailed NLP procedures. Most studies included only the techniques used without providing sufficient detailed information on how the procedure is done.

The most widely used NPL technique is the POS tag based on the fact that the study objectives usually require the verbs and the nouns of the user story. Other techniques, such as preprocessing, syntactic parse tree, dependency, lemmatization, term frequency–inverse document frequency, and bag-of-word, are also referred to in the primary studies. The NER and semantic role labeling are usually used to obtain the aspects of who in the text. Techniques, such as clustering, machine learning, and vector space models, are used to acquire the semantic similarity in a user story.

Most primary studies are still preliminary studies; hence, it is not surprising that the evaluation technique still uses a case study/by example. Ideally, evaluation is done using precision and recall because it is widely used in NLP research.

SpaCy, Stanford CoreNLP, NLTK, and word2vec are the main tools used by researchers along with other supporting tools, such as WordNet, PropBank, and Stanford POS tagger. These tools do not stand alone. Researchers sometimes use more than one tool in accordance with the requirements.

D. FINDINGS RELATED TO RQ3

Contextual knowledge is needed when processing user stories [75]. Different problem domains often introduce new terms/words in user stories, including tacit knowledge (information understood by domain experts), which makes it difficult to obtain a general pattern. As reported by several studies, the domain context influences the scope of results. Some studies have reported changes when applying their methods to broader and more complex user stories.

The main advantage of using NLP is that it helps system analysts understand and manage user stories. In general, the processing time can be improved compared with the manual method [40]. Using NLP also helps system analysts more quickly understand the context of requirements, especially when handling a large collection of user stories [52]. NLP can be applied to provide suggestions on how to complete user stories [33].

E. LIMITATION OF THE REVIEW

Some papers might be missed, which could affect the incompleteness of our results. We used a defined protocol, performed a rigorous search, and used multiple databases to reduce this risk. We also applied forward and backward snowballing to obtain a comprehensive primary study in the study search and selection process.

We used the spreadsheet tool and Mendeley software to manage the study results and avoid primary study duplication. We also implemented a phased search strategy to manage text duplication. In the inclusion and exclusion process, we only scanned based on the title, abstract, and keywords in each database, which might affect irrelevant, relevant, or unrelated papers on the list. We added the stages of full-text articles assessed for eligibility to avoid irrelevant papers. For untracked relevant papers, we accepted the risk with the argument that the core context of the paper should be available in the title, abstract, and keywords.

VII. CONCLUSION

This study presented an SLR of the implementation of NLP in user stories. We identified 287 studies on the initial search and produced 30 primary studies after applying the inclusion and exclusion criteria. We complemented this count with additional three primary studies after employing forward and backward snowballing. We then evaluated the primary studies through quality assessment.

The main findings of the SLR are as follows:

(i) Many studies are position papers expressing ideas by displaying examples of the application of concepts, indicating that more research would imerge in the near future.

(ii) The category of studies mostly performed is key abstraction identification of user stories and generation of models or artifacts from user stories.

(iii) POS tags are the most widely used NLP techniques, but semantic approaches (e.g., vector space models and machine learning) are starting to gain a place.

(iv) A case study is widely used for study evaluation. However, the precision-recall method would be widely used as research maturity increases.

(v) In line with NLP research in general, understanding the context of a sentence is still a major issue herein. We believe that the study findings can help researchers conduct research in the field of user stories with NLP.

Our review showed that this research field is still immature and requires deeper exploration. The NLP application could be developed such that it can produce more diverse and useful results. Some NLP studies on user stories have shown good foundations, such as conceptual models of user story extraction, software artifacts from user stories, user story similarity, priority and size estimation, user quality stories, and user story extraction. We hope that the ASD would also thrive in NLP and user story research. Research in broader aspects, such as management and requirement security maintenance may also be another area of interest. Industry involvement also needs to be encouraged for the mutual benefit of researchers and practitioners.

ACKNOWLEDGMENT

The authors would like to thank Mutia Rahmi Dewi and Nafingatun Ngaliah, who assisted in the process of gathering and selecting the primary studies. Both are Master's students in informatics at the Institut Teknologi Sepuluh Nopember, Indonesia.

REFERENCES

[1] E. M. Schön, J. Thomaschewski, and M. J. Escalona, ''Agile requirements engineering: A systematic literature review,'' *Comput. Stand. Interface*, vol. 49, pp. 79–91, 2017, doi: 10.1016/j.csi.2016.08.011.

[2] R. Noel, F. Riquelme, R. M. Lean, E. Merino, C. Cechinel, T. S. Barcelos, R. Villarroel, and R. Munoz, ''Exploring collaborative writing of user stories with multimodal learning analytics: A case study on a software engineering course,'' *IEEE Access*, vol. 6, pp. 67783–67798, 2018, doi: 10.1109/ACCESS.2018.2876801.

[3] Y. Wautelet, S. Heng, M. Kolp, and I. Mirbel, ''Unifying and extending user story models,'' in *Advanced Information Systems Engineering* (Lecture Notes in Computer Science), vol. 8484. New York, NY, USA: Springer, 2014.

[4] G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, and S. Brinkkemper, ''Improving agile requirements: The quality user story framework and tool,'' *Requirements Eng.*, vol. 21, no. 3, pp. 383–403, Sep. 2016, doi: 10.1007/s00766-016-0250-x.

[5] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, ''A systematic literature review on agile requirements engineering practices and challenges,'' *Comput. Hum. Behav.*, vol. 51, pp. 915–929, Oct. 2015, doi: 10.1016/j.chb.2014.10.046.

[6] M. Younas, D. N. A. Jawawi, M. A. Shah, A. Mustafa, M. Awais, M. K. Ishfaq, and K. Wakil, ''Elicitation of nonfunctional requirements in agile development using cloud computing environment,'' *IEEE Access*, vol. 8, pp. 209153–209162, 2020, doi: 10.1109/access.2020.3014381.

[7] H. Meth, M. Brhel, and A. Maedche, ''The state of the art in automated requirements elicitation,'' *Inf. Softw. Technol.*, vol. 55, no. 10, pp. 1695–1709, Oct. 2013, doi: 10.1016/j.infsof.2013.03.008.

[8] A. R. da Silva, ''Linguistic patterns and linguistic styles for requirements specification (I): An application case with the rigorous RSL/business-level language,'' in *Proc. 22nd Eur. Conf. Pattern Lang. Programs*, Jul. 2017, pp. 1–27, doi: 10.1145/3147704.3147728.

[9] H. Dar, M. I. Lali, H. Ashraf, M. Ramzan, T. Amjad, and B. Shahzad, ''A systematic study on software requirements elicitation techniques and its challenges in mobile application development,'' *IEEE Access*, vol. 6, pp. 63859–63867, 2018, doi: 10.1109/ACCESS.2018.2874981.

[10] M. Arias, A. Buccella, and A. Cechich, ''A framework for managing requirements of software product lines,'' *Electron. Notes Theor. Comput. Sci.*, vol. 339, pp. 5–20, Jul. 2018, doi: 10.1016/j.entcs.2018.06.002.

[11] A. Al-Hroob, A. T. Imam, and R. Al-Heisa, ''The use of artificial neural networks for extracting actions and actors from requirements document,'' *Inf. Softw. Technol.*, vol. 101, pp. 1–15, Sep. 2018, doi: 10.1016/j.infsof.2018.04.010.

[12] T. Johann, C. Stanik, A. M. B. Alizadeh, and W. Maalej, ''SAFE: A simple approach for feature extraction from app descriptions and app reviews,'' in *Proc. IEEE 25th Int. Requirements Eng. Conf.*, 2017, pp. 21–30, doi: 10.1109/RE.2017.71.

[13] V. Garousi, S. Bauer, and M. Felderer, ''NLP-assisted software testing: A systematic mapping of the literature,'' *Inf. Softw. Technol.*, vol. 126, Oct. 2020, Art. no. 106321, doi: 10.1016/j.infsof.2020.106321.

[14] I. K. Raharjana, F. Harris, and A. Justitia, ''Tool for generating behavior-driven development test-cases,'' *J. Inf. Syst. Eng. Bus. Intell.*, vol. 6, no. 1, p. 27, Apr. 2020, doi: 10.20473/jisebi.6.1.27-36.

[15] Y. Wautelet, S. Heng, S. Kiv, and M. Kolp, ''User-story driven development of multi-agent systems: A process fragment for agile methods,'' *Comput. Lang., Syst. Struct.*, vol. 50, pp. 159–176, Dec. 2017, doi: 10.1016/j.cl.2017.06.007.

[16] E. Cambria and B. White, ''Jumping NLP curves: A review of natural language processing research,'' *IEEE Comput. Intell. Mag.*, vol. 9, no. 2, pp. 48–57, May 2014, doi: 10.1109/MCI.2014.2307227.

[17] N. H. Bakar, Z. M. Kasirun, N. Salleh, and H. A. Jalab, ''Extracting features from online software reviews to aid requirements reuse,'' *Appl. Soft Comput.*, vol. 49, pp. 1297–1315, Dec. 2016, doi: 10.1016/j.asoc.2016.07.048.

[18] A. Ferrari and A. Esuli, ''An NLP approach for cross-domain ambiguity detection in requirements engineering,'' *Automated Softw. Eng.*, vol. 26, no. 3, pp. 559–598, Sep. 2019, doi: 10.1007/s10515-019-00261-7.

[19] C. Li, L. Huang, J. Ge, B. Luo, and V. Ng, ''Automatically classifying user requests in crowdsourcing requirements engineering,'' *J. Syst. Softw.*, vol. 138, pp. 108–123, Apr. 2018, doi: 10.1016/j.jss.2017.12.028.

[20] W. Maalej, Z. Kurtanović, H. Nabil, and C. Stanik, ''On the automatic classification of app reviews,'' *Requirements Eng.*, vol. 21, no. 3, pp. 311–331, Sep. 2016, doi: 10.1007/s00766-016-0251-9.

[21] E. Guzman and W. Maalej, ''How do users like this feature? A fine grained sentiment analysis of app reviews,'' in *Proc. IEEE 22nd Int. Requirements Eng. Conf. (RE)*, Aug. 2014, pp. 153–162. [Online]. Available: https://mast.informatik.uni-hamburg.de/wp-content/uploads/2014/06/FeatureSentiments.pdf

[22] W. Behutiye, P. Karhapä, L. López, and X. Burgués, ''Management of quality requirements in agile and rapid software development: A systematic mapping study,'' *Inf. Softw. Technol.*, vol. 123, Apr. 2019, Art. no. 106225, doi: 10.1016/j.infsof.2019.106225.

[23] S. Tiwari and A. Gupta, ''A systematic literature review of use case specifications research,'' *Inf. Softw. Technol.*, vol. 67, pp. 128–158, Nov. 2015, doi: 10.1016/j.infsof.2015.06.004.

[24] G. Loniewski, E. Insfran, and S. Abrahão, ''A systematic review of the use of requirements engineering techniques in model-driven development,'' in *Model Driven Engineering Languages and Systems* (Lecture Notes in Computer Science), vol. 6395. New York, NY, USA: Springer, pp. 213–227, doi: 10.1007/978-3-642-16129-2_16.

[25] P. Heck and A. Zaidman, *A Systematic Literature Review on Quality Criteria for Agile Requirements Specifications*, vol. 26. New York, NY, USA: Springer, 2018.

[26] N. H. Bakar, Z. M. Kasirun, and N. Salleh, ''Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review,'' *J. Syst. Softw.*, vol. 106, pp. 132–149, Aug. 2015, doi: 10.1016/j.jss.2015.05.006.

[27] F. Nazir, W. H. Butt, M. W. Anwar, and M. A. K. Khattak, ''The applications of natural language processing (NLP) for software requirement engineering—A systematic literature review,'' in *Information Science and Applications* (Lecture Notes in Electrical Engineering), vol. 424. New York, NY, USA: Springer, Mar. 2017, pp. 485–493, doi: 10.1007/978-981-10-4154-9_56.

[28] B. Kitchenham and S. Charters, ''Guidelines for performing systematic literature reviews in software engineering,'' Dept. Comput. Sci., Durham, U.K., Tech. Rep. EBSE-2007-01, 2007.

[29] A. Pollock and E. Berge, ''How to do a systematic review,'' *Int. J. Stroke*, vol. 13, no. 2, pp. 138–156, Feb. 2018, doi: 10.1177/1747493017743796.

[30] A. Liberati, D. G. Altman, J. Tetzlaff, C. Mulrow, P. C. Gotzsche, J. P. A. Ioannidis, M. Clarke, P. J. Devereaux, J. Kleijnen, and D. Moher, ''The PRISMA statement for reporting systematic reviews and meta-analyses of studies that evaluate healthcare interventions: Explanation and elaboration,'' *BMJ*, vol. 339, no. 1, p. b2700, Dec. 2009, doi: 10.1136/bmj.b2700.

[31] D. Badampudi, C. Wohlin, and K. Petersen, ''Experiences from using snowballing and database searches in systematic literature studies,'' in *Proc. 19th Int. Conf. Eval. Assessment Softw. Eng.*, vols. 27–29, Apr. 2015, pp. 1–10, doi: 10.1145/2745802.2745818.

[32] M. E. Falagas, E. I. Pitsouni, G. A. Malietzis, and G. Pappas, ''Comparison of PubMed, scopus, Web of science, and Google scholar: Strengths and weaknesses,'' *FASEB J.*, vol. 22, no. 2, pp. 338–342, Feb. 2008, doi: 10.1096/fj.07-9492lsf.

[33] F. S. Bäumer and M. Geierhos, ''Running out of words: How similar user stories can help to elaborate individual natural language requirement descriptions,'' in *Commun. Comput. Inf. Sci.*, vol. 639, pp. 549–558, Oct. 2016, doi: 10.1007/978-3-319-46254-7_44.

[34] F. Dalpiaz, I. van der Schalk, S. Brinkkemper, F. B. Aydemir, and G. Lucassen, ''Detecting terminological ambiguity in user stories: Tool and experimentation,'' *Inf. Softw. Technol.*, vol. 110, pp. 3–16, Jun. 2019, doi: 10.1016/j.infsof.2018.12.007.

[35] M. Galster, F. Gilson, and F. Georis, ''What quality attributes can we find in product backlogs? A machine learning perspective,'' in *Software Architecture* (Lecture Notes in Computer Science), vol. 11681. New York, NY, USA: Springer, Sep. 2019, pp. 88–96.

[36] H. Villamizar, M. Kalinowski, A. Garcia, and D. Mendez, ''An efficient approach for reviewing security-related aspects in agile requirements specifications of Web applications,'' *Requirement Eng.*, vol. 25, no. 4, pp. 439–468, Dec. 2020, doi: 10.1007/s00766-020-00338-w.

[37] R. Barbosa, A. E. A. Silva, and R. Moraes, ''Use of similarity measure to suggest the existence of duplicate user stories in the srum process,'' in *Proc. 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshop (DSN-W)*, Jun. 2016, pp. 2–5, doi: 10.1109/DSN-W.2016.27.

[38] M. Landhäußer and A. Genaid, ''Connecting user stories and code for test development,'' in *Proc. 3rd Int. Workshop Recommendation Syst. Softw. Eng.*, 2012, pp. 33–37, doi: 10.1109/RSSE.2012.6233406.

[39] R. Elghondakly, S. Moussa, and N. Badr, ''Waterfall and agile requirements-based model for automated test cases generation,'' in *Proc. IEEE 7th Int. Conf. Intell. Comput. Inf. Syst. (ICICIS)*, Dec. 2015, pp. 607–612, doi: 10.1109/IntelCIS.2015.7397285.

[40] K. Athiththan, S. Rovinsan, S. Sathveegan, N. Gunasekaran, K. S. A. W. Gunawardena, and D. Kasthurirathna, ''An ontology-based approach to automate the software development process,'' in *Proc. IEEE Int. Conf. Inf. Automat. Sustainability (ICIAfS)*, Dec. 2018, pp. 1–6, doi: 10.1109/ICIAFS.2018.8913339.

[41] S. C. Allala, J. P. Sotomayor, D. Santiago, T. M. King, and P. J. Clarke, ''Towards transforming user requirements to test cases using MDE and NLP,'' in *Proc. Int. Comput. Softw. Appl. Conf.*, vol. 2, 2019, pp. 350–355, doi: 10.1109/COMPSAC.2019.10231.

[42] N. Mulla and N. Jayakumar, ''The potent combo of software testing and NLP,'' in *ICDSMLA* (Lecture Notes in Electrical Engineering). New York, NY, USA: Springer, 2020, pp. 1623–1632.

[43] J. Fischbach, A. Vogelsang, D. Spies, A. Wehrle, M. Junker, and D. Freudenstein, ''SPECMATE: Automated creation of test cases from acceptance criteria,'' in *Proc. IEEE 13th Int. Conf. Softw. Test., Validation Verification (ICST)*, Oct. 2020, pp. 321–331, doi: 10.1109/ICST46399.2020.00040.

[44] W. Dahhane, A. Zeaaraoui, E. H. Ettifouri, and T. Bouchentouf, ''An automated object-based approach to transforming requirements to class diagrams,'' in *Proc. 2nd World Conf. Complex Syst.*, 2014, pp. 158–163, doi: 10.1109/ICoCS.2014.7060906.

[45] S. Nasiri, Y. Rhazali, M. Lahmer, and N. Chenfour, ''Towards a generation of class diagram from user stories in agile methods,'' *Procedia Comput. Sci.*, vol. 170, pp. 831–837, Dec. 2020, doi: 10.1016/j.procs.2020.03.148.

[46] M. Elallaoui, K. Nafil, and R. Touahni, ''Automatic generation of UML sequence diagrams from user stories in Scrum process,'' in *Proc. 10th Int. Conf. Intell. Syst., Theor. Appl.*, 2015, pp. 1–6, doi: 10.1109/SITA.2015.7358415.

[47] Y. Wautelet, S. Heng, D. Hintea, M. Kolp, and S. Poelmans, ''Bridging user story sets with the use case model,'' in *Advances in Conceptual Modeling* (Lecture Notes in Computer Science), vol. 9975. New York, NY, USA: Springer, 2016, pp. 127–138, doi: 10.1007/978-3-319-47717-6_11.

[48] M. Elallaoui, K. Nafil, and R. Touahni, ''Automatic transformation of user stories into UML use case diagrams using NLP techniques,'' *Procedia Comput. Sci.*, vol. 130, pp. 42–49, Dec. 2018, doi: 10.1016/j.procs.2018.04.010.

[49] A. Gupta, G. Poels, and P. Bera, ''Creation of multiple conceptual models from user stories—A natural language processing approach,'' in *Advances in Conceptual Modeling* (Lecture Notes in Computer Science), vol. 11787. New York, NY, USA: Springer, 2019, pp. 47–57, doi: 10.1007/978-3-030-34146-6_5.

[50] F. Gilson, M. Galster, and F. Georis, ''Generating use case scenarios from user stories,'' in *Proc. Int. Conf. Softw. Syst. Processes*, Jun. 2020, pp. 31–40, doi: 10.1145/3379177.3388895.

[51] Y. Wautelet, S. Heng, M. Kolp, and C. Scharff, ''Towards an agent-driven software architecture aligned with user stories,'' in *Proc. 8th Int. Conf. Agents Artif. Intell.*, 2016, pp. 337–345, doi: 10.5220/0005706103370345.

[52] R. Barbosa, D. Januario, A. E. Silva, R. Moraes, and P. Martins, ''An approach to clustering and sequencing of textual requirements,'' in *Proc. IEEE Int. Conf. Dependable Syst. Netw. Workshops*, Jun. 2015, pp. 39–44, doi: 10.1109/DSN-W.2015.20.

[53] G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, and S. Brinkkemper, ''Visualizing user story requirements at multiple granularity levels via semantic relatedness,'' in *Conceptual Modeling* (Lecture Notes in Computer Science), vol. 9974. New York, NY, USA: Springer, pp. 463–478, Nov. 2016.

[54] S. Sharma and D. Kumar, ''Agile release planning using natural language processing algorithm,'' in *Proc. Amity Int. Conf. Artif. Intell. (AICAI)*, Feb. 2019, pp. 934–938, doi: 10.1109/AICAI.2019.8701252.

[55] K. J. Gulle, N. Ford, P. Ebel, F. Brokhausen, and A. Vogelsang, ''Topic modeling on user stories using word mover's distance,'' *Proc. 7th Int. Work. Artif. Intell. Requir. Eng. AIRE*, vol. 2020, pp. 52–60, 2020, doi: 10.1109/AIRE51212.2020.00015.

[56] M. R. Resketi, H. Motameni, H. Nematzadeh, and E. Akbari, ''Automatic summarising of user stories in order to be reused in future similar projects,'' *IET Softw.*, vol. 14, no. 6, pp. 711–723, Dec. 2020, doi: 10.1049/iet-sen.2019.0182.

[57] T. Gunes and F. B. Aydemir, ''Automated goal model extraction from user stories using NLP,'' in *Proc. IEEE 28th Int. Requirements Eng. Conf. (RE)*, Aug. 2020, pp. 382–387, doi: 10.1109/RE48521.2020.00052.

[58] C. Thamrongchote and W. Vatanawood, ''Business process ontology for defining user story,'' in *Proc. IEEE/ACIS 15th Int. Conf. Comput. Inf. Sci. (ICIS)*, Jun. 2016, pp. 3–6, doi: 10.1109/ICIS.2016.7550829.

[59] Y. Wautelet, S. Heng, M. Kolp, I. Mirbel, and S. Poelmans, ''Building a rationale diagram for evaluating user story sets,'' in *Proc. IEEE 10th Int. Conf. Res. Challenges Inf. Sci. (RCIS)*, Jun. 2016, pp. 1–12, doi: 10.1109/RCIS.2016.7549299.

[60] G. Lucassen, M. Robeer, F. Dalpiaz, J. M. E. M. van der Werf, and S. Brinkkemper, ''Extracting conceptual models from user stories with visual narrator,'' *Requirements Eng.*, vol. 22, no. 3, pp. 339–358, Sep. 2017, doi: 10.1007/s00766-017-0270-1.

[61] L. Müter, T. Deoskar, M. Mathijssen, S. Brinkkemper, and F. Dalpiaz, ''Refinement of user stories into backlog items: Linguistic structure and action verbs,'' in *Requirements Engineering: Foundation for Software Quality* (Lecture Notes in Computer Science), vol. 11412. New York, NY, USA: Springer, 2019, pp. 109–116.

[62] F. E. Castillo-Barrera, M. Amador-Garcia, H. G. Perez-Gonzalez, F. E. Martinez-Perez, and F. J. Torres-Reyes, ''Adapting Bloom's taxonomy for an agile classification of the complexity of the user stories in SCRUM,'' in *Proc. 6th Int. Conf. Softw. Eng. Res. Innov. (CONISOFT)*, Oct. 2018, pp. 139–145, doi: 10.1109/CONISOFT.2018.8645899.

[63] M. Ecar, F. N. Kepler, and J. P. S. da Silva, ''AutoCosmic: COSMIC automated estimation and management tool,'' in *Proc. XIV Brazilian Symp. Inf. Syst.*, 2018, pp. 481–488, doi: 10.1145/3229345.3229409.

[64] I. K. Raharjana, D. Siahaan, and C. Fatichah, ''User story extraction from online news for software requirements elicitation: A conceptual model,'' in *Proc. 16th Int. Joint Conf. Comput. Sci. Softw. Eng. (JCSSE)*, Jul. 2019, pp. 342–347, doi: 10.1109/jcsse.2019.8864199.

[65] P. Rodeghero, S. Jiang, A. Armaly, and C. Mcmillan, ''Detecting user story information in developer-client conversations to generate extractive summaries,'' in *Proc. IEEE/ACM 39th Int. Conf. Softw. Eng. (ICSE)*, May 2017, pp. 49–59, doi: 10.1109/ICSE.2017.13.

[66] A. Henriksson and J. Zdravkovic, ''A data-driven framework for automated requirements elicitation from heterogeneous digital sources,'' in *The Practice of Enterprise Modeling*, vol. 400. New York, NY, USA: Springer, 2020.

[67] B. Plank, T. Sauer, and I. Schaefer, ''Supporting agile software development by natural language processing,'' in *Trustworthy Eternal Systems via Evolving Software, Data and Knowledge* (Communications in Computer and Information Science), vol. 379. New York, NY, USA: Springer, 2013, pp. 91–102.

[68] A. Soni and V. Gaur, ''A methodological approach to identify type of dependency from user requirements,'' in *Computational Science and Its Applications* (Lecture Notes in Computer Science), vol. 9789. New York, NY, USA: Springer, 2016, pp. 374–391.

[69] G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, S. Brinkkemper, and D. Zowghi, ''Behavior-driven requirements traceability via automated acceptance tests,'' in *Proc. IEEE 25th Int. Requirements Eng. Conf. Workshops (REW)*, Sep. 2017, pp. 431–434, doi: 10.1109/REW.2017.84.

[70] D. Berry, R. Gacitua, P. Sawyer, and S. F. Tjong, ''The case for dumb requirements engineering tools,'' *Requirements Engineering: Foundation for Software Quality* (Lecture Notes in Computer Science), vol. 7195. New York, NY, USA: Springer, 2012, pp. 211–217, doi: 10.1007/978-3-642-28714-5_18.

[71] M. Masud, M. Iqbal, M. U. Khan, and F. Azam, ''Automated user story driven approach for Web-based functional testing,'' *Int. J. Comput. Inf. Eng.*, vol. 11, no. 1, pp. 91–98, 2017.

[72] A. Goknil, I. Kurtev, K. van den Berg, and J.-W. Veldhuis, ''Semantics of trace relations in requirements models for consistency checking and inferencing,'' *Softw. Syst. Model.*, vol. 10, no. 1, pp. 31–54, Feb. 2011, doi: 10.1007/s10270-009-0142-3.

[73] F. Dalpiaz, "Requirements data sets (user stories)," Mendeley Ltd., London, U.K., Tech. Rep. 7zbk8zsd8y.1, 2018, doi: 10.17632/7zbk8zsd8y.1.

[74] A. Menkveld, "Applying the requirements engineering for software architecture model in software products: A case study using crowdsourcing," Utrecht Univ., Utrecht, The Netherlands, Tech. Rep., Mar. 2019. [Online]. Available: http://dspace.library.uu.nl/handle/1874/380343, doi: 10.17632/7r9j67wxzb.1.

[75] F. Dalpiaz, A. Ferrari, X. Franch, and C. Palomares, "Natural language processing for requirements engineering: The best is yet to come," *IEEE Softw.*, vol. 35, no. 5, pp. 115–119, Sep. 2018, doi: 10.1109/MS.2018.3571242.

INDRA KHARISMA RAHARJANA (Graduate Student Member, IEEE) received the bachelor's degree in informatics from the Institut Teknologi Sepuluh Nopember, Indonesia, in 2005, and the master's degree in informatics from the Institut Teknologi Bandung, Indonesia, in 2011. He is currently pursuing the Ph.D. degree in computer science with the Institut Teknologi Sepuluh Nopember. In 2006, he joined the Information Systems Study Programme, Universitas Airlangga, Indonesia, as an Assistant Professor. His research interests include software engineering, agile software development, natural language processing, and information systems. He is also an Editor-in-Chief of *Journal of Information Systems Engineering and Business Intelligence*.

DANIEL SIAHAAN (Member, IEEE) received the master's degree in software engineering from the Technische Universiteit Delft, in 2002, and the Ph.D. degree in software engineering from the Technische Universiteit Eindhoven, in 2004. He is currently an Associate Professor with the Department of Informatics, Institut Teknologi Sepuluh Nopember. He has published more than 50 journal articles and conference papers related to software engineering. His research interests include software engineering, requirements engineering, and natural language processing.

CHASTINE FATICHAH (Member, IEEE) received the Ph.D. degree from the Tokyo Institute of Technology, Japan, in 2012. She is currently an Associate Professor with the Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia. She has published more than 110 journal articles and conference papers related to computer science. Her research interests include artificial intelligence, data mining, and image processing.