A Framework using NLP to automatically convert User-Stories into Use Cases in Software Projects

Ahmad Azzazi

Faculty of Information Technology, Applied Science Private University, Amman, Jordan

Abstract
To develop quality software within time and cost we have to follow the available rules and methods of modern software engineering. There are many different approaches used. At one hand we have in the analysis stage of a software engineering project the so called user stories that capture what the user does or needs to do within the developed Software System. User stories provide the functions that the system should have. User Stories are considered as a tool for rapid requirements gaining in the Analysis Phase of a Software Engineering Project. They are considered as a very effective way to communicate requirements but still inexpensive and easy to develop. But when we are going from the analysis phase to the design phase we face the lack of details in the requirements with user stories, therefore we better use the called Use Case instead of user stories. The use cases are a list of steps to achieve functionality or a goal within a software system. This interaction is normally done between a user and the software system. Therefore we want to combine both techniques in the approach of developing and validating requirements without extra efforts to rewrite the user stories into use cases. In this work I will provide a framework to automatically convert the user stories into use cases with the use of Natural language processing techniques. A case study is developed for this framework with results showing the percentage of correctly detected actors and use cases.

Keywords:
User Stories, Use Cases, Conversion from user stories to use case, NLP

1. Introduction

Developing quality software is an emergent research field where scientists are trying to put standards and methods to achieve a better development of software systems without the losses of time and cost.
There are many methods for developing software systems with different approaches.
Among the approaches is the agile approach which uses the user stories to summarize the functions of the software system. We will make a higher level of abstraction and more useful tool to understand. There is a challenge to make a smooth conversion from user stories into use cases, which should be studied and done within this work through the use of Natural language processing techniques.
The motivation for this work is to develop an intelligent framework system make it easier to gain rapid requirement for both the analysis phase and the design phase.

1.1 User Stories

User Stories are a very helpful tool when gaining the requirements from customers or users of a new software intensive system. They are simple, clear and short description of functional and nonfunctional requirements of the new system [1].
This short lingual description has the advantage of reducing the amount of effort done to describe or understand the features given by the users or customers of the new system capabilities.
There are different ways to write the user stories like sticky notes, cards arranged together in a box within a table or a wall (story map) to obtain discussion or planning of overall system features [1]. The advantage of such arrangement helps to obtain the understanding and the discussion of the new systems features in a rapid and clear way.
A User Story should be a lightweight substitution of the software requirements specification, where it depicts the most important artifacts in the software development processes like the agile development process, which depends on the rapid requirements development.
A Story map shows the branches and the sequence of the different user stories within the system work flow [2].
A User story represents a simple functionality of the software system. There are many ways to write user stories, like " As a [role] I want [Feature or Function] so that [value delivered to the business]"
Examples of user stories in a Student Registration System:
As a Student I want to see the course available so that I can register my courses.
As a Teacher I want to see the registered students in my course so that I send them the course material
As we can see from the previous example, we have the main three elements of a user story, first the user, second the functionality delivered to this user and finally the valued delivered to the business when using this functionality.
Finally; user stories should provide a common language making the understanding between users and the technical team. It should easily remove the gap between developers

Manuscript received May 5, 2017
Manuscript revised May 20, 2017

and users of the system when understanding the users' requirements .
Quality user stories must have quality attributes like negotiable, small, estimable, independent, verifiable, and testable [2].There are many tips to write good user stories, which should be considered in the proposed framework in this paper [2].
Figure 1 shows an example of user stories for the student registration system.

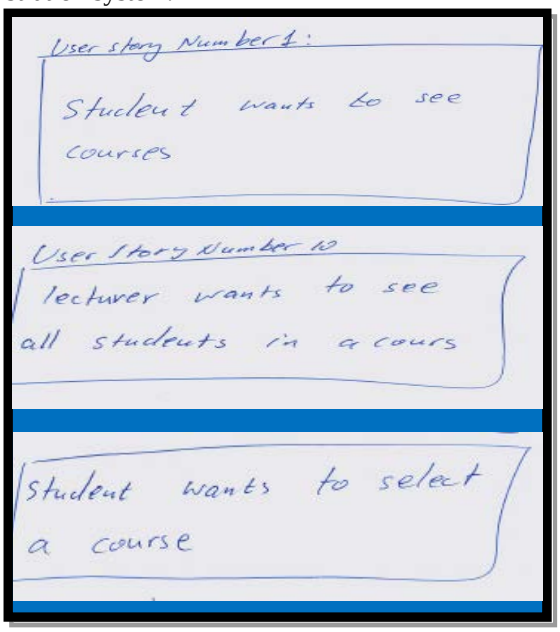

Figure 1: User Stories for a Student Registration System.

1.2 Use case and Use case diagram

Use cases are considered to be used for modelling requirements. They were developed by Ivar Jacobson in the year 1980; he had the idea using use cases in developing software requirements. Use cases describe the ways in which a user uses a system [3].
A use case could be described through three main parts, first the actor of the use case, who interacts actually with the developed software system. Secondly; The actor interacts with the system through the different use cases. The third thing of the use case is the functional achievement that the actor achieves when using the system (the reason for using the system for a specific functionality) [3].There are many benefits when using use cases like; Find all actors of the system, linking multiple stakeholders' needs the requirements, developing the desired behavior of the system, finding boundaries of a software system, validating system's, etc.
Use cases are easy to understand for different stakeholders involved in the development process of a system, they are also helpful to make agreement with customers and are reusable at different stages of the software development life cycle like; design phase, testing phase and in the documentation.
Figure (2) shows an example of a use cases; when a student wants to register his semester hours at the student registration system.

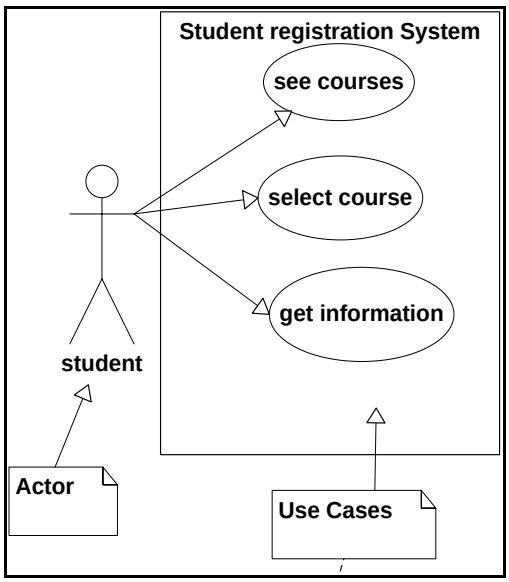

Figure 2: Use Cases for a Student Registration System.

2. Related Work

Olga Liskin and Co. [4] described in their work User stories as a guide for communication in Software Projects; they described the problem facing the developer of user stories such that not capturing all the requirement of the software project. They suggested how to utilize requirements artifacts effectively, what their benefits and challenges are, and how their scope granularity affects their utility. They showed how user Stories have helped the participants throughout the project.

Dan X. Houston et al [5] used for the measure the dominant factors in user story productivity to improve the software.

Mathias Landhäußer and Co. [6] presented in their work a tool that builds User Stories with natural language processing elements which contain links to API components to implement functionality testing with the result that that the links can be used to recommend the reusable test steps for new User Stories.

The Natural Language Processing Group and Co. [7] demonstrated a set of tools for language analysis systems for wide range of languages. Their System is modular with

important features for different natural language processing goals.

Venera Arnaoudova and Co. [5] presented the different and Natural Language Processing and Text Retrieval techniques used in Software Engineering projects. The described the application fields of NLP in software engineering projects.

Shuang Liu et al. [9] introduced automatic defect detection in use case documents by leveraging on advanced parsing techniques. They parsed the use case document using dependency parsing techniques. The parsing result is used then to forma an activity diagram. Then error detection technique is used to find error on activity diagram.

Leonard W. D'Avolio and Co. [10] introduced the Automated Retrieval Console (ARC) which builds a natural language processing framework by streamlining the many processes surrounding the development, evaluation, and deployment of natural language processing technologies. It is a graphical user interface that facilitates corpus import, reference set creation, annotation, and inter-annotator agreement calculation with many features helpful for machine learning testing capabilities with performance evaluation against a set of preferences.

Davide Falessi et al. [11] characterized in their work a comprehensive set of NLP techniques. Through their work they provided a level to the software analyst to detect the equivalent requirements. They concluded that simple natural language processing is more precise for requirement detection than complex ones.

Deva Kumar Deeptimahanti et al. [12] provided a semi-automated assistance for developers to generate UML models from normalized natural language requirements using Natural Language Processing techniques. . Their technique initially focuses on generating use-case diagram and analysis class model (conceptual model) followed by collaboration model generation for each use-case, which also provides requirement traceability.

Ian G. Harris [13] described in his work the extraction of behavioral information from a natural language specification to generate a formal behavioral description with clear and unambiguous semantics. In his work he employed the semantic parsing to identify key information describing bus transactions in the natural language specification

Neil Maiden et al. [14] introduced in their paper the application of one scenario-based approach - RESCUE - to discover requirements for DMAN, an air traffic management system for the UK's National Air Traffic Services. They revealed the importance of diverse information sources in the specification of use cases that enabled systematic requirements discovery.
Parastoo M. et al [15] described in their work through use cases the effort estimation. The study showed important factors influencing effort when developing large projects.

Chunhui Wang et al. [16] proposed a Use Case Modelling for System Tests Generation (UMTG), an approach that automatically generates executable system test cases from use case specifications.

Roman Pichler [2] showed some tips for writing good user stories, which we will user as a guide when developing user stories. The following is a summary of them; where the developer of good user stories should start first with user, collaborate with user effectively, write simple user stories, the developer should also start with epics, then decompose them into smaller parts, putting criteria to accept them after that, putting everything on paper, where they are visibly kept.

3. The proposed framework

Use Cases and Use Case Diagrams and nowadays more used, more understandable, more effective, more standardized and beloved by Software Engineers, Customers and IP Professionals. Therefore we want to remove in an automatic manner from the written user stories to Use Cases.
In this section we will propose our newly developed frame work to automatically convert User Stories into Use Cases. Figure (3) shows the different stages of this framework, in which different stages of different functionalities have to be considered.

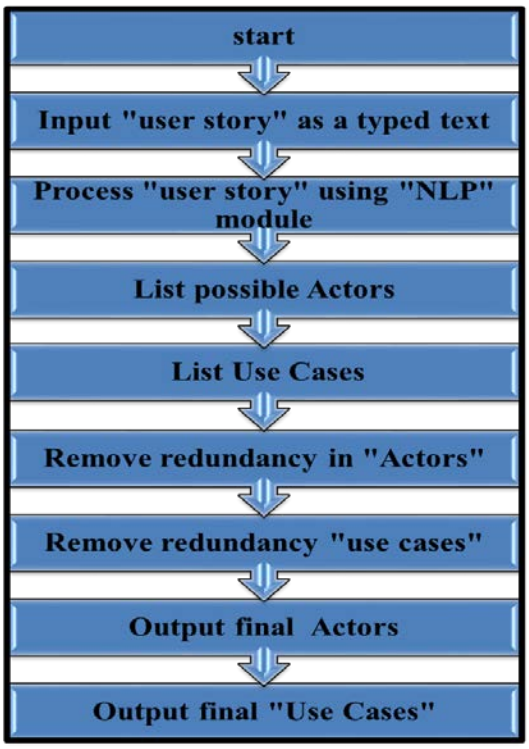

Figure [3]: The proposed Framework

As a preprocessing step before using this framework, a user story is either handwritten or typed using a guideline for writing effective User Stories, see Figure (4).

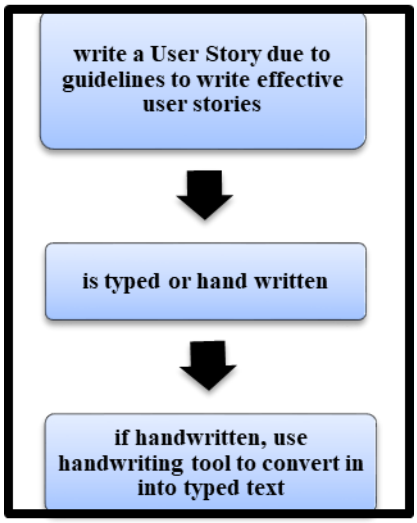

Figure [4]: Preprocessing stage

As a first step in this framework we begin with a typed "User Story" as input to the automatic converter. The core work of this converter is the "NLP" module, which takes the User Story as stage input and gives out two lists; a list possible "Actors" and another list of possible "Use Cases". It is the core or in other words the brain of the automatic conversion framework, therefore we will discuss some important issues of its design.

3.1 Natural Language Processing (NLP) Module

The general term Natural Language Processing has the meaning of using computer science in conjunction with linguistic rules to automatically understand and analyze human languages. NLP involves many disciplines like Artificial Intelligence, computational linguistic, Speech Processing, Speech synthesis and many other disciplines [ref for def. of NLP]. Natural Language Processing is one the most recent and widely used areas of Artificial Intelligence in a sentence that the computer should react to understand issues related to the human language production and processing like humans themselves.
Among widely known areas of application of NLP are automatic text summarizations, automatic machine translation, Optical character recognition, Morphological segmentation, Named entity recognition, Natural language generation, Natural language understanding, automatic language parsing, Part of speech tagging, Question answering systems, automatic test relationship extraction, Speech segmentation and recognition, Word segmentation, Information retrieval and extraction and many other common application areas[ref about NLP2].
Our NLP Module is built like in Figure (5).

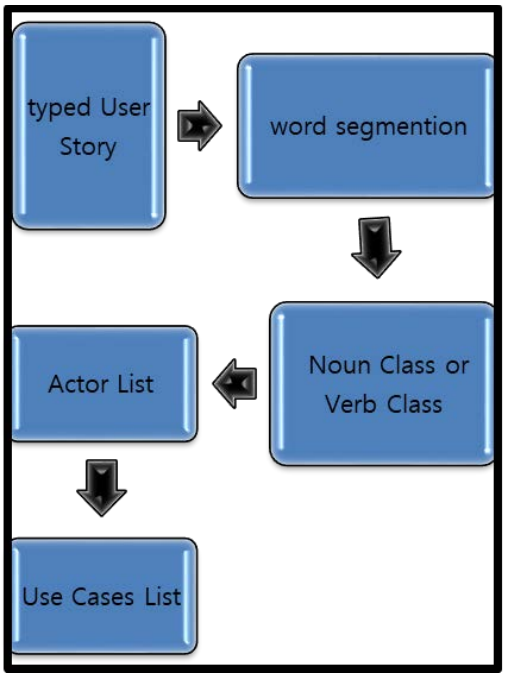

Figure [5]: NLP Module

First of all the typed User Story is segmented into separated word. Each word is processed through a dictionary of words that categorizes segmented word. One of the guides of building the effective User Stories is to "write a simple verbal sentences; Therefore our categories of word are reduced into two classes, the Noun Class and the verb Class. If the word in a sentence from the user story is classified as a noun then this noun is added to an "Actor List" otherwise if the word is classified as a verb then it should be added to a verb list named as "Use Cases". Therefore the main goal of the NLP Module is done by the extraction of nouns as actors and the extraction of verbs as use cases. If we now go back to our framework then we should have a list of possible actor and a list possible use cases. These two lists are sent into a further processing step, which should remove any duplication in the two lists by removing duplicated actors from the actor list and removing the duplicated use cases from the use case list with final redundant Actors list and Use Case Lists .

The User Story undergoes anther processing stages these time in conjunction with each found "actor-use case-sentence" to bind a use case with an actor to get a first draft of a "Use Case Diagram". Furthermore we will get from a simple semantic of the sentence a simple use case description.

4. Case study and Results

A university software system is used to help universities to automate different aspects on a university campus. Among these things is the course registration software system. It helps universities intensively in the registration process. Due to the specified field and scope of the solution, all developed user stories cover the aspect of the student course registration system. About 30 user stories where developed to describe different requirements of this system. The 30 user stories were putted into the automated framework to be converted into user stories. The following table (Table 1) shows the results of the automatic conversion.

Table 1: Results of the automatic conversion

Number of user stories	Number of Actors actually in the user stories	Number of use cases actually in the user stories	Number retrieved Actors through the frame work	Number retrieved Use Cases through the frame work
30	5	33	5	27

From the above table we find the following:
We have a total number of 30 user stories written after "user story writing criteria"; all of these user stories are related to student registration system. The actual number of Actors in these user stories are five. We have 33 different user stories within these 30 user stories.
The total number of Actors retrieved through automated frame work is five, which equals the actual number of Actors in all user stories, where we have a 100% retrieval rate.
The Number of Use Cases retrieved through the frame work are 27, which represents a rate of 82% of the actual use case within the total number of use cases.
The final actor list and use case list undergo anther processing stage namely the "actor-use case- sentence" stage to bind a use case with an actor to get a first draft of a "Use Case Diagram".
Table (2) shows that the Percentage of Actors mapped correctly to their use cases is 75% for the found 27 use cases and the 5 Actors.

Table 2: Percentage of Actors mapped correctly to their use cases

Number of user stories	Percentage of Actors mapped correctly to their use cases
27	75%

5. Conclusion and future work

In this paper we showed the importance of using user stories to get quick requirements that are helpful in building software products in an agile way. These user stories are not known for all software developers in some way; but use cases and use case diagrams are better known; therefore we proposed a frame work using NLP to automatically convert User-Stories into Use Cases in Software Projects. The proposed framework consists of different stages and modules. A case study showed that this framework is very helpful to automatically convert user stories into use cases.
For further researches the framework could be modified to make wider studies on different software systems.

Acknowledgement

The author is grateful to the Applied Science Private University, Amman, Jordan, for the financial support granted to cover the publication fee of this research article.

References

[1] Agile Alliance, "User Stories", URL: https://www.agilealliance.org/glossary/user-stories, retrieved 5/6/2016.

[2] Roman Pichler, "10 Tips for Writing Good User Stories", URL: www.romanpichler.com/blog/10-tips-writing-good-user-stories/ ", retrieved 7/7/2016.

[3] Ivar Jacobson, Object-Oriented Software Engineering: A Use Case Driven Approach, Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, 2004.

[4] Olga Liskin, Kurt Schneider, Fabian Fagerholm, Jürgen Münch ,"Understanding the role of requirements artifacts in Kanban", Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering, Pages 56-63,ACM New York, NY, USA, 2014.

[5] Dan X. Houston, Douglas J. Buettner," Modeling user story completion of an agile software process", Proceedings of the 2013 International Conference on Software and System Process, Pages 88-97, ACM New York, NY, USA, 2013.

[6] Mathias Landhäußer, Adrian Genaid," Connecting user stories and code for test development", Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering, Pages 33-37,IEEE Press Piscataway, NJ, USA,2012.

[7] Proceedings of the COLING-2000 Workshop on Efficiency in Large-Scale Parsing Systems, Pages 54-54, Association for Computational Linguistics Stroudsburg, PA, 2000.

[8] Venera Arnaoudova, Richardson, FLAndrian Marcus, TXGiuliano Antoniol, "The use of text retrieval and natural language processing in software engineering",ICSE '15 Proceedings of the 37th International Conference on Software Engineering - Volume 2,Pages 949-950,IEEE Press Piscataway, NJ, 2015.

[9] Shuang Liu,Jun Sun,Yang Liu,Yue Zhang,Bimlesh Wadhwa," Automatic early defects detection in use case documents ", Proceeding ASE '14 Proceedings of the 29th ACM/IEEE international conference on Automated software engineering .

[10] Leonard W. D'Avolio, Thien Nguyen, Louis Fiore," the automated retrieval console (ARC): open source software for streamlining the process of natural language processing ", Proceedings of the 1st ACM International Health Informatics Symposium, Pages 469-473, ACM New York, NY, 2010.

[11] Davide Falessi, Giovanni Cantone,Gerardo Canfora,"A comprehensive characterization of NLP techniques for identifying equivalent requirements", Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Article No. 18 ,ACM New York, NY,2010.

[12] Deva Kumar Deeptimahanti, Ratna Sanyal,"Semi-automatic generation of UML models from natural language requirements", Proceedings of the 4th India Software Engineering Conference, Pages 165-174,ACM New York, NY,2011..

[13] Ian G. Harris,"Extracting design information from natural language specifications", Proceedings of the 49th Annual Design Automation Conference, Pages 1256-1257, ACM New York, NY, 2012.

[14] Neil Maiden and Suzanne Robertson, "Developing use cases and scenarios in the requirements process", Proceeding ICSE '05 Proceedings of the 27th international conference on Software engineering, Pages 561-570, ACM New York, NY, 2005.

[15] Parastoo Mohagheghi,Bente Anda,Reidar Conradi,"Effort estimation of use cases for incremental large-scale software development", ProceedingICSE '05 Proceedings of the 27th international conference on Software engineering,Pages 303-311,ACM New York, NY, 2005.

[16] Chunhui Wang,Fabrizio Pastore,Arda Goknil,Lionel Briand,Zohaib Iqbal,"Automatic generation of system test cases from use case specifications", Proceeding ISSTA 2015 Proceedings of the 2015 International Symposium on Software Testing and Analysis, Pages 385-396, ACM New York, NY, 2015.

[17] Shane Hastie & Angela," User Stories and Use Cases - Don't Use Both!" Wick [http://www.batimes.com/articles/user-stories-and-use-cases-dont-use-both.html], retrieved 6/62016.