

# Visualisation of User Stories in UML Models: A Systematic Literature Review

Mohammad Nazrul Mornie , Nurfaeza Jali , Syahrul Nizam Junaini , Edwin Mit ,  
Cheah Wai Shiang , Suhaila Saeed 

Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Kota Samarahan, Malaysia

Corresponding author: Mohammad Nazrul Bin Mornie (21020048@siswa.unimas.my)

## Abstract

The use of agile methodology in software development projects is growing rapidly among industry professionals and academia. The Unified Modelling Language (UML) conventionally accompanies agile software development to model the software requirements. The user story is fundamental and should be identified to communicate the basic requirements between the development team and the stakeholders before the UML model such as the use case diagram, class diagrams and many others can be designed. However, there are several challenges associated with this process such as poorly organised user stories, natural language complexity and high time consumption to create them. A systematic literature review is conducted to grasp more knowledge about the utilisation of natural language processing (NLP) for UML model generation. A total of 198 papers were initially found in four online databases, namely Scopus, IEEE Xplore, ScienceDirect and ACM Digital Library, from the period 2018–2022. After removing duplicates, applying inclusion and exclusion criteria, and conducting the full-text assessment, only 20 papers are included as the primary studies. The primary studies are reviewed to discover several important pieces of information, namely the challenges of designing UML models, NLP tools and techniques used to generate UML models, UML models generated, and validation methods used for measuring the accuracy of generated models. Finally, this study discusses important elements related to UML model generation using NLP tools and techniques.

## Keywords

Unified modelling language; Conceptual model; Natural language processing; Agile software development; User stories.

# 1 Introduction

The agile methodology has been widely used in software development projects since it was introduced (Thesing et al., 2021). It offers useful methods for software development and ensures that the end product satisfies the stakeholders. Additionally, agile permits the development process to be more responsive to requirement changes from stakeholders, enabling the development team to break solutions down into numerous parts while producing solutions step-by-step, and directly involving stakeholders in the development process to achieve the best results (Thesing et al., 2021). There are several elements associated with agile software development, such as user stories and Unified Modelling Language (UML) (Dalpiaz & Brinkkemper, 2018; Gallud & Fardoun, 2019)

A user story is usually a medium used in agile software development to communicate the software project's requirements between the stakeholders and the development team. The process of acquiring user stories from stakeholders may improve the functionality of the software experienced by the users (Cohn, 2004). An important aspect of the agile requirement collecting technique is the user narrative, which provides early insight into how actual end-users of the software will interact with it. User stories have grown to become more popular nowadays to record the functional requirements from the user's point of view (Elallaoui et al., 2018). User stories generally serve three functions: to identify the types of users that will use the software, what goal they want to achieve with the software, and why they want to achieve it (Bik et al., 2017). Based on the user stories' roles, it is understandable that developers and other professionals must coordinate and monitor stakeholder interactions carefully. It helps meet users' software needs. User stories do not fully indicate absolute requirements for the software because they are a starting point of conversation between professionals and stakeholders. The requirements themselves are established based on the stories provided by stakeholders. Gilson and Irwin (2018) state that user stories can provide long listings of software requirements since they may contain detailed functional decomposition of the software. This leads to a crucial effort to adequately document user stories properly because poorly written user stories will lead to several problems such as neglect of non-functional requirements and confusion of requirement priority.

Next, Unified Modelling Language (UML) is a type of visual language used to document and visualise software requirements. The UML allows professionals such as developers to understand how software works. Moreover, UML can greatly help organise users' requirements. The agile methodology makes extensive use of several kinds of UML models. Class diagrams, use case diagrams, activity diagrams, sequence diagrams, system sequence diagrams, object diagrams, domain-model diagrams, state machine diagrams, communication diagrams and package diagrams are all instances of UML diagrams (Koç et al., 2021). These diagrams serve different features and functions. For example, a use case diagram is used to determine the use cases or scenarios that a system should accomplish, while a class diagram describes the objects and their relationships within the system (Seidl et al., 2015). The use case diagram is a form of UML model or diagram that is used to describe the use cases, categories of users (also known as actor), and relationships between users, whether it is between one user and another, or between users and use cases of a system or software. There are three basic components of a UML use case diagram: actors, use cases and interaction between actors. Furthermore, a use case diagram can be regarded as an important aspect of requirement documentation in agile development because relying on user stories to represent the software requirement is not sufficient in this context. Gilson and Irwin (2018) state that a graphical model and a visual notation such as a use case diagram can increase the efficiency of information communicated between stakeholders and developers. Since the content of a use case diagram represents the stakeholders' expectations of how the system should work, a use case diagram must include all of the crucial information to be fully effective (Seidl et al., 2015).

Regarding an agile software development project, UML models may support the development process in a number of ways. Gallud and Fardoun (2019) stated that UML models such as a class diagram, sequence

diagram and deployment diagram can be very useful for the agile team. For example, class diagram can help the agile team decide on the best design to implement the features required by the stakeholders. Besides, a deployment diagram can be used in an agile software development project to determine the overall architecture of the project. These diagrams and other UML diagrams in the form of a use case diagram, activity diagram and many more can greatly contribute to agile software development. However, these diagrams need user stories as the base because user stories provide stakeholders' general and most basic requirements. The process of transforming the user stories into UML diagrams, however, is not an easy task. This process requires huge effort, time commitment and expertise because user stories are usually inconsistent, incomplete and come with incorrect tasks (Elallaoui et al., 2015). Additionally, Reuter et al. (2020) conducted a study to identify the problems faced by software engineering students in generating UML models. They found that generating UML diagrams is a complex and confusing process for the students where they found a lot of problems such as determining connections between elements, chronological orders of elements, the interaction between objects, and many others. On a more expert level, creating a UML diagram for a software development project has a number of drawbacks even if it is advantageous for the projects (Gebretsadik, 2020). Some examples include their high complexity and time requirements, as well as the knowledge lost during model construction. This is the point at which it becomes necessary to include tools or techniques that can help with either completely automated or semi-automated ways of generating UML diagrams from user stories. This can be achieved by utilising the tools and techniques offered by natural language processing (NLP).

In addition, NLP is a crucial part of this research. It is a branch of artificial intelligence that allows the computer to learn, understand and process human language (Zhao et al., 2021). Humans speak a wide variety of languages. However, they are all classified as forms of "natural language", which are beyond the comprehension of computers. In the context of user stories, this holds true. User stories are usually collected from the communication between stakeholders and professionals such as analysts or developers. NLP is required for this type of communication since it entails using natural language and processing that language (Alzayed & Al-Hunaiyyan, 2021). Many types of NLP techniques may be used to train a computer to understand natural language and communicate with humans. This includes tokenisation, POS tagging, stemming and lemmatisation, and dependency parsing. Each of the techniques provides a different outcome. For example, tokenisation such as in a word tokeniser will break down the words into tokens. Last but not least, there is a wide range of NLP software services accessible, including Stanford CoreNLP, spaCy, NLTK and OpenNLP.

This research is conducted to discover knowledge surrounding the topic of UML model generation from user stories that utilise the implementation of NLP. This is done to obtain insights into how previous studies have conducted their research, which includes challenges associated with the creation of UML models in software development projects, especially agile ones, methods and tools implemented, and validation used to measure the accuracy of their approaches. However, there will also be some studies that generate other types of conceptual models used in the structured systems analysis and design method (SSADM) such as the entity relationship diagram (ERD), business process model and many others, as long as they apply the tools and techniques of NLP. The review was conducted based on the approach and methods of Raharjana et al. (2021). The scope of this research, however, was limited to examining studies that involve the generation of conceptual models in the form of UML. Conversely, the study conducted by Raharjana et al. (2021) reviewed studies that involved the use of NLP for user stories such as identification of NLP tools and methods used for user story specification together with challenges associated with using NLP in user stories.

This paper is organised as follows. Section 2 discusses the related works. Section 3 introduces the methods used for conducting this review in detail. Section 4 provides the results obtained from the review. Section 5 discusses the findings and limitations of the review and finally, Section 6 concludes the paper.

## 2 Related Works

A few secondary studies are related to reviewing the application of NLP that leads to generation of different software development process components such as user stories, conceptual models and test cases. Relevant review papers have been collected to conduct this study. That the review procedure for this study is directed in the proper direction by the results of prior studies. There does not appear to be a review paper that focuses on the development of UML models from user stories anywhere in the available web databases. Bozyiğit et al. (2021), Loniewski et al. (2010), Mustafa et al. (2021), Nazir et al. (2017) and Raharjana et al. (2021) are some of the related works that are similar to this study. All the mentioned secondary studies produce great insight into the knowledge related to user stories, requirement engineering, NLP and automation/transformation/generation of software development components. However, this study focuses more specifically on reviewing research that involves the generation of conceptual models, whether UML or any other types of models, from user stories using NLP approaches. Table 1 summarises the related works that have been found to be useful for this study.

**Table 1.** Summary of related works.

Reference	Goal-based on research questions
Raharjana et al. (2021)	<ul style="list-style-type: none"> <li>• Study uses of NLP for user stories.</li> <li>• Figure out approaches of NLP used for user stories.</li> <li>• Identify challenges of associating NLP with user stories.</li> </ul>
Nazir et al. (2017)	<ul style="list-style-type: none"> <li>• Study areas of software requirements that frequently employ NLP techniques.</li> <li>• Identify primary NLP activities in the context of software requirement engineering.</li> <li>• Determine tools used by researchers for software requirement engineering.</li> </ul>
Mustafa et al. (2021)	<ul style="list-style-type: none"> <li>• Identify existing ways for generating test cases from requirements.</li> <li>• Identify challenges in requirement-based testing.</li> </ul>
Bozyiğit et al. (2021)	<ul style="list-style-type: none"> <li>• Study languages that are analysed for transforming software requirements into a conceptual model.</li> <li>• Determine techniques for translating needs into a conceptual model.</li> <li>• Find out types of conceptual models that are transformed from previous research.</li> <li>• Understand details of datasets used in the reviewed studies.</li> <li>• Determine evaluation methods used for accuracy assessment.</li> </ul>
Loniewski et al. (2010)	<ul style="list-style-type: none"> <li>• Identify requirement engineering techniques that have been employed in model-driven development approaches and their level of automation.</li> </ul>

## 3 Review Method

Three main stages are used in the review: review planning, conducting the review and reporting the review. Each of these stages will be discussed in detail, which is an essential part of a systematic literature review. Besides, information from Kitchenham (2004) is used to help conduct this review.

### 3.1 Review Planning

For the planning of this study, it is necessary to identify the research questions that will be used to extract valuable data from the related research. In addition to planning, it is essential to define the databases from which the literature will be gathered, while also determining relevant keywords that should be used to perform the search. The last part of the review planning is to set the inclusion and exclusion criteria which will be used to filter the search results to retain only useful studies.

### 3.1.1 Research Questions

The generation of UML models such as the use case diagrams, class diagrams, activity diagrams and others, whether it is fully automated or semi-automated, needs to be explored and applied in the industry, which can help fasten and ease the process of software development, especially agile software development, which requires quick output with great quality. In order to discover exactly the knowledge that can help with the understanding of this topic, there are three research questions (RQs) that should be answered:

- What are the challenges faced by academics and the industry in designing UML models?
- What are the NLP tools used to assist the generation of UML models from user stories?
- How to measure the accuracy of UML models or conceptual models generated from user stories?

## 3.2 Search Strategy

The strategy to conduct the search is important to ensure that the desired literature can be found to help the study. There are two stages of search strategy, namely source selection and keyword or search term selection.

### 3.2.1 Source Selection

The search sources, which are mainly online databases, are identified. Many online databases are available, such as Scopus, ScienceDirect, IEEE Xplore, SpringerLink, Google Scholar and ACM Digital Library. For this study, four online databases are selected to find the literature as primary studies: Scopus, ScienceDirect, IEEE Xplore and ACM Digital Library. Since these databases have huge numbers of publications per year, the literature to be included in the primary study will only be taken from a period of five years (2018–2022). This is to ensure that the information gathered through the review is up to date, which will also be helpful for this study.

### 3.2.2 Keyword/Search Term Selection

Once the sources have been identified, it is necessary to identify the keywords or terms that will be used to search for the literature. The keywords that will be used to find the primary studies will evolve around the topic of this study, which is to look for the knowledge of visualising user stories in UML models by utilising NLP. Besides, similar words or synonyms should also be considered in order to ensure that more primary studies can be found. Table 2 shows the keywords or search terms along with their synonyms that will be used to conduct the search for primary studies.

**Table 2.** Keywords or search terms used for searching selected databases.

Keyword/term	Synonyms
Natural language processing	NLP
UML model	Unified modelling language/conceptual model/UML diagram/UML
User stories	User story
Generate	Transform/visualise/automation/analyse

### 3.2.3 Inclusion and exclusion criteria

As the literature is collected, it needs to be assessed and evaluated to make sure that it will be useful for the review. There are two types of criteria that will be considered to select relevant literature. The criteria are grouped into inclusion and exclusion criteria. These criteria are used to filter the literature found to ensure that it can be used as the primary studies in this review. The inclusion criteria are a set of conditions

that have been identified where the literature will be included in the review, while the exclusion criteria are a set of conditions to exclude the literature from the review. The inclusion and exclusion criteria are listed in Table 3.

**Table 3.** *Inclusion and exclusion criteria.*

Inclusion criteria	Exclusion criteria
The paper is written in English	The paper is written in other languages
Publication year between 2017 until 2022/23	Publication year is before 2017
Involves model generation using NLP (UML model or other types of conceptual model) – RQ2, RQ3	Duplicates
Discusses challenges faced by the industry in designing UML model/conceptual model – RQ1	Review paper
Includes types of UML model/conceptual model that are produced – RQ3	Lecture notes
Includes types of NLP tools used for model generation – RQ2	Proposals
Contains information about NLP techniques used in their approach	Does not contain any information regarding research questions (RQs)

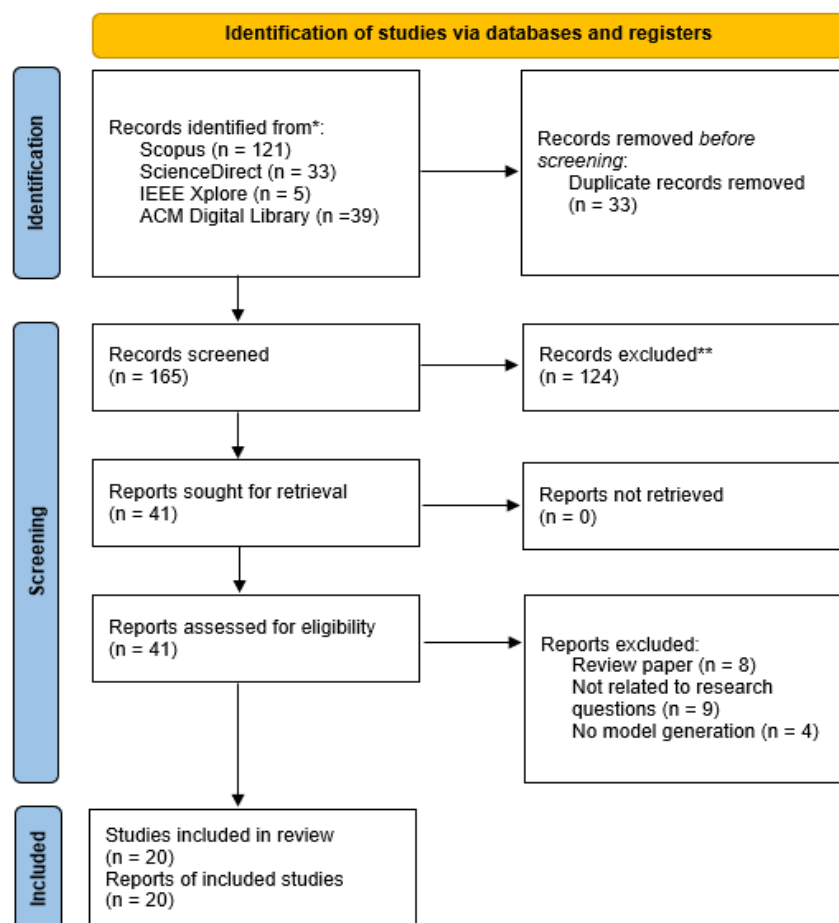
### 3.3 Conducting the Review

The review is conducted by first performing the search in the selected databases, namely Scopus, ScienceDirect, IEEE Xplore and ACM Digital Library, using the identified keywords and search terms. The searches are performed through title, abstract, keywords on ScienceDirect and Scopus, and through all metadata for IEEE Xplore and ACM Digital Library. These databases were chosen for the searches since the authors were rather familiar with those. Additionally, other databases, such as Web of Science (WoS), Taylor & Francis and SpringerLink, were also searched. However, the results returned by the searches had little to do with this review's needs or overlapped heavily. Therefore, the review does not include these databases.

#### 3.3.1 Search and study selection

The first search was conducted on Scopus followed by ScienceDirect, ACM Digital Library and IEEE Digital Xplore. The search was done from one database to another sequentially to ensure that the recording process goes in an organised manner. Once the search was done in all the selected databases, it was necessary to apply the inclusion and exclusion criteria to make sure that only relevant and non-duplicate studies are selected for the primary studies. The final step for study selection was full-text assessment of articles selected with the inclusion and exclusion criteria while also being non-duplicates. Figure 1 presents the study search and selection process.

Based on the search in the four databases, 20 papers were selected as primary studies. The initial search returned 121 results in Scopus, 33 in ScienceDirect, 39 in ACM Digital Library, and five in IEEE Xplore. This made a total of 198 papers. Then, 33 papers were removed since they were duplicates. The papers were further screened, excluding 124 papers. This gave a total of 41 papers after the initial screening. Inclusion and exclusion criteria were then applied to the papers found, excluding 21 papers from the process. Last but not least, the remaining papers were assessed in full text to ensure their relevance for the review. Thus, this review includes 20 papers as primary studies. The list of included papers is shown in Table A1 of Appendix A.



**Figure 1.** Study selection process for primary studies.

### 3.3.2 Data Extraction and Synthesis

The data extraction form in Table 4 was used to extract relevant information needed to conduct this review. The data extraction form was created based on the research questions. Thus, it is possible to extract as much relevant information for the review by following the details of the form.

**Table 4.** Data extraction form.

No	Data	Description	Relevant RQ
1	Identifier	Unique study ID	Overview
2	Title	Title of the article	Overview
3	Authors	Authors that contributed to the study	Overview
4	Publication year	Year of publication (last five years)	Overview
5	Source	Database	Overview
6	Research goal	Findings and contribution of the study	RQ2
7	Problem statement	Challenges faced by the industries (or academics) while designing UML models	RQ1
8	Challenge and limitations	Limitations present after the research was completed	RQ1
9	NLP tool	Types of NLP tools used to generate UML models	RQ2
10	NLP techniques	Types of NLP techniques used to generate UML models	RQ2
11	UML model	Types of UML models generated	RQ3

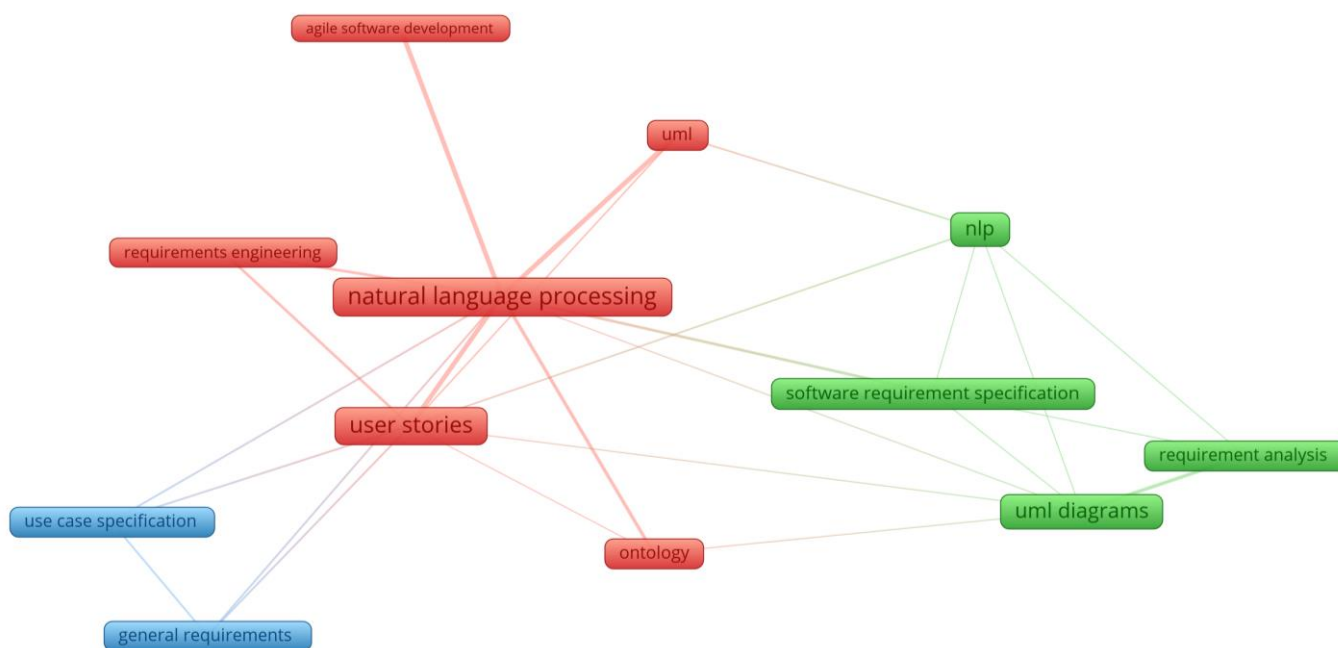
No	Data	Description	Relevant RQ
12	Requirement category	Alternatives to user stories	RQ3
13	Results	Results and accuracy from validation	RQ3

### 3.3.3 Reporting the Review

The reporting of this review is based on the checklist provided by Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) 2020 checklist (Page et al., 2021). The results of the review are reported in detail in the next section. The reporting of the review involves a detailed explanation of the results, findings related to the research questions, as well as a discussion of the results.

## 4 Results

The publication year of the primary studies included is between 2018 and 2022. Papers that are older than from 2018 were not selected. Scopus returned most of the primary studies compared to other databases. Information was extracted from the primary study using the data extraction form. Figure 2 shows the network visualisation of keywords from selected primary studies created using VOSviewer, which visualises the relationship between the keywords used by the researchers.



**Figure 2.** Visualisation of keywords used to search primary studies.

### 4.1 What are the challenges faced by academics and the industry in designing UML models?

A UML model or any other conceptual model is usually designed during the design and analysis phase in agile software development. Designing UML diagrams such as sequence diagrams poses lots of challenges: it is time-consuming, requires a lot of effort, and is very costly (Abdelnabi et al., 2020; Alami et al., 2020; Fischbach et al., 2020; Maatuk & Abdelnabi, 2021). Besides, there are several problems associated with the process of designing UML diagrams which come from the complexity and ambiguity of natural language (Abdelnabi et al., 2021; Javed & Lin, 2018, 2021).



For a goal model, which is another type of conceptual model, the great efforts needed to create it are one of the main concerns (Gunes et al., 2021). Since the design of UML models is based on the natural language user story or written requirements, it comes with some challenges such as information duplicity, incompleteness, redundancy and ambiguity. This includes difficulty determining the types of actors along with their interactions (Vasques et al., 2019). Based on empirical research, professional data modellers such as system analysts faced difficulty finding meaningful identifiers (Ternes et al., 2021). Some of these professionals need some time to learn and develop expertise in their field of work. In addition, the manual transformation of user stories to UML models is error-prone since it is done by human analysts (Alami et al., 2020). The mistakes are easy to present because there is usually a large amount of user requirements that need to be read by the analysts. This kind of mistakes likely leads to incorrect UML model generated by the analyst. Furthermore, the user stories are high-level requirements and each user's action is recorded in a separate story. Designing a UML model manually from a lot of user stories may cause the analyst to miss some important part of the stories (Gilson et al., 2020). Table 5 summarises the challenges associated with designing UML models or other conceptual models found in the primary studies.

**Table 5.** Challenges associated with designing UML models or other conceptual models.

No	Challenges/problems	Primary study
1	High time consumption	Abdelnabi et al. (2021); Alami et al. (2020); Fischbach et al. (2020); Maatuk & Abdelnabi (2021)
2	Requires lots of effort	Abdelnabi et al. (2021); Alami et al. (2020); Fischbach et al. (2020); Gunes & Aydemir (2020); Maatuk & Abdelnabi (2021)
3	High costs	Abdelnabi et al. (2021); Alami et al. (2020); Fischbach et al. (2020); Maatuk & Abdelnabi (2021)
4	Complexity of natural language	Javed & Lin (2018, 2021); Maatuk & Abdelnabi (2021)
5	Ambiguity of natural language	Javed & Lin (2018, 2021); Maatuk & Abdelnabi (2021)
6	Information duplicity, incompleteness, redundancy, ambiguity	Vasques et al. (2019)
7	Actors and their action confusion	Vasques et al. (2019)
8	Difficulty finding important identifiers	Ternes et al. (2021)
9	Process of manually transforming user stories to UML models is error-prone due to high volume of user stories	Alami et al. (2020)
10	Missing important information from user stories	Gilson et al. (2020)

## 4.2 What are NLP tools and techniques used to assist the generation of UML models from user stories?

To answer this research questions, there are two main components of focus in order to gather appropriate and relevant information for the review. First is the NLP tools used in the primary study and the next is the NLP techniques applied to ensure that the desired outcome can be achieved. There are a number of NLP tools that can be used to assist the generation of UML models from user stories. Some examples are Spacy, Stanford CoreNLP and NLTK.

First, TreeTagger Parser is used to generate part-of-speech (POS) tags as a preparation for the user stories to generate the use case diagram (Elallaoui et al., 2018). A set of rules is also used to ensure that the POS tags are useful for the extraction of use case diagram elements. Then, Abdelnabi et al. (2021) proposed an

approach to generate a sequence diagram and a collaboration diagram from unrestricted natural language requirements, using Stanford CoreNLP as the NLP tool. They applied five different NLP techniques: tokenisation, POS tagging, lemmatisation and stemming, parse tree and type dependencies, and open information extraction (OpenIE). Their approach is a combination of the use of NLP and application of heuristic rules to increase the accuracy of generated elements. Athiththan et al. (2018) applied NLP techniques to extract the elements of conceptual models from user stories. They used the Stanford CoreNLP tool to conduct this process. However, the specific NLP techniques were not mentioned in the papers. The extraction process in the study focused on user story details, such as the format of word capitalisation for user stories, while the main attributes of the conceptual model must have to be written in camel case format.

Sequence diagrams are generated from user requirements in Arabic using the MADA+TOKAN tool (Alami et al., 2020). It is a type of NLP tool that can be used to process natural language written in Arabic. The NLP techniques applied in the study were tokenisation, POS tagging, discretisation, morphological disambiguation, and stemming and lemmatisation. Maatuk and Abdelnabi (2021) generate a use case diagram and an activity diagram by adopting the generation approach of Abdelnabi et al. (2021). This means that the study also applied five different NLP techniques which are sentence and word tokenisation, POS tagging, lemmatisation and stemming, parse tree and type dependencies, and OpenIE using StanfordCore NLP.

Visualisation of use case scenarios in the form of a robustness diagram from textual user stories was proposed by Gilson et al. (2020). They generated robustness diagrams over use case diagrams because a use case diagram provides richer notation in terms of behavioural issues. They used an NLP tool named spaCy with the application of stemming and lemmatisation, POS tagging, and dependency tree. Javed and Lin (2021) proposed an automated approach to generate two types of conceptual models: entity-relationship diagram (ERD) and business process model. The conceptual models were generated from three different forms of requirements: general requirements, use case specifications and user stories. The study used the Stanford CoreNLP as the NLP tool to assist the generation of the mentioned conceptual models. Several NLP techniques were applied in the study. For the purpose of pre-processing the requirements, the study applied NLP techniques such as tokenisation, lemmatisation and POS tagging, while for the syntactic analysis, it applied the NLP technique of type dependency to study the relationships of words within a sentence. Javed and Lin (2018) proposed automated generation of ERD by using an NLP approach based on unrestricted requirement format, such as use case specifications, user stories and general requirements.

Furthermore, sequence diagrams and class diagrams can also be generated automatically using Stanford CoreNLP (Alashqar, 2021). The NLP techniques applied in their approach were tokenisation, POS tagging and dependency parsing. The study did not use any user stories for the base of their generation; instead, they used the longer and more detailed textual requirements in the form of use case scenarios. Gunes and Aydemir (2020), on the other hand, built a pipeline to automatically extract a goal model from user stories using spaCy. The techniques applied in their study were tokenisation and POS tagging, while the rest of the generation process was conducted with three sets of algorithms.

In addition, some of the papers identified do not involve the generation of a UML model. For example, Tiwari et al. (2019) proposed a systematic approach to extract use case scenarios from textual requirements. They used an NLP tool named Stanford NL Parser to conduct POS tagging and type dependencies. Besides, Allala et al. (2019) transformed user requirements in the form of user stories, use cases and source models into test cases. Their approach used Stanford CoreNLP to perform the processing of natural language from the requirements. In their study, they applied NLP techniques such as tokenisation and POS tagging. The rest of NLP tools and techniques used in primary studies are shown in

Table 6 and Table 7 respectively, while Table 8 shows the frequency of NLP tools used in the selected primary studies.

**Table 6.** NLP tools used in primary studies.

No.	NLP tool	Primary study
1	TreeTagger Parser	Elallaoui et al. (2018)
2	Stanford CoreNLP	Alashqar (2021); Allala et al. (2019); Athiththan et al. (2018); Javed & Lin (2018, 2021); Lano et al. (2021); Maatuk & Abdelnabi (2021); Nasiri et al. (2020, 2021); Schlutter & Vogelsang (2020); Shweta et al. (2018)
3	MADA+TOKAN	Alami et al. (2020)
4	spaCy	Gilson et al. (2020); Gunes & Aydemir (2020); Kochbati et al. (2021)
5	Word2vec	Kochbati et al. (2021)
6	Apache OpenNLP	Lano et al. (2021)
7	DeepSRL	Schlutter & Vogelsang (2020)
8	WordNet	Nasiri et al. (2020)
9	Stanford NL Parser	Tiwari et al. (2019)

**Table 7.** NLP techniques used in primary studies.

No.	NLP techniques	Primary studies
1	POS Tagging	Elallaoui et al. (2018)
2	Tokenisation, POS Tagging, Lemmatisation and Stemming, Parse Tree and Type Dependencies, OpenIE	Abdelnabi et al. (2021)
3	POS Tagging, Type Dependencies	Tiwari et al. (2019)
4	Tokenisation, POS Tagging, Disambiguation, Discretisation, Morphological Disambiguation, And Stemming, Lemmatisation	Alami et al. (2020)
5	Tokenisation, POS Tagging, Lemmatisation and Stemming, Parse Tree and Type Dependencies, OpenIE	Maatuk & Abdelnabi (2021)
6	Stemming And Lemmatisation, POS Tagging, Dependencies Tree	Gilson et al. (2020)
7	Tokenisation, Lemmatisation, POS Tagging	Javed & Lin (2018, 2021)
8	Tokenisation, POS Tagging, Dependency Parsing	Alashqar (2021)
9	Tokenisation, POS Tagging	Allala et al. (2019); Gunes & Aydemir (2020)
10	Word level semantic, tokenisation	Kochbati et al. (2021)
11	Text Splitting, Tokenisation, POS Tagging, Lemmatisation, Type Dependencies	Nasiri et al. (2021)
12	POS Tagging, Syntax Tree	Lano et al. (2021)
13	POS Tagging, Named-Entity Recognition	Gilson & Irwin (2018)
14	Tokenisation, Sentence Splitting, POS Tagging, Lemmatisation, Dependency Parsing, Coreference Resolution, Semantic Role Labelling	Schlutter & Vogelsang (2020)
15	Sentence Splitting, Text Parser, Universal Dependency, Lemmatisation	Shweta et al. (2018)
16	Tokenisation, POS Tagging, Conference Resolution, Stemming	Nasiri et al. (2020)

**Table 8.** Frequency of NLP tools used from primary studies.

No.	NLP tool	Frequency
1	Stanford CoreNLP	12
2	spaCy	4
3	MADA+TOKAN	1
4	Word2Vec	1
5	Stanford NL Parser	1
6	Apache OpenNLP	1
7	DeepSRL	1
8	WordNet	1
9	TreeTagger Parser	1

### 4.3 How to measure the accuracy of UML models or conceptual models generated from user stories?

The main concern of this RQ is to study the types of UML models or any other conceptual models that are usually generated from user stories. Besides, the validation method of the approach for generation of models proposed in the primary studies is also observed.

Elallaoui et al. (2018) generated use case diagrams from a set of user stories. The accuracy of the approach in their study was measured with a validation method using precision and recall score. In order to perform this validation, the automatically extracted elements were compared to the manually extracted elements, resulting in the categorisation of true positive (TP), false positive (FP) and false negative (FN) for the actors, use cases and association relationships. The study did not fully generate every element of the use case diagram, which obviously can be seen from the type of relationship. The only type of relationship that could be generated using the approach is the association relationship. However, the accuracy of the elements generated is satisfactory where the values of precision and recall involving the actors are 98% each, the precision value for both use cases and relationships is 87%, and the recall value for both use cases and relationships is 85%.

Next, the generation of a sequence diagram and a collaboration diagram from natural language requirements was proposed by Abdelnabi et al. (2021). Their study applied a set of heuristic rules to assist the extraction of the sequence and collaboration diagram elements. The elements generated using the heuristic rules were actors, senders, receivers and messages. These elements are crucial parts of sequence diagrams and collaboration diagrams. Once the elements were generated, the selected UML diagrams were drawn manually using a UML drawing tool. As for the validation of the approach, the study used case study experiments named the qualification verification system (QVS). Based on their experimental results, their approach is not only able to generate elements for sequence diagrams and collaboration diagrams, but also to assist the generation of class diagrams, use case diagrams and activity diagrams.

Using an NLP tool called MADA+TOKAN, sequence diagrams were generated from user requirements in Arabic by Alami et al. (2020). They used a set of heuristics rules in addition to an NLP tool to collect the participants, messages and workflow transitions that make up a sequence diagram. The participants of the sequence diagram that were identified in this approach included senders, main actors and receivers. The accuracy of their approach is measured by conducting experiments from a set of case studies validated by students and industry professionals familiar with the use of sequence diagrams. The results of the case studies were compared and the accuracy was calculated. Their approach is able to generate better participants of the sequence diagram compared to the students and achieve similar score to that of the experts. However, the approach cannot generate much better messages compared to students and experts since these researchers believe that the messages are more complicated than the participants.

To create use case diagrams and activity diagrams, a set of heuristic rules was used in conjunction with Stanford CoreNLP by Maatuk & Abdelnabi (2021). Stanford CoreNLP was used to pre-process the requirements, while the heuristics rules were used to extract the elements of use case diagrams and sequence diagrams. The rules used for identifying the elements of use case diagrams are actor identification rules, use case identification rules, and relationship identification rules. In contrast, for the identification of activity diagram elements, the rules used were activity diagram identification rules, decision node identification rules, action name identification rules and activity group identification rules. The study used the same method of validation as Abdelnabi et al. (2021). As for the results of the experiment, their approach is able to generate class diagrams, use case diagrams and activity diagrams.

In addition, an approach was proposed by Alashqar (2021) to automatically generate sequence diagrams and class diagrams from scenario-based user requirements. This approach combined the use of Stanford CoreNLP with several NLP techniques together with a set of algorithms. As for the validation to measure the accuracy of the approach, the author used a method that compared a manually generated sequence diagram and an automatically generated one. The accuracy of the proposed approach ranged from 77% to 90%, where actors achieved an accuracy score of 80%, caller objects achieved 90%, receiver objects 88%, operations achieved 90%, and messages achieved a score of 77%. The results were affected by the POS tags that were generated from the case study, where some actors could not be identified.

Furthermore, some of the primary studies do not include any validation method, while some only include validation plans. To summarise the findings relevant to this RQ, the details for the rest of the selected primary studies are shown in Table 9. This includes the generated UML diagram, validation method and score for each of the approaches in the primary studies.

**Table 9.** UML models or conceptual models generated in primary studies.

No	Model generated	Validation method	Score/results	Primary study
1	Use case diagram	Precision and recall	Precision: (actors = 98%, use cases = 87%, relationships = 87%) Recall: (actors = 98%, use cases = 85%, relationships = 85%)	Elallaoui et al. (2018)
2	Sequence diagram and collaboration diagram	Case study and comparison study	Able to generate sequence, collaboration, class, use case and activity diagrams	Abdelnabi et al. (2021)
3	Use case scenarios	Case study and comparison study	86.1% completion 89.7% accuracy 92.8% consistency 87% non-redundance	Tiwari et al. (2019)
4	Sequence diagram	Case study and comparison study	Able to generate participant element better than students, but lesser in terms of message elements	Alami et al. (2020)
5	Use case diagram and activity diagram	Case study and comparison study	Able to generate class, use case and activity diagrams	Maatuk & Abdelnabi (2021)
6	Robustness diagram	Case study and comparison study	Able to generate 81.4% syntactically valid diagrams	Gilson et al. (2020)
7	Entity relationship diagram	Precision, recall and over generation	Recall = 95% Precision = 93 Over generation = 9% Overall accuracy = 96%	Javed & Lin (2018)

No	Model generated	Validation method	Score/results	Primary study
8	Sequence diagram, class diagram	Case study and comparison study	Overall accuracy: 77-90%	Alashqar (2021)
9	Goal model	Prototype demonstration	Planning stage	Gunes & Aydemir (2020)
10	Use case diagram	Case study, precision, recall and F-measure	Precision: (actors = 100%, use cases = 95-97%, relationships = 75-97%) Recall: (actors = 100%, use cases = 89-94%, relationships = 84-90%) F-measure: (actors = 100%, use cases = 93-94%, relationships = 79-93%)	Kochbati et al. (2021)
11	Robustness diagram	-	Planning stage	Gilson & Irwin (2018)
12	Class diagram	Case study, precision and recall	Precision = 82% Recall = 94%	Shweta et al. (2018)
13	Class diagram	Case study, comparison study and precision	Overall precision = 98%	Nasiri et al. (2020)

## 5 Discussion

The findings of the primary studies are discussed in detail in this section, along with the limitations present while conducting the review. As for the general findings, it is observable that studies involving generation of UML models or any other conceptual models using natural language processing (NLP) are still not very numerous. The relevant primary studies found for this review were not very many. However, some of the studies provide very useful information, which adds to the body of knowledge and helps boost overall comprehension of the subject matter. A discussion of the findings based on the study questions is presented below.

### 5.1 Findings related to RQ1

Several challenges were found when dealing with the process of designing UML models or other types of conceptual models. The most prominent challenges concerned the complexity of designing UML models itself. This is present when generating models manually, which is done most of the time by industry professionals such as system analysts and requirement engineers; another group that deals with a similar situation is students, mainly in the field of computer science.

The challenge found when designing the models manually is the complexity of the work itself. It demands great efforts and lots of time. This challenge is the most mentioned in the primary studies, e.g., Abdelnabi et al. (2021) and Gunes et al. (2021). In these studies, the researchers aimed to help professionals and students ease their work on generating UML models.

Besides, the second most frequently observed challenges stated in the primary studies concern natural language. Some studies have found that requirements such as user stories are mostly written in natural language, e.g., English, Malay, Arabic, etc. (Abdelnabi et al., 2021; Javed & Lin, 2018, 2021). This leads to another challenge while designing a UML model. Natural language is broad and complex. User stories written in natural language pose a great challenge for the UML model designing process due to the complexity and ambiguity of natural language. Therefore, unwanted situations, such as missing information, information redundancy and incompleteness are foreseeable. To overcome these challenges, the researchers in selected primary studies propose various approaches to generating UML models, whether semi-automated or fully automated.

## 5.2 Findings related to RQ2

NLP tools are helpful for generation of UML models from user stories. This can be semi-automated or fully automated, depending on the approach proposed by the researchers. In order to answer the second research question, which is to identify the NLP tools and techniques used to generate UML models from user stories, details for both are discovered.

Different types of NLP tools are available to assist generation of UML models from requirements such as user stories. Some examples are spaCy, Stanford CoreNLP, WordNet, TreeTagger Parser and word2vec. These tools offer different functionalities. Some of them, such as spaCy and Stanford CoreNLP, can work standalone, while others need other tools to work, as shown by WordNet.

NLP techniques include tokenisation, stemming and lemmatisation, part-of-speech (POS) tagging, dependency parsing, constituency parsing, and many more. All of these techniques do not necessarily match a certain approach, so the researcher should plan carefully which techniques to apply in theirs. To relate the NLP tools with NLP techniques, the tools offer their own sets of NLP techniques. The NLP tool that offers the most techniques is Stanford CoreNLP.

From the review conducted on the primary studies, Stanford CoreNLP is the most commonly used NLP tool. This is because it offers many techniques and the researcher can pick those that suit their approach. Besides, the high efficiency and accuracy of generating UML models or other types of conceptual models using Stanford CoreNLP provides great results, as can be seen from several primary studies (Abdelnabi et al., 2021; Alashqar, 2021; Javed & Lin, 2018; Shweta et al., 2018).

Therefore, it can be said that to generate UML models from textual requirements such as user stories, suitable NLP tools and techniques must be considered. The main reason is to ensure that the desired results such as generated elements of the UML model or even the model itself can be achieved.

## 5.3 Findings related to RQ3

When it comes to model generation, many different types of models can be explored. Although the focus in this review is mainly on the UML model, other types of conceptual models have also been found that use approaches similar to that of UML model generation.

Based on the review, different types of UML models have been generated in both fully automated and semi-automated manner. UML models are generated from various types of requirements, where the user story is in the focus. The types of UML models generated found in the primary studies are use case diagrams, sequence, collaboration, activity, class, package, interaction and state machine diagrams. As for other types of conceptual models, the generated models include entity relationship diagrams, business process models, goal models, robustness diagrams, knowledge graphs and test case models.

The UML models most generated in the selected primary studies are the class diagram and sequence diagram, which were made in several studies (Alami et al., 2020; Alashqar, 2021; Maatuk & Abdelnabi, 2021; Nasiri et al., 2020; Shweta et al., 2018). Other types of UML models, such as use case diagrams, activity diagrams and collaboration diagrams, have not got much attention in the last five years. These UML models are usually generated using a semi-automated approach since parts of the generation process require human intervention.

As for the evaluation methods used in the primary studies, the most dominant method used by the researchers is case studies to measure the accuracy of their generated model. Case studies are used to conduct the evaluation in many papers (Abdelnabi et al., 2021; Alami et al., 2020; Alashqar, 2021; Gilson et al., 2020; Kochbati et al., 2021; Maatuk & Abdelnabi, 2021; Nasiri et al., 2020; Shweta et al., 2018; Tiwari et al., 2019). A set of case studies is gathered and used to conduct the experiment, and the results are observed. Once the observation is done, some researchers use a method such as a comparison study, where

experts are asked to manually generate a related model and it is compared with the automatically generated model using the researchers' approach. The outcome from this validation is compared and the results are used to measure the accuracy. Next, the F-measure, precision and recall value are also used in some primary studies to validate their approach (Elallaoui et al., 2018; Javed & Lin, 2018; Kochbati et al., 2021; Nasiri et al., 2020; Shweta et al., 2018). The researchers set some conditions for getting the value of true positive (TP), false positive (FP) and false negative (FN). These values are then used to obtain the scores of F-measure, precision and recall.

Reviewing the results of the approaches proposed in the primary studies, most of them produce the intended results. However, in some studies, such as Elallaoui et al. (2018), which aimed to generate a use case diagram from user stories, the proposed approach was not able to fully extract the elements of the use case diagram. To be more specific, the proposed approach was only able to generate an association relationship rather than all types of relationships used in the use case diagram, e.g., include relationship, extend relationship and generalisation relationship.

In a nutshell, it is difficult to achieve perfect results when it comes to automated generation of UML diagrams from user stories by means of NLP due to the nature of natural language itself. However, the best results can be achieved if every detail such as selection of tools, techniques, rules and algorithms is planned and organised carefully.

## 5.4 Limitations

The most obvious limitation that is present in this review is that there might be some useful papers that are not included. This is because only four databases were selected as the source, which makes a sum of 20 papers. Besides, some papers might as well be missed during the selection mainly due to an unsuitable choice of title, since the screening for paper selection starts with the paper title, followed by its abstract and keywords. Next, one of the most common methods for selecting papers, which is the forward and backward snowballing technique, was not used because most of them are older than five years, i.e., before 2018. However, these limitations can be mitigated if the publication year range is widened and more sources are added to the list for paper selection.

## 6 Conclusion

In a nutshell, very important information was found during this review regarding the generation of UML models from user stories through the use of natural language processing. From a total of 198 papers that were identified, 20 primary studies were selected from four different databases, namely Scopus, ScienceDirect, IEEE Xplore and ACM Digital Library. The number of the selected primary studies was reduced due to the application of inclusion and exclusion criteria, as well as a full-text assessment.

From the review, there are a few important findings crucial for answering the research questions (RQs). The important findings beneficial for this review are as follows.

- The manual designing process of the UML model, or any other conceptual model, comes with many challenges, the most common one being high time consumption.
- Different types of NLP tools can be used to assist the automated or semi-automated generation of UML models from user stories. These tools include Spacy, Stanford CoreNLP, NLTK, etc.
- Each NLP tool comes with different features and applies different techniques compared to others.
- Different types of UML models can be generated using similar NLP tools as long as different approaches are applied such as the use case diagram, class diagram, sequence diagram, etc.
- Evaluation methods are an important part of similar research to measure the effectiveness and accuracy of the approach.



Although plenty of information has been gathered through this review related to the generation of UML models from user stories using NLP, few researchers are interested in conducting this kind of research. The number of studies related to this field should be increased in order to solve existing gaps.

## Additional Information and Declarations

**Acknowledgements:** The authors appreciate the reviewers' thoughtful and rigorous comments, which enhanced the manuscript. The authors would also like to thank everyone who contributed, especially the *Faculty of Computer Science and Information Technology, UNIMAS*, for this research opportunity and financial assistance.

**Funding:** This research is fully supported by the Universiti Malaysia Sarawak (UNIMAS), Pilot Research Grant Scheme, UNI/F08/PILOT/85043/2022. The authors fully acknowledged the Universiti Malaysia Sarawak for the approved fund, which makes this important research viable and effective.

**Conflict of Interests:** The authors declare no conflict of interest.

**Author Contributions:** M.N.M.: Data curation, Formal Analysis, Investigation, Writing – Original draft preparation. N.J.: Conceptualization, Funding acquisition, Project administration, Supervision, Validation, Writing – Reviewing and Editing. S.N.J.: Methodology, Resources, Software. C.W.S.: Validation. E.M.: Writing – Reviewing and Editing. S.S.: Writing – Reviewing and Editing.

## Appendix A

**Table A1.** Primary studies included for review.

No	Title	Authors	Year
1	An Algorithmic Approach for Generating Behavioral UML Models Using Natural Language Processing	Esra A. Abdelnabi, Abdelsalam M. Maatuk, Tawfig M. Abdelaziz	2021
2	An approach to identify use case scenarios from textual requirements specification	Saurabh Tiwari, Deepti Ameta, Asim Banerjee	2019
3	An Ontology-based Approach to Automate the Software Development Process	Kathirgamasegaran Athiththan, Selvaratnam Rovinsan, Srijeevahan Sathveegan, Nahanaa Gunasekaran, Kamila S. A. W. Gunawardena, Dharshana Kasthurirathna	2018
4	Automated Goal Model Extraction from User Stories Using NLP	Tugce Gunes, Fatma Basak Aydemir	2020
5	Automated Requirements Formalisation for Agile MDE	Kevin Lano, Sobhan Yassipour-Tehrani, M. A. Umar	2021
6	Automatic Extraction of Structural Model from Semi Structured Software Requirement Specification	Shweta, Prof. Ratna Sanyal, Dr. Bibhas Ghosal	2018
7	Automatic Generation of UML Diagrams from Scenario-based User Requirements	Abdelkareem M. Alashqar	2021
8	Automatic Transformation of User Stories into UML Use Case Diagrams using NLP Techniques	Meryem Elallaoui, Khalid Nafil, Raja Touahni	2018
9	From User Stories to Models: A Machine Learning Empowered Automation	Takwa Kochbati, Shuai Li, Sébastien Gérard, Chokri Mraidha	2021
10	From User Stories to UML Diagrams Driven by Ontological and Production Model	Samia Nasiri, Yassine Rhazali, Mohammed Lahmer, Amina Adadi	2021

No	Title	Authors	Year
11	From user stories to use case scenarios towards a generative approach	Fabian Gilson, Calum Irwin	2018
12	Generating sequence diagrams from Arabic user requirements using MADA+TOKAN tool	Nermeen Alami, Nabil Arman, Faisal Khamayseh	2020
13	Generating UML Use Case and Activity Diagrams using NLP Techniques and Heuristics Rules	Abdelsalam M. Maatuk, Esra A. Abdelnabi	2021
14	Generating Use Case Scenarios from User Stories	Fabian Gilson, Matthias Galster, François Georis	2020
15	iMER: Iterative process of entity relationship and business process model extraction from the requirements	Muhammad Javed, Yuqing Lin	2021
16	Iterative Process for Generating ER Diagram from Unrestricted Requirements	Muhammad Javed, Yuqing Lin	2018
17	Knowledge Extraction from Natural Language Requirements into a Semantic Relation Graph	Aaron Schlutter, Andreas Vogelsang	2020
18	Towards a Generation of Class Diagram from User Stories in Agile Methods	Samia Nasiri, Yassine Rhazali, Mohammed Lahmer, Nouredine Chenfour	2020
19	Towards Transforming User Requirements to Test Cases Using MDE And NLP	Sai Chaithra Allala, Juan P. Sotomayor, Dionny Santiago, Tariq M. King, and Peter J. Clarke	2019
20	Use Case Extraction through Knowledge Acquisition	D. G. Vasques, G.S Santos, F. D. Gomes, J. F. Galindo, P. S. Martins	2019


## References

- Abdelnabi, E. A., Maatuk, A. M., & Abdelaziz, T. M. (2021). An Algorithmic Approach for Generating Behavioral UML Models Using Natural Language Processing. In *ICEMIS'21: The 7th International Conference on Engineering & MIS 2021*. ACM. <https://doi.org/10.1145/3492547.3492612>
- Abdelnabi, E. A., Maatuk, A. M., Abdelaziz, T. M., & Elakeili, S. M. (2020). Generating UML Class Diagram using NLP Techniques and Heuristic Rules. In *2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering*, (pp. 277–282). IEEE. <https://doi.org/10.1109/STA50679.2020.9329301>
- Alami, N., Arman, N., & Khamayseh, F. (2019). Generating Sequence Diagrams from Arabic User Requirements using MADA+TOKAN Tool. *International Arab Journal of Information Technology*, 17(1), 65–72. <https://doi.org/10.34028/iajit/17/1/8>
- Alashqar, A. M. (2021). Automatic generation of uml diagrams from scenario-based user requirements. *Jordanian Journal of Computers and Information Technology*, 7(2), 180–191. <https://doi.org/10.5455/jcit.71-1616087318>
- Allala, S. C., Sotomayor, J. P., Santiago, D., King, T. M., & Clarke, P. J. (2019). Towards transforming user requirements to test cases using MDE and NLP. In *International Computer Software and Applications Conference*, (pp. 350–355). IEEE. <https://doi.org/10.1109/COMPSAC.2019.10231>
- Alzayed, A., & Al-Hunaiyyan, A. (2021). A Bird's Eye View of Natural Language Processing and Requirements Engineering. *International Journal of Advanced Computer Science and Applications*, 12(5), 81–90. <https://doi.org/10.14569/IJACSA.2021.0120512>
- Athiththan, K., Rovinsan, S., Sathveegan, S., Gunasekaran, N., Gunawardena, K. S. A. W., & Kasthurirathna, D. (2018). An Ontology-based Approach to Automate the Software Development Process. In *2018 IEEE 9th International Conference on Information and Automation for Sustainability, ICIAfS 2018*, (pp. 1–6). IEEE. <https://doi.org/10.1109/ICIAfS.2018.8913339>
- Bik, N., Lucassen, G., & Brinkkemper, S. (2017). A Reference Method for User Story Requirements in Agile Systems Development. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*. IEEE. <https://doi.org/10.1109/REW.2017.83>
- Bozyiğit, F., Aktaş, Ö., & Kilinc, D. (2021). Linking software requirements and conceptual models: A systematic literature review. *Engineering Science and Technology, an International Journal*, 24(1), 71–82. <https://doi.org/10.1016/j.jestch.2020.11.006>
- Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison Wesley Longman Publishing.
- Dalpiaz, F., & Brinkkemper, S. (2018). Agile requirements engineering with user stories. In *2018 IEEE 26th International Requirements Engineering Conference*, (pp. 506–507). IEEE. <https://doi.org/10.1109/RE.2018.00075>

- Elallaoui, M., Nafil, K., & Touahni, R. (2015). Automatic generation of UML sequence diagrams from user stories in Scrum process. In *2015 10th International Conference on Intelligent Systems: Theories and Applications, SITA 2015*. IEEE. <https://doi.org/10.1109/SITA.2015.7358415>
- Elallaoui, M., Nafil, K., & Touahni, R. (2018). Automatic Transformation of User Stories into UML Use Case Diagrams using NLP Techniques. *Procedia Computer Science*, 130, 42–49. <https://doi.org/10.1016/j.procs.2018.04.010>
- Fischbach, J., Vogelsang, A., Spies, D., Wehrle, A., Junker, M., & Freudenstein, D. (2020). SPECMATE: Automated Creation of Test Cases from Acceptance Criteria. In *2020 IEEE 13th International Conference on Software Testing, Verification and Validation, ICST 2020*, (pp. 321–331). IEEE. <https://doi.org/10.1109/ICST46399.2020.00040>
- Gallud, J. A., & Fardoun, H. M. (2019). UML and agile methods: Looking for an agreement. In *Proceedings of the 13th International Conference on Software Technologies*, (pp. 780–785). ScitePress. <https://doi.org/10.5220/0006940907800785>
- Gebretsadik, K. K. (2020). Challenges and Opportunity of UML Diagram for Software Project development as a complete Modeling Tool. *IOSR Journal of Mobile Computing & Application*, 7(3), 46–48.
- Gilson, F., Galster, M., & Georis, F. (2020). Generating use case scenarios from user stories. In *2020 IEEE/ACM International Conference on Software and System Processes, ICSSP 2020* (pp. 31–40). ACM. <https://doi.org/10.1145/3379177.3388895>
- Gilson, F., & Irwin, C. (2018). From user stories to use case scenarios towards a generative approach. In *Proceedings of the 25th Australasian Software Engineering Conference, ASWEC 2018*, (pp. 61–65). IEEE. <https://doi.org/10.1109/ASWEC.2018.00016>
- Gunes, T., & Aydemir, F. B. (2020). Automated Goal Model Extraction from User Stories Using NLP. In *Proceedings of the IEEE International Conference on Requirements Engineering*, (pp. 382–387). IEEE. <https://doi.org/10.1109/RE48521.2020.00052>
- Gunes, T., Oz, C. A., & Aydemir, F. B. (2021). ArTu: A Tool for Generating Goal Models from User Stories. In *Proceedings of the IEEE International Conference on Requirements Engineering*, (pp. 436–437). IEEE. <https://doi.org/10.1109/RE51729.2021.00058>
- Javed, M., & Lin, Y. (2018). Iterative process for generating ER diagram from unrestricted requirements. In *ENASE 2018 – Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering*, (pp. 192–204). ScitePress. <https://doi.org/10.5220/0006778701920204>
- Javed, M. A., & Lin, Y. (2021). iMER: Iterative process of entity relationship and business process model extraction from the requirements. *Information & Software Technology*, 135, 106558. <https://doi.org/10.1016/j.infsof.2021.106558>
- Kitchenham, B. (2004). *Procedures for Performing Systematic Reviews*. Keele University Technical Report TR/SE-0401. <https://www.inf.ufsc.br/~aldo.vw/kitchenham.pdf>
- Koç, H., Erdoğan, A. M., Barjakly, Y., & Peker, S. (2021). UML Diagrams in Software Engineering Research: A Systematic Literature Review. *Proceedings*, 74(1), 13. <https://doi.org/10.3390/proceedings2021074013>
- Kochbati, T., Li, S., Gérard, S., & Mraidha, C. (2021). From user stories to models: A machine learning empowered automation. In *Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development*, (pp. 28–40). ScitePress <https://doi.org/10.5220/0010197800280040>
- Lano, K., Yassipour-Tehrani, S., & Umar, M. A. (2021). Automated Requirements Formalisation for Agile MDE. Companion In *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, (pp. 173–180). <https://doi.org/10.1109/MODELS-C53483.2021.00030>
- Loniewski, G., Insfran, E., & Abrahão, S. (2010). A Systematic Review of the use of Requirements Engineering Techniques in Model-Driven Development. In Petriu, D.C., Rouquette, N., Haugen, Ø. (eds) *Model Driven Engineering Languages and Systems, MODELS 2010* (pp. 213–227). [https://doi.org/10.1007/978-3-642-16129-2\\_16](https://doi.org/10.1007/978-3-642-16129-2_16)
- Maatuk, A. M., & Abdelnabi, E. A. (2021). Generating UML use case and activity diagrams using NLP techniques and heuristics rules. In *DATA'21: International Conference on Data Science, E-learning and Information Systems 2021*, (pp. 271–277). ACM. <https://doi.org/10.1145/3460620.3460768>
- Mustafa, A., Kadir, W. H. W., Ibrahim, N., Shah, M. A., Younas, M., Khan, A. A., Zareei, M., & Alanazi, F. (2021). Automated Test Case Generation from Requirements: A Systematic Literature Review. *Computers, Materials & Continua*, 67(2), 1819–1833. <https://doi.org/10.32604/cmc.2021.014391>
- Nasiri, S., Rhazali, Y., Lahmer, M., & Adadi, A. (2021). From User Stories to UML Diagrams Driven by Ontological and Production Model. *International Journal of Advanced Computer Science and Applications*, 12(6), 333–340. <https://doi.org/10.14569/IJACSA.2021.0120637>
- Nasiri, S., Rhazali, Y., Lahmer, M., & Chenfour, N. (2020). Towards a Generation of Class Diagram from User Stories in Agile Methods. *Procedia Computer Science*, 170, 831–837. <https://doi.org/10.1016/j.procs.2020.03.148>
- Nazir, F., Butt, W. H., Anwar, M. W., & Khan Khattak, M. A. (2017). The applications of natural language processing (NLP) for software requirement engineering – A systematic literature review. In: Kim, K., Joukov, N. (eds) *Information Science and Applications 2017*, (pp. 485–493). Springer. [https://doi.org/10.1007/978-981-10-4154-9\\_56](https://doi.org/10.1007/978-981-10-4154-9_56)
- Page, M. J., Moher, D., Bossuyt, P. M., Boutron, I., Hoffmann, T., Mulrow, C. D., Shamseer, L., Tetzlaff, J., Akl, E. A., Brennan, S., Chou, R., Glanville, J., Grimshaw, J., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E., Mayo-Wilson, E., McDonald, S., ... McKenzie, J. E. (2021). PRISMA 2020 explanation and elaboration: updated guidance and exemplars for reporting systematic reviews. *BMJ*, 372, n160. <https://doi.org/10.1136/bmj.n160>

- Raharjana, I. K., Siahaan, D., & Fatichah, C.** (2021). User Stories and Natural Language Processing: A Systematic Literature Review. *IEEE Access*, 9, 53811–53826. <https://doi.org/10.1109/ACCESS.2021.3070606>
- Schlutter, A., & Vogelsang, A.** (2020). Knowledge Extraction from Natural Language Requirements into a Semantic Relation Graph. In *Proceedings – 2020 IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSEW 2020*, (pp. 373–379). ACM. <https://doi.org/10.1145/3387940.3392162>
- Seidl, M., Scholz, M., Huemer, C., & Kappel, G.** (2015). The Use Case Diagram. In *UML @ Classroom. Undergraduate Topics in Computer Science* (pp. 23–47). Springer. [https://doi.org/10.1007/978-3-319-12742-2\\_3](https://doi.org/10.1007/978-3-319-12742-2_3)
- Shweta, Sanyal, R., & Ghoshal, B.** (2018). Automatic Extraction of Structural Model from Semi Structured Software Requirement Specification. In *Proceedings – 17th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2018*, (pp. 543–548). IEEE. <https://doi.org/10.1109/ICIS.2018.8466406>
- Ternes, B., Rosenthal, K., & Strecker, S.** (2021). Automated assistance for data modelers combining natural language processing and data modeling heuristics: A prototype demonstration. In *CEUR Workshop Proceedings, vol. 2958*, (pp. 25–30). <https://ceur-ws.org/Vol-2958/paper5.pdf>
- Thesing, T., Feldmann, C., & Burchardt, M.** (2021). Agile versus Waterfall Project Management: Decision Model for Selecting the Appropriate Approach to a Project. *Procedia Computer Science*, 181, 746–756. <https://doi.org/10.1016/j.procs.2021.01.227>
- Tiwari, S., Ameta, D., & Banerjee, A.** (2019). An approach to identify use case scenarios from textual requirements specification. In *ISEC'19: Proceedings of the 12th Innovations on Software Engineering Conference*. ACM. <https://doi.org/10.1145/3299771.3299774>
- Vasques, D. G., Santos, G. S., Gomes, F. D., Galindo, J. F., & Martins, P. S.** (2019). Use Case Extraction through Knowledge Acquisition. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2019*, (pp. 624–631). IEEE. <https://doi.org/10.1109/IEMCON.2019.8936279>
- Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K. J., Ajagbe, M. A., Chioasca, E., & Batista-Navarro, R. T.** (2021). Natural language processing for requirements engineering. *ACM Computing Surveys*, 54(3), 1–41. <https://doi.org/10.1145/3444689>

---

**Editorial record:** The article has been peer-reviewed. First submission received on 13 December 2022. Revisions received on 13 February 2023 and 7 March 2023. Accepted for publication on 12 March 2023. The editor in charge of coordinating the peer-review of this manuscript and approving it for publication was Zdenek Smutny .

---

Acta Informatica Pragensia is published by Prague University of Economics and Business, Czech Republic.

ISSN: 1805-4951

---