SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Visualisation of User Stories to UML use Case Diagram

Mohammad Nazrul Mornie[1], Nurfauza Jali[1,*], Cheah Wai Shiang[1], Edwin Mit[1], Suhaila Saee[1], Suriati Khartini Jali[2], Desmond Greer[3]

[1] Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia
[2] Institute for Tourism Research and Innovation (ITRI), Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia
[3] School Of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT7 1NN, United Kingdom

ARTICLE INFO

Keywords:
Requirement engineering; user stories; use case diagram; UML model; natural language processing

ABSTRACT

The growing usage of Agile methodology in software development projects among industry professionals (software engineers, system analysts, requirement engineers, etc.) and academia (software engineering students) leads to the need for the implementation of UML diagram for requirements modelling. Use case diagram, an example of UML diagram, is a very powerful tool to model the requirements specified by the users while also helping the development teams understand the functionalities and interactions between users and the system. However, there is a lack of a system or tool that can perform the operation to visualise the use case diagram directly from user stories because generating this diagram manually requires a deep understanding of the requirements and effective communications with stakeholders and it consume lots of time while previous studies which relate to this study are unable to fulfil the relationship elements of use case diagram. This study will introduce a method to visualise the use case diagram from structured textual user stories by utilising Natural Language Processing (NLP) and application of logical rules which will be done in four stages, namely Requirement Gathering, Natural Language Processing, Application of Logical Rules and UML Diagram Generation. A tool named Stanford CoreNLP will be used to perform four techniques of NLP: tokenisation, stemming and lemmatisation, POS tagging and dependency parsing to process the textual user stories, followed by applying the logical rules before generating the use case diagram. This study will propose a method to solve the gap, which is the problem with the generation of relationship elements, while contribute a semi-automated approach to generate a use case diagram from user stories.

1. Introduction

Unified Modelling Language (UML) is a type of modelling language used during the design phase in an Agile methodology. It is used to model the requirements gathered from the stakeholders. There are many UML diagram types, such as use case diagram, activity diagram, sequence diagram, class diagram and many others [1]. These diagrams serve a different function for each of them. The usage

* *Corresponding author.*
E-mail address: jnurfauza@unimas.my

https://doi.org/10.37934/araset.63.3.6880

of UML diagrams, including its associated challenges, in a software development project among industry professionals and Software Engineering students will be one of the concerns in this research. There are many benefits of using the UML diagram to model the requirements, but the most obvious benefit is it can help software developers to visualise complex and relatively large systems. Different types of UML models are used in Agile methodology, including use case diagram, class diagram, sequence diagram and activity diagram.

In this research, the type of UML diagram that will be focused on is the use case diagram. Use case diagram is a type of UML diagram used to describe a system's high-level functions. This means that it is used to visualise and organise the functionalities of a system. Besides, the use case diagram is also a very important part of requirements engineering because it will help the developers understand how the system should work as the stakeholder's desire. There are three main components of a use case diagram which are actors, use cases and relationship. These components are associated with the purpose of the use case diagram which are to visualise and understand the possible interactions between the users and the system [2].

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 [3]. 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 [3]. Many types of NLP techniques may be used to train a computer to understand natural language and communicate with humans.

On the other hand, user stories are also significant in Agile methodology. This is because the user stories help the developer get ideas about the software from their end-user's perspective. The user stories are also crucial to specify the requirements of the software. Once the requirements for software have been specified, there comes the need to design the software, including the preparation of UML models such as use case diagram, activity diagram and sequence diagram.

Consider software development from the viewpoint of an Agile project. There is a design phase that demands the to team members create a UML model such as use case diagram, to visualise the requirements of the software application they are proposing to construct. However, there is an absence of a system that can automatically help to generate UML use case diagrams through user stories. Besides, the traditional method of manually generating the diagram during the design phase is also less effective and highly time-consuming. Although there are similar studies that generate UML use case diagrams from user stories, relationship elements of the diagram still yet to be fulfilled. This element is one of the most crucial components of a use case diagram.

To effectively overcome the problem, a set of objectives and questions have been formulated for this research. There are three research questions (RQs) that have been identified: (RQ1) What challenges do academia and industry professional face in designing UML Model, (RQ2) How can a framework and prototype be developed for automatically generating a UML use case diagram from user stories and (RQ3) How to test the framework's efficacy with a developed prototype. As a response to the questions, proper objectives have been set for this research: to formulate and compile typical challenges faced by academia and industry professionals when designing UML use case diagrams and the best method for automating the generation of UML use case diagram from user stories, to develop a framework of an automated system that generates UML use case diagram from user stories and to test and evaluate the effectiveness of the framework using a prototype.

So, this paper will present a semi-automated approach to generate UML use case diagram from structured user stories. Natural Language Processing (NLP) will be utilised in order to generate the UML model from the text of user stories.

2. Literature Review

A systematic review has been conducted to review relevant studies conducted to achieve related goals, problem solving and answering relevant questions with this study. The first highlight of the review method is the 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.

2.1 Review Results

As mentioned earlier, the results of the review are specified based on the research questions. There are three research questions that need to be figured which are:

i. RQ1: What challenges do academia and industry professional face to design UML Model?
ii. RQ2: How to develop a framework and prototype for automatically generating UML use case diagram from user stories?
iii. RQ3: How to test framework's efficacy with a developed prototype?

2.1.1 What are the challenges faced by academia 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 [2,3,9,18]. Besides, there are several problems associated with the process of designing UML diagrams which come from the complexity and ambiguity of natural language [1,14,15]. 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 [25]. Based on empirical research, professional data modellers such as system analysts faced difficulty finding meaningful identifiers [23]. 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 [3]. The mistakes are easy to present because there is usually a large number 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 [10]. Table 1 summarises the challenges associated with designing UML models or other conceptual models found in the primary studies.

Table 1
Challenges associated with designing UML models or other conceptual models

No	Challenges/problems	Primary study
1	High time consumption	[1,3,9,18]
2	Requires lots of effort	[1,3,9,12,13,18]
3	High costs	[1,3,9,18]
4	Complexity of natural language	[14,15,18]
5	Ambiguity of natural language	[14,15,18]
6	Information duplicity, incompleteness, redundancy, ambiguity	[25]
7	Actors and their action confusion	[25]
8	Difficulty finding important identifiers	[23]
9	Process of manually transforming user stories to UML models is error-prone due to high volume of user stories	[3]
10	Missing important information from user stories	[10]

2.1.2 How to develop a framework and prototype for automatically generating UML use case diagram from user stories?

To answer this research question, there are two main components of focus to gather appropriate and relevant information for the review. First, 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 several NLP tools that can be used to assist the generation of UML models from user stories. Some examples are Spacy, Stanford CoreNLP and NLTK. Table 2, Table 3 and Table 4 summarises the results for this question.

Table 2
NLP used in primary studies

No.	NLP tool	Primary study
1	TreeTagger Parser	[7]
2	Stanford CoreNLP	[4-6,14,15,17-19,21,22]
3	MADA+TOKAN	[3]
4	spaCy	[10,12,16]
5	Word2vec	[16]
6	Apache OpenNLP	[17]
7	DeepSRL	[21]
8	WordNet	[20]
9	Stanford NL Parser	[24]

Table 3
NLP techniques used in primary studies

No.	NLP techniques	Primary studies
1	POS Tagging	[8]
2	Tokenisation, POS Tagging, Lemmatisation and Stemming, Parse Tree and Type Dependencies, OpenIE	[1]
3	POS Tagging, Type Dependencies	[24]
4	Tokenisation, POS Tagging, Disambiguation, Discretisation, Morphological Disambiguation and Stemming, Lemmatisation	[3]
5	Tokenisation, POS Tagging, Lemmatisation and Stemming, Parse Tree and Type Dependencies, OpenIE	[18]
6	Stemming And Lemmatisation, POS Tagging, Dependencies Tree	[10]
7	Tokenisation, Lemmatisation, POS Tagging	[14,15]
8	Tokenisation, POS Tagging, Dependency Parsing	[4]

9	Tokenisation, POS Tagging	[5,12]
10	Word level semantic, tokenisation	[16]
11	Text Splitting, Tokenisation, POS Tagging, Lemmatisation, Type Dependencies	[19]
12	POS Tagging, Syntax Tree	[17]
13	POS Tagging, Named-Entity Recognition	[11]
14	Tokenisation, Sentence Splitting, POS Tagging, Lemmatisation, Dependency Parsing, Coreference Resolution, Semantic Role Labelling	[21]
15	Sentence Splitting, Text Parser, Universal Dependency, Lemmatisation	[22]
16	Tokenisation, POS Tagging, Conference Resolution, Stemming	[20]

Table 4
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

2.1.3 How to test framework's efficacy with a developed prototype?

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.,* [8] 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.,* [1]. 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.,* [3]. 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 *et al.,* [18]. 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 by Abdelnabi *et al.,* [1]. 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 [4] 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.

3. Methodology

This section will discuss the methodology used for this research. There are four main stages in this research which are Stage 1: Prior investigation and content analysis, Stage 2: Identification of model design consideration and attributes, Stage 3: Further investigation and Stage 4: Evaluation of framework. These stages are crucial for this research because they will be used as the main guideline to identify processes, procedures and techniques required. Figure 1 shows the stages and activities for each of this research.

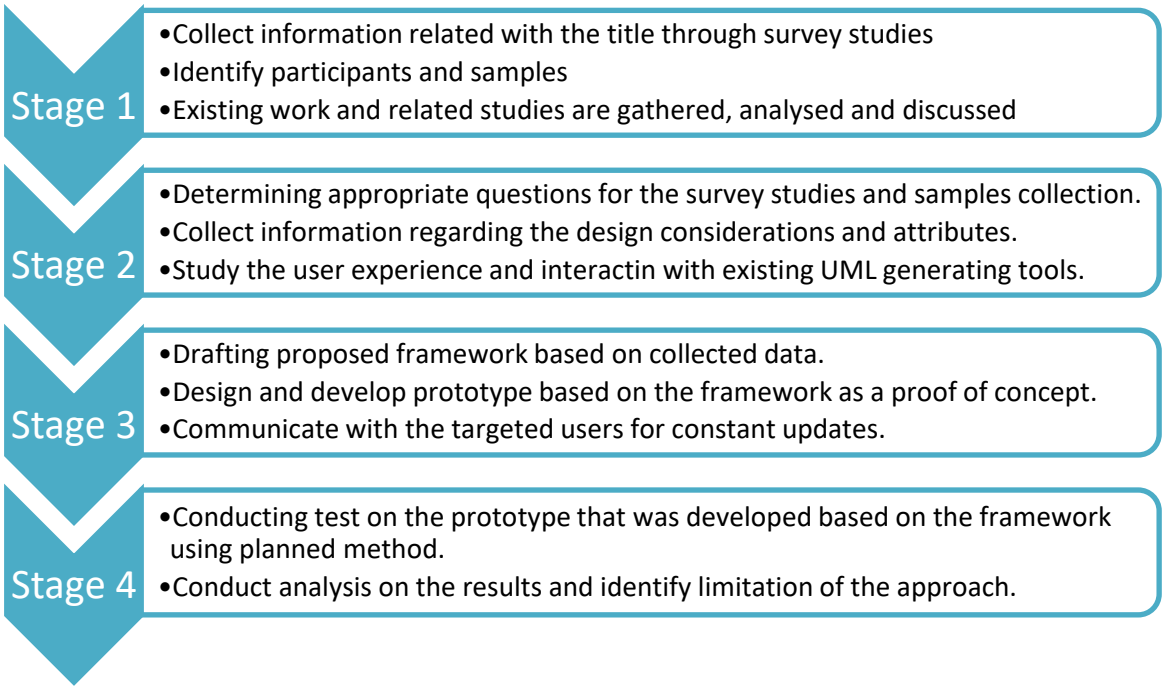

Fig. 1. Methodology

4. Preliminary Data

For the collection of preliminary data, a survey study has been conducted where the respondents can be grouped into two which are industry professionals and internship students from Universiti Malaysia Sarawak. The purpose of this survey study is to obtain ideas and information about the challenge faced by industry professionals such as Software Engineers, System Analysts, Software Developer, Requirement Engineer, as well as internship students from FCSIT, UNIMAS. This group of people are selected as the participants of this survey study because they are regularly exposed and dealt with the requirement design which may involves UML Model generation. The survey was conducted using an online survey administration software named Google Form. In addition, the question used for conducting this survey is built based on the research questions.

Besides, a web-based application has been developed as part of the method to collect the preliminary data. The web-based application will be used to collect UML use case diagram samples mainly from the software engineering students at UNIMAS. The samples collected will be used to conduct test and evaluation of prototype which is done as proof of the framework's concept.

4.1 Data Collected from Survey Studies

For the survey's results, several questions will be selected to highlight the main purpose of this study. For the first section which the background information of the participant, Scrum is the most used development method by the students which shows a percentage of 70 percent, followed by Kanban (7.8 percent), Waterfall (5.2 percent) and Rapid Application Development (5.2 percent). The results for the most used methodology among the students are presented in Figure 2.

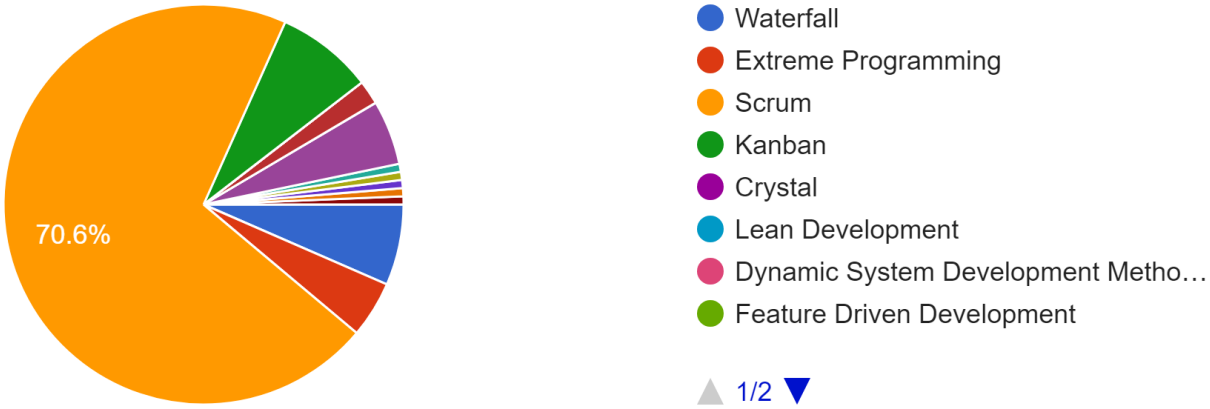

Fig. 2. Results of most used methodology among the students

For the requirement gathering method, interview is the most popular among the students which show a percentage of 89.7 percent followed by brainstorming with 70.3 percent and prototyping with 48.4 percent. Figure 3 shows the results of the requirement gathering method used by the students.

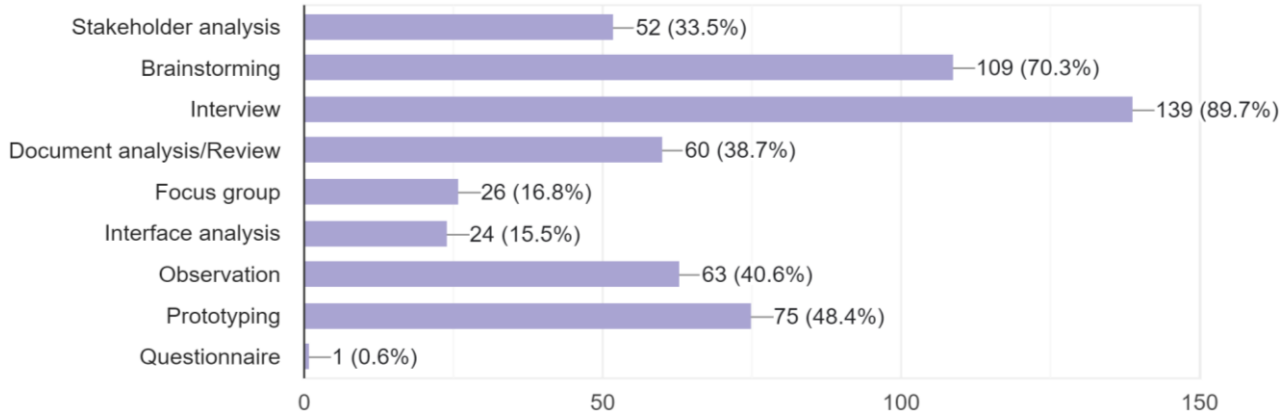

Fig. 3. Results of most used requirement gathering methods of the students

4.2 Samples Collected from US2UML Website

A total of 17 samples of UML and Goal model have been collected through the website. The samples are submitted along with the user stories. The website is accessible at https://www.us2uml.org/.

5. A Framework for UML Use Case Diagram Generation

This subsection will introduce the proposed approach for this research. The approach of this research is proposed based on the gap and knowledge found from previous research. There are several components that are considered crucial for this proposed approach which are the selection of UML diagram, chosen NLP tools and the process of UML use case diagram generation.

5.1 Selected UML Diagram

The use case diagram has been selected as the UML model to be used in the suggested approach for the development of the framework. The use case diagram was chosen since it is one of least frequently used diagrams for automation by previous researchers' studies. Besides, the use case

diagram is also selected because some of the previous research cannot fully include the important features of use case diagrams especially relationships such as include, extend and generalisation.

5.2 Natural Language Processing (NLP) Tools

The tools that will be used in this research will be the Stanford CoreNLP. It is chosen because it produces the best result from previous research. Alashqar [4] use Stanford CoreNLP to generate sequence diagrams elements from scenario-based user requirements. The accuracy of the generated elements is very high ranging from 77% to 90% for elements such as actors, messages, operations, caller object and receiver object. While for Nasiri *et al.,* [19], they use Stanford CoreNLP to extract object-oriented design elements for the generation of UML class diagram. By using Stanford CoreNLP in their proposed approach, the accuracy of generated class diagram elements is 98% which is very satisfactory. For those reason, the Standford CoreNLP will also be used in this research.

5.3 Generation of UML Model

In this research, the UML model, a use case diagram, will be generated through four main phases. The phases are requirement gathering, Natural Language Processing, Application of Logical Rules and UML diagram generation. Each phase will involve different activities. Detailed explanation about each phase will be further discussed in the following sub-section.

5.4 Requirement Gathering

The first phase, requirement gathering involves collecting requirements from the users. The requirement that is collected here is in the form of user-stories. In order to accomplish this, a template will be created in order to record the requirements from the users. Users will be able to fill in only specific information using this template. The template sample for collecting the user requirements are shown in Figure 4.

As a (1) _____, I want to (2) _____. (1) _____ is a/may be type of (4) _____. (2) _____ will cause/require (3) _____. (5) _____ occurs/present if (1) _____ (6) _____.

Fig. 4. Template for collecting user stories

5.5 Natural Language Processing

Natural language processing (NLP) techniques will be applied to the requirements or stories that have been collected. These techniques are tokenisation, stemming and lemmatisation, Part-of-Speech (POS) tagging and dependency parsing. The NLP itself is the most crucial part of this approach because it will be the phase where the elements of the use case diagram are generated. This will be followed by POS tagging, stemming and lemmatisation and dependency parsing, which will complete the cycle of natural language processing. Figure 5 depicts the NLP steps in order to aid with visualising them.

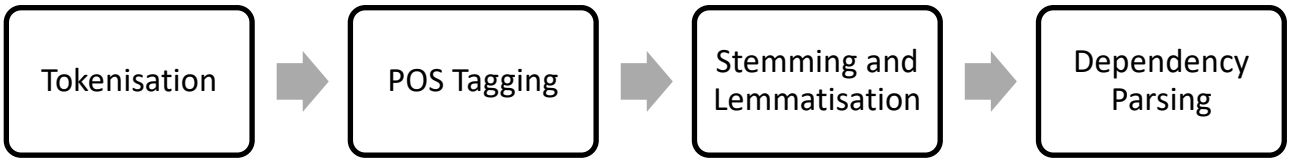

Fig. 5. NLP pipeline used to process the user stories

Tokenisation is the first NLP technique that will be performed on the structured textual user stories. There are three types of tokenisation which are word tokenisation, sub-word tokenisation and character tokenisation. For this research, the type of tokenisation that will be applied is word tokenisation because the textual user stories will be converted into tokens by each word. This process is performed using Stanford CoreNLP.

After the text has been converted to tokens, it needs to be labelled or tagged to indicate the part of speech and categorising it based on its grammatical and lexical categories such as tense, verb, noun, etc. Since this research is using Stanford CoreNLP to perform the POS tagging, Universal POS tags will be used to refer to the result of POS tagging.

The next NLP technique that will be applied is stemming and lemmatisation. This technique is done to reduce the word into its base form. Natural language is complex in nature. Some similar words are used to represent the same meaning which are the synonyms and others are inflectional and derivational forms of the word. The goal of stemming and lemmatisation is to reduce these words into their base form so it will be more conventional to produce a parse tree in the next NLP technique that will be applied.

The last NLP technique that will be used in this proposed approach is dependency parsing. Dependency parsing is a type of NLP technique that is used to identify the grammatical structure of the sentences in the structured textual user stories input. This technique is crucial in this proposed approach because it will be used to identify the relationship between each word of the user stories so it will be easier to apply the logical rules before the elements of the UML use case diagram can be extracted.

5.6 Application of Logical Rules

The output from the NLP will be applied with a set of logical rules. These rules will be used to increase the accuracy of generated elements from NLP. There are three types of rules that will be applied to the output from NLP which are actor identification rules, use case identification rules and relationship identification rules. The output from this phase will be an Extensible Markup Language (XMI) file containing the elements of the use case diagram. After the rules have been applied, the process will move to the last phase which is UML diagram generation. The code snippets used to apply the rules are shown in Table 5, Table 6 and Table 7.

Table 5
Code snippets for actor identification

Code Snippets 1: Actor Identification

```
for sentence in doc.sentences:
  for word in sentence.words:
    if word.deprel == 'obl':
     if word.text in nouns:
       continue
     else:
       actor = word.text
       nouns.append(actor)
```

Table 6
Code snippets for use case identification

Code Snippets 2: Use Case Identification

```
for sentence in doc.sentences:
  for word in sentence.words:
    if word.deprel == 'root':
      col_verb = word.text
      verbs.append(word.text)
      for childrens in word.childen:
        if childrens.deprel == 'obj':
          use_case = f"{col_verb}{childrens.text} {actor}"
          verbs.append(use_case)
```

Table 7
Code snippets for relationship identification

Code Snippets 3: Relationship Identification

```
for sentence in doc.sentences:
  if 'include' in sentence.text:
    include_action = sentence.verbs[0].text
    included_actions = [verb.text for verb
          in sentence.verbs[1:]]
    includes.append((include_action,
          included_actions))
  elif 'extend' in sentence.text:
    extend_action = sentence.verbs[0].text
    extended_action = sentence.verbs[1].text
    extends.append((extend_action,
          extended_action))
```

5.7 UML Diagram Generation

For the last phase in this proposed approach, which is the UML diagram generation, the use case diagram elements produced from previous phases will be fully utilised. The XMI file produced in the last phase will be used to visualise the use case diagram. It will be done using an Application Programming Interface (API) named PlantUML API. This API will perform the generation of use case diagram from the produced XMI file.

6. Statement of Limitations

At the completion of this research, there are three questions that will be answered which are RQ1, RQ2 and RQ3. This research will identify the challenge and difficulties faced by the industry professionals and academics while designing UML Model. Besides, methods to automate generation of UML Model from user stories will be figured out while at the same time a prototype and a framework will be developed to demonstrate the automatic generation of UML Model from user stories. Last but not least, this research will figure out the best way to evaluate the effectiveness of the framework using the developed prototype.

Although there are several benefits that can be obtained through this research, some limitations can be presented. The first limitation is this research will only propose the automation of user stories to the use case diagram. This is because including other UML models such as class diagrams and sequence diagrams will require more time and process. However, as an improvement or future

research idea, the knowledge and information from this research can be used to propose the generation for another type of UML models.

7. Conclusion

In conclusion, this study proposes a method for transforming user stories into UML models use case diagrams. This project aims to close a significant gap found in earlier studies where the creation of use case diagrams generally fell short of capturing all the essential aspects, especially the relationship elements. The development of a system capable of automatically generating UML use case diagrams represents a significant advancement of the work currently practiced by the professionals. By streamlining diagram creation, this system offers the potential to significantly enhance diagram accuracy, so freeing up precious time that would have been used for manual design and lessening the effort required to arrange complicated requirements. Moreover, this research goes beyond the mere implementation of NLP tools and techniques. It also lays the foundation for future investigations into various UML diagram types. With the logical rules proposed along with the NLP implementations and insights gained from this study, it will serve as a valuable resource for researchers generating other types of UML diagrams such as sequence diagrams, class diagrams and more. In a nutshell, this research has the potential to have a profound impact on the research community as a whole, as well as on students and professionals in the sector. Its contributions will extend beyond the immediate application, offering knowledge and insights that will continue to benefit and inspire future researchers.

Acknowledgement

This research is sponsored by Universiti Malaysia Sarawak (UNIMAS) under PILOT research grant, UNI/F08/PILOT/85043/2022.

References

[1] A. Abdelnabi, Esra, Abdelsalam M. Maatuk and Tawfig M. Abdelaziz. "An algorithmic approach for generating behavioral UML models using natural language processing." In *The 7th International Conference on Engineering & MIS 2021*, pp. 1-6. 2021. https://doi.org/10.1145/3492547.3492612

[2] Abdelnabi, Esra A., Abdelsalam M. Maatuk, Tawfig M. Abdelaziz and Salwa M. Elakeili. "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 (STA)*, pp. 277-282. IEEE, 2020. https://doi.org/10.1109/STA50679.2020.9329301

[3] Alami, Nermeen, Nabil Arman and Faisal Khamayseh. "Generating sequence diagrams from arabic user requirements using mada+ tokan tool." *Int. Arab J. Inf. Technol.* 17, no. 1 (2020): 65-72. https://doi.org/10.34028/iajit/17/1/8

[4] Alashqar, Abdelkareem M. "Automatic generation of uml diagrams from scenario-based user requirements." *Jordanian Journal of Computers and Information Technology* 7, no. 2 (2021). https://doi.org/10.5455/jjcit.71-1616087318

[5] Allala, Sai Chaithra, Juan P. Sotomayor, Dionny Santiago, Tariq M. King and Peter J. Clarke. "Towards transforming user requirements to test cases using MDE and NLP." In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, pp. 350-355. IEEE, 2019. https://doi.org/10.1109/COMPSAC.2019.10231

[6] Athiththan, Kathirgamasegaran, Selvaratnam Rovinsan, Srijeevahan Sathveegan, Nahanaa Gunasekaran, Kamila SAW Gunawardena and Dharshana Kasthurirathna. "An ontology-based approach to automate the software development process." In *2018 ieee international conference on information and automation for sustainability (iciafs)*, pp. 1-6. IEEE, 2018. https://doi.org/10.1109/ICIAFS.2018.8913339

[7] Elallaoui, Meryem, Khalid Nafil and Raja Touahni. "Automatic transformation of user stories into UML use case diagrams using NLP techniques." *Procedia computer science* 130 (2018): 42-49. https://doi.org/10.1016/j.procs.2018.04.010

[8] Elallaoui, Meryem, Khalid Nafil and Raja Touahni. "Automatic Transformation of User Stories into UML Use Case Diagrams using NLP Techniques." (2018): 42-49. https://doi.org/10.1016/j.procs.2018.04.010

[9] Fischbach, Jannik andreas Vogelsang, Dominik Spies andreas Wehrle, Maximilian Junker and Dietmar Freudenstein. "Specmate: Automated creation of test cases from acceptance criteria." In *2020 IEEE 13th international conference on software testing, validation and verification (ICST)*, pp. 321-331. IEEE, 2020. https://doi.org/10.1109/ICST46399.2020.00040

[10] Gilson, Fabian, Matthias Galster and François Georis. "Generating use case scenarios from user stories." In *Proceedings of the international conference on software and system processes*, pp. 31-40. 2020. https://doi.org/10.1145/3379177.3388895

[11] Gilson, Fabian and Calum Irwin. "From user stories to use case scenarios towards a generative approach." In *2018 25th Australasian Software Engineering Conference (ASWEC)*, pp. 61-65. IEEE, 2018. https://doi.org/10.1109/ASWEC.2018.00016

[12] Güneş, Tuğçe and Fatma Başak Aydemir. "Automated goal model extraction from user stories using NLP." In *2020 IEEE 28th International Requirements Engineering Conference (RE)*, pp. 382-387. IEEE, 2020. https://doi.org/10.1109/RE48521.2020.00052

[13] Günes, Tugçe and Fatma Basak Aydemir. "Automated Goal Model Extraction from User Stories Using NLP."

[14] Javed, Muhammad and Yuqing Lin. "Iterative Process for Generating ER Diagram from Unrestricted Requirements." In *ENASE*, pp. 192-204. 2018. https://doi.org/10.5220/0006778701920204

[15] Javed, Muhammad and Yuqing Lin. "iMER: Iterative process of entity relationship and business process model extraction from the requirements." *Information and Software Technology* 135 (2021): 106558. https://doi.org/10.1016/j.infsof.2021.106558

[16] Kochbati, Takwa, Shuai Li, Sébastien Gérard and Chokri Mraidha. "From user stories to models: A machine learning empowered automation." In *MODELSWARD 2022-9th International Conference on Model-Driven Engineering and Software Development*, vol. 1, pp. 28-40. SCITEPRESS-Science and Technology Publications, 2021. https://doi.org/10.5220/0010197800280040

[17] Lano, Kevin, Sobhan Yassipour-Tehrani and Muhammad Aminu Umar. "Automated requirements formalisation for agile MDE." In *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 173-180. IEEE, 2021. https://doi.org/10.1109/MODELS-C53483.2021.00030

[18] M. Maatuk, Abdelsalam and Esra A. Abdelnabi. "Generating UML use case and activity diagrams using NLP techniques and heuristics rules." In *International Conference on Data Science, E-learning and Information Systems 2021*, pp. 271-277. 2021. https://doi.org/10.1145/3460620.3460768

[19] Nasiri, Samia, Yassine Rhazali, Mohammed Lahmer and Amina Adadi. "From user stories to UML diagrams driven by ontological and production model." *International Journal of Advanced Computer Science and Applications* 12, no. 6 (2021). https://doi.org/10.14569/IJACSA.2021.0120637

[20] Nasiri, Samia, Yassine Rhazali and Mohammed Lahmer. "Towards a generation of class diagram from user stories in agile methods." In *Advancements in Model-Driven Architecture in Software Engineering*, pp. 135-159. IGI Global, 2021. https://doi.org/10.4018/978-1-7998-3661-2.ch008

[21] Schlutter, Aaron and Andreas Vogelsang. "Knowledge extraction from natural language requirements into a semantic relation graph." In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, pp. 373-379. 2020. https://doi.org/10.1145/3387940.3392162

[22] Sanyal, Ratna and Bibhas Ghoshal. "Automatic extraction of structural model from semi structured software requirement specification." In *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, pp. 543-58. IEEE, 2018. https://doi.org/10.1109/ICIS.2018.8466406

[23] Ternes, Benjamin, Kristina Rosenthal and Stefan Strecker. "Automated Assistance for Data Modelers combining Natural Language Processing and Data Modeling Heuristics: A Prototype Demonstration." In *ER Demos/Posters*, pp. 25-30. 2021.

[24] Tiwari, Saurabh, Deepti Ameta and Asim Banerjee. "An approach to identify use case scenarios from textual requirements specification." In *Proceedings of the 12th Innovations in Software Engineering Conference (formerly known as India Software Engineering Conference)*, pp. 1-11. 2019. https://doi.org/10.1145/3299771.3299774

[25] Vasques, Dildre G., Glaucia S. Santos, Franciene D. Gomes, Juan F. Galindo and Paulo S. Martins. "Use case extraction through knowledge acquisition." In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pp. 0624-0631. IEEE, 2019. https://doi.org/10.1109/IEMCON.2019.8936279