Designing a Pipeline to extract User Stories from Requirements Engineering Meeting Transcripts

Hans van Tuin
6872727

Utrecht University
Business Informatics, Faculty of Science
Department of Information and Computing Sciences

2022

First supervisor: Prof. dr. Sjaak Brinkkemper*
Second supervisor: Dr. Fabiano Dalpiaz*
External supervisor: Dr. ir. Marcela Ruiz†
External supervisor: MSc. Tjerk Spijkman*‡

* Utrecht University
† Zurich University of Applied Sciences
‡ Fizor

Abstract

[**Context & Motivation**] The quality of Requirements Engineering meetings and the resulting requirements correlate with the quality of a software product. These requirements can be written in the standardized form of a User Story. [**Question/Problem**] It takes time to manually extract these requirements from the meeting transcripts and write these down as User Stories. The thesis goal is to explore how to automatically extract User Stories from software Requirements Engineering (RE) meeting transcripts. [**Principal ideas/results**] A Transcript to User Story Pipeline (TUSP) is designed. This pipeline combines already existing Requirement Specification Algorithms (RSAs). The TUSP also uses a novel RSA, specifically designed for this thesis, called the Fit-Gap Searcher. The TUSP consists of 2 configurations: a Machine Learning Configuration and a Lexical Configuration. These two configurations are validated qualitatively and quantitatively on two real-world test cases: a Greenfield test case and a Customization test case. Both configurations predict a couple of good quality User Story fragments, but no whole User Story of good quality is found. The configurations are assessed on the precision, accuracy, recall and F1 metrics, all of which score no higher than 0.1 for both of the configurations. [**Contribution**] This thesis presents a novel way of automatically extracting User Stories from RE meeting transcripts. The future research section provides suggestions on improving the RSAs and a future view for combining the TUSP with issue-tracking and project management systems.

Contents

1 Introduction

1.1 Problem statement

Every product has certain requirements that are to be met. This goes for software products as well. As Dick, Hull, and Jackson (2017) describe, can requirements be seen as the basis for a project. Requirements Engineering (RE) is a field of interest where one is concerned with *what* needs to be designed, in contrast to *how* it is designed (Macaulay, 2012). Thus, RE for software products means working out what a software product needs to be capable of, and in what way. These RE sessions are not to be underestimated, as Mund, Fernandez, Femmer, and Eckhardt (2015) show that the quality of RE sessions correlates with the quality of the resulting software product. However, as Ruiz and Hasselman (2020) state, these RE sessions take time, and this time could be reduced through automatization.

This thesis is based on the project (forward: 'the project') proposed in the paper by Ruiz and Hasselman (2020). The project focuses on reducing "[...]*the time-to-market of software products by automating the task of requirements specification while requirements are discussed*" (Ruiz & Hasselman, 2020, p.328). With this goal in mind Ruiz and Hasselman (2020) propose a framework for automated requirements specification, shown in figure 1. This framework should result in a Requirements Specification Algorithm (RSA). A framework consisting of multiple sections, namely the: 'requirements engineering room', 'automatic specification of user stories' and 'software prototype generation', is shown in figure 1. These sections consist of components, such as an 'automatic transcription tool' and a 'code compiler'.

The project envisions online/offline or combined RE meetings where stakeholders discuss the requirements. The definition of stakeholders is best described by Freeman (2010, p. 46): "A stakeholder in an organization is (by definition) any group or individual who can affect or is affected by the achievement of the organization objectives". As this is the definition preferred in academic circles (Fontaine, Haarman, & Schmid, 2006), this is the definition used in this thesis project. During the RE discussion, everything that is said will be transcribed. From this transcription, the requirements are then automatically extracted in the form of User Stories and transformed into the desired prototypes. User Stories are a standard way of formulating requirements and more on User Stories can be found in section 3.3. The goal of the project is to reduce manual tasks, thus shifting the focus of the software analysts to tasks that are harder to automate, such as analysis of User Stories and User Story prioritization. The project should also incorporate user reviews to make new User Stories (Panichella & Ruiz, 2020). One of the challenges is that both transcribed RE meetings and written user reviews consist of so-called 'unstructured text' (Panichella & Ruiz, 2020). Unstructured text is so-called free form and meaning arises from its context (Miner, Elder IV, Fast, Hill, Nisbet, & Delen, 2012). This unstructured

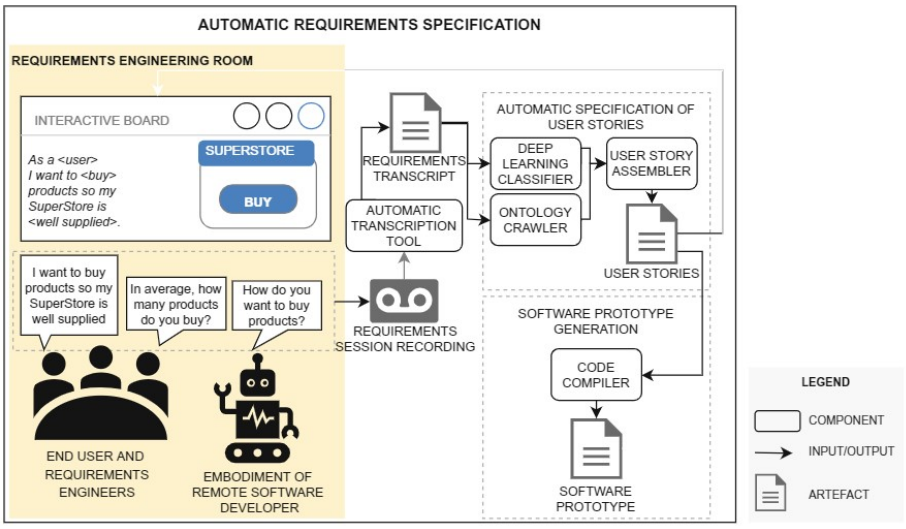


Figure 1: Automated requirements specification (Ruiz & Hasselman, 2020, p.329).

text then has to be transformed into User Stories. This means that the project has to be able to derive meaning from the transcripts, and 'summarize' it in a way. As it is estimated that about 80% of all text written is unstructured (Miner et al., 2012), being able to automatically summarize or extract meaning from an unstructured text can potentially be very valuable.

As not all User Stories are found in the first meeting, and some User Stories develop over multiple meetings and thus time, traceability is of big importance. Although traceability is sometimes seen as burdensome and of little value (Cleland-Huang, 2012) it is critical when working on large projects (Lee, Guadagno, & Jia, 2003). Thus, developing a method of incorporating traceability in a project can also be very valuable.

Based on the project, Keller et al. (2020) wrote a BSc thesis. This thesis tried to automate the specification of User Stories from RE meeting transcriptions. The BSc thesis resulted in two AI scripts, a coordinator that links these two scripts together while also managing input, output and loading of pre-trained models, and a transcript sentence producer, which was used for creating data to train the models on. Due to the lack of real-world data and the time frame which fitted a BSc thesis, the authors were not able to automatically produce meaningful User Stories from real RE meeting transcriptions. This thesis project is a continuation of the work done by Keller et al. (2020). A more thorough summary is given in section 3.8.

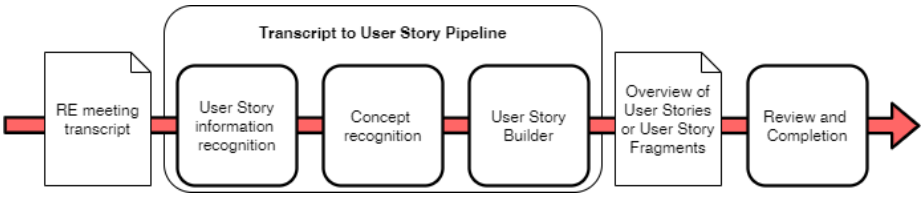


Figure 2: High-level view of the hypothesized TUSP.

The hypothesised result of this thesis project will be a prototype of a Transcript to User Story Pipeline (TUSP). The pipeline will take the transcript from an RE meeting or interview as input and output a list of User Stories or fractions of User Stories. The definition of a pipeline in software architecture has been around since the start of the 20th century (Booch, 2018). Booch (2018) describes the first (organic) computers as a lot of humans sitting in a room doing computations. The results of one would be used as input for the next. In a way, this can be seen as a pipeline, where a large computation is divided into sub computations which are completed by autonomous units (pipeline segments) (Ramamoorthy & Li, 1977). Pipe-lining in software architecture is a way of introducing parallelism. By dividing one process into multiple pipeline segments these segments can be executed independently from each other. The TUSP will consist of multiple segments and is thus called a pipeline. How a hypothesized TUSP can look in a high-level view is depicted in figure 2. It shows how the TUSP takes a transcript from an RE meeting as input and uses multiple pipeline segments to output an overview of User Stories. The hypothesis is that the pipeline needs to recognize which parts of the transcript contain information on User Stories. The pipeline also needs to be able to recognize certain concepts which can help build the User Stories. All the available information is then used by the User Story builder to build User Stories or fragments of User Stories. It is hypothesised that the pipeline will not achieve a 100% precision and recall and thus the overview User Stories and User Story fragments will have to be reviewed for completeness and be complemented manually.

1.2 Research goal

The main research goal of the thesis presented is to automatically specify User Stories from software RE meeting transcripts. This main research goal is divided into three aims. First of all, there is an aim to expose any gaps in the literature on how to automatically extract User Stories from software RE meeting transcripts. This aim also focuses on gaining knowledge on traceability. Secondly, an aim is to review, extend and improve on existing RSAs and to produce a new pipeline with incorporated traceability. The last aim is to test this pipeline in two case studies. The research goal and aims are set to add to the current understanding, technical and scientific knowledge of how to automatically extract User Stories from software RE meeting transcripts. The scientific

relevance comes from the improvement of currently available RSAs and providing a new way of possibly extracting User Stories from RE meeting transcripts, which should provide a foundation for future research.

1.3 Thesis structure

The first part of this thesis project is structured as follows. Section 2 will first consider the main research question and corresponding research questions in section 2.2. Then the research method will be explained in section 2.1. Section 2.1.3 explains the used test cases. Section 3 is about the problem investigation, where the literary framework is explained. The pipeline is designed in section 4 and is subsequently validated in section 5. The thesis is finished with a discussion, future recommendations, future vision and a conclusion in section 6.

2 Research approach

The following section describes Wieringa's design cycle (section 2.1) and how this cycle is used to answer the research questions provided in section 2.2. The chapter is concluded in section 2.3 with a Process Deliverable Diagram.

2.1 Research method: Wieringa's design cycle

To structure the research, Wieringa's design cycle, pictured in figure 3, will be used (Wieringa, 2014). The design cycle consists of three phases: problem investigation, treatment design and treatment validation. These phases are discussed in sections 2.1.1, 2.1.2 and 2.1.3 respectively. The phases can be seen as circular and are iterated over during research.

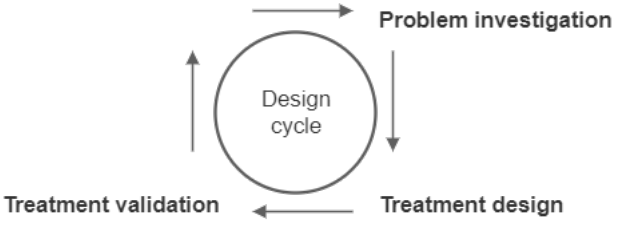


Figure 3: Wieringa's design cycle. (Wieringa, 2014).

Knowledge questions are of importance in understanding how the problem can be solved with the correct treatment. These knowledge questions are divided into 'descriptive' and 'explanatory' questions. Descriptive knowledge questions purely describe the events that happened, explanatory knowledge questions ask the additional question of why something happened. Explanatory knowledge questions can be further divided into lower-level question types, dependent on the research goal. Knowledge questions are answered by iterating over the empirical cycle, shown in figure 4.

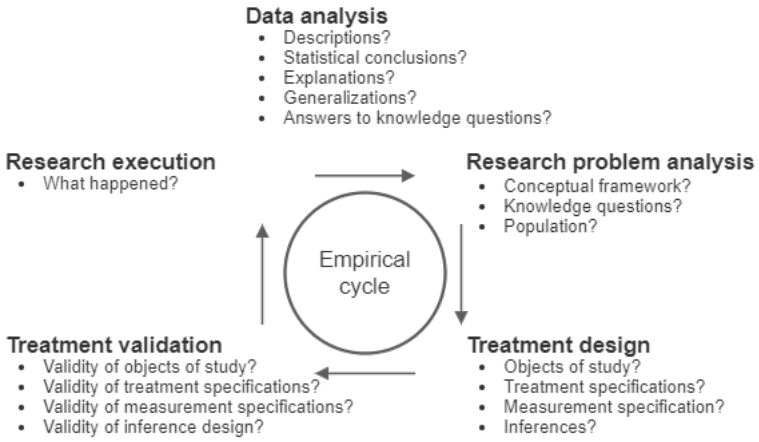


Figure 4: Wieringa's emperical cycle. (Wieringa, 2014, p. 112).

The first phase, the *research problem analysis*, is where the research problem is defined. This is done in chapter 1 and section 2.2. The second phase, *research and inference design* is about the methodology of how to answer the research problem. This is done in section 2.1. During the *validation* phase it is checked if the methodology and the desired outcomes are a match. This is also done in section 2.1. Chapter 3 concerns the *research execution* and *data analysis* phases for knowledge questions. During these phases, the data is collected and analysed.

Next to knowledge questions, there are design problems. Design problems are problems that need solutions to improve the associated artefact. In a way, design problems can be seen as the requirements of an artefact. They depend on the stakeholder goals in a way that the interaction of the artefact context and the artefact itself, should produce effects that help achieve the stakeholder goals.

2.1.1 Problem investigation

In the problem investigation phase, one explores the phenomenon to be improved and why it should be improved. An important part of the problem investigation phase consists of doing literature research, where scientific and grey literature regarding the available knowledge and gaps in the knowledge on RSA and traceability will be investigated. The literature study will not be a systematic one (Cronin, Ryan, & Coughlan, 2008), as this is outside the scope of the thesis. Instead, a traditional literature review (Cronin et al., 2008) will done.

In a traditional literature review, the author collects, summarizes and critiques multiple pieces of literature that concern a specific field of interest (Cronin et al., 2008). This type of literature review fits well with a thesis project that

had a small scope for background literature because as Cronin et al. (2008, p. 38) mention: "Its primary purpose is to provide the reader with a comprehensive background for understanding current knowledge and highlighting the significance of new research. It can inspire research ideas by identifying gaps or inconsistencies in a body of knowledge [...]". The traditional literature review will be combined with the snowballing method (Wohlin, 2014). Even though Wohlin (2014) defines snowballing as a systematic literature review technique, it is not systematic in the way that Cronin et al. (2008) and Badger, Nursten, Williams, and Woodward (2000) define it, where one tries to obtain a complete set of all the topic-specific literature available. With the snowballing method the researcher does not only review a paper itself, but also looks at the reference list of the paper (backwards snowballing), and in what other papers the paper itself is referenced (forward snowballing). The papers that fit the inclusion criteria are then used as a new point for backwards and forward snowballing. This continues until no new papers are found. The key to snowballing is the start set of initial papers and the inclusion criteria for papers found through backwards or forward snowballing.

Traditional literature research will be done on requirements engineering in general, Functional and Non-Functional Requirements, User Stories, User Story traceability and requirements specification algorithms in the context of software requirements meetings. The snowball method will be used for papers with information on automatic requirement or User Story extraction from RE session transcripts.

Next to this, the inclusion criteria for the start set, backward and forward snowballing are as follows:

- The paper should be written in English or Dutch.
- The paper should be available to the author, as not all papers are present in the Utrecht University or ZHAW library.
- The paper should contain a list of references.

The start set for automatic requirement or User Story extraction from RE session transcripts consists of the following papers:

- Keller et al. (2020)
- Panichella and Ruiz (2020)
- Rodeghero, Jiang, Armaly, and McMillan (2017)
- Spijkman, Winter, Bansidhar, and Brinkkemper (2021)

2.1.2 Treatment design

In the treatment design phase, one starts designing one or multiple treatments that should help fulfil the research goals. The treatment investigation phase will have resulted in multiple algorithms already designed to extract User Stories from transcriptions. From these algorithms, one will be chosen and thus will these algorithms have to be compared on quality to make a supported decision.

The algorithm quality will be defined by four quantitative metrics: accuracy, precision, recall and the F1-score (Lipton et al., 2014; Tharwat, 2020). Important for these measurements are the definitions of true and false positives and true and false negatives. If one wants to predict if a turn (section 3.5) contains or does not contain information on a User Story, then the following definitions are of importance:

- **True positives (TP)**: The correctly predicted turns which actually **do** contain User Story information.
- **True negatives (TN)**: The correctly predicted turns which actually **do not** contain User Story information.
- **False positives (FP)**: The turns that are predicted to **do** have User Story information, but they actually **do not**.
- **False negatives (FN)**: The turns that are predicted to **not** have User Story information, but they actually **do**.

Using these definitions the following formulas can be made (Abualhaija et al., 2020, p. 5476), (Lipton et al., 2014, p. 3):

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{2}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{3}$$

$$F1 = \frac{2TP}{(2TP + FP + FN)} \tag{4}$$

Still using the example of turns that do or do not contain User Story information, these formulas show that accuracy is the number of correctly made predictions divided by the total number of prediction candidates (turns). Precision is the number of correctly predicted turns with User Story information divided by all the turns that were predicted to have User Story information. The Recall is the number of correctly predicted turns with User Story information divided by all the actual turns that contain User Story information. The F1 score is the harmonic mean of precision and recall which can be written as F1-score = 2TP/(1/recall + 1/precision). The F1-score function above is a refactored version that can be used without calculating the recall and precision (Lipton et al., 2014).

As Abualhaija et al. (2020) describe, is the cost of misclassification not symmetrical. It is theorized that the cost of removing a false positive (something that is not a requirement but is classified as one) is much lower than missing a false negative (not classifying a requirement as a requirement). Thus a high recall combined with an acceptable precision is sought after (Abualhaija et al.,

2020; Winkler et al., 2019). A high recall means that preferably no false negatives are classified and an acceptable precision is one where the deletion of false positives is considered doable.

For this thesis project, one treatment will be designed, this treatment will be a TUSP. A hypothesised TUSP design will be discussed in the introduction of the treatment design phase chapter. The treatment design phase chapter will also focus on a different part of the to be designed TUSP per section.

2.1.3 Treatment validation

During treatment validation, it is checked if the designed artefact produces the desired effects (Wieringa, 2014). In the thesis presented the effects will be measured qualitatively and quantitatively. The qualitative desired effects are User Story completeness, User Story quality and traceability. Quantitative desired effects are again measured as accuracy, precision, recall and the F1-score, see subsection 2.1.2 for a detailed explanation of these metrics.

The model will be tested on the two test cases, further explained below. The case studies will be set up according to the methodology found in the book by Wohlin, Runeson, Höst, Ohlsson, Regnell, and Wesslén (2012). A more detailed description of the case study methodology is described in chapter 5.

Greenfield Test Case - Extracting User Stories for a greenfield software development project. The first test case will be a case study on the Zurich University of Applied Sciences (ZHAW) Formula Student project (Formula Student ZHAW, n.d.). Formula Student (FS) is an international competition held annually at the Silverstone Circuit in the United Kingdom. Worldwide university teams build racing cars that compete in three classes: the regular FS, concept class and FS-Artificial Intelligence (AI) (Institution of Mechanical Engineers, n.d.). The goal of Formula Student ZHAW is to design a driverless vehicle for the FS-AI class. The team consists of multiple groups of students, ranging from management to mechanics, electrics, students working on the motor and the driverless group. To test the software of the vehicle, a part of the team focuses on developing a simulation tool. The development of the simulation software started in 2019 and because of its relatively short development time, it can be seen as a greenfield software development project. This means that there were no requirements before the development started and the requirements were defined along the way. For this test case, the author will meet with different stakeholders of the simulation tool. These meetings will be transcribed and used as input for the TUSP prototype to check the performance.

Customization Test Case - Extracting User Stories for customization of an existing software product. The second test case is a case study on interviews held by master students following the Requirements Engineering course given at Utrecht University. These interviews were conducted as part of an assignment where the students were randomly given one out of three system

descriptions. This description gives an overview of the organization that is the 'customer', the system as-is and a vision for the product to-be. The students interview a stakeholder from the company, a role which is performed by the course organiser or assistant. The interviews will be transcribed by the thesis project author and will serve as input.

2.2 Research questions

The main research question (MQ) is as follows:

> *MQ: How can User Stories automatically be extracted from software Requirements Engineering meeting transcripts?*

To structure the research, and based on the research goal given in section 1.2 and the MQ, three main research questions (RQs) have been formulated, which correspond with the three phases of Wieringa's design cycle, as explained in section 2.1. It is hypothesised that the MQ can be answered by combining multiple RSAs into a TUSP. Answering the RQs will help with designing and testing the TUSP. The sub research questions (SQs) are either design problems or knowledge questions, which are explained in section 2.1 as well. The main research question is divided into the following research questions:

> *RQ1: What are the problems for developing a Requirements Specification Algorithm with incorporated traceability in the context of software requirements meetings?*
>
> *SQ1.1: What knowledge is available in the context of requirements engineering and software requirements meetings?*
>
> *SQ1.2: What knowledge is available on developing a Transcript to User Story Pipeline in the context of software requirements meetings, and what knowledge gaps can be found?*
>
> *SQ1.3: What knowledge is available on User Story traceability, and what knowledge gaps can be found?*
>
> *SQ1.4: What Requirements Specification Algorithms in the context of software requirements meetings are currently available?*

RQ1 is a knowledge question and will be answered in the problem investigation phase, through literature research. This literature research reveals what knowledge is already available and what gaps exist in current literature, as well as showing what algorithms can be used as a basis for answering RQ2.

Answering SQ1.1 will include gathering literature from which knowledge on general requirements engineering and requirements engineering meetings will be extracted, to explain what the context is that the RSA will be designed for and used in. The expected output of SQ1.1 will be an overview of knowledge on general RE and RE meetings.

To answer SQ1.2, literature concerning RSAs will be gathered. To broaden the search, both publications about automatically extracting requirements and User Stories will be included. The expected output of SQ1.2 will be an overview of knowledge concerning RSAs, such as different methods, use cases and solutions.

SQ1.3 will be answered by collecting literature on User Story traceability. The expected output will be an overview of available knowledge concerning User Story traceability.

SQ1.4 will be answered by collecting literature that contains readily available RSAs. The expected output of this sub-question will be an overview of the available RSAs and a discussion on which RSAs will be used in the treatment design phase.

A summarization of the above can be found in table 1.

Table 1: Overview of expected inputs and outputs for RQ1 and its sub questions.

	Exp. input	**Exp. output**
RQ1	• Literature on RE, traceability and RSAs	• Overview of the problems and available solutions for developing a TUSP
SQ1.1	• Literature on general RE and RE meetings	• Overview of knowledge on general RE and RE meetings
SQ1.2	• Literature on RSAs	• Overview of knowledge on RSAs
SQ1.3	• Literature on User Story traceability	• Overview of knowledge on User Story traceability
SQ1.4	• Literature on readily available RSAs	• Overview of readily available RSAs • Discussion of RSAs chosen for experimentation

> *RQ2: How can a Transcript to User Story Pipeline that extracts traceable User Stories from transcripts be designed?*
>
> *SQ2.1: How do the Requirements Specification Algorithms perform and how can they be extended?*
>
> *SQ2.2: How can the gained knowledge and available Requirements Specification Algorithms be combined into a Transcript to User Story Pipeline?*

RQ2 is partly a knowledge question and partly a design problem. Thus RQ2 is divided into one knowledge-focused sub research question, SQ2.1, and one design-focused sub research question, SQ2.2. RQ2 and its sub-questions will be answered in the treatment design phase.

The output from SQ1.4 will be used to answer SQ2.1. The chosen RSAs will be tested and bench-marked. The expected output of SQ2.1 will be an overview of the metrics used to bench-mark the algorithms along with suggestions on how the RSAs can be extended.

SQ2.2 will use the algorithm with the best performance found in SQ2.1 and combine this with the gathered knowledge from RQ1 to output a new TUSP.

A summarization of the above is shown in table 2.

Table 2: Overview of expected inputs and outputs for RQ2 and its sub questions.

	Exp. input	**Exp. output**
RQ2	• Gathered knowledge on RE, traceability and RSAs • Readily available RSAs	• TUSP
SQ2.1	• Readily available RSAs	• Bench-marked RSAs • Suggestions for extensions on RSAs
SQ2.2	• Gathered knowledge on RE, traceability and RSAs • The best performing RSA	• TUSP

RQ3: What is the performance of the developed Transcript to User Story Pipeline in terms of completeness, quality and traceability of the produced User Stories?

SQ3.1: How does the developed Transcript to User Story Pipeline perform in the Greenfield Test Case?

SQ3.2: How does the developed Transcript to User Story Pipeline perform in the Customization Test Case?

RQ3 is about testing the developed TUSP on two test cases in the treatment validation phase. These test cases are explained in section 2.1.3. Both test cases are from real-world data and are incorporated into this thesis to check the validity of the designed TUSP.

The expected input for SQ3.1 and SQ3.2 will be the TUSP developed in RQ2, and the data of the Greenfield Test Case and the Customization Test Case respectively. Both SQs will output an overview of the performance of the TUSP in the respective data.

A summarization of the above is found in table 3.

Table 3: Overview of expected inputs and outputs for RQ3 and its sub questions.

	Exp. input	**Exp. output**
RQ3	• TUSP	• Overview of performance on completeness, quality and traceability of the User Stories
SQ3.1	• TUSP • Greenfield Test Case data	• Overview of the performance on the Greenfield Test Case
SQ3.2	• TUSP • Customization Test Case data	• Overview of the performance on the Customization Test Case

2.3 Process Deliverable Diagram

To give a structured overview of the thesis project a Process Deliverable Diagram (PDD) has been made. The PDD meta-modelling technique is developed by van de Weerd and Brinkkemper (2009) to support situational method engineering. The PDD in figure 5 is a high-level overview of the activities to be completed in the problem investigation, treatment design and treatment validation phases, and which concepts are to be delivered for each activity. The associated Activities table and Concept table can be found in Appendix A. The PDD has three main activities that all contain sub-activities: the problem investigation, the treatment design and the treatment validation. The process starts with the problem investigation which has 4 sub-activities. The "Conduct literature research on readily available RSAs" is a closed complex activity because it contains multiple activities that are too detailed for this high-level view. The same goes for all the sub-activities of the treatment design and the treatment validation activities. The most important deliverables are the "TUSP" and the "TUSP WITH TRACEABILITY". These are also complex closed deliverables because the details are out of scope for this high-level view. Important is that the three main activities follow a pre-defined order meaning that the treatment validation follows the treatment design, which in turn follows the problem investigation. However, the sub-activities of the problem investigation and the treatment validation do not have to follow a pre-defined order. These sub-activities can be done simultaneously or in another order if the opportunity arises. The sub-activities of the treatment design do have to follow the order as shown in figure 5.

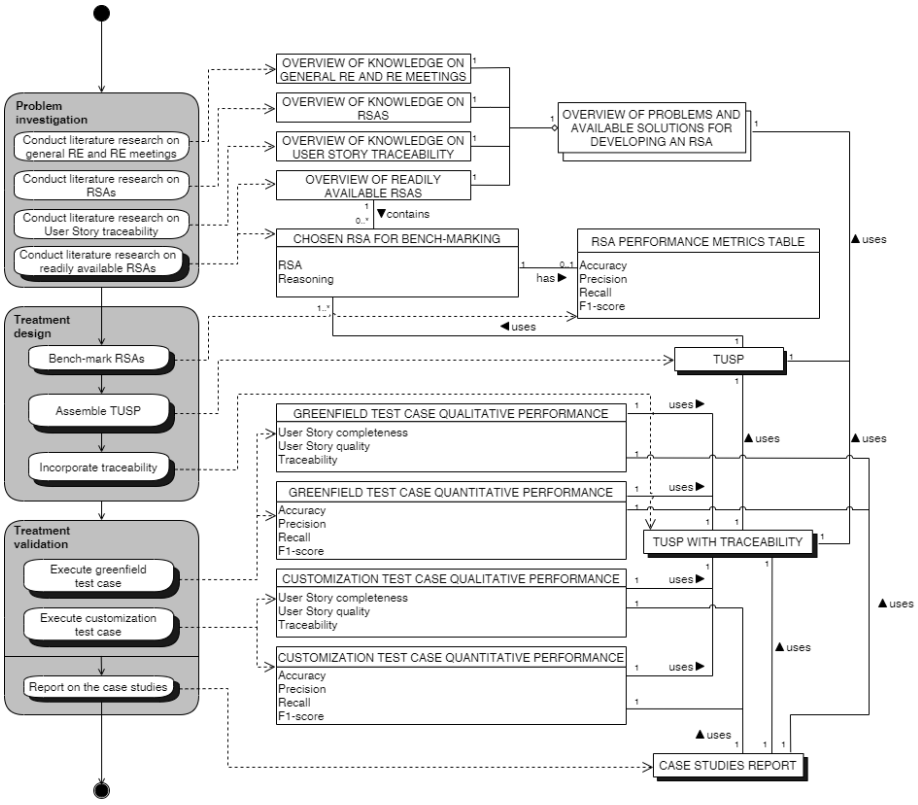


Figure 5: High level Process Deliverable Diagram of the problem investigation, treatment design and treatment validation phases.

3 Theoretical framework

The first part of the problem investigation phase is focused on requirements engineering. This starts with a general introduction to the field of requirements engineering (section 3.1). The requirements are an important result of requirements engineering and can be divided into functional or non-functional requirements. This division is discussed in section 3.2. These requirements can be written down in a standardized form, called User Stories, which is explained in section 3.3. Section 3.4 gives more insight into the context in which requirements engineering is practised and in which the TUSP will be used.

The second part of the problem investigation is focused on the technical side of the to be designed TUSP and will explain important elements. It starts with section 3.5 which explains the 'turn', a unit of speech used by some RSAs to differentiate between multiple actors. Section 3.6 then explains the basics of the field of traceability and how it will be applied in the TUSP. Sections 3.8 and 3.9 then discuss the available knowledge on automatic requirement or User Story extraction and the currently available RSAs respectively. The problem investigation phase is concluded in section 3.10. An overview of the sections can be seen in figure 6. The deliverables as discussed in section 2.3 are also shown in this figure.

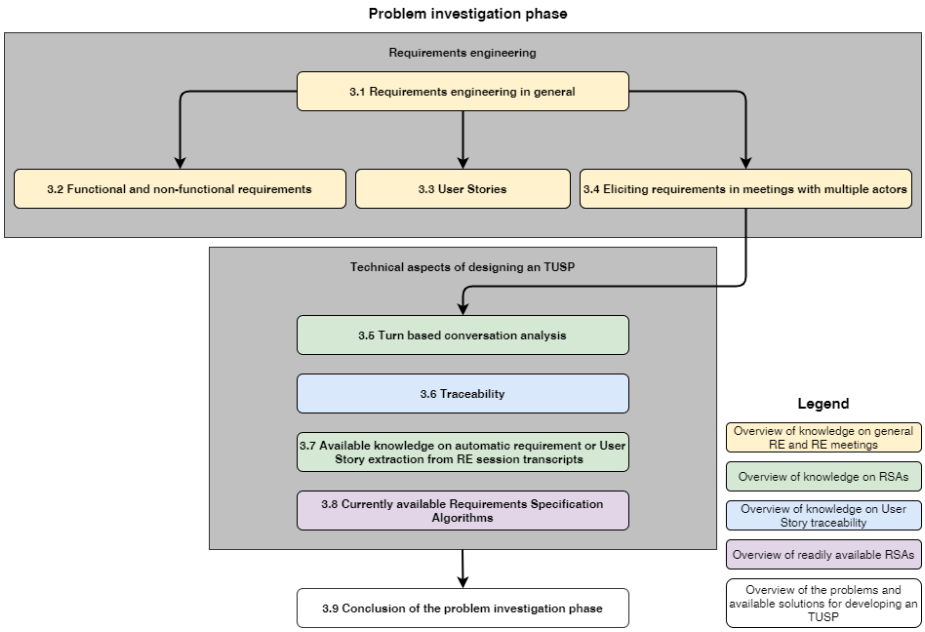


Figure 6: Overview of the problem investigation phase.

3.1 Requirements engineering in general

RE started in the '90s when two groups of software engineering recognized the growing importance of RE Mead, 2013. Both groups organized the first congresses on the subject, namely the International Symposium on RE, 4-6 January 1993 in San Diego, the U.S.A., and the First International Conference on RE, 18-22 April 1994 in Colorado, U.S.A. and Taipei, Taiwan (Mead, 2013; Pohl, 1994). Although it is hard to pinpoint a standard definition in RE literature (Zowghi & Coulin, 2005), this thesis will use the following definition by Van Lamsweerde (2009):

> **Requirements Engineering:** *"We need to discover, understand, formulate, analyse and agree on what problem should be solved, why such a problem needs to be solved and who should be involved in the responsibility of solving that problem. Broadly, this is what requirements engineering is all about." (Van Lamsweerde, 2009, p. 3)*

Dick et al. (2017) explain that requirements are often defined in natural language, to ensure it is well understood by everybody involved in the project. However, using natural language also poses the following problem: "to capture the need or problem completely and unambiguously without resorting to specialist jargon or conventions." (Dick et al., 2017, p. 2). The requirements engineer needs to formulate the requirements in a way that is understandable for all the stakeholders and solve the conflicts between requirements that arise. Doing this correctly and getting the right requirements, as a result, is considered to be very difficult (Zowghi & Coulin, 2005).

RE has evolved as an "interdisciplinary research area" (Pohl, 1994, p. 243). Dick et al. (2017) emphasize the interdisciplinary aspect with an example of a train. A requirement of a certain train connection between Amsterdam and Utrecht could be that the journey does not take longer than 15 minutes. Just a few examples of the many components that would require cooperation are:

- The trains (mechanical/machine).
- The train operators (human).
- Software that operates the railroad switches (software).

This shortlist shows that many different areas need to be considered for just one high-level requirement to be full-filled. When the total of all these components work together in an organized way, it is called a *'system'*. The system can then output, achieve or support certain results, which can be seen as the requirements (Dick et al., 2017). Two important versions of these systems are the *'system as-is* and the *'system to-be'*. A system-as-is can be seen as the current system that needs to be improved. The system-to-be is the resulting system after changes have been made according to the elicited requirements (Van Lamsweerde, 2009).

Among the many tasks that a requirements engineer is expected to do, such as exploring, facilitating, mediating, developing and validating, an impor-

tant part is to correctly document the RE session and resulting requirements. (Zowghi & Coulin, 2005).

3.2 Functional and non-functional requirements

Traditionally, requirements are split into two categories: functional requirements (FR) and non-functional requirements (NFR) (Glinz, 2007). FRs are easier to define. These describe the functions a system should have or the behaviour that a system should perform (Glinz, 2007). Chung and do Prado Leite (2009) describe how most of the time focus is laid upon the FRs, as the actual functions of a system are deemed most important. NFRs are harder to define, and multiple definitions are given in the literature. Glinz (2007) provides an overview of different NFR definitions from before 2007. A partly adaptation of this overview, with additions of the author can be seen in table 4.

Table 4: Different definitions of NFR's, in alphabetical order. *Adapted from Glinz (2007, p. 2).*

Reference	**Definition**
(Anton, 1997)	Describe the nonbehavioral aspects of a system, capturing the properties and constraints under which a system must operate.
(Chung & do Prado Leite, 2009)	NFRs refer to all words ending in "-ilities" (e.g., transferability) or "-ities" (e.g., security). However, words as "performance, user-friendliness and coherence" (Chung & do Prado Leite, 2009, p. 364) are also NFRs.
(Davis, 1993)	The required overall attributes of the system, including portability, reliability, efficiency, human engineering, testability, understandability, and modifiability.
(IEEE Standards Coordinating Committee, 1990)	Term is not defined. The standard distinguishes design requirements, implementation requirements, interface requirements, performance requirements, and physical requirements.

continues on next page

(Software Engineering Standards Committee and IEEE-SA Standards Board, 1998)	Term is not defined. The standard defines the categories functionality, external interfaces, performance, attributes (portability, security, etc.), and design constraints. Project requirements (such as schedule, cost, or development requirements) are explicitly excluded.
(Jacobson et al., 1999)	A requirement that specifies system properties, such as environmental and implementation constraints, performance, platform dependencies, maintainability, extensibility, and reliability. A requirement that specifies physical constraints on a functional requirement.
(Kotonya & Sommerville, 1998)	Requirements which are not specifically concerned with the functionality of a system. They place restrictions on the product being developed and the development process, and they specify external constraints that the product must meet.
(Mylopoulos et al., 1992)	"... global requirements on its development or operational cost, performance, reliability, maintainability, portability, robustness, and the like. (...) There is not a formal definition or a complete list of nonfunctional requirements."
(Ncube, 2000)	The behavioural properties that the specified functions must have, such as performance, usability.
(Paech & Kerkow, 2004)	NFRs focus on 'how good' software does something, FRs focus on 'what' the software does.
(Robertson & Robertson, 1999)	A property, or quality, that the product must have, such as an appearance, or a speed or accuracy property.

continues on next page

(Wiegers & Beatty, 2013)	A description of a property or characteristic that a software system must exhibit or a constraint that it must respect, other than an observable system behaviour.

Multiple interpretations of what an NFR is are shown in table 4. Globally, the definitions clarify that an NFR is everything that an FR is not. This means that NFR's concern qualities or properties that are not direct results of the system behaviour. Most of the definitions contain the words 'property' or 'constraint'. According to the Cambridge dictionary, a property is "a quality in a substance or material, especially one that means that it can be used in a particular way" (Cambridge Dictionary, 2019b). A constraint is defined as "something that controls what you do by keeping you within particular limits" (Cambridge Dictionary, 2019a) which can also be applied to a system or software product.

Next to the classification of functional or non-functional, requirements can also be classified as hard or soft (Glinz, 2007). One classifies a requirement as a hard requirement when it has a clear distinction of being satisfied or not. A hard requirement can be 0 or 1 (Glinz, 2005). A requirement is soft when it has no definitive distinction for satisfaction. Thus, it can be 0, 1 or anything in between (Li et al., 2014). "Hence, it makes sense for soft requirements to have both a planned degree of satisfaction and a minimum acceptable degree of satisfaction" (Glinz, 2005, pp. II–59).

As Inayat et al. (2015) point out in their systematic literature review on the practices and challenges of agile RE, is dealing with NFRs still a major challenge in the software industry. However, neglecting NFRs can pose a major threat to the quality of the end product, resulting in rework and a lapse in progress.

This thesis project labels a requirement as an NFR if it concerns: a constraint, a property, an "-ility", performance, user-friendliness or coherence. Grey zone cases where a requirement is not explicitly functional or fits the just described definition will be judged per case based on the definitions in table 4.

3.3 User Stories

User Stories are a relatively new method that is often used when working with Agile methods (Lucassen, Dalpiaz, van der Werf, & Brinkkemper, 2016; Madanayake, Dias, & Kodikara, 2017). It is a method of standardizing the formulation of requirements.

The basic User Story form exists as a sentence (Lucassen et al., 2016):

> **An User Story:** *As a **[role]**, I want **[goal]**, so that **[benefit]**. (Lucassen et al., 2016)*

This is further explained by (Dalpiaz & Brinkkemper, 2018): the part 'as a [role]' is about **who** wants the functionality/requirement, this part is called [**the role**] (Wautelet, Heng, Kolp, & Mirbel, 2014). The part 'I want [goal]' is about **what** functionality/requirement the stakeholder wants from the system and this part is called [**the goal**] (Wautelet et al., 2014). Finally, the *optional* part 'so that [benefit]' explains **why** the stakeholder wants/needs the functionality/requirement, so this part is called [**the benefit**] (Wautelet et al., 2014). Please note that other authors use different terms for goal and benefit in this context, such as 'function' and 'rationale' respectively (Rodeghero et al., 2017; Ruiz & Hasselman, 2020).

An example of a User Story could be as follows: 'as a Java developer, I want my IDE to give an error when a semicolon is missing at the end of a line so that I can prevent syntax errors.'

Lucassen, Dalpiaz, Van Der Werf, and Brinkkemper (2015) developed a Quality User Story Framework (QUSF), as seen in figure 7. The QUSF shows that a good User Story is dependent on a range of characteristics. The 'syntactic' characteristics are concerned with how the User Stories are written, the 'semantic' characteristics concern the meaning of parts of and complete User Stories and the 'pragmatic' characteristics concern practical form and uses of the User Story. These characteristics are needed for a User Story to be valuable. The terms given in figure 7 are explained in table 5.

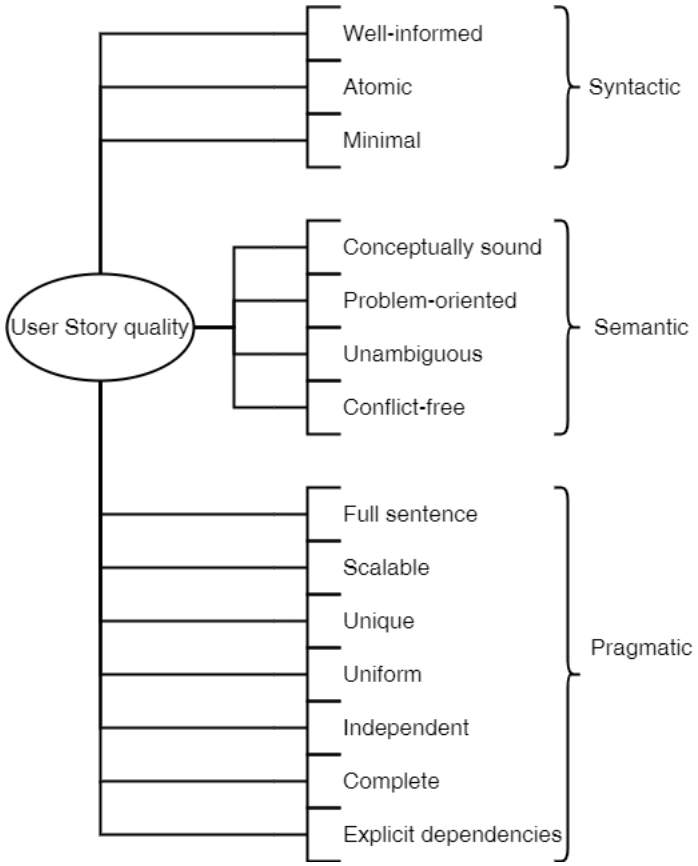


Figure 7: Quality User Story Framework. *Lucassen et al. (2015, p. 2).*

Table 5: Quality User Story Framework descriptions. *Lucassen et al. (2015, p. 2).*

Criteria	**Description**
Well-formed	A User Story includes at least a role and a goal
Atomic	A User Story expresses a requirement for exactly one feature
Minimal	A User Story contains nothing more than role, goal and reason

continues on next page

Conceptually sound	The goal expresses a feature and the reason expresses a rationale, not something else
Problem-oriented	A User Story only specifies the problem, not the solution to it
Unambiguous	A User Story avoids terms or abstractions that may lead to multiple interpretations
Conflict-free	A User Story should not be inconsistent with any other User Story
Full sentence	A User Story is a well-formed full sentence
Scalable	User stories do not denote too coarse-grained requirements that are difficult to plan and prioritize
Unique	Every User Story is unique, duplicates are avoided
Uniform	All user stories follow roughly the same template
Independent	The User Story is self-contained, avoiding inherent dependencies on other user stories
Complete	Implementing a set of user stories creates a feature-complete application, no steps are missing
Explicit Dependencies	Link all unavoidable, non-obvious dependencies on user stories

3.4 Eliciting requirements in meetings with multiple actors

Although there are multiple methods of eliciting requirements, this thesis project will focus on methods that likely contain multiple people in the same online or offline room. An overview of methods can be seen in table 6.

Table 6: Methods for eliciting requirements with multiple stakeholders.

Method/tool	**Description**	**Reference**
RE meeting	Can be formal or informal. Covers a large area such as basic meeting tools. Can be especially useful when a conflict arises among stakeholders or to bring stakeholders together from multiple areas	(Sharma & Pandey, 2013; Zowghi & Coulin, 2005)
Interviews	Can be unstructured, semi-structured and structured. Used to gather domain knowledge, knowledge about the problems, the boundaries of knowledge available and the stakeholders that possess the right knowledge.	(Bano et al., 2018; Dick et al., 2017; Khan et al., 2014; Sharma & Pandey, 2013; Tiwari et al., 2012; Van Lamsweerde, 2009; Zowghi & Coulin, 2005)
Focus groups	4 to 9 stakeholders with diverse backgrounds will discuss their requirements from the system.	(Tiwari et al., 2012)
Brainstorming sessions	A method where a lot of ideas can be generated without going in-depth on an idea in particular.	(Dick et al., 2017; Sharma & Pandey, 2013; Tiwari et al., 2012; Zowghi & Coulin, 2005)
Storyboarding	A method of combining pictures, audio, video, text and animation to provide the stakeholders with an overview of the system's functionalities.	(Tiwari et al., 2012; Zowghi & Coulin, 2005)

continues on next page

Workshops	A method where all the stakeholder groups are present. The stakeholders are taken through the whole process of brainstorming on scenarios to the end product of the complete set of requirements.	(Dick et al., 2017; Tiwari et al., 2012; Zowghi & Coulin, 2005)
JAD/RAD sessions	Joint Application Development (JAD) and Rapid Application Development (RAD) sessions are sessions where the high-level purpose of the system is already known, but the stakeholders will work out the details together. These sessions are more structured than brainstorming.	(Sharma & Pandey, 2013; Tiwari et al., 2012; Zowghi & Coulin, 2005)

This thesis project will focus on RE meetings and interviews because these can be small proceedings where only an interviewer and an interviewee are present. This is opposed to focus groups, workshops and JAD/RAD sessions which inherently need multiple stakeholders. A brainstorming session is not in-depth enough to extract meaningful User Stories and storyboarding is inherently multi-media, which is beyond the scope of this thesis project.

Creating a viable prototype of the system that can handle all the methods mentioned in table 6 is depend on the way the system distinguishes multiple actors. Ruiz and Hasselman (2020) do this by the way of 'turns', which are explained in the following section.

3.5 Turn based conversation analysis

Human conversational language is different from the human written language in, among many other differences, that the written form is neatly ordered by punctuation and full sentences. Conversational language tends to be messy, without punctuation and filled with repetitions, mistakes, half sentences and "uuhhss" and "aaahhs". Recorded dialogue needs to be transcribed, at which point punctuation is added. However, the way of adding punctuation is subjective to the transcriber or tool (Rodeghero et al., 2017). For this reason, do researchers in the field of conversation analysis prefer the analysis of the turn above the analysis of the sentence (Ford et al., 2002; Hutchby & Wooffitt, 2008; Nishida, 2008; Schegloff, 2007; Ten Have, 2007). In daily dialogue each participant has their turn to speak while the other participants are quiet (Rodeghero

et al., 2017). Thus, in this thesis, is a turn defined as follows:

> **A Turn:** *A unit of speech done by a participant in between two other participants.*

Turn-based conversation analysis is useful for detecting User Stories because, as Rodeghero et al. (2017) describe, there is a big possibility that there is not one sentence available that contains all the information on a User Story. Much rather, would a participant try to explain his or her thoughts on the requirements during multiple sentences. Thus the probability is higher that a turn contains all the information needed for a complete User Story.

3.6 Traceability

As not all User Stories are found in the first meeting, and some User Stories develop over multiple meetings and thus time, traceability is of big importance. Although traceability is sometimes seen as burdensome and of little value (Cleland-Huang, 2012) it is critical when working on large projects (Lee et al., 2003) and everyday software development, where requirements can evolve quickly and can become obsolete in no time (Espinoza & Garbajosa, 2011). Thus, developing a method of incorporating traceability in an RSA can also be very valuable. Although there is a large corpus of literature on traceability in general, there is no literature available on traceability between different stages of User Stories and how these trace back to the origin of the User Story. Therefore this section defines basic traceability and applies this to the User Story problem.

Gotel and Finkelstein (1994) give the most referenced definition of requirements traceability as:

> **Requirements Traceability:** *"Requirements traceability refers to the ability to describe and follow the life of a requirement, in both forward and backward direction, ideally through the whole system life cycle (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases)." (Gotel & Finkelstein, 1994, p. 4)*

Gotel et al. (2012) give definitions of the basic terminology used in traceability:

- **Trace artefact :** A traceable unit of data.
- **Source artefact :** The origin of a trace.
- **Target artifact:** The destination of a trace.
- **Trace link:** An association between the source and target artefacts. The trace link can go in both directions (i.e. if A precedes B then B evolves from A).
- **Trace** *(noun)*: A combination of a source artifact, a target artifact and a trace link.

- **Trace** *(verb)*: Actively following a trace link from a source artifact to a target artifact or vice-versa.
- **Trace element:** One of the parts of a trace. Thus a trace element can be the source artefact, the target artefact or the trace link.
- **Trace attribute:** Extra information for the trace or trace elements.
- **Atomic trace:** A combination of one source artefact, one target artefact and one trace link.
- **Chained trace:** A sequence of multiple atomic traces, where the target artefact of the first atomic trace becomes the source artefact of the second atomic trace etcetera.

As Cleland-Huang et al. (2014) describe, is there no "one-size-fits-all" solution to traceability. The traceability envisioned for this project is rather simple as it only considers three types of artefacts. A trace artifact can be a User Story (*User Story artifact*), a piece of transcript (*transcript artifact*) or a piece of documentation (*documentation artifact*). Between User Story artefacts there is one type of trace link, one that shows the evolution of the User Stories. A part of a User Story artefact can also have a trace link to a transcript artefact or a documentation artefact. The goal is to create a data frame filled with chained traces of User Story artefacts. A User Story artefact has the trace attributes of UserStory_state, UserStory_Artifact_ID and User_Story_ID. A transcripts artefact has the trace attributes of transcript_artifact_ID, RE_session_ID and meeting_participant_ID. A documentation artefact has the trace attributes of Document_artifact_ID and Document_ID.

3.7 Fit-Gap analysis

> **Fit-Gap analysis:** *A requirements elicitation technique that, based on matching a customer's needs with the functionality of a software product, identifies needs that are supported by the current functionality as fits, and needs that are not as gaps. (Spijkman, Dalpiaz, et al., 2021, p. 4)*

Spijkman, Dalpiaz, et al. (2021) combined the fit-gap analysis with the grounded theory. This way the authors defined fit-gap categories and provide keywords and phrases to identify these categories. For the study, the authors transcribed 9 interviews, for a total of 12 hours and 79.938 words. This fit-gap analysis through the grounded theory is interesting because the resulting keywords and phrases can be used for categorizing important turn parts in a transcript.

3.8 Available knowledge on automatic requirement or User Story extraction from RE session transcripts

The literature search through the use of the snowballing method resulted in one extra paper found, besides the starting papers. However, this paper is written

by Ruiz and Hasselman (2020) and was already known to the author. The small number of papers on the subject shows that automatically extracting requirements or User Stories from (RE session) transcripts is a novel concept. However, the search did result in a lot of interesting literature indirectly connected to the subject. These papers are discussed in later sections.

Keller et al. (2020) developed a prototype for automatically extracting User Stories from transcripts. The authors used multiple modules of TensorFlow, but most importantly its module ELMO (Embeddings from Language Models), to make predictions based on the word context. As the authors did not have enough real-life data to properly train the AI, a script called 'create transcript' was created to artificially create data. Create transcript produces combinations of prewritten sentence parts and also labels the sentence parts based on the User Story standard format. It can also produce sentences that do not hold any User Story information. Next to 'create transcript' the study resulted in two AI scripts: the 'sentence classifier' script and the 'word classifier' script. The sentence classifier uses binary labels to predict whether a sentence has information regarding a potential User Story. The word classifier predicts whether a sentence part is linked to a User Story role, function or rationale. Create transcript, sentence classifier and word classifier are linked together by the 'User-story Creation Coordinator'. This coordinator uses transcripts as input. It will check for preexisting models and when not available it will train a model.

The authors found that when configuring the ELMO model, more Epochs and a higher LSTM Unit size, lead to higher accuracy. The highest accuracy of 79.17% was achieved with 1024 LSTM Units, running for 12 Epochs.

The authors reported that the results were "[...] meaningless regarding a real-world case right now." (Keller et al., 2020, p. 16). The automatically generated transcripts would not compare to real-world transcripts, and thus the authors recommend to start configuring the AI scripts using real-world data. The authors make the following recommendations for future work:

- Use real-world data to validate the existing performance.
- Modify the create transcript script to use real-world sentence parts.
- Use a different model or approach for the word classifier. Right now the model needs to be re-trained every time, which is costly both time and computing power-wise.
- Train the AI in-dept on a specific field to improve accuracy.
- Make general improvements to the code.
- Add randomization to the create transcripts script.

Ruiz and Hasselman (2020) and **Panichella and Ruiz (2020)** are discussed together since these are part of the same project. As described in section 1 Ruiz and Hasselman (2020) propose a framework consisting of a requirements engineering discussion room, where the audio is recorded. This audio is transcribed and used as an input for a *deep learning classifier* and an *ontology*

crawler. These two components are then used as input for the *User Story assembler*, which should output the actual User Stories. The deep learning (DL) classifier component was trained to classify a turn (section 3.5) as 'None (0)', 'Non-functional (1)', or 'Functional (2)'. As Ruiz and Hasselman (2020) mention, is the downside of this way of classifying that a turn can be 1 and 2 at the same time. When interpreting the results as probabilities, both 1 and 2 can have probabilities of around 0.5, which is hard for machine learning (ML) classifier evaluation techniques to evaluate properly. The ontology crawler component is still in the early stages and needs more development. Panichella and Ruiz (2020) expand upon the previous paper by testing multiple ML and one DL strategies. The best performing was the ML technique SMO, with an F-measure of 77%.

Rodeghero, Jiang, Armaly, and McMillan (2017) wrote a paper consisting of two parts. For the first part, the authors recorded 27 conversations between developers and customers with a total of approximately 24 hours of audio. The transcriptions of this audio were manually annotated for containing information on the User Story role, function or rationale. Rodeghero et al. (2017) also used turn-based conversation analysis (section 3.5 and found that 0.5% of the turns contained information on the role, 5.5% of the turns contained information on the function and 2.9% of the turns contained information on the rationale. An explanation given for the low percentage of the role being discussed is that the role is often already known beforehand, thus there being no need to discuss this during the meeting.

For the second part of the paper, Rodeghero et al. (2017) build multiple classifiers, two classifying turns with function information and two for classifying turns with rationale information. Because of the low percentage of role information available in the transcriptions, the authors decided not to build classifiers for role information. The authors used 25 attributes, divided into four categories: length, structural, participants and lexical. Two of the classifiers were based on Logistic Regression, and two were based on Support Vector Machines (SVM) with an RBF kernel. The Logistic Regression classifiers outperformed the SVM-RBF classifiers for both the classification of function and rationale. A total of seven attributes were the best predictors, which are given and shortly explained in table 7.

Table 7: The seven best performing attributes from Rodeghero et al. (2017, p. 6).

Attribute	**Description**
cent1	cos. of turn and convo., w/ Sprob
cent2	cos. of turn and convo., w/ Tprob

continues on next page

slen	word count in turn, globally normalized
sms	sum of Sprob scores
smt	sum of Tprob scores
spau	time btwn. current and next turn
wc	word count in turn, not normalized

Two of the attributes are categorized as length attributes, one as a structural attribute, four as lexical attributes and zero as a participants attribute.

Spijkman, Winter, Bansidhar, and Brinkkemper (2021) try to design a tool that can extract known and unknown concepts from a RE session transcription. The authors make two hypothesises:

> *"**H1** Unknown concepts indicate the need for customization of the product.*
> ***H2** Known concepts indicate the need for configurations of existing features in the product." (Spijkman, Winter, et al., 2021, p. 2)*

Checking if a concept is known or unknown is done by comparing extracted concepts against a software-product ontology. Spijkman, Winter, et al. (2021) use the following definition of an ontology:

> **An ontology:** *"A set of concepts and relationships in a software product domain, which describe functionalities, artifacts and related software systems." (Spijkman, Winter, et al., 2021, p. 2)*

The authors have written a prototype in Python using the Python packages NLTK, Pandas and TextBlob.

The tool was validated on the transcripts of 11 hours of conversation between an RE analyst and multiple customers. The tool managed to identify 205 concepts, from which 53 were tagged as known and 152 as unknown. Out of the 152, 114 were manually tagged as irrelevant, 17 as additions for the ontology, 5 as an indication for customisation and 16 as customer domain-specific.

The tool was presented to RE experts for feedback. These suggested the following improvements (Spijkman, Winter, et al., 2021, p. 7):

1. Addition of a word counter for each speaker. Combined with the expertise of the speaker this would show the importance of certain requirements.
2. Matching of concepts with the ontology. This provides a larger domain context.
3. Direct linking of concepts to their relevant transcript content.

3.8.1 The definition of a concept

The algorithm of Spijkman, Winter, et al. (2021) heavily relies on the definitions of a concept, however, no definition of a 'concept' is given. This thesis project defines a concept by making use of an NLP technique known as 'text chunking' which is also known as 'shallow parsing' (Arora et al., 2016; Ramshaw & Marcus, 1999; Subhashini & Kumar, 2010). The core of this technique is dividing a sentence into non-overlapping sub-sentences which are given a label, with the most important labels being (Arora et al., 2016; Subhashini & Kumar, 2010):

- **Noun phrases (NPs):** "A noun phrase consists of a noun or pronoun, which is called the head, and any dependent words before or after the head. Dependent words give specific information about the head." (Cambridge Dictionary, n.d.-a)
- **Verb phrases (VPs):** "A verb phrase consists of a main verb alone, or a main verb plus any modal and/or auxiliary verbs." (Cambridge Dictionary, n.d.-c)
- **Prepositional phrases (PPs):** "Prepositional phrases consist of a preposition and the words which follow it (a complement). The complement ... is most commonly a noun phrase or pronoun, but it can also be, an adverb phrase (usually one of place or time), a verb in the -ing form or, less commonly, a prepositional phrase or a wh-clause" (Cambridge Dictionary, n.d.-b)

For example, the sentence 'the boy was running in the park' is split into [NP the boy NP][VP was running VP][PP in PP][NP the park NP].

For this thesis project a concept is defined as follows:

> **A Concept:** *A concept is an NP that indicates a new requirement, customization or configuration of an existing feature.*

Whether a concept indicates either a new requirement or customization or configuration is dependent on:

- Is the concept known (present in product glossary) or unknown (not present in product glossary)?
- VPs and PPs present in the same turn the concept is.

3.9 Currently available Requirements Specification Algorithms

Following are papers that contain useful algorithms for developing a TUSP. Each algorithm is given a label (A1 to A7) to help provide an easy overview at the end of this section in table 8.

Ruiz and Hasselman (2020) build a framework for automatic specification of User Stories. It consists of a Deep Learning Classifier Component (A1)

and an Ontology Crawler Component (A2). The Deep Learning Classifier Component uses turns as the unit of analysis. The words of each turn are embedded as multidimensional vectors, according to the GloVe representation developed by Pennington, Socher, and Manning (2014). Ruiz and Hasselman (2020) used the "Wikipedia 2014 + Gigawords" pre-trained embedding model to develop the classifier. The Ontology Crawler Component tries to extract information about the <role> part of a User Story from the context (the ontology) of a product. Based on this ontology, the component outputs a list of foundational User Stories. Both of the components are available as public GitHub repositories[1][2].

Spijkman, Winter, Bansidhar, and Brinkkemper (2021) build a prototype key concept extraction tool (A3) in Python using NLTK, Pandas and TextBlob. The pseudocode can be seen in algorithm 1. This tool first preprocesses the transcripts by splitting by speaker, effectively splitting on turns. After removing the timestamps, punctuation and stop-words the tool calls the functions for known and unknown concepts. The function for known concepts iterates over the words in the turn and compares the words to the known concepts in the ontology. If known, it adds the word to the data frame and increases the counter for how many known concepts have been found. The function for unknown concepts first removes all the known concepts from the transcript and then iterates over the turns again. It checks if a noun is mentioned enough and if so it is added to the data frame for unknown concepts. The tool is available online[3]. The tool can also handle glossary documents. In this case, glossary documents are .txt files with an overview of words known and relevant to the project or a domain.

[1] https://github.com/lmruizcar/Requirements-Collector-DL-Component
[2] https://github.com/lmruizcar/ontology_crawler
[3] https://bowis.github.io/keyextractor/

Algorithm 1: Pseudocode of the key abstraction extraction tool prototype. *(Spijkman, Winter, et al., 2021, p. 4).*

```
input  : T: RE session transcript
         O: Ontology of the software product
output: E: Set of extracts where each e ∈ E is a tuple of <concept,
         frequency, speaker> categorized in known and unknown
         concept tables

function preprocessing;
for T do
    split by speaker tag s;
    remove timestamps;
    remove punctuation & stop-words;
    initiate function known concepts;
    initiate function unknown concepts;
end

function known concepts;
for speaker s
        for words in s do
            compare words to concepts in O;
            if known then
                add to dataframe;
                increase concept count;
            else
                discard words;
combine dataframes;
export combined dataframe to E (known);
end

function unknown concepts;
remove concepts in O from T;
for speaker s
        for noun phrases in s do
            determine number of mentions;
            if mentions > configured frequency then
                add to table;
                increase noun phrase count;
            else
                discard noun phrase;
combine dataframes;
export combined dataframe to E (unknown);
end
```

Keller, Lütolf, and Ruiz (2020) wrote a tool running on Ubuntu Linux as some TensorFlow functions are not available on Windows. The tool focuses on sentences instead of turns. The tool consists of a sentence classifier (A4) and a word classifier (A5). The tool is available online[4].

Abualhaija, Arora, Sabetzadeh, Briand, and Traynor (2020) build the tool DemaRQ (A6) for classifying requirements in a Requirements Specifications (RS) document. As it concerns a document and not a transcript, sentence analysis instead of turn analysis is used. Still, this tool is deemed useful because of the extensive documentation and feature list. The authors aim for as high recall as possible, so as to not miss any requirements. The RS document was first preprocessed by tokenization, then split into sentences by a sentence splitter and then ran through a part-of-speech (POS) tagger, which tags tokens with noun, verb or adjective. The results of these steps are then processed by a constituency parser, a dependency parser and a semantic parser. Abualhaija et al. (2020, p. 5460) explains the following about these parsers: "(1) constituency parsing for delineating the structural units of sentences, notably Noun Phrases (NPs) and Verb Phrases (VPs), (2) dependency parsing for inferring grammatical dependencies between the words in sentences, e.g., subject-object relations, and (3) semantic parsing for building a representation of the meaning of a sentence based on the meaning of the sentence's constituents". For semantic parsing, cognition verbs, action verbs and stative verbs are used. Especially cognition verbs are used to find sentences that indicate the rationale or thought behind requirements, and thus could indicate the User Story rationale part. After preprocessing and parsing the output is an RS document with annotations, which is used to extract metadata on e.g. verb distributions and frequencies. These results are used to build a frequency matrix after which the ML algorithm classifies if sentences contain a requirement or not. These results are refined in post-processing. The tool is available online[5].

Rodeghero, Jiang, Armaly, and McMillan (2017) build two Logistic Regression classifiers and two SVM-RBF classifiers (A7) to predict if a turn had information on User Story function or rationale. The Logistic Regression showed the best performance. Unfortunately, the classifiers are not available online and thus not a lot can be inquired about the way these are built.

The consideration of which RSAs to use for experimenting is based on the following factors:

- Availability of the source code
- Availability of original train and test data
- Relevance to the TUSP

An overview is given in table 9.

[4]https://github.zhaw.ch/kelles13/PA$_{AI4US}$
[5]https://zenodo.org/record/3881549.YPH2feTuLDs

Table 8: Overview of algorithms

Algorithm	**Prog. language**	**Labels (I/A)**	**Trained on (I/A)**	**Input**	**Output**
DL Classifier Component (A1)	Java	• None [0,NULL] • Non-Functional [1,A] • Functional [2,F]	• .txt file with 388 labelled turns from a real transcript	• .txt file with 3 tab separated columns [turn number, turn, label]	Overview of metrics: • Accuracy • Precision • Recall • F1 Score
Ontology Crawler (A2)	Java	N/A	N/A	• Ontology in .owl format	• List of User Stories in .txt format
Concept Extraction Tool(A3)	Python	N/A	N/A	• Ontology in a non-specified format • Transcript in .txt format, with a turn per line	• Set of tuples, categorized in known and unknown concept tables
Sentence Classifier (A4)	Python	• Irrelevant for User Story [0] • Relevant for User Story [1]	• .txt file of 20.000 generated sentences • .txt file with corresponding labels	• .txt file with 1 sentence per line • .txt file with 1 label per line	• Sentence classification model in .h5 format • Accuracy metric
Word Classifier (A5)	Python	• None [0] • Role [1] • Action [2] • Rational [3] • Pronoun [4] • Linking-word [9]	• .txt file of 20.000 generated sentences • .txt file with corresponding labels	• Transcript in .txt format. Turns do not have to be on individual lines	• .txt file of User Stories • .txt file of corresponding labels
DemaRQ (A6)	Java	• None • Requirement statement	• Undefined training set T	• RS document in .docx format	• Demarcated RS file in .arff format Overview of metrics: • Accuracy • Precision • Recall
LR and SVM-RBF (A7)	N/A	• Irrelevant for User Story • Relevant for User Story	• Transcripts in undefined format	• Transcript in undefined format	• Classification model

Table 9: RSA considerations

Algorithm	Source code available	Data available	Relevant
A1	+	+	+
A2	+	-	+
A3	+	+	+
A4	+	+	+
A5	+	+	+
A6	+	-	-
A7	-	-	+

To conclude, the Deep Learning Classifier Component from Ruiz and Hasselman (2020) (A1), the concept extraction tool from Spijkman, Winter, et al. (2021) (A3) and both the word and sentence classifiers from Keller et al. (2020) (A4 & A5) meet all considerations, as is shown in table 9.

3.10 Conclusion of the problem investigation phase

In conclusion, do sections 3.1 to 3.4 describe the general knowledge on requirements engineering needed to understand the importance and the context of developing a TUSP. Understanding and correctly eliciting requirements are of importance in developing a system. These requirements can be functional or non-functional and can be reported in a standardized form, the User Story. These requirements can be elicited by multiple methods but this thesis project focuses on the RE meeting and the interview method.

Sections 3.5 to 3.9 show the technical knowledge available to develop an RSA. It is explained why turns are used to define a unit of speech in this thesis project. Section 3.6 shows that there is no knowledge available on User Story traceability. However, this section does hypothesise how User Story traceability could work, based on basic traceability literature. Section 3.8 provides an overview of the available knowledge on automatic requirements or User Story extraction from RE meeting transcripts. It shows that there is no method or tool available yet that can extract requirements or User Stories without human interaction or supervision. There is also no knowledge available on creating traceable requirements or User Stories. The section also defines a 'concept'. Finally, section 3.9 discusses the currently available RSAs. Out of these, the decision has been made to focus on the designs by Ruiz and Hasselman (2020), Keller et al. (2020) and Spijkman, Winter, et al. (2021).

4 Designing a Transcript to User Story Pipeline prototype

This chapter will explain how the components found in the Problem Investigation Phase are adjusted and fitted together to create the TUSP. The chapter starts with a description of a scenario in which the TUSP can be used. The choice has been made to make two configurations for the TUSP which are discussed in section 4.1. This section also give a more detailed view of the TUSP. The rest of the chapter will explain the different components based on that view in sections 4.2 to 4.8. The chapter is concluded with a technical view of the TUSP in section 4.9.

An example of how the TUSP could be put to practice in a hypothetical scenario is given in figure 8. In this scenario a client and an RE engineer discuss requirements in an RE meeting. This meeting is recorded (step 1) and then transcribed (step 2) by hand or through a preferred transcription tool such as Google Speech-to-Text[6], Amazon Transcribe[7] or otter.ai[8], which is used in this example. This transcript is then exported to a .txt type file (step 3), the required type to be used as input for the TUSP. In step 4 the transcript has to be transformed to the correct form, meaning 1 turn per line and with correct turn labelling. See section 4.2 for an explanation on this turn labelling, also called Turn Traceability IDs. The transformed .txt type transcription is then run in the preferred TUSP configuration (step 5), which are explained in section 4.1. Both of these configurations output a .txt type file with US candidates, which can be exported to the preferred US management software (step 6). In step 7 these US candidates are filtered on relevant information. An important note is that both configurations require the same data preparations and formatting and also output in the same format, as can be seen in the figure, while the actual outputted User Stories may be different for each configuration.

[6] https://cloud.google.com/speech-to-text
[7] https://aws.amazon.com/transcribe/
[8] https://otter.ai/

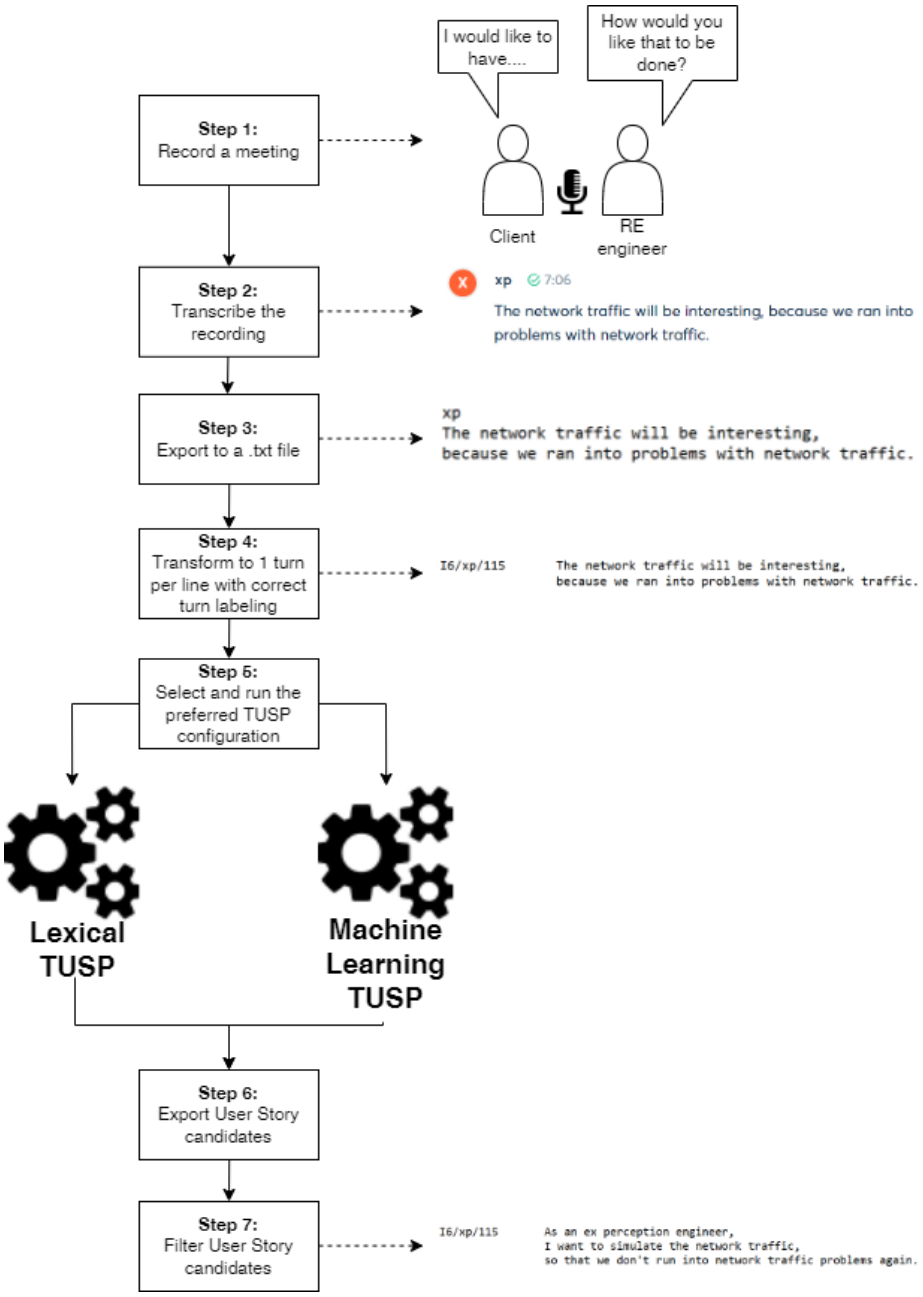


Figure 8: From RE meeting to User Story scenario.

4.1 Different configurations of the Transcript to User Story Pipeline

The final design of the TUSP contains 5 components: the Requirement Sentence Classifier, the Domain Concept Extractor, the Fit-Gap Searcher, the Requirement Word Classifier and the User Story Builder. These different components of the TUSP influence one another, and therefore it is envisioned that a user may not want to use all the components. This can be dependent on the result that the user is looking for. Therefore two TUSP configurations have been made, the Machine Learning Configuration (section 4.1.1 and the Lexical Configuration (section 4.1.2).

4.1.1 The Machine Learning Configuration (MLC)

The Machine Learning Configuration uses the Requirement Sentence Classifier, the Domain Concept Extractor, the Requirement Word Classifier and the User Story Builder components. It is called the Machine Learning Configuration because it uses both a Deep Learning and a Machine Learning component. The transcript is first processed by the Requirement Sentence Classifier, which extracts all the turns that are supposed to have information on a User Story. These turns are then processed by the Domain Concept Extractor, which extracts the known and unknown concepts. Then, the Requirement Word Classifier indicates which words in a turn belong in which User Story part. If nothing of interest is found then a User Story part is left blank.

4.1.2 The Lexical Configuration (LC)

The Lexical Configuration uses the Domain Concept Extractor, the Fit-Gap Searcher, the Requirement Word Classifier and the User Story Builder components. The transcription is first processed by the Domain Concept Extractor, which extracts the known and unknown concepts. Then, the Fit-Gap Searcher extracts the relevant turn sections. If the Fit-Gap Searcher does not find anything, then the Requirement Word Classifier will be used to try and fill in the gaps. If the Requirement Word Classifier also does not result in anything then that part of the User Story is left blank.

The whole TUSP and its two configurations can be seen in figure 9. There are four ways that data can flow: from algorithm to algorithm, from artifact to artifact, from algorithm to artifact and vice versa. All of data flows from artifact to algorithm and from artifact to artifact have to be done manually at the moment. All of the data flows from algorithm to artifact and from algorithm to algorithm are automated and do not require manual interference. For simplicity and overview is the User Story Builder depicted as 1 component, even though there are two User Story Builders, as described in section 4.7.

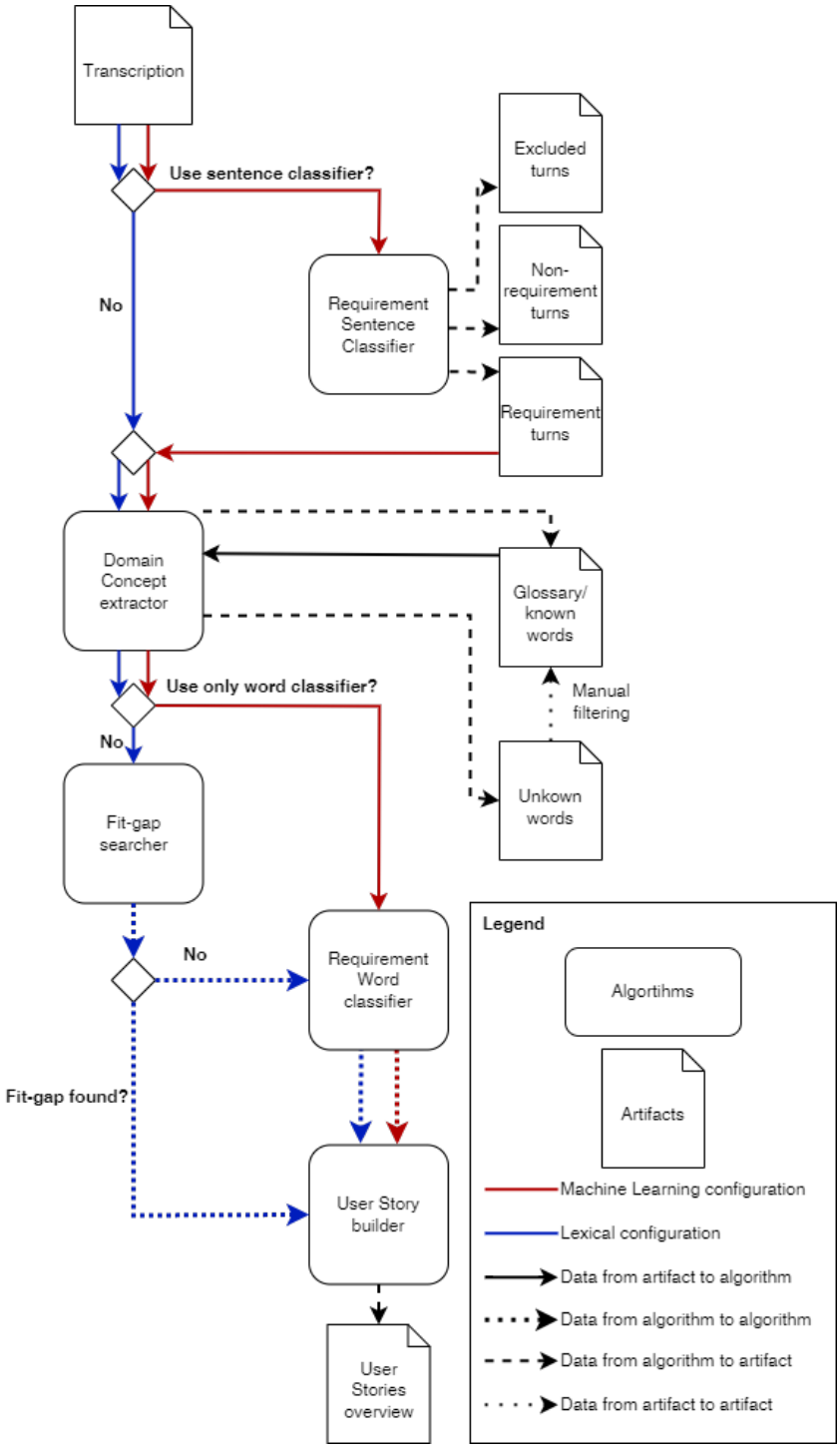


Figure 9: Overview of the TUSP and its two configurations.

4.2 Turn Traceability IDs

All lines and thus turns in every transcript are coded with a unique ID to ensure traceability. This ID consists of the file name, the abbreviation of the role of the actor whose turn it is and the line number in the whole transcript. The different parts of the ID are divided by a "/". An example of the Turn Traceability IDs can be seen in figure 10. The abbreviation for the role is included as a backup for when the Requirement Word Classifier cannot find an actor in a specific turn. These abbreviations consist of two letters and are case sensitive. The abbreviations are saved in an excel (.xlsx) worksheet, an example of the used abbreviations for this thesis project can be seen in figure 11. More on the usage of actor abbreviations is explained in section 4.5.

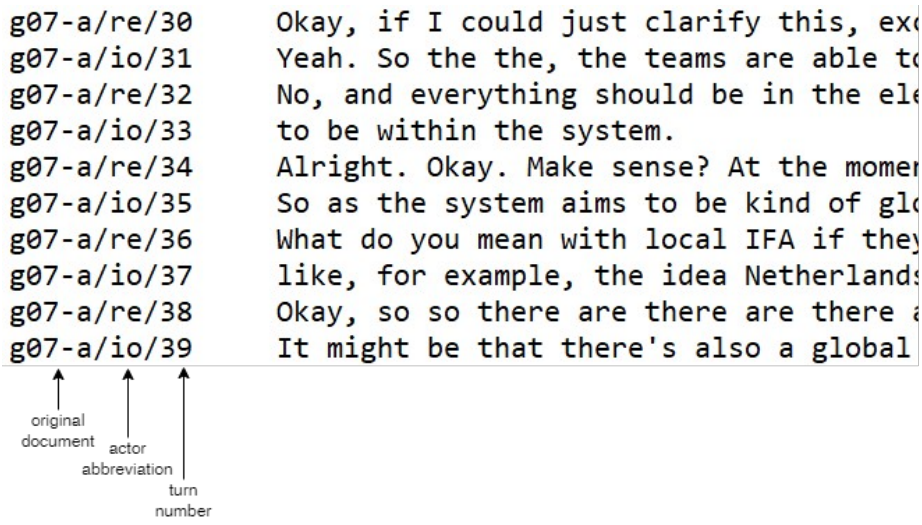


Figure 10: Example of the Turn Traceability IDs. re stands for Requirements Officer, io stands for IFA information officer.

	A	B	C	D
1	abbreviation	name		
2	re	requirements engineer		
3	io	IFA information officer		
4	hu	head of urban planning		
5	ea	enterprise architect		
6	ct	CTO		
7	xt	ex CTO		
8	sd	simulation tool developer		
9	ce	CEO		
10	pe	perception engineer		
11	ee	ECU engineer		
12	xp	ex perception engineer		
13	ae	autopilot engineer		
14				

Figure 11: Example of the abbreviations used in this thesis project.

4.3 The Requirement Sentence Classifier

The problem investigation found two Requirement Sentence Classifier candidates, a Classifier based on Deep Learning (DL) and a Classifier based on Machine Learning (ML). In the following section, these candidates are compared after which one is chosen on showing the best results. This Requirement Sentence Classifier is then modified to fit with the overall TUSP prototype. Please note that the name Requirement Sentence Classifier is adopted from the work of Keller et al. (2020), but that both the Classifiers process turns. A Requirement Sentence Classifier component is included in the TUSP to allow the RE engineer to filter redundant turns. In theory, this could reduce processing time when further in the pipeline only turns that contain relevant User Story information are processed.

4.3.1 Data used for testing Requirement Sentence Classifiers

The following data files are used to benchmark the Requirement Sentence Classifiers.

File: 1
Authors: Ruiz and Hasselman (2020)
Name: test_set.txt
File 1 contains fake testing data. It consists of 15 lines of turns all labelled with a turn number and either the label “A” when the sentence contains non-functional requirement information, “F” when the sentence contains functional requirement information or “NULL” when the sentence contains nothing of interest. A part of the file is shown in figure 12.

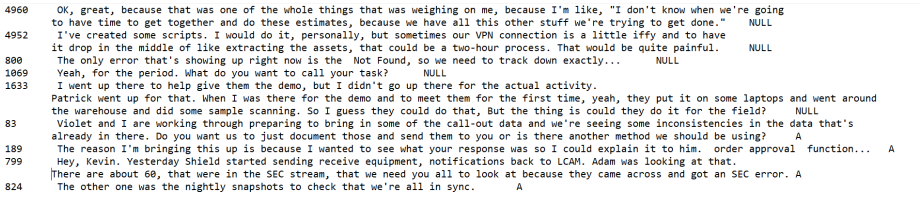


Figure 12: Example of file 1.

File: 2
Authors: Keller et al. (2020)
Name: GenTextData.txt
File 2 contains fake data. It consists of 20 lines of sentences either containing or not containing information about User Stories. These sentences can also be seen as turns. A part of the file is shown in figure 13.

```
investor is able to interact with Colour to close the Window and it would be rather nice if she wants to be able to
        interact with the Time to close the Window.
buyer is able to press a Toggle Switch to see meal time on the other hand it would be rather nice if she is able to
        watch the Screen to close the Window.
client can interact with the Screen to read the news or it would be rather nice if she wants to be able to see the
        Time to see meal time.
End user wants to be able to see a Toggle Switch to get an Overview on the other hand it would be rather nice if he can
        read the Screen to read the news.
Hanna will have a toilett breeak
Marcela wants a coffee
Simon needs a terrible idea
Marcela is having a coffee
Hanna needs a coffee
```

Figure 13: Example of file 2.

File: 3
Authors: Keller et al. (2020)
Name: interview_jonas.txt
File 3 contains a transcribed RE interview with 2 participants: the interviewer and Jonas. It consists of 95 lines each containing a turn. A part of the file is shown in figure 14.

```
Okay, the next question, pretty much answered.
All your live citizen from ran in the output?
Maybe?
Yeah.
For now, and nearly all teams did it for us.
I think it's also the being at the beginning.
I also thought why Ross, we can also do just without Ross and just write everything in c++ or so.
But I think it's really, it's really nice for developing because it's easy to split the code.
And then you just have to define the, the message types.
Yeah.
And so it's very easy to test different things.
```

Figure 14: Example of file 3.

File: 4
Authors: Keller et al. (2020)
Name: interview_yasmine.txt
File 4 contains a completed questionnaire. This means that the data is more structured than an interview. It consists of 72 lines each containing a turn. A part of the file is shown in figure 15.

```
How accurate does the system need to be?
Very accurate as this needs to be valid for real-time systems.

In what shape or form do we receive tracks?
Please ask Mapping team (Nicolas Borla or Jonas Langenegger).

What kind of sensors do you use?
GPS, IMU, Wheel Encoders, Actuators for steering, drives and brakes, camera, lidar

How do they work?
How detailed does this question need to be answered?
```

Figure 15: Example of file 4.

4.3.2 Guideline for data preparation

Both the ML and the DL Classifiers required the data to be supplied in different forms. The ML algorithm takes text files. All the text can be on one line, the algorithm splits the sentences on punctuation marks. A separate file containing the labels, either 0 for "nothing of interest" or 1 for "contains requirement information". The DL Classifier also takes text files but requires each turn to be on one line and the data to be written as line-number/turn/label. The labels consist of "F", "A" or "NULL", meaning a functional requirement, non-functional requirement or nothing of interest respectively.

File 2, File 3 and File 4 were not in the required format for the DL Classifier, and all 4 files were not labelled in a way that was compatible with the ML Classifier, so these files were manually prepared. Data was labelled with the hypothesis in mind that to gain a high recall, everything that connected to a possible functional or non-functional requirement was labelled. One problem that the author ran into was how to label question-answer constructions. An example: the interviewer asks the question *"would you want a button to click?"*. This question becomes a requirement dependent on whether the client answers with either *"yes"* or *"no"*. The decision has been made to label both the question and answer turns as a (non-)functional requirement if the answer is "yes" and to label both the turns as nothing of interest if the answer is "no". The decision if a requirement is functional or non-functional is based on the literature research done in section 3.2.

4.3.3 Comparison of the Deep Learning and Machine Learning Classifiers

Both the DL and the ML Classifiers are bench-marked with files 1 to 4. Both Classifiers are trained on their original training sets used for their respective papers. The results are shown in tables 10 and 11.

Results: The Deep Learning Classifier

Table 10: DL Classifier bench-mark metrics.

Test file	No. of epochs	Accuracy	Precision	Recall	F1-score
File 1	10	0.2667	0.1429	0.2667	0.2105
	25	0.3333	0.4038	0.3333	0.3651
	50	0.3333	0.4038	0.3333	0.3651
	100	0.2667	0.1429	0.2667	0.2105
File 2	10	1.000	1.000	1.000	1.000
	25	1.000	1.000	1.000	1.000
	50	1.000	1.000	1.000	1.000
	100	0.9500	0.9500	0.9545	0.9499
File 3	10	0.3474	0.4255	0.3831	0.3961
	25	0.4526	0.4678	0.4261	0.5097
	50	0.4316	0.4580	0.4125	0.4887
	100	0.3789	0.4646	0.4035	0.4355
File 4	10	0.5938	0.5518	0.4621	0.6026
	25	0.6875	0.6474	0.4829	0.6758
	50	0.6875	0.6474	0.4829	0.6758
	100	0.5000	0.5000	0.3956	0.5119

Results: The Machine Learning Classifier

Table 11: ML Classifier bench-mark metrics.

Test file	No. of epochs	Accuracy	Precision	Recall	F1-score
File 1	10	0.3333	0.0000	0.0000	0.0000
	25	0.4167	0.5000	0.1429	0.2222
	50	0.4167	0.5000	0.1429	0.2222
	100	0.3333	0.0000	0.0000	0.0000
File 2	10	1.0000	1.0000	1.0000	1.0000
	25	1.0000	1.0000	1.0000	1.0000
	50	1.0000	1.0000	1.0000	1.0000
	100	1.0000	1.0000	1.0000	1.0000
File 3	10	0.5579	0.6667	0.0465	0.0869
	25	0.5368	0.4545	0.1163	0.1851
	50	0.5895	0.7000	0.1628	0.2642
	100	0.5684	0.5833	0.1628	0.2545
File 4	10	0.5833	0.7500	0.0938	0.1667
	25	0.5833	0.7500	0.0938	0.1667
	50	0.5694	0.5714	0.1250	0.2051
	100	0.5972	0.6000	0.2813	0.3830

The results in tables 10 and 11 show that the generated data in file 2 is too simple for accurately testing Sentence Classifiers, as apparently both of the classifiers are almost perfect in predicting the sentences. Even though the DL classifier was originally tested on file 1, does it not perform better on this file than the ML classifier does. No explanation for this was found. Furthermore are differences in accuracy, precision and F1 scores different per file and classifier. The ML classifier scores higher accuracy scores for files 1 and 3 but the DL classifier achieves an overall higher accuracy on files 2 and 4. The ML classifier is also able to achieve a relatively high precision on files 3 and 4. However, the results also show that the DL Classifier performs best on recall for all test files when classifying sentences. For this reason, the DL Classifier is chosen as the main Requirement Sentence Classifier for this project. The results also show that more epochs do mean better results. No file showed an improvement when the number of epochs was higher than 25 epochs.

4.3.4 Adjusting the Deep Learning Classifier

To be able to use the DL Classifier in the TUSP prototype some adjustments had to be made. No adjustments to the training data or method have been made. The original classifier was not able to make predictions on files that were not already labelled, so a new predict function was added. The original classifier made a distinction between non-functional requirements (classified as

"A"), functional requirements (classified as "F") or no requirements information (classified as "NULL"). The adjusted DL classifier for the TUSP groups all "A" and "F" classifications as *containing requirement information* and all "NULL" classification as *containing no requirement information*. Furthermore, would the program crash if a turn did not contain any word present in the Glove vocabulary. Therefore these turns are filtered and saved in a separate file: excluded_results_[fileName]. In the end the DL Requirement Sentence Classifier outputs three files: the excluded turns (excluded_results_[fileName], the turns that were classified as not containing any User Story related information (non_requirement_results_[fileName] and the turns classified as containing User Story information (requirement_results_[fileName]). An example of the DL classifier having finished a run is shown in figure 16.

```
==========Scores========================================
 # of classes:    3
 Accuracy:        0.6875
 Precision:       0.6474        (1 class excluded from average)
 Recall:          0.4829
 F1 Score:        0.6758        (1 class excluded from average)
Precision, recall & F1: macro-averaged (equally weighted avg. of 3 classes)
========================================================================
35258 [main] INFO ch.zhaw.hassebjo.Word2VecTest - Number of lines in TEST_SET excluded from evaluation: 8
35260 [main] INFO ch.zhaw.hassebjo.Word2VecTest - Predicting sentences
37542 [main] INFO ch.zhaw.hassebjo.Word2VecTest - Evaluation and Prediction done
resulting files can be found in: output\results\
```

Figure 16: Example of the DL classifier having finished a run

4.4 The Domain Concept Extractor

The tool that Spijkman, Winter, et al. (2021) developed for extracting concepts from a transcript originally consisted of 4 files: a file for extracting known concepts (known.py), a file for extracting unknown concepts (unknown.py), a file for providing statistics on the extracted words (nltk.py) and a file for splitting raw text files into multiple sections (splitter.py). nltk.py and splitter.py have been discarded for use in the TUSP as statistics are deemed out of the scope and the files are assumed to already be properly prepared, making splitting unnecessary. For the TUSP, known.py and unknown.py have been adjusted and a coordinator file (ce_coordinator.py) for coordination of known.py and unknown.py has been added. Originally, known.py and unknown.py extracted on speakers and outputted a file listing the known or unknown concepts and which speaker has used those concepts. This has been adjusted to output a file where speakers are ignored but it is recorded which turns contain the known/unknown concepts. This is done to ensure traceability. The ontology used as input, next to the transcript, is a .txt file filled with all the known concepts. The Domain Concept Extractor is included in the TUSP to provide the RE engineer with additional artefacts which help to support the understanding of the client's wants and needs. The output of the unknown.py extractor on one of the validation files is shown in figure 17. It shows that the abbreviation "ifa" is found 25 times, with an overview of the turns it was extracted from. This artefact with unknown extracted concepts can be manually filtered by the RE engineer. The

concepts that remain after filtering are added to the known concepts artefact, to gradually produce a more complete overview of the concepts associated with the project. This can help improve the domain knowledge and in understanding the processes surrounding the project.

Noun phrase	Frequency	Turn indexes
ifa	25	['g01-a/re/1', 'g01-a/re/12', 'g01-a/io/15', 'g01-a/io/19', 'g01-a/re/31',
it	5	['g01-a/re/86', 'g01-a/re/92', 'g01-a/re/103', 'g01-a/re/114']
real time	3	['g01-a/io/19', 'g01-a/io/82', 'g01-a/io/106']
good question	2	['g01-a/io/89', 'g01-a/io/91']
exactly	2	['g01-a/re/90', 'g01-a/re/100']
new system	2	['g01-a/re/1', 'g01-a/re/103']
scheduling algorithm	2	['g01-a/io/34', 'g01-a/io/38']
currently	2	['g01-a/re/24', 'g01-a/io/113']
technicalities system	1	['g01-a/re/79']
sounds sounds	1	['g01-a/io/78']
specific game	1	['g01-a/re/77']
minutes game	1	['g01-a/re/77']
example remind	1	['g01-a/re/77']

Figure 17: Example of output from the unknown.py program

4.5 The Requirement Word Classifier

A Requirement Word Classifier component is added to the TUSP to classify the words in a turn important for the User Stories. This way the irrelevant words and information in a turn is filtered out. The Requirement Word Classifier originally designed by Keller et al. (2020) made use of a trained ELMO model to classify a word in a turn as one out of five labels. These labels can be seen in table 12. The pronoun label is used to find the correct pronoun context. Keller et al. (2020) introduced the "linking word" to check if a turn contains one or multiple User Stories.

Table 12: The five labels used by the Requirement Word Classifier.

Label	Meaning
0	Nothing of interest
1	Role
2	Goal
3	Benefit
4	Pronoun
9	Linking word

No alterations have been made to the training procedure of the ELMO model. The model is trained on 40.000 labelled sentences that are generated by a program written by Keller et al. (2020). The model is trained over 4 epochs. As the training is computationally intensive, the choice has been made for 4 epochs. As each epoch takes on average 353 seconds to train, means that training on 4 epochs takes around 24 minutes. More epochs do not improve the results in a way that justifies even more training time. The Requirement Word Classifier training itself is shown in figure 18.

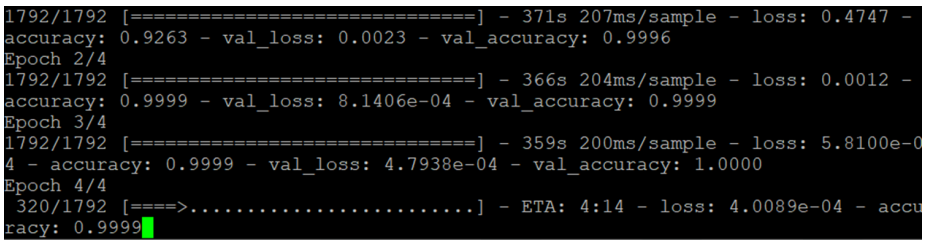


Figure 18: The Requirement Word Classifier in training

4.6 The Fit-Gap Searcher

The Fit-Gap Searcher is an algorithm developed for this thesis project based on the keywords and phrases extracted by Spijkman, Dalpiaz, et al. (2021), which have been acquired from the authors. The Fit-Gap Searcher uses the keywords and phrases which have been identified to indicate a *requirement*. Examples of these keywords and phrases are: "we need", "what would be good" or "it would be nice". The Fit-Gap Searcher is a lexical, hard-coded algorithm and uses regular expressions to recognize these keywords and phrases and if the *requirement* come before or after. These *requirements* are used as input for the Goal part of a User Story. The Fit-Gap Searcher also recognizes possible turn sections for the Benefit part of the User Story. The Fit-Gap Searcher is not yet able to filter sentence parts that are of importance for the Role part of a User Story.

This component is added to the TUSP because during testing of the Sentence and Requirement Word Classifiers a different approach was envisioned, one that was not dependent on training data as classifiers based on Machine- and Deep learning techniques are. The Fit-Gap Searcher provides this solution. Part of the trigger words written as regal expressions that the Fit-Gap Searcher uses to recognize important sentence parts in a transcript are shown in figures 19, 20 and 21. The regal expressions are divided into expressions that search for ensuing, preceding and in-between sentence parts.

```
# sentences that indicate a requirement that ensues
req_re1 = re.compile(r"(i|we|they) need (.*)", re.I)
req_re2 = re.compile(r"(what would be good )(.*)", re.I)
req_re3 = re.compile(r"(if we have )(.*)", re.I)
req_re4 = re.compile(r"(i|we|they) want (.*)", re.I)
req_re5 = re.compile(r"(want to )(.*)", re.I)
req_re6 = re.compile(r"(it would be nice )(.*)", re.I)
req_re7 = re.compile(r"(it should )(.*)", re.I)
req_re8 = re.compile(r"(can i just )(.*)", re.I)
req_re9 = re.compile(r"(insist that )(.*)", re.I)
```

Figure 19: Fit-Gap Searcher code example 1

```
# sentences that indicate a requirment that precedes
req_re11 = re.compile(r"(.*)would be an improvement", re.I)
req_re12 = re.compile(r"(.*)that would be awesome", re.I)
```

Figure 20: Fit-Gap Searcher code example 2

```
# sentences that indicate something inbetween
req_re17 = re.compile(r"if (.*) we will be fine", re.I)
```

Figure 21: Fit-Gap Searcher code example 3

4.7 The User Story Builder

The User Story builder component does not do any classification. Instead, it fits the classified words and sentence fragments found by the Requirement Word Classifier and the Fit-Gap Searcher into the standardized form of a User Story: As a [role], I want [goal], so that [benefit]. The component also creates the .txt file filled with User Story candidates including the turn labels from which these candidates were extracted. Two User Story Builders are developed. One is originally developed by Keller et al. (2020) and is used solely by the Requirement Word Classifier component. This User Story Builder can build User Stories which contain words from multiple turns. The other User Story Builder is developed by the author and uses the output from the Fit-Gap Searcher. This User Story Builder produces User Stories which contain words from one turn. The reason there are two User Story Builders is because of the different ways

the Requirement Word Classifier and the Fit-Gap Searcher are coded. Both of the User Story builders do essentially the same thing, which is why they are depicted as one in figures to come.

4.8 Fitting the components

In the end, there are five components to create the TUSP. In theory, if all components are used sequentially, 5! = 120 TUSP configurations should be possible. But not all of these configurations make sense. For instance, if the User Story Builder were to be placed at the beginning of the pipeline, then the User Story builder would not be able to produce sensible USs, as it does not know which words contain US information. The User Story Builder needs the input from either the Fit-Gap Searcher or the Requirement Word Classifier to function. And, since the goal of this thesis is to automatically produce User Story candidates, it makes the most sense to have the User Story builder as the last component. The Fit-Gap Searcher and the Requirement Word Classifier can be used separately or the Requirement Word Classifier can use the output from the Fit-Gap Searcher as input. In this case would the Requirement Word Classifier extract words from the sentence parts filtered out by the Fit-Gap Searcher. This approach is hypothesised to be sub-optimal because the Requirement Word Classifier could miss out on words not filtered by the Fit-Gap Searcher, such as words depicting a Role. At the moment the Fit-Gap Searcher is not yet able to extract sentence parts on Roles. Since the Requirement Word Classifier output a list of words will the Fit-Gap Searcher not be able to work with the output from the Requirement Word Classifier.

The User Story builder is not able to use the output from the Requirement Sentence Classifier as input. Therefore does this component always has to come before the Fit-Gap Searcher or the Requirement Word Classifier. In theory could the Fit-Gap Searcher be used before the Requirements Sentence Classifier in which case the Requirements Sentence Classifier would predict if sentence fragments actually contain User Story information. The Requirement Sentence Classifier filters the original data to exclude turns that do not have information on User Stories, which makes it useful to reduce the amount of data that has to be processed by the Requirement Word Classifier or the Fit-Gap Searcher. This component is therefore placed at the beginning of the pipeline.

The Domain Concept Extractor can in theory be used at any moment in the pipeline as it does not mutate any data but only creates new artefacts. However, it is recommended to use it either at the beginning of the pipeline or after the Requirement Sentence Classifier. This way most unknown concepts are caught and can be used during the process of Requirements Engineering.

Taken all of the above, it shows why the most logical build-up of the TUSP consists of the Requirement Sentence Classifier or the Domain Concept Extractor at the beginning, followed by the Fit-Gap Searcher or the Requirement Word

Classifier in the middle and ended with the User Story Builder.

4.9 Concluding the treatment design phase

This chapter has shown the choices made to build the finalized TUSP prototype. The complete TUSP includes a Requirement Sentence Classifier, a Domain Concept Extractor, a Requirement Word Classifier and a Fit-Gap Searcher. These can be used according to different configurations, of which two have been described in sections 4.1.1 and 4.1.2.

Finally, The technical architecture of the TUSP is depicted in figure 22. The TUSP is currently run on a client and a server. This division is made because of the need for the Requirement Word Classifier to run on a Linux machine. Theoretically the whole TUSP can run on one machine/server, if it is a Linux server. The Requirement Sentence Classifier written in Java and therefore runs on a Windows machine, all the other TUSP components are written in Python and can therefore be run on the Linux server. Any files have to be transferred to and from the Linux server manually at the moment. The main libraries used are also depicted in the figure. The Requirements Sentence Classifier uses the Deeplearning4j library for setting up the multi layer network used to predict the turns that contain US information. The Deeplearning4j library is also used to create the Word2Vec model which turns the words in a turn into vectors which the multi layer network can understand. The Known and Unknown components of the Domain Concept Extractor use TextBlob for extracting noun phrases and use NLTK for tokenization and detokenization of words next to filtering out stop words. The Requirement Word classifier uses TensorFlow to set up the ELMO model with an extra Long Short-Term Memory (LSTM) layer in order to predict the words that contain US information. To give an example of how data would flow in the MLC: the user selects a transcripts through Eclipse and the Requirement Sentence Classifier outputs a file with the predicted turns that contain US information. The user then uploads this file through the file management component (for instance FileZilla or any other File Transfer Protocol solution) to the Ubuntu server. The transcript now exists as a copy in the "Copies of client files". The user then interacts with the TUSP coordinator component through a frontend (for instance PuTTY or any other SSH client). Once the correct file and the MLC instructions are given will the TUSP coordinator take over and coordinate the Domain Concept Extractor, the Requirement Word Classifier and the User Story Builder. The output from the TUSP can then be found in the Copies of client files again, from where it can be downloaded through the file management component.

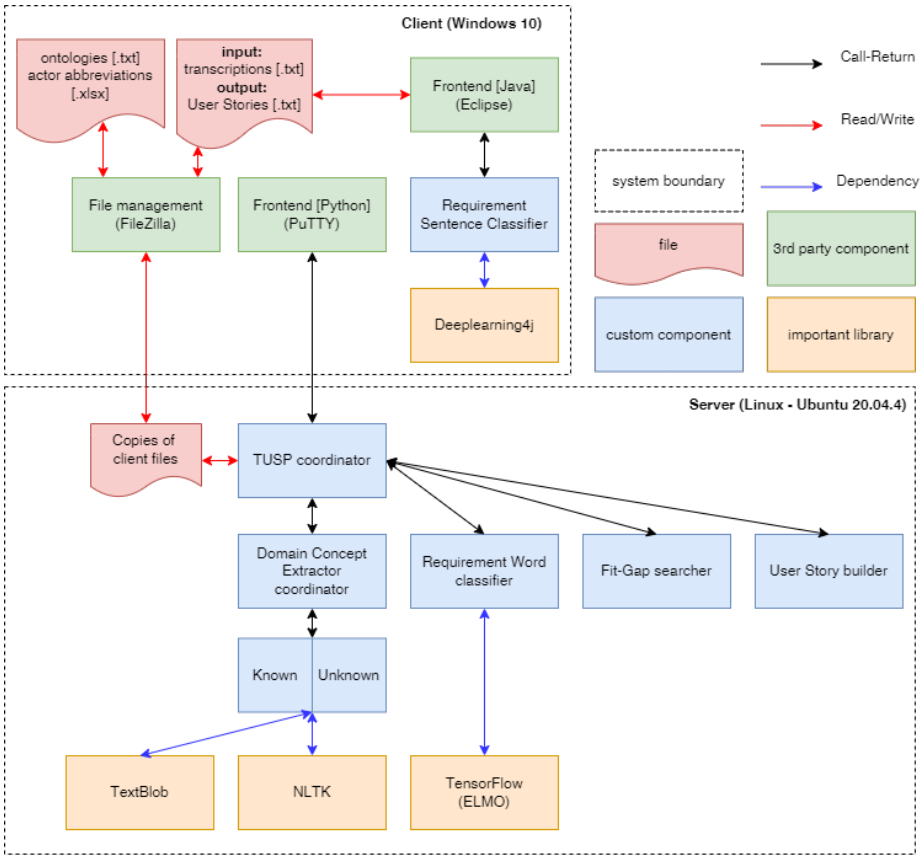


Figure 22: Technical architecture of the TUSP.

5 Validating the Transcript to User Story Pipeline

In this chapter, the TUSP will be validated in two ways. The TUSP is validated in two case studies: the Greenfield Case and the Customization Case. First, the case study design is explained in section 5.1. This is followed by an explanation of the two case studies and their respective meta-data in sections 5.2 and 5.3. Then the validation results are analysed in sections 5.2 and 5.3.

5.1 Case study design

The case study design is simplified version of the case study design as described by Wohlin et al. (2012). It is simplified because Wohlin et al. (2012) also describes the need for describing the research questions and the theoretical framework, which are for this thesis already described in section 2.2 and chapter 2.1.1 respectively. Thus this section will focus on the *objective*, the *case*, the *methods* and the *selection strategy*:

- **Objective:** To conduct two exploratory case studies. Exploratory because the TUSP is only a prototype and in this section it is explored how well it performs on the two cases.
- **Case:** Two cases are handled, the Greenfield case and the Customization case, further explained in sections 5.2 and 5.3 respectively.
- **Methods:** Data is retrieved through interviews and received from other people.
- **Selection strategy:** Data is retrieved in the contexts of the two cases.

5.2 The Greenfield Case

For the Greenfield case the author collaborated with the Formula Student (FS) team of the *Zürcher Hochschule für Angewandte Wissenschaften* (ZHAW). Keller et al. (2021) started on a Simulation Tool for the FS ZHAW team. For this Simulation Tool, they extracted 46 User Stories from an RE interview and a survey with two FS ZHAW team members. Keller et al. (2021) used these User Stories and the open-source simulation tool made by the Edinburgh FS team[9] to create a prototype for the ZHAW FS simulation tool. It was hypothesised that the User Stories found by Keller et al. (2021) were not an extensive list and that more could be found if more interviews with different team members were held. Thus, for the Greenfield case, five extra interviews were held, bringing the total to six interviews and one survey. These seven files act as the Units of Analysis for the Greenfield Case. An overview of the interviews/survey and the roles of the participating FS ZHAW members can be seen in table 13.

[9] https://gitlab.com/eufs/eufs$_sim$

Table 13: The roles and number of people present during the interviews for the Greenfield case. I Stands for Interview, S stands for Survey.

ID	Roles
I1	1 RE engineer, 1 Path-finding Engineer
I2	1 RE engineer, 1 CTO, 1 ex-CTO, 1 Simulation Tool Developer
I3	1 RE engineer, 1 CEO, 1 CTO, 1 Perception engineer
I4	1 RE engineer, 1 CTO, 1 ECU Engineer
I5	1 RE engineer, 1 Path-finding Engineer
I6	1 RE engineer, 1 CTO, 1 Perception Engineer, 1 ex-Perception Engineer
S1	1 RE engineer, 1 Ex-CTO/System Architect

5.2.1 Interview protocol

Multiple RE meeting interviews with members of the FS team were held. The interviews were recorded to be transcribed. These transcriptions were as input for the Greenfield Case. Next to the recordings the interviewer made notes during the interviews to assist the interview and prepare follow-up questions. The interview protocol had multiple iterations. Parts of the protocol that turned out to be sub-optimal were changed. All versions of the protocol can be found in Appendix B.

5.2.2 Greenfield case files metadata

All the transcriptions for the Greenfield Case consist of 1472 lines and the survey contains 72 lines. From these 7 files the User Stories are extracted, and these are used as so-called Ground Truth. These Ground Truth User Stories are what the results from the TUSP configurations are compared against. An overview of the number of turns per file and the number of Ground Truth User Stories that were extracted are shown in table 14. An overview of the actual User Stories extracted from the files can be found in Appendix D.

Table 14: Greenfield case files metadata

ID	# of turns	# of User Stories
I1	95	27
I2	370	8
I3	340	42
I4	193	17

continues on next page

I5	119	12
I6	157	8
S1	72	14

5.3 The Customization Case

For the Customization Case, the author received ten audio files recorded by students following the Requirements Engineering course held at Utrecht University in the year 2019/2020. These ten files were transcribed and acted as the Units of Analysis for the Customization case. For the RE course, the students had to work on one out of three hypothetical cases and interview a stakeholder played the course organizers or assistants. Two students played the role of RE engineer each time. These cases are further explained below.

Case A - International Football Association (IFA) portal For Case A the students had to interview the chief information officer of the IFA. This chief information officer wanted an operational and analytical system to combine all the core information systems that the IFA used before. The students had one interview to extract as much information regarding the system to-be as possible. Four interviews of Case A were received, which totalled 740 lines when transcribed and prepared. The extracted User Stories that are used as ground truth can be found in Appendix E. The roles of the people present during the meetings can be found in table 15 and the meta-data of the files can be seen in table 16.

Table 15: The roles and number of people present during the interviews for the IFA portal.

ID	**Roles**
A1	2 RE engineers, 1 IFA chief information officer
A2	2 RE engineers, 1 IFA chief information officer
A3	2 RE engineers, 1 IFA chief information officer
A4	2 RE engineers, 1 IFA chief information officer

Table 16: Customization case A files metadata

ID	# of turns	# of User Stories
A1	133	44
A2	153	38
A3	241	37
A4	213	36

Case B - Urban Mobility Simulator For Case B the students had to meet up with the head of urban planning of a fantasy city. This head of urban planning required an urban traffic simulation tool that was to be based on existing solutions, as the municipality did not have the money to build one from scratch. In one interview the students had to find out as much as possible about the wants and needs of this head of urban planning. Two interviews of Case B were received, which totalled 353 lines when transcribed and prepared. The extracted User Stories that are used as ground truth can be found in Appendix E. The roles of the people present during the meetings can be found in table 17 and the meta-data of the files can be seen in table 18.

Table 17: The roles and number of people present during the interviews for the urban mobility simulator.

ID	Roles
B1	2 RE engineers, 1 head of urban planning
B2	2 RE engineers, 1 head of urban planning

Table 18: Customization case B files metadata

ID	# of turns	# of User Stories
B1	247	28
B2	106	18

Case C - Hospital Management System For Case C the students interviewed the enterprise architect of a hypothetical hospital to extract the requirements for a new hospital management system. The system as-is consisted of more than 32 hospital management systems which all operated independently. Four interviews of Case C were received, which totalled 493 lines when transcribed and prepared. The extracted User Stories that are used as ground truth can be found in Appendix E. The roles of the people present during the meetings can be found in table 19 and the meta-data of the files can be seen in table

20.

Table 19: The roles and number of people present during the interviews for the IFA portal.

ID	Roles
C1	2 RE engineers, 1 enterprise architect
C2	2 RE engineers, 1 enterprise architect
C3	2 RE engineers, 1 enterprise architect
C4	2 RE engineers, 1 enterprise architect

Table 20: Customization case C files metadata

ID	# of turns	# of User Stories
C1	71	26
C2	145	42
C3	171	43
C4	106	46

To conclude, the validation will be an embedded case study (Yin, 2009), where seventeen Units of Analysis, the different transcripts, are studied within two contexts and two cases. A visualization of this is given in figure 23. The contexts are everything surrounding the case studies that is not studied explicitly. So for the Greenfield case, this means the FS ZHAW and for the Customization case, this means the RE course.

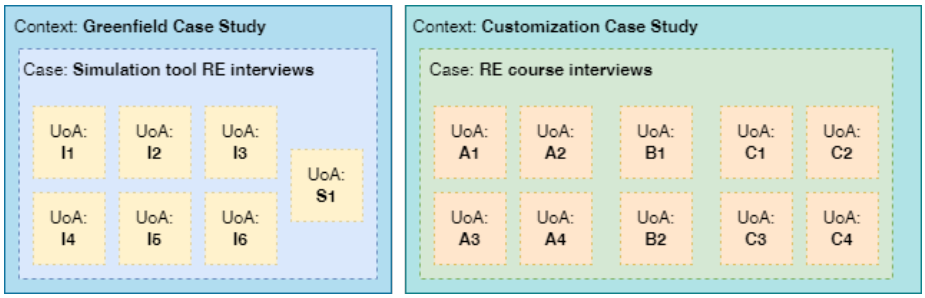


Figure 23: Case Study designs visualized. UoA stands for Unit of Analysis.

5.3.1 Results of the Machine Learning Configuration

The MLC starts with running a transcript through the Sentence Classifier where all turns that do not contain User Story information should be filtered out. The Sentence Classifier outputs 3 files: one that contains turns that could not be classified and thus have been excluded, one that contains turns that have been predicted to **not** contain User Story information and one that contains turns that have been predicted to contain User Story information. Thus, the 7 files from the Greenfield case and the 10 files from the Customization case resulted in 51 files. The turns in each file have been coded in Nvivo 12 in either a "US information" or a "no US information" category. This way, the True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN) could be counted and the accuracy, precision, recall and F1 scores could be calculated. These scores have been calculated on the Customization Case and the Greenfield Case separately, which can be seen in tables 21 and 22 respectively.

Table 21: Metric scores of the Sentence Classifier on the Customization Case

Metric	**Score**
Accuracy	0.509
Precision	0.360
Recall	0.699
F1	0.475

Table 22: Metric scores of the Sentence Classifier on the Greenfield Case

Metric	**Score**
Accuracy	0.498
Precision	0.180
Recall	0.399
F1	0.248

These results show that the Sentence Classifier performed worse on the Greenfield Case for the precision, recall and F1 metrics. Although further research has to be conducted, it is hypothesised that this difference is due to the nature of the interviews. The interviews from the Customization Case tend to be more structured, with more turns containing information on User Stories. Some of the Greenfield Case interviews spanned a lot of turns but contained not that many turns with User Story information, due to multiple people with different interests being present for the interviews.

The turns that were predicted by the Sentence Classifier to contain User Story information were then used as input for the Word Classifier. The output is analysed both qualitatively and quantitatively in the following sections.

MLC - Qualitative analysis. The results of the MLC configuration are analysed on User Story completeness, User Story quality and traceability.

User Story Completeness. In both Cases no predicted User Story exactly matched a complete Ground Truth User Story. Only fragments were predicted, ranging from 0 to 2 US fragments in a predicted US. In file B2 not a single Ground Truth fragment is predicted.

User Story Quality. The US quality is analysed based on the Quality User Story Framework as described in section 3.3. Some US candidates do not have a Traceability ID, as can be seen by the "NoTurnIndex" label. This goes for both the Customization Case and the Greenfield Case. The reason for this is a bug in the User Story builder that could not be fixed within the time frame for this thesis project. In the Customization Case, four User Story candidates were counted which included at least a role and a goal (*well-formed*). One of these expresses a requirement for exactly one feature (*atomic*). The other three predicted USs ("A3/io/166 A3/io/170 A3/io/172", "A4/io/92 A4/io/94" and "NoTurnIndex") describe more than one feature, as shown below in color. Each of these predicted USs is analysed below:

A3/io/166 A3/io/170 A3/io/172

> *A3/io/166 A3/io/170 A3/io/172*
> *As a I seek for a player I, I can do which is kind of maintaining its own social network page managed within the system, I can I, I can kind of communicate with my fans Player and teams Player and Player and are not adding They dont have their own pages within the*

A3/io/166 A3/io/170 A3/io/172 predicts two features: managing pages within the system and communicating with the fans. However, as can be seen in the Ground Truth US A3.30 below, should communicating with the fans actually be a benefit. This shows that the MLC configuration cannot make a good distinction between what should be classified as a goal and what should be classified as a benefit.

> *A3.30*
> *As a player, I want to have my own social network page, so that I can kind of communicate with my fans.*

> *A3/io/166*
> *I seek for a player, there is only one things that he can do, which*

is kind of maintaining its own social network page that is managed within the system. So we can that way he can kind of communicate with his fans.

Both of the features originate from the original turn A3/io/166. In this case, the User Story builder changed the words "he" and "we" from the original turn into "I" in the predicted turn and changed "his fans" into "my fans". While sometimes this way of embellishing the predicted US can be a good thing, as with the part about owning a social network page, it can also lead to every fragment becoming a goal fragment. Even though, sometimes it should be predicted as a benefit fragment.

A4/io/92 A4/io/94

A4/io/92 A4/io/94
As a and the main referee, I can the time and the main referee, I can finaiize the game want to be finalized the game a month after , I can insert their constraints regarding the scheduling , I can participate in are able to they are available in certain dates are available In certain hours

A4/io/92 A4/io/94 also predicts two features that appear in the Ground Truth USs: finalizing the game and inserting constraints regarding the schedule. The first and the last predicted features ("I can the time and the main referee" and "I can participate in [...]") do not make sense, are not complete and do not appear in a Ground Truth US and are thus ignored. The feature about finalizing the game appears as a benefit fragment in Ground Truth US 4.25. This shows the impact of the RE engineer that makes the Ground Truth USs because this feature could also have been written down as a goal fragment. The feature about constraints appears in Ground Truth US A4.26 as a goal fragment. Thus, this feature is correctly predicted.

A4.25
As a main referee, I want to upload a final report after the game ends, so that the game is finalized.

A4.26
As a referee, I want to insert constraints regarding the schedule, so that I only get scheduled on days that I'm available.

The predicted feature about finalizing the game appears in the original turn A4/io/90. In this case, the User Story builder changed "he" to "I". The predicted feature about inserting constraints appears in the original turn A4/io/94 and here the User Story builder inserted "I" before "can". In these original turns, the MLC configurations predicted the role fragment partly correct, as in both of the original turns the conversation concerned referees. However, the TUSP did miss the goal fragment from Ground Truth US A4.25 about uploading the final report. This shows room for improvement.

A4/io/90
and the main referee might have some kind of final report after the game ends. And this as well can have some certain threshold between the end of the game and till when they can the time he can finalize the the game.

A4/io/94
Excellent question. So there might be an option. So the referee can insert their constraints regarding the scheduling they can participate in. For example, they are able to they are available in certain dates. They are available In certain hours, and then the scheduling should take this into account.

NoTurnIndex

NoTurnIndex
As a doctor, I can request the test and what test type

This predicted US comes from file C3. The predicted US contains a feature to request a test and a feature to request a test type. The feature on requesting a test matches with the benefit fragment of Ground Truth US C3.19, which can be seen below. The second predicted feature, regarding the test type, cannot be found in the Ground Truth USs. Thus, this can be an example of where the RE engineer missed a requirement and the TUSP can complement the list of USs.

C3.19
As a doctor, I want to be connected to the laboratory, so that I can request a test.

Both of the predicted features originate from the original turn C3/ea/117. The User Story builder changed "you" to "I", which in this case created a good prediction. However, the configuration did fail to recognize the possibility of two USs being present. This is indicated by the word "and" (shown in red in the original turn. This shows where an improvement can be made when further developing the MLC.

C3/ea/117
no, I would just be within the system. So for example, the doctor opens his system. And then within the patient file, you can choose laboratory tests, and then you can request the test, and what test type and if it's urgent or by when it needs to be done, and all that information. And that information can be accessed by the laboratory. So the laboratory laboratory system is connected to that. And they then have a list of all the tests that need to be performed. And, you know, as soon as they enter the result, the doctors can see it within that form. Okay, it's not always a new message or email or anything like that.

The predicted US that turned out the best is from file C3:

NoTurnIndex
As a doctor and the doctor, I am able to see if the test is in progress

This User Story candidate is considered *minimal* meaning it contains nothing more than role, goal and reason (even though it contains the same role twice). The goal fragment of the candidate expresses a feature (*conceptually sound*) and is *problem-oriented*. It is also *conflict-free*, *scalable*, *unique* and *independent*. However, it is not *unambiguous* as some doubt could arise about what kind of test is meant. It is also not a *full sentence* in the sense that it is well-formed, the role contains a repetition of the fact that it concerns the doctor.

Regarding all the User Story candidates it can be said that they are not *complete* and do not account for *explicit dependencies* and most of all, are not *uniform*. Some US candidates only consist of a role:

NoTurnIndex
As a the team

And some consist of multiple goals:

A1/io/48 A1/io/50 A1/io/54 A1/io/56 A1/io/58 A1/io/60 A1/io/62 A1/io/64
As a one case case changes only in case of change okay will be part of the policy of the scheduling mentioned your own way in the you should need to be to be able to modify these parameters into the system to be built any the fans would like nowadays is going to the internet to pull the information need to look for a game changes the changed schedule any need to look for it post should look for also, I can register to that Okay who will follow oh follow its like register for each of the information , I can subscribe to it and then get notification

Why the TUSP outputs USs with only is role is not known but it could be that after the role words there was a "linking word". The User Story builder cuts USs in two when such a linking word is found. The US candidate containing multiple goals is because every time the word "can" appears, the User Story builder put ", I " in front of it, creating a new goal fragment.

For the Greenfield Case no candidate was *well-formed*. The predictions for file I3 contained 2 US candidates that were *atomic* and *minimal* but also contained only a goal fragment:

NoTurnIndex
As a , I can test the perception

NoTurnIndex
As a , I can test the logic

These candidates are also *conceptually sound*, *problem oriented*, *conflict-free*, *unique* and *independent*.

Lastly, a bug was noticed in the MLC User Story Builder. To separate the actor fragment from the goal fragment does the MLC US Builder parse in ', I ' if it finds the word 'can' in a sentence. However, this also hits if 'can' is part of a larger word, such as 'cancelled' or 'cancellation'. See an example below:

> *A2/io/48*
> *As a are set changes should be rarely be done example cause the, I cancellation of a game due to weather*

When checking back in the original turn, it shows that the ', I ' comes out of nowhere and should not have been put in:

> *A2/io/48*
> *And again, if something for example, cause the cancellation of a game due to weather conditions or things like that, this should be again, be explicated within the system.*

Traceability. To ensure traceability should every US candidate should have a Turn Traceability ID that refers back to the original turn or turns that the candidate was based on. However, as already shown in the previous examples of this section, not every US candidate outputted by the MLC configuration has a Turn Traceability ID due to a bug present in the User Story Builder. For other US candidates, the MLC configuration gives some appropriate Turn Traceability IDs but also forgets some. An example is given below. This example shows predicted US candidate I1/pa/21 I1/pa/23 I1/pa/25 I1/pa/27 and the original turns the words have been extracted from. The colours show the matches from the predicted US with the original turns. This way one can see where the words classified by the Requirements Word Classifier came from. Some turns have been shortened for readability. An interesting side note is that some turns were skipped between I1/pa/12 and I1/pa/21. The turns following (I1/pa/[21, 23, 25, 27]) all have one turn in between where the RE engineer is talking. But the gap between I1/pa/12 and I1/pa/21. Turn 15, 16, 17, 18 and 20 were excluded by the Requirements Sentence Classifier. Turn 13, 14 and 19 did not contain any words that contained US information.

> *I1/pa/21 I1/pa/23 I1/pa/25 I1/pa/27*
> *As a just run it overnight then have an output of read something about it , I can improve them the model that is used right now in just any dynamic, I can always make it more accurate also have to identify those parameters on the race cars be still open to be to define think add with a CSV file asked us returns a point cloud*
>
> *I1/pa/12*
> *You ever thought would be really nice if we if it could have something*

like that we could store like, three or four different configurations, like with different controller parameters or something, then just run it overnight? And do it like 1000 times and then have an output of Yeah. I don't know, maybe the overall time or the tracking performance? [...]

I1/pa/21
Yeah. So read something about it that it works somewhat, but there are still some problems. Maybe we can improve them.

I1/pa/23
Yeah, it's actually been really difficult because I mean, the model that is used right now in Edinburgh is also just any dynamic model. And you can always make it more accurate. But then you also have to identify those parameters. And that's really depends on the race cars. So first, we actually would have to be able to take some measurements. Yeah, yeah. Accurate? Yeah, it's really hard to answer. But I think the dynamic model that they use so far is already quite good.

I1/pa/25
I think this will be still open to be to define I don't know, I think add with a CSV file. I think that's okay. Yeah.

I1/pa/27
And I mean, maybe that's asked first. And what kind of sensors, I think the most important for us is the LIDAR. LIDAR just returns a point cloud of, of points. I mean, the one we use has, like 32 layers. [...]

This example shows that words have been extracted from turn I1/pa/12, but that this turn is not referenced by the Turn Traceability ID of the candidate turn. Thus, it can be concluded that the MLC configuration has some traceability but that it is not conclusive.

MLC - Quantitative analysis. The resulting files with predicted User Stories were loaded into Nvivo12 so they could be coded and thus quantified. Both the Ground Truth User Stories and the predicted User Stories were split into the separate User Story fragments: *role, goal and benefit*. If a predicted fragment matched a Ground Truth fragment it would be counted as a True Positive. To give an example: if a predicted User Story contained a goal that matched with a goal of a Ground Truth User Story, this would be counted as a True Positive. Since the Word Classifier only predicts positives, only the True Positives and False Positives were counted. To calculate the recall the total number of Ground Truth User Story fragments that appeared in the predicted User Stories were counted as the sum of the True Positives and all the fragments that did

appear in a prediction but as a wrong fragment. For instance, if a fragment was predicted as a role fragment, but in the Ground Truth, it appears as a goal fragment. This sum is then divided by the number of relevant fragments in the Ground Truth. An example of this can be seen in figure 24. The US fragments that matched with a Ground Truth US fragment can be seen, so for instance 4 fragments were predicted as a role fragment but should have been a goal fragment. TP stands for True Positives and FP stands for False Positives. The True False Positives are the total predicted fragments minus the TP because, in theory, every prediction is a positive even if they don't match with a Ground Truth Fragment.

File	Fragment	Role	Goal	Benefit	Total US fragments	Total predicted fragments	TP	FP	True FP	Accuracy:	Precision:	Recall:	F1:
A4	Role	2	4		36	75	2	4	73	0.056	0.027	0.167	0.046
	Goal		1	1	36	32	1	1	31	0.028	0.031	0.056	0.040
	Benefit				36	0	0	0	0	0.000	#DIV/0!	0.000	#DIV/0!

Figure 24: An example of the Excel file used for the MLC analysis.

The total number of Ground Truth User Story fragments per case and the total number of predicted User Story fragments per case are shown in table 23. The accuracy, precision, recall and F1 metrics per file are depicted in table 24. There are also tables where the metrics per US fragment are shown, these can be found in appendix F. The accuracy, precision, recall and F1 metrics per case are shown in table 25.

Table 23: Number of Ground Truth User Story fragments against the number of predicted User Story fragments that resulted from the MLC configuration

Case	**Ground Truth US**	**Predicted US**
Customization	1074	571
Greenfield	384	362

Table 24: Quantitative metrics of the MLC configuration per file. US frag. are the Ground Truth US fragments, Pred. frag. are the predicted US fragments. Acc. is accuracy, Pre. is precision and Rec. is recall.

File	**GT frag.**	**Pred. frag.**	**TP**	**FP**	**Acc.**	**Pre.**	**Rec.**	**F1**
A1	132	107	1	1	0.008	0.009	0.015	0.012
A2	114	48	0	4	0	0	0.035	0

continues on next page

A3	111	65	3	3	0.027	0.046	0.054	0.050
A4	108	107	3	5	0.028	0.028	0.074	0.041
B1	84	38	0	1	0	0	0.012	0
B2	54	28	0	0	0	0	0	0
C1	78	37	1	1	0.013	0.027	0.026	0.026
C2	126	53	3	0	0.024	0.057	0.024	0.034
C3	129	37	2	5	0.016	0.054	0.054	0.054
C4	138	51	0	4	0	0	0.029	0
I1	81	47	0	1	0	0	0.012	0
I2	24	37	0	0	0	0	0	0
I3	126	68	1	0	0.008	0.015	0.008	0.010
I4	51	100	0	2	0	0	0.039	0
I5	36	55	2	0	0.056	0.036	0.056	0.044
I6	24	51	1	4	0.042	0.020	0.208	0.036
S1	42	4	1	0	0.024	0.250	0.024	0.043

The numbers in table 24 show that files I5 and I6 had relatively high accuracy. An explanation could be that there were relatively few Ground Truth US fragments for these files (36 and 24 respectively). When compared to low scoring files such as A1 (132 Ground Truth US fragments) and I3 (126 Ground Truth US fragments). The files A1, I3, I5 and I6 all had about the same amount of True Positives, 1, 1, 2 and 1 respectively. This could thus have led to lower accuracy. It also means that files with a smaller amount of Ground Truth User Stories produce a higher accuracy. The survey (S1) had the highest precision, again due to small numbers. The TUSP had predicted 4 US fragments, from which 1 was correct. This led to a precision number of 0.250. The next best precision comes from file C2. This number is relatively high because of a relatively high number of True Positives (3) and a relatively low number of total predicted fragments (53). The highest recall is reached by file I6. Again this is due to a low number of Ground Truth US fragments and a relatively high number of TPs and FPs (number of fragments predicted that match a Ground Truth fragment but appear in the wrong place). Thus these results show the importance of the Ground Truth USs on the performance when measured with these the metrics used in this validation. However, file I2 also performed bad, even with a small number of Ground Truth USs. A possible explanation for this is that this was a meeting with a lot of people present where not only requirements were discussed. This created a lot of turns containing no US information and the TUSP probably had trouble finding the important parts. The files A2, B1, B2, C4, I1, I2 and I4 all perform badly. Why this performance is that bad

for these files is hard to pinpoint as the files are all different and with not 1 obvious common trait that infers with the TUSP performance.

Table 25: Quantitative metrics of the MLC configuration per case

Case	Accuracy	Precision	Recall	F1
Customization	0.012	0.023	0.034	0.027
Greenfield	0.013	0.014	0.031	0.019

Using a two-sided t-test with unequal variances and an alpha of 0.05 shows no significant difference in the accuracy, precision, recall and F1 numbers of the two cases.

5.3.2 Results of the Lexical Configuration

LC - Qualitative analysis. The results of the LC configuration are analysed on User Story completeness, User Story quality and traceability.

User Story Completeness. In both Cases no predicted User Story exactly matched a complete Ground Truth User Story. File I2 produced no correct predictions for any Ground Truth fragment. However, in some cases, the predicted US contained all the elements of the matching Ground Truth US but were these elements not placed in their correct respective US fragment. An example is the predicted US B2/hu/41:

> *B2/hu/41*
> *As a head of urban planning, I want to tell people I need to be able to convince everyone that we rely on real data I cannot tell them we just invented how many cars are moving around , so that I need*

Which matches Ground Truth US B2.8:

> *B2.8*
> *As a head of urban planning, I want to convince everybody that we rely on real data, so that I show that we didn't just invent how many cars are moving around.*

In this case, the role was correctly predicted, the predicted goal contains both the Ground Truth goal fragment and the Ground Truth reason fragment and the predicted reason contains nonsense. However, concerning a high recall and filtering all the relevant information from a whole transcript, B2/hu/41 can be seen as a good prediction. The same goes for C3/ea/123:

> *C3/ea/123*
> *As a the patient , I want to be able to schedule their appointments for example So with their login they should be able to so the appointment*

scheduling part should be integrated with the whole patient account or access And they should also be able to access their medical records so to see , so that for example So with their the whole patient account so to see

Which contains all the elements of the Ground Truth US C3.40:

C3.40
As a patient, I want to access the appointment scheduling, so that I can schedule my own appointment.

In this case, some redundant information needs to be scrapped from the predicted US and again does the predicted goal contain the information on both the Ground Truth goal and the Ground Truth reason.

User Story Quality. The US quality is analysed based on the Quality User Story Framework as described in section 3.3. For the Customization Case were 12 predicted User Stories *well-formed*. 8 of these 12 predicted User Stories only contain a requirement for 1 feature and are thus *atomic*. The description for a *minimal* US is that the US contains nothing more than a role, goal and reason. But since there are no User Stories that contain all three of these elements correctly the choice has been made to filter the predicted User Stories on whether the role and the goal are minimal. This leaves 4 of the 8 predicted User Stories that are *well-formed*, *atomic* and *minimal*:

A2/io/99

A2/io/99
As an IFA information officer, I want to be a mechanism for controlling the access of the other people indeed , so that access of the other people indeed

The predicted US A2/io/99 matches with the Ground Truth US A2.36:

A2.36
As an IFA chief information officer, I want a mechanism for controlling the access of other people, so that different stakeholders can not access the whole system.

This shows that the LC is pretty good at extracting goal US fragments, but has more trouble with extracting a good benefit US fragment.

C2/ea/45

C2/ea/45
As an enterprise architect, I want that the data is secure secure that every time leaves the computer the need to be locked out, so

that whats important is every time someone leaves the computer the tablet the phone whenever you need to be locked out So especially for nurses

This predicted US matches with two Ground Truth USs:

C2.16
As an enterprise architect, I want all data to be secure, so that all privacy issues are considered.

C2.17
As an enterprise architect, I want that every time someone leaves their computer, tablet or phone that person is locked out, so that security of the system is ensured.

In this case, the LC had trouble differentiating between two separate USs. This is a problem that happens more often: an actor lists multiple requirements and the TUSP extracts these as one requirement.

C3/ea/22

C3/ea/22
As an enterprise architect, I want definitely support mobile devices Because , so that some doctors do have tablets It should also support different devices with stationary computers we have tablets will be nice to be able to access that from the phone But we also want to integrate or to allow patients to use

The predicted US above matches with Ground Truth US C3.4:

C3.4
As an enterprise architect, I want the system to support mobile devices, so that I can access it from my phone.

In this case, do the role and goal fragments match but in the benefit fragment another Ground Truth US is mentioned:

C3.3
As an enterprise architect, I want the system to support stationary devices, so that I can access it from my pc.

Again, this shows that the LC captures too much of what an actor says and creates one US instead of two. The original turn below shows why two USs have been captured:

C3/ea/22
Yeah, so it should definitely support mobile devices. Because some doctors do have tablets. It should also support different devices with

stationary computers, we have tablets, will be nice to be able to access that from the phone. But we also want to integrate or to allow patients to use it, at least part of it. So it needs to be something that is compatible, also for the users at home. So I assume that would be web based, then

The yellow marked words indicate the trigger words for the goal part of the predicted US. The cyan word indicates the trigger word for the benefit fragment of the predicted US. The two hits were originally overlapping if that happens, the LC will cut one part from the other and use the shortest sentence. This can be seen by the trailing "because" in the goal fragment of the predicted US.

C4/ea/54

C4/ea/54
As an enterprise architect, I want to the system to be available on mobile devices , so that has kind of two sides So once we need

This predicted US matches with the Ground Truth US C4.25:

C4.25
As an enterprise architect, I want the system to be available on mobile devices, so that I can use it on my phone or tablet.

In this case, the LC did capture exactly one requirement but still failed to capture the correct benefit fragment.

None of these four is *conceptually sound* as all four contain a reason fragment that describes something else than a rationale. US candidates A2/io/99, C3/ea/22 and C4/ea/54 can all be seen as *problem-oriented*, whereas C2/ea/45 already gives a solution to the requirement of being secure. A2/io/99 is not *unambiguous* as it is not clear what the thing is that access should be restricted to. C3/ea/22 is not *unambiguous* as it is not clear what should be supported by the mobile devices. All four of the candidates are *conflict-free*, *unique* and *independent*. However, none of the candidates is a *full sentence* as they all contain grammatical errors and loose words or uncompleted sentences. Except for C2/ea/45 are all candidates reasonably *scaleable*, 'I want that the data is secure' (C2/ea/45) can be seen as too course-grained. All of the candidates are *uniform* in the sense that they follow the same template as the Ground Truth User Stories: the role starts with 'as a ...', the goal starts with 'I want ...' and the reason starts with 'so that ...'. All of the candidates together can not be seen as *complete* and no *explicit dependencies* have been accounted for.

The results of the Greenfield Case contain one candidate that contains both a role and a goal and that is thus *well-formed*:

I5/ae/26
As an autopilot engineer, I want just the bit the dynamic model the

adapted to our is to just the , so that we adapted to our model car to Yeah just some the kinematic model is to just the weight and mass inertia and these things

This candidate is also *atomic*, *problem-oriented*, *conflict-free*, *scalable*, *unique* and *uniform*. It is not *conceptually sound* as the reason expresses something else than a rationale. It is also not *unambiguous* as it is not clear what the dynamic model should be of. The candidate is not made up of *full sentences* because of the grammar errors and halve sentences the candidate is made up of.

All of the candidates together are not *complete* not do not contain *explicit dependencies*.

Traceability. In both of the cases do all of the US candidates contain a Turn Traceability ID. For instance US candidate A1/re/120:

A1/re/120
As an else fan , I want register to the system We are looking for internet so we can be able to get notification Otherwise it will be just a guest , so that system be just a guest

Refers back to the original turn in file A1, where all the elements from the candidate can be found:

A1/re/120
else? So, a fan should register to the system. We are looking for internet so we can be able to get notification Otherwise, it will be just a guest

This same example can be used to show how the Fit-Gap Searcher and the Word Classifier work together in the LC configuration. It shows that for the role, the Word Classifier predicted the words 'else' and 'fan' to be a role. The Fit-Gap Searcher got a hit on 'should' and labelled all the following words to be part of the goal fragment. Finally, did the Word Classifier predict the words 'system', 'be', 'just', 'a' and 'guest' to be of a reason fragment. To give an example that only uses the Fit-Gap Searcher please see the following US candidate:

B1/hu/48
As a head of urban planning, I want to be able to take a more informed tto ake more informed decisions based on real data real time data Yes I want to be able to say the reason , so that why I suggest making these roads one way or inverting the one way direction or whatever is

Which is extracted from the following turn, which is shortened for readability:

B1/hu/48
Well I want to be able to take a more informed tto ake more informed decisions based on real data, real time data? Yes, I want to be able to say, the reason why I suggest making these roads, one way or inverting the one way direction, or whatever, is that, you know, we created the reason we can simulate it, [...]

In this case was the role extracted from the actor abbreviation in the turn index, both shown in yellow. The Fit-Gap Searcher had hits on 'I want' and 'the reason' (both shown in green) for the goal and the reason respectively. This led to the Fit-Gap Searcher extracting the red part for the goal and the cyan part for the reason.

LC - Quantitative analysis. The resulting files with predicted User Stories were loaded into Nvivo12 so they could be coded and thus quantified. Both the Ground Truth User Stories and the predicted User Stories were split into the separate User Story fragments: *role, goal and benefit*. If a predicted fragment matched a Ground Truth fragment it would be counted as a True Positive. The metrics were calculated in the same way as for the MLC configuration. An example of this can be seen in figure 25. The US fragments that matched with a Ground Truth US fragment can be seen, so for instance 4 fragments were predicted as a goal fragment but should have been a benefit fragment. TP stands for True Positives and FP stands for False Positives. The True False Positives are the total predicted fragments minus the TP because, in theory, every prediction is a positive even if they don't match with a Ground Truth Fragment.

File	Fragment	Role	Goal	Benefit	Total US fragments	Total predicted fragments	TP	FP	True FP	Accuracy:	Precision:	Recall:	F1:
C3	Role	4			43	34	4	0	30	0.093	0.118	0.093	0.104
	Goal		10	4	43	34	10	4	24	0.233	0.294	0.326	0.309
	Benefit			2	43	34	2	0	32	0.047	0.059	0.047	0.052

Figure 25: An example of the Excel file used for the LC analysis.

The total number of Ground Truth User Story fragments per case and the total number of predicted User Story fragments per case are shown in table 26. Notable is that for the Greenfield case almost double the amount of US fragments were predicted, compared to the number of Ground Truth US fragments. It seems like a lot of trigger words were used but not all these trigger words were connected to USs. The accuracy, precision, recall and F1 metrics per file are depicted in table 27. A more in-depth version of this table can be found in appendix G, where the scores per US fragments are depicted. The accuracy, precision, recall and F1 metrics of the two cases as a whole can be found in table 28.

Table 26: Number of Ground Truth User Story fragments against the number of predicted User Story fragments that resulted from the LC configuration

Case	Ground Truth US	Predicted US
Customization	1074	1047
Greenfield	384	735

Table 27: Quantitative metrics of the LC configuration per file. US frag. are the Ground Truth US fragments, Pred. frag. are the predicted US fragments. Acc. is accuracy, Pre. is precision and Rec. is recall.

File	GT frag.	Pred. frag.	TP	FP	Acc.	Pre.	Rec.	F1
A1	132	93	7	2	0.053	0.075	0.068	0.072
A2	114	135	11	1	0.096	0.081	0.105	0.092
A3	111	108	4	1	0.036	0.037	0.045	0.041
A4	108	147	8	5	0.074	0.054	0.120	0.075
B1	84	141	7	0	0.083	0.050	0.083	0.062
B2	54	60	3	2	0.056	0.050	0.093	0.065
C1	78	51	10	0	0.128	0.196	0.128	0.155
C2	126	159	10	6	0.079	0.063	0.127	0.084
C3	129	102	16	4	0.124	0.157	0.155	0.156
C4	138	51	4	2	0.029	0.078	0.043	0.056
I1	81	63	4	0	0.049	0.063	0.049	0.056
I2	24	171	0	0	0	0	0	0
I3	126	162	5	1	0.040	0.031	0.048	0.037
I4	51	117	2	1	0.039	0.017	0.059	0.026
I5	36	66	2	1	0.056	0.030	0.083	0.044
I6	24	99	2	0	0.083	0.020	0.083	0.033
S1	42	57	3	0	0.071	0.053	0.071	0.061

Files C1 and C3 had the best overall scores on these metrics and with file C3 having the most TP hits (16) of all the files. Apparently, the client in this interview used a lot of trigger words on which the Fit-Gap Searcher could have hits. This configuration performed very badly on file I2, with no TPs at all.

This shows that the configuration does not work well on a meeting with a lot of people where other than requirements are discussed as well.

Table 28: Quantitative metrics of the LC configuration per case

Case	Accuracy	Precision	Recall	F1
Customization	0.074	0.076	0.096	0.085
Greenfield	0.047	0.024	0.055	0.034

Using a two-sided t-test with unequal variances and an alpha of 0.05 shows no significant difference in the accuracy, precision, recall and F1 numbers of the two cases. The results above show that for the Customization Case almost as many US fragments were predicted as there were Ground Truth fragments. For the Greenfield Case, however, almost double the fragments were predicted than were possible. A possible explanation is that even though actors were not talking about requirements, there were possibly still using a lot of trigger words on which the Fit-Gap searcher had a hit.

5.4 Conclusion of the Transcript to User Story Pipeline validation

Both the LC configuration and the MLC configuration did not predict any complete User Story that exactly matched a Ground Truth User Story. However, both configurations did extract several US fragments that also appeared in the Ground Truth User Stories. Both of the configurations also suggested some User Stories that had some US quality components, but no perfect US was found. The main difference between the two configurations was that the LC had perfect traceability and the MLC has cases where original turns are not registered. Also, did the MLC perform better on small transcripts with not a lot of potential USs to be extracted. For the LC the size of the transcript and the number of Ground Truth USs seemed to matter less. Quantitatively do both configurations score low on accuracy, precision, recall and F1. The reason for this is the low amount of True Positives, or in other words, good predictions, combined with a large amount of Ground Truth US fragments and large amounts of predicted fragments that contained no useful information. There was no significant difference between any of the metrics for the two configurations. Both of the configurations performed very bad on file I2, which shows that the way a meeting is held matters. More research needs to be done but there is an indication that doing a meeting where other subjects besides requirements are discussed can negatively impact the performance of the TUSP.

6 Discussion, future recommendations and conclusion

6.1 Discussion

This chapter will handle possible threats to the validity of the research done. There was some ambiguity when labelling the test files for the comparison of the two sentence classifiers. There were no guidelines written down by Ruiz and Hasselman (2020) and thus it was assessed by the author himself if a turn contained no requirement information, functional requirement information or non-functional requirement information. This also poses the next challenge which is that the definition of a non-functional requirement still varies depending on what literature is checked. The training data for both the Deep Learning Sentence Classifier and the Machine Learning Sentence Classifier was already labelled, while some of the validation data had to be relabelled to fit the for needed to be used as input. This could have led to cases with both of the classifiers where wrong classifications are the result of a different way of labelling instead of how well the classifier performs. This is thus a threat to internal validity. The biggest threat to internal validity is how the RE interviews were conducted. There is a possibility that if the interviews were held differently or with different questions, the results would have been different. Therefore it is impossible to say if the results are causally produced by the TUSP. The second threat to internal validity is that the USs that were used as a Ground Truth have not been checked by an independent source. Since the User Stories were made after the construction of the TUSP, these are susceptible to bias. These Ground Truth USs had a big impact on both the quantitative and qualitative validation of TUSP. In the same spirit has the validation of the US candidates not been checked by an independent source. This makes the validation susceptible to confirmation bias. For instance, the author can label a fragment as a correct prediction of a Ground Truth User Story fragment, while in reality, this was not what the actor talking during the turn meant.

The biggest threat to external validity is that the TUSP has been validated in only 2 contexts. To get a better idea of how generalizable the TUSP is, it is recommended to apply the TUSP to more contexts. Another threat to external validity is that the TUSP has not been compared with other tools. However, since the TUSP is so innovative and at the moment there do not exist any other tools that encompass the whole process from transcript to User Story it is not possible to compare the TUSP with other tools at the moment.

6.2 Future recommendations

A lot of improvements to the TUSP can be done for future research. First, the points of improvement on the TUSP itself will be discussed and afterwards, there will be some suggestions for combining the TUSP with other tools.

6.2.1 Improvements on the TUSP

The following list contains points that can be used as a way of improving the qualitative and quantitative performance of the TUSP. The main points are listed below:

- **Research on other Requirements Engineering methods for eliciting requirements with multiple stakeholders:** The used elicitation methods for this thesis, interviews and meetings, have proven to be very unstructured. Therefore, it can be worthwhile to research elicitation methods which follow a structured question-answer format. Or perhaps a method which follows a more strict role-goal-reason format. This could help ML classifiers with training in a more structured manner or lexical classifiers with finding more precise hits.

- **Use external databases for giving semantic information on words:** There are readily available online databases such as Wordnet[10] for lexical information and Framenet[11] for semantic information. Using lexical and semantic information could lead to the algorithm's better understanding of the words used in a transcript and the relationships between these words. Because as of right now do words not have any value or meaning for an algorithm. Research into combining these databases with the TUSP or TUSP components could show if these databases could lead to improvement.

- **Take environmental impacts into account:** Training a neural network for NLP can be very computationally intensive and thus cost a lot of energy, as Strubell et al. (2019) show in their paper. As the ELMO model from the Word Classifier cannot be saved and thus has to be trained every time the TUSP runs, it is not hard to imagine that over time an environmental impact is created which has to be taken seriously. Rule-based models, such as the Fit-Gap Searcher do not require any form of training and can therefore be a solution when one wants to look for an alternative instead of machine-learning-based models.

- **Focus on unsupervised models:** If one chooses to ignore the previous point due to various reasons, it is recommended to focus on unsupervised learning models.

- **Allow the Concept Extractor usage of .owl format files:** Currently can the concept extractor only read known concepts from a .txt format file. The .owl format is the most often used type for writing down ontologies. If the concept extractor is modified to use .owl format files it could be used as it was intended: as an ontology crawler which can get an ontology as input and output all the matching concepts.

[10] https://wordnet.princeton.edu/
[11] https://framenet.icsi.berkeley.edu/fndrupal/

- **Process multiple turns simultaneously:** Whereas the MLC configuration can create User Story candidates from multiple turns, is the Fit-Gap Searcher currently not able to. Being able to process multiple turns is for instance important when the RE engineer rewords a requirement or gives an example and asks for a confirmation. Whether the rewording or example can then be seen as a new requirement then depends on the answer of the client.

- **Process multiple requirements in one turn:** In opposition to the previous point refers this point to when a turn contains multiple requirements. For instance, if a client lists multiple small features or if the client lists the most important features of the system to-be.

- **Research on Fit-Gap trigger words:** The current Fit-Gap Searcher is based on the research by Spijkman, Dalpiaz, et al. (2021). For this research, the authors analysed 12 hours of RE meetings at the company of one of the authors. This means that the fit-gap trigger words extracted are context-dependent on the company that these words were extracted from. Thus, more research on other RE meetings in other contexts could perhaps lead to more trigger words which can in turn be used to improve the Fit-Gap Searcher.

6.2.2 Future vision for the TUSP

The TUSP has been designed to function as a tool to aid the Requirements Engineer in the extraction of User Stories from transcripts. However, the User Stories and User Story fragments produced by the TUSP are not the finished product. The output from the TUSP can be used in issue-tracking systems and project management systems such as Jira[12] and Backlog[13] or perhaps a self-developed system such as Storyscreen[14], which is made by students from the ZHAW Zurich specifically for managing User Stories. For instance, an idea can be that the TUSP gets hooked up to the back-end of the Storyscreen. Thus, when a transcript gets uploaded, all User Story candidates would automatically be extracted and uploaded to the Storyscreen where they would appear in the backlog as can be seen in figure 26.

[12] https://www.atlassian.com/nl/software/jira
[13] https://www.backlog.com/
[14] https://www.storyscreen.ch/

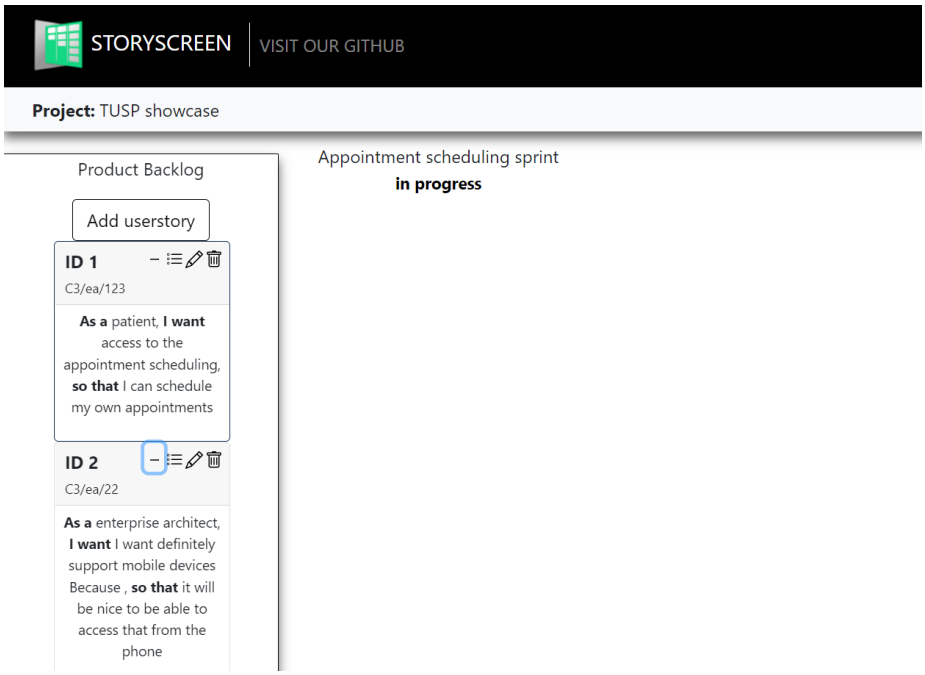


Figure 26: Example of TUSP results loaded into Storyscreen

Once all the User Story candidates have been loaded into the backlog these can be filtered and adjusted and finally they can be dragged into the appropriate sprint column as can be seen in figure 27.

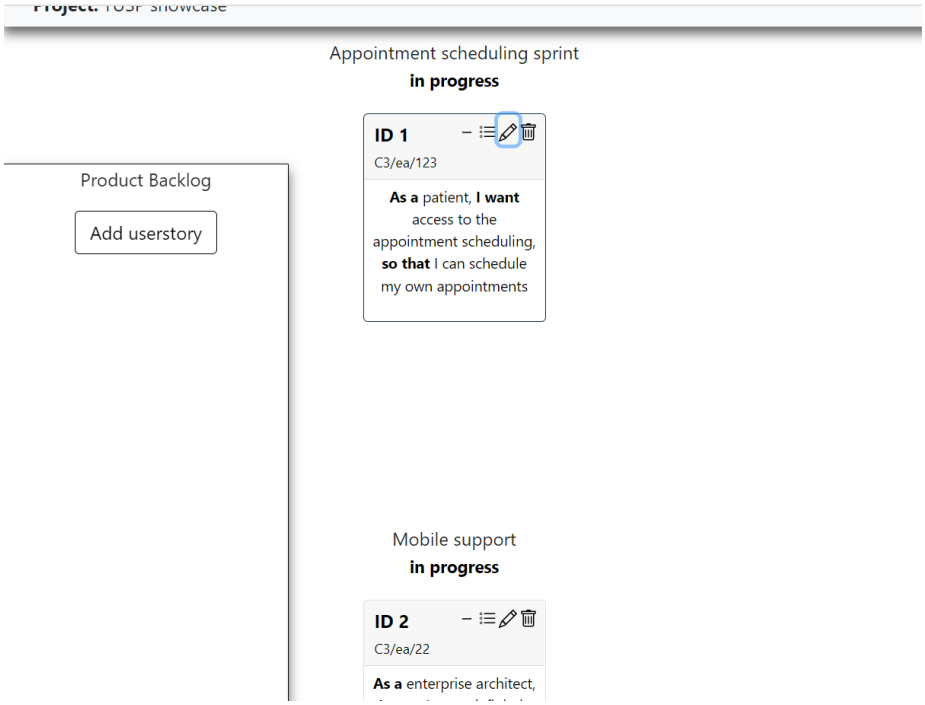


Figure 27: Example of TUSP results used in Storyscreen

Another example is being able to automatically load the US candidates into Jira. In this case, the candidates would appear in the To-Do list, as can be seen in figure 28. Ultimately it should be possible to automatically add the Epics (in this example 'appointment scheduling' and 'mobile support') to which the US candidates belong.

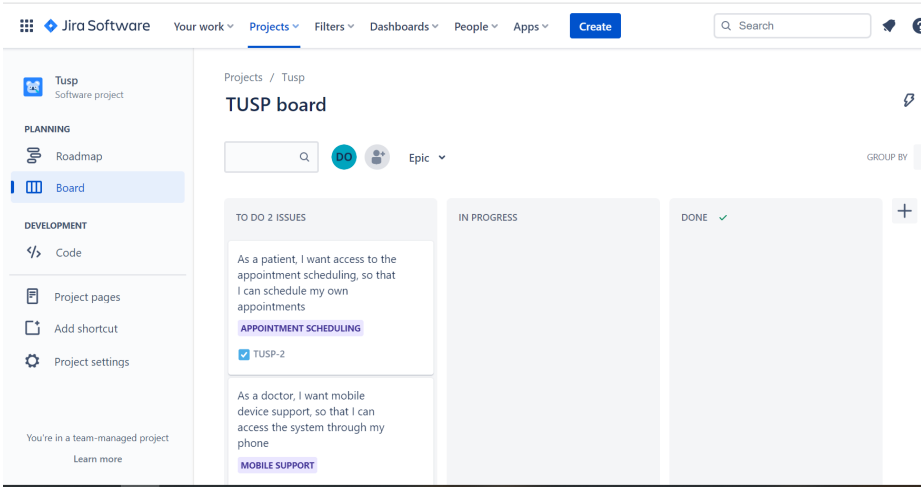


Figure 28: Example of TUSP results loaded into Jira

Regarding traceability, it is envisioned that the TUSP will output a set of US candidates, which can be compared against the original transcript to show where the US or US fragment originates from. A mock-up of a program where the TUSP results can be shown on the left and the original text is shown on the right can be seen in figure 29. If the user then hovers with their mouse over a US fragment it is shown by highlighting where the fragment originates from.

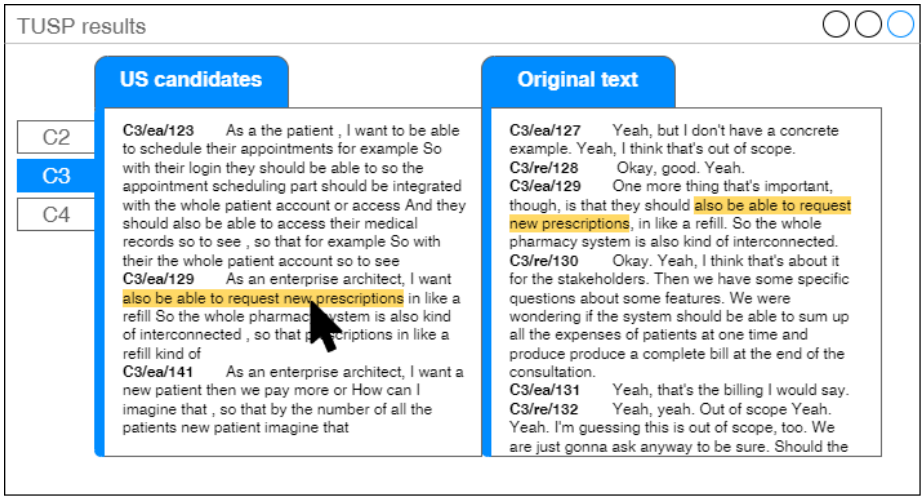


Figure 29: TUSP mock-up.

Finally, there is a vision to track changes in User Stories over multiple interviews. This would mean being able to recall and combine with US candidates

from previous interviews when analysing a new interview. The example above would then show the traces to multiple transcripts and show how the US has evolved.

6.3 Conclusion

First off, **RQ1** was answered in the problem investigation phase. It has been shown that understanding and correctly eliciting requirements is of importance when developing a system. Writing these requirements down in the standardized form of a User Story can help to understand the requirements. User Stories can also be assessed on quality with the help of the Quality User Story Framework. This all answered **SQ1.1**. For **SQ1.2** have multiple requirement elicitation methods been discussed and the decision was made to focus on interviews and meetings. Traceability was discussed to answer **SQ1.3** and it was shown how traceability could benefit the system next to how US traceability could look like. Finally, for **SQ1.4** were multiple Requirement Specification Algorithms discussed, however, none of these tried to elicit requirements through the use of a pipeline, where multiple components are fitted together.

Secondly, **RQ2** was answered in the design phase. At the start of the design phase, the Turn Traceability IDs were discussed, which made it possible to reference back to a turn in a specific document and to show which actor talked during that turn. A Deep Learning sentence classifier and a Machine Learning sentence classifier were compared by tests on the same data to answer **SQ2.1** and the choice has been made to continue with the Deep Learning sentence classifier for building the TUSP. The Concept Extractor and Word Classifier components were improved. The Fit-Gap Searcher was developed as a lexical, hard-coded alternative to the Machine Learning based Word Classifier. In the end, were the components split into two configurations: the Machine Learning Configuration and the Lexical Configuration. This answered **SQ2.2**.

Thirdly, was **RQ3** answered in the validation phase. The validation phase showed that both of the configurations performed poorly on the two test cases. Only a few User Story candidates were found that contained multiple good User Story qualities as described in the Quality User Story Framework and those only consisted of User Story fragments. Not a single whole User Story was found for both the configurations in any of the test cases. None of the configurations had any metric scores above the 0.1 for both of the test cases. Thus were both **SQ3.1** and **SQ3.2** answered.

In conclusion, there is a lot that can be improved on the TUSP. However, it has been shown that the use of a pipeline with multiple components to extract User Stories from software Requirements Engineering meeting transcripts can result in multiple correct User Story fragments which can assist in the RE process, thus answering the Main Research Question.

Special thanks to my supervisors Sjaak, Marcela, Fabiano and Tjerk for their input and ideas, to Fabiano in particular for sharing the course information with me, to Simon for helping me with the Ubuntu server and to the FS ZHAW team for temporarily accepting me into the team so that I could interview them.

References

Abualhaija, S., Arora, C., Sabetzadeh, M., Briand, L. C., & Traynor, M. (2020). Automated demarcation of requirements in textual specifications: A machine learning-based approach. *Empirical Software Engineering*, *25*(6), 5454–5497.

Anton, A. I. (1997). *Goal identification and refinement in the specification of software-based information systems*. Georgia Institute of Technology.

Arora, C., Sabetzadeh, M., Briand, L., & Zimmer, F. (2016). Automated extraction and clustering of requirements glossary terms. *IEEE Transactions on Software Engineering*, *43*(10), 918–945.

Badger, D., Nursten, J., Williams, P., & Woodward, M. (2000). Should all literature reviews be systematic? *Evaluation & Research in Education*, *14*(3-4), 220–230.

Bano, M., Zowghi, D., Ferrari, A., Spoletini, P., & Donati, B. (2018). Learning from mistakes: An empirical study of elicitation interviews performed by novices. *2018 IEEE 26th International Requirements Engineering Conference (RE)*, 182–193.

Booch, G. (2018). The history of software engineering. *IEEE Software*, *35*(5), 108–114.

Cambridge Dictionary. (2019a). Constraint — meaning in the cambridge english dictionary. *Cambridge.org*. Retrieved March 17, 2021, from https://dictionary.cambridge.org/dictionary/english/constraint

Cambridge Dictionary. (2019b). Property — meaning in the cambridge english dictionary. *Cambridge.org*. Retrieved March 17, 2021, from https://dictionary.cambridge.org/dictionary/english/property

Cambridge Dictionary. (n.d.-a). Noun phrases. *Cambridge.org*. Retrieved September 29, 2021, from https://dictionary.cambridge.org/de/grammatik/britisch-grammatik/noun-phrases

Cambridge Dictionary. (n.d.-b). Prepositional phrases. *Cambridge.org*. Retrieved September 29, 2021, from https://dictionary.cambridge.org/de/grammatik/britisch-grammatik/prepositional-phrases

Cambridge Dictionary. (n.d.-c). Verb phrases. *Cambridge.org*. Retrieved September 29, 2021, from https://dictionary.cambridge.org/us/grammar/british-grammar/verb-phrases

Chung, L., & do Prado Leite, J. C. S. (2009). On non-functional requirements in software engineering. *Conceptual modeling: Foundations and applications* (pp. 363–379). Springer.

Cleland-Huang, J. (2012). Traceability in agile projects. *Software and systems traceability* (pp. 265–275). Springer.

Cleland-Huang, J., Gotel, O. C., Huffman Hayes, J., Mäder, P., & Zisman, A. (2014). Software traceability: Trends and future directions. *Future of software engineering proceedings* (pp. 55–69).

Cronin, P., Ryan, F., & Coughlan, M. (2008). Undertaking a literature review: A step-by-step approach. *British journal of nursing*, *17*(1), 38–43.

Dalpiaz, F., & Brinkkemper, S. (2018). Agile requirements engineering with user stories. *2018 IEEE 26th International Requirements Engineering Conference (RE)*, 506–507.

Davis, A. M. (1993). *Software requirements: Objects, functions, and states*. Prentice-Hall, Inc.

Dick, J., Hull, E., & Jackson, K. (2017). *Requirements engineering*. Springer.

Espinoza, A., & Garbajosa, J. (2011). A study to support agile methods more effectively through traceability. *Innovations in Systems and Software Engineering*, *7*(1), 53–69.

Fontaine, C., Haarman, A., & Schmid, S. (2006). The stakeholder theory. *Edlays education*, *1*(4), 1–33.

Ford, C. E., Fox, B. A., & Thompson, S. A. (2002). *The language of turn and sequence*. Oxford University Press.

Formula Student ZHAW. (n.d.). Formula student zhaw – racing. Retrieved May 6, 2021, from https://fszhaw.ch/en/homepage/

Freeman, R. E. (2010). *Strategic management: A stakeholder approach*. Cambridge university press.

Glinz, M. (2005). Rethinking the notion of non-functional requirements. *Proc. Third World Congress for Software Quality*, *2*, 55–64.

Glinz, M. (2007). On non-functional requirements. *15th IEEE International Requirements Engineering Conference (RE 2007)*, 21–26.

Gotel, O., Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grünbacher, P., Dekhtyar, A., Antoniol, G., Maletic, J., & Mäder, P. (2012). Traceability fundamentals. *Software and systems traceability* (pp. 3–22). Springer.

Gotel, O., & Finkelstein, C. (1994). An analysis of the requirements traceability problem. *Proceedings of IEEE International Conference on Requirements Engineering*, 94–101.

Hutchby, I., & Wooffitt, R. (2008). *Conversation analysis*. Polity.

IEEE Standards Coordinating Committee. (1990). Ieee standard glossary of software engineering terminology (ieee std 610.12-1990). los alamitos. *CA: IEEE Computer Society*, *169*, 132.

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. *Computers in human behavior*, *51*, 915–929.

Institution of Mechanical Engineers. (n.d.). Formula student - uk competition faqs. *www.imeche.org*. Retrieved May 6, 2021, from https://www.imeche.org/events/formula-student/team-information/faqs

Jacobson, I., Booch, G., & Rumbaugh, J. (1999). The unified software development process.

Keller, S., Lütolf, J., & Ruiz, M. (2020). Project work it ai for automatic creation of user stories.

Keller, S., Lütolf, J., & Ruiz, M. (2021). From requirements to code: Visualising traceability links among software artefacts for self-driving simulation tools.

Khan, S., Dulloo, A. B., & Verma, M. (2014). Systematic review of requirement elicitation techniques. *India*.

Kotonya, G., & Sommerville, I. (1998). *Requirements engineering: Processes and techniques*. John Wiley & Sons, Inc.

Lee, C., Guadagno, L., & Jia, X. (2003). An agile approach to capturing requirements and traceability. *Proceedings of the 2nd International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE 2003)*, *20*.

Li, F.-L., Horkoff, J., Mylopoulos, J., Guizzardi, R. S., Guizzardi, G., Borgida, A., & Liu, L. (2014). Non-functional requirements as qualities, with a spice of ontology. *2014 IEEE 22nd International Requirements Engineering Conference (RE)*, 293–302.

Lipton, Z. C., Elkan, C., & Narayanaswamy, B. (2014). Thresholding classifiers to maximize f1 score. *ArXiv*, 1402–1892.

Lucassen, G., Dalpiaz, F., Van Der Werf, J. M. E., & Brinkkemper, S. (2015). Forging high-quality user stories: Towards a discipline for agile requirements. *2015 IEEE 23rd international requirements engineering conference (RE)*, 126–135.

Lucassen, G., Dalpiaz, F., van der Werf, J. M. E., & Brinkkemper, S. (2016). The use and effectiveness of user stories in practice. *International working conference on requirements engineering: Foundation for software quality*, 205–222.

Macaulay, L. A. (2012). *Requirements engineering*. Springer Science & Business Media.

Madanayake, R., Dias, G., & Kodikara, N. (2017). Transforming simplified requirement in to a uml use case diagram using an open source tool. *International Journal of Computer Science and Software Engineering*, *6*(3), 61.

Mead, N. R. (2013). A history of the international requirements engineering conference (re) re@ 21. *2013 21st IEEE International Requirements Engineering Conference (RE)*, 21–221.

Miner, G., Elder IV, J., Fast, A., Hill, T., Nisbet, R., & Delen, D. (2012). *Practical text mining and statistical analysis for non-structured text data applications*. Academic Press.

Mund, J., Fernandez, D. M., Femmer, H., & Eckhardt, J. (2015). Does quality of requirements specifications matter? combined results of two empirical studies. *2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 1–10.

Mylopoulos, J., Chung, L., & Nixon, B. (1992). Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Transactions on software engineering*, *18*(6), 483–497.

Ncube, C. (2000). *A requirements engineering method for cots-based systems development* (Doctoral dissertation). City University London.

Nishida, T. (2008). *Conversational informatics: An engineering approach* (Vol. 9). John Wiley & Sons.

Paech, B., & Kerkow, D. (2004). Non-functional requirements engineering-quality is essential. *10th International Workshop on Requirments Engineering Foundation for Software Quality*.

Panichella, S., & Ruiz, M. (2020). Requirements-collector: Automating requirements specification from elicitation sessions and user feedback. *2020 IEEE 28th International Requirements Engineering Conference (RE)*, 404–407.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Pohl, K. (1994). The three dimensions of requirements engineering: A framework and its applications. *Information systems*, *19*(3), 243–258.

Ramamoorthy, C. V., & Li, H. F. (1977). Pipeline architecture. *ACM Computing Surveys (CSUR)*, *9*(1), 61–102.

Ramshaw, L. A., & Marcus, M. P. (1999). Text chunking using transformation-based learning. *Natural language processing using very large corpora* (pp. 157–176). Springer.

Robertson, S., & Robertson, J. (1999). Mastering the requirements process. *ACM Press, Addison-Wesley*.

Rodeghero, P., Jiang, S., Armaly, A., & McMillan, C. (2017). Detecting user story information in developer-client conversations to generate extractive summaries. *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, 49–59.

Ruiz, M., & Hasselman, B. (2020). Can we design software as we talk? *Enterprise, business-process and information systems modeling* (pp. 327–334). Springer.

Schegloff, E. A. (2007). *Sequence organization in interaction: A primer in conversation analysis i* (Vol. 1). Cambridge university press.

Sharma, S., & Pandey, S. (2013). Revisiting requirements elicitation techniques. *International Journal of Computer Applications*, *75*(12).

Software Engineering Standards Committee and IEEE-SA Standards Board. (1998). *Ieee recommended practice for software requirements specifications* (Vol. 830). IEEE.

Spijkman, T., Dalpiaz, F., & Brinkkemper, S. (2021). Requirements elicitation via fit-gap analysis: A view through the grounded theory lens.

Spijkman, T., Winter, B., Bansidhar, S., & Brinkkemper, S. (2021). Concept extraction in requirements elicitation sessions: Prototype and experimentation.

Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Subhashini, R., & Kumar, V. J. S. (2010). Shallow nlp techniques for noun phrase extraction. *Trendz in Information Sciences & Computing (TISC2010)*, 73–77.

Ten Have, P. (2007). *Doing conversation analysis*. Sage.

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*.

Tiwari, S., Rathore, S. S., & Gupta, A. (2012). Selecting requirement elicitation techniques for software projects. *2012 CSI Sixth International Conference on Software Engineering (CONSEG)*, 1–10.

van de Weerd, I., & Brinkkemper, S. (2009). Meta-modeling for situational analysis and design methods. *Handbook of research on modern systems analysis and design technologies and applications* (pp. 35–54). IGI Global.

Van Lamsweerde, A. (2009). *Requirements engineering: From system goals to uml models to software* (Vol. 10). Chichester, UK: John Wiley & Sons.

Wautelet, Y., Heng, S., Kolp, M., & Mirbel, I. (2014). Unifying and extending user story models. *International Conference on Advanced Information Systems Engineering*, 211–225.

Wiegers, K., & Beatty, J. (2013). *Software requirements*. Pearson Education.

Wieringa, R. J. (2014). *Design science methodology for information systems and software engineering*. Springer.

Winkler, J. P., Grönberg, J., & Vogelsang, A. (2019). Optimizing for recall in automatic requirements classification: An empirical study. *2019 IEEE 27th International Requirements Engineering Conference (RE)*, 40–50.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 1–10.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). sage.

Zowghi, D., & Coulin, C. (2005). Requirements elicitation: A survey of techniques, approaches, and tools. *Engineering and managing software requirements* (pp. 19–46). Springer.

Appendices

A - Activities and Concept tables

Table 29: Activities and sub-activities associated with figure 5

Activity	Sub-Activity	Description
Problem investigation	Conduct literature research on general RE and RE meetings	Literature research on Requirements Engineering in general and specifically on Requirements Engineering meetings is conducted. Delivers the OVERVIEW OF KNOWLEDGE ON GENERAL RE AND RE MEETINGS.
	Conduct literature research on RSAs	Literature research on RSAs is conducted. Delivers the OVERVIEW OF KNOWLEDGE ON RSAS.
	Conduct literature research on User Story traceability	Literature research on User Story traceability is conducted. Delivers the OVERVIEW OF KNOWLEDGE ON USER STORY TRACEABILITY.
	Conduct literature research on readily available RSAs	Literature research on readily available Requirement Specification Algorithms is conducted. This delivers the OVERVIEW OF READILY AVAILABLE RSAS. From the OVERVIEW OF READILY AVAILABLE RSAS the RSAs are chosen to be bench-marked in the Bench-mark RSAs activity. This delivers the CHOSEN RSA FOR BENCH-MARKING.

continues on next page

Treatment design	Bench-mark RSAs	The chosen RSAs from CHOSEN RSA FOR BENCH-MARKING are bench-marked on the accuracy, precision, recall and F1-score metrics. Delivers RSA PERFORMANCE METRICS TABLE.
	Assemble TUSP	One or more chosen RSAs which performed best in the bench-marking experiment are combined with the knowledge found in OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA to deliver the TUSP.
	Incorporate traceability	The TUSP is combined with the knowledge concerning User Story traceability to deliver the TUSP WITH TRACEABILITY.
Treatment validation	Execute greenfield test case	The TUSP WITH TRACEBILITY is tested on the greenfield test case and delivers GREENFIELD TEST CASE QUALITATIVE PERFORMANCE and GREENFIELD TEST CASE QUANTITATIVE PERFORMANCE.
	Execute customization test case	The TUSP WITH TRACEBILITY is tested on the customization test case and delivers CUSTOMIZATION TEST CASE QUALITATIVE PERFORMANCE and CUSTOMIZATION TEST CASE QUANTITATIVE PERFORMANCE.

continues on next page

	Report on the case studies	The case studies are reported on, discussed and concluded. Delivers CASE STUDIES REPORT.

Table 30: Concepts associated with figure 5

Concept	Description
OVERVIEW OF KNOWLEDGE ON GENERAL RE AND RE MEETINGS	The OVERVIEW OF KNOWLEDGE ON GENERAL RE AND RE MEETINGS is an overview of all the relevant extracted knowledge concerning general Requirements Engineering and Requirements Engineering meetings. Is aggregated in 1 OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA.
OVERVIEW OF KNOWLEDGE ON RSAS	The OVERVIEW OF KNOWLEDGE ON RSAS is an overview of all the relevant extracted knowledge concerning Requirements Specification Algorithms. Is aggregated in 1 OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA.
OVERVIEW OF KNOWLEDGE ON USER STORY TRACEABILITY	The OVERVIEW OF KNOWLEDGE ON USER STORY TRACEABILITY is an overview of all the relevant extract knowledge concerning User Story traceability. Is aggregated in 1 OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA.

continues on next page

OVERVIEW OF READILY AVAILABLE RSAS	The OVERVIEW OF READILY AVAILABLE RSAS is an overview of all the found Requirements Specification Algorithms that are available as a tool or GitHub repository. It contains 0 to many CHOSEN RSA FOR BENCH-MARKING. Is aggregated in 1 OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA.
OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA	The OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA is an aggregation of 1 OVERVIEW OF KNOWLEDGE ON GENERAL RE AND RE MEETINGS, 1 OVERVIEW OF KNOWLEDGE ON RSAS, 1 OVERVIEW OF KNOWLEDGE ON USER STORY TRACEABILITY and 1 OVERVIEW OF READILY AVAILABLE RSAS.
CHOSEN RSA FOR BENCH-MARKING	A Requirements Specification Algorithm that has been chosen for bench-marking based on features extracted from the literature. Has the properties: • RSA: The name of the RSA. • Reasoning: The reasoning behind the choice to use the RSA in bench-marking. Is contained by 1 OVERVIEW OF READILY AVAILABLE RSAS. Has 0 to 1 RSA PERFORMANCE METRICS TABLE.

continues on next page

RSA PERFORMANCE METRICS TABLE	A table containing the following metrics as properties: • Accuracy • Precision • Recall • F1-score Is part of 1 CHOSEN RSA FOR BENCH-MARKING.
TUSP	The Requirements Specification Pipeline is the pipeline that takes a transcript as input and outputs User Stories. As a deliverable it uses 1 to many CHOSEN RSA FOR BENCH-MARKING and 1 OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA. It is used by 1 TUSP WITH TRACEABILITY.
TUSP WITH TRACEABILITY	The Requirements Specification Pipeline with added traceability uses 1 TUSP to extend upon. It also uses 1 OVERVIEW OF PROBLEMS AND AVAILABLE SOLUTIONS FOR DEVELOPING AN RSA. It is used by 1 GREENFIELD TEST CASE QUALITATIVE PERFORMANCE, 1 GREENFIELD TEST CASE QUANTITATIVE PERFORMANCE, 1 CUSTOMIZATION TEST CASE QUALITATIVE PERFORMANCE and 1 CUSTOMIZATION TEST CASE QUANTITATIVE PERFORMANCE.
GREENFIELD TEST CASE QUALITATIVE PERFORMANCE	Contains the qualitative results of the greenfield test case. Has the following properties: • User Story completeness • User Story quality • Traceability Uses 1 TUSP WITH TRACEABILITY.

continues on next page

GREENFIELD TEST CASE QUANTITATIVE PERFORMANCE	Contains the quantitative results of the greenfield test case. Has the following properties: • Accuracy • Precision • Recall • F1-score Uses 1 TUSP WITH TRACEABILITY.
CUSTOMIZATION TEST CASE QUALITATIVE PERFORMANCE	Contains the qualitative results of the customization test case. Has the following properties: • User Story completeness • User Story quality • Traceability Uses 1 TUSP WITH TRACEABILITY.
CUSTOMIZATION TEST CASE QUANTITATIVE PERFORMANCE	Contains the quantitative results of the customization test case. Has the following properties: • Accuracy • Precision • Recall • F1-score Uses 1 TUSP WITH TRACEABILITY.
CASE STUDIES REPORT	Contains the report and the summary of the two case studies. In this report the case studies are discussed and concluded. Uses 1 TUSP WITH TRACEABILITY, 1 GREENFIELD TEST CASE QUALITATIVE PERFORMANCE, 1 GREENFIELD TEST CASE QUANTITATIVE PERFORMANCE, 1 CUSTOMIZATION TEST CASE QUALITATIVE PERFORMANCE and 1 CUSTOMIZATION TEST CASE QUANTITATIVE PERFORMANCE

B - Interview protocols

B.1 Interview 22/09/2021

The interviews will be semi-structured (Zowghi & Coulin, 2005). This means that the interviews will follow a predetermined list of questions, but the interviewer is able to ask follow-up questions and move away from the predetermined questions may they find a reason to do so. The interviewees will receive the questions and an explanation about the purpose of the interview beforehand so that they are able to prepare for the interview. This should help with eliciting more requirements. At the end of the interview a feedback question will be asked to inquire whether this really was the case.

> The **purpose** of the first meeting is to get an overview of the domain and required domain knowledge. The meeting will gain insight into the system as-is and define the system boundaries. It will also elicit the most important requirements.

The focus of the interview will be on eliciting User Stories. This means that the interviewer will specifically ask follow-up questions regarding [Role], [Goal] and [Benefit] may they deem this not clear enough. Feature importance, relative to other features, is also of interest and will be asked for.

The interview will consist of open and closed questions. Most of the predetermined questions are open, to get as much information as possible. Follow-up questions can be both. Questions are designed to answer domain and process knowledge. The interview and interview questions are structured as follows:

1. Introduction. The introduction will start with a short introduction of the interviewer, their background and a recap of the purpose of the interview. The interviewer will emphasize that their knowledge concerning the topic is minimal and would like to be seen as a student learning about the domain and system. Then the interviewer will state their goals, expectations and assumptions:

- *Goal:* the goal is to create an as complete document as possible of the software requirements (in the form of User Stories) of the FSZHAW. The transcripts that created from the interviews will be used for the thesis project of the author.
- *Expectations:* personal expectations are that a lot will be learned about the FS domain and designing a software product to aid in developing a driverless car. It is also expected that more than 1 meeting will be necessary.
- *Assumptions:* I will make the following assumption:
  - The goal of the FSZHAW is to win the FS-AI competition.
  - Everything that is being simulated (the car, tracks etc.) in the simulation software will have to adhere to the rules found online at the FS website.

Followed by the first question:

- Do you have any concerns about this goal, these expectations and the assumption?

2. Problem Domain. (what is FS and FSZHAW? Stakeholders? Time frame?) This phase will try to elicit information on the domain of FS, the FSZHAW and how the simulation software will aid in designing the driverless car.

- Next to the FS-AI, are there other competitions the FSZHAW will compete in?
- Will the simulation software also be used in other contexts?
- Will the FSZHAW compete in the ADS or the DDT class?
- Can you explain in short how the car gets controlled?
- Am I correct in that there will also be points received for the simulation software during the static events?
- For the dynamic events there are the skid pad, acceleration, autocross/sprint and the trackdrive missions?
  - Could you explain in short what each of these missions mean?
  - Is there a different algorithm for each mission?
- In the rules it is stated that the simulation tool has to simulate at least one of the acceleration, skid pad or trackdrive missions. Could you rate these in order of priority, where the highest priority is the mission that absolutely has to be simulated?
- Are you building the actual physical car at the same time that you are developing the simulation software?
- In the team, who will develop the simulation software?
- In the team, who will make use of the simulation software?
- Next to the users and developers are there any other stakeholders that could be of importance?
- Will the program already be used to run simulations by the team during the development of the program, or would you want there to be a set release?
- Is there a deadline when the simulation software has to be finished?

3. Existing System. (system as-is, system boundaries) This phase will try to elicit information about the simulation software as it is now, and in what boundaries it operates/ will operate.

- Which of the missions can the current software simulate?
- Do you have an idea of what features are most often used?
- Do you have or have you heard of any complaints that have been made about the lack of functionality of the software?
- Do you have or have you heard of any complaints about the lack of user friendliness?
- Does it run local on your laptop/PC or is it accessible from everywhere?

- If you could change 1 thing on the current software what would it be?
- If you could add 1 thing to the current software what would it be?
- Do you ever have trouble finding out where to find certain features, where to click or where to enter data?

  For the following question the interviewer will hand over part S4.1.3 (see appendix C) of the FS-AI rule book. The interviewer will also read these out loud.
- Can these all be seen as requirements for the simulation software?
- Can you say which ones are already included and which ones still have to be included?

4. Feedback and conclusion. The feedback phase is for eliciting whether other stakeholders are of importance to be interviewed and to receive feedback on the process of the interview. The following questions will be asked:

- If this interview were a group meeting to discuss the project, who, besides us, do you think should participate to the meeting?
- Did receiving the interview overview and purpose beforehand help with orienting for the interview?
- Did the interview went to your expectations? What would you have liked to see different?
- Do you have any concerns regarding the process that has been or will come?

Then the interview will be concluded. The interviewer will give a short recap in his own words using his memory and the notes he made during the interview. He will ask for confirmation regarding this recap. Any items that may need follow-up will be discussed. Finally, the interviewee will be thanked for their time.

Follow-up phase After the interview has been conducted the follow-up phase starts. This phase is meant to reinforce what was achieved and to find depth in the process. Interviewees will be send a document containing a summarisation of the interview and will be asked if they have remarks on incorrect information. If deemed necessary they will be asked for a follow-up interview.

B.2 Interview 05/10/2021

The interviews will be semi-structured. This means that the interviews will follow a predetermined list of questions, but the interviewer is able to ask follow-up questions and move away from the predetermined questions may they find a reason to do so. The interviewees will receive the questions and an explanation about the purpose of the interview beforehand so that they are able to prepare for the interview. This should help with eliciting more requirements. At the end of the interview a feedback question will be asked to inquire whether this really was the case.

> The **purpose** of this meeting is to extract the requirements that Jonas has for a simulation tool.

The focus of the interview will be on eliciting User Stories. This means that the interviewer will specifically ask follow-up questions regarding [Role], [Goal] and [Benefit] may they deem this not clear enough. Feature importance, relative to other features, is also of interest and will be asked for.

The interview will consist of open and closed questions. Most of the predetermined questions are open, to get as much information as possible. Follow-up questions can be both. Questions are designed to answer domain and process knowledge. The interview and interview questions are structured as follows:

1. Introduction. The introduction will start with a short introduction of the interviewer, their background and a recap of the purpose of the interview. The interviewer will emphasize that their knowledge concerning the topic is minimal and would like to be seen as a student learning about the domain and system. Then the interviewer will state their goals, expectations and assumptions:

- *Goal:* the goal is to create an as complete document as possible of the software requirements (in the form of User Stories) of the FSZHAW. The transcripts that created from the interviews will be used for my personal thesis project.
- *Expectations:* my expectations are that Jonas will have a lot of information on the ECU and what needs to be simulated. I want to constrain the meeting to about an hour.

Followed by the first question:

- Do you have any concerns about this goal and these expectations?

2. Problem Domain. (what is the ECU and how does it work?) This phase will try to elicit information on the domain of the Engine Control Unit and how the ECU can be simulated.

- How long have you been working on the ECU?
- How do you test the ECU at the moment?
- What does the ECU control?

- [optional] Can I see the ECU and the engines as two separate things?
- What parts of the car influence the ECU? (battery, steering, breaking etc.)
- Does the ECU use any sensory input? (speed, steering angle, GPS etc.)
- Is it correct that there is a physical part and a digital part?
  - How much can you alter/adjust on the physical ECU?
  - What software is used to control the ECU?
- How is the ECU controlled by the driver?
- How is the ECU controlled when driving driver-less?
- Does the ECU log any information?

3. Simulating the ECU. (near and far future, wants and needs.) This phase will try to elicit information on how to simulate the ECU and what is most important.

- Do you know of any existing ECU simulations?
- If the tool was being developed right now, what part of the ECU would be the first thing you would like to see simulated?
- On the long term (longer than a year), what would you like to see simulated?
- What input variables would you like to be able to alternate?
- What output variables do you want to be able to simulate/read? (temperature, humidity etc.)
- Do you know of stopping criteria for the simulation to stop?
- Do you have any rules that you would like to see simulated? Perhaps in the form of restrictions on input variables?

4. Feedback and conclusion. The feedback phase is for eliciting whether other stakeholders are of importance to be interviewed and to receive feedback on the process of the interview. The following questions will be asked:

- Is there anything you would like to add?
- If this interview were a group meeting to discuss the project, who, besides us, do you think should participate to the meeting?
- Did the interview went to your expectations? What would you have liked to see different?
- Do you have any concerns regarding the process that has been or will come?

Then the interview will be concluded. The interviewer will give a short recap in his own words using his memory and the notes he made during the interview. He will ask for confirmation regarding this recap. Any items that may need follow-up will be discussed. Finally, the interviewee will be thanked for their time.

B.3 Interview 12/10/2021

The interviews will be semi-structured. This means that the interviews will follow a predetermined list of questions, but the interviewer is able to ask follow-up questions and move away from the predetermined questions may they find a reason to do so. The interviewees will receive the questions and an explanation about the purpose of the interview beforehand so that they are able to prepare for the interview. This should help with eliciting more requirements. At the end of the interview a feedback question will be asked to inquire whether this really was the case.

> The **purpose** of this meeting is to extract the requirements that Nicolas has for a simulation tool and to talk about knowledge gained from the model car.

The focus of the interview will be on eliciting User Stories. This means that the interviewer will specifically ask follow-up questions regarding [Role], [Goal] and [Benefit] may they deem this not clear enough. Feature importance, relative to other features, is also of interest and will be asked for.

The interview will consist of open and closed questions. Most of the predetermined questions are open, to get as much information as possible. Follow-up questions can be both. Questions are designed to answer domain and process knowledge. The interview and interview questions are structured as follows:

1. Introduction. The introduction will start with a short introduction of the interviewer, their background and a recap of the purpose of the interview. The interviewer will emphasize that their knowledge concerning the topic is minimal and would like to be seen as a student learning about the domain and system. Then the interviewer will state their goals, expectations and assumptions:

- *Goal:* the goal is to create an as complete document as possible of the software requirements (in the form of User Stories) of the FSZHAW. The transcripts that created from the interviews will be used for my personal thesis project.
- *Expectations:* my expectations are that Nicolas will have information learned from the model car that could be useful for the simulation. I want to constrain the meeting to about an hour.

Followed by the first question:

- Do you have any concerns about this goal and these expectations?

2. Problem Domain. (what is the model car and how does it work?) This phase will try to elicit information on the domain of the model car, how it works, biggest obstacles and what the future will hold.

- What was/is your role in the FS ZHAW?

- If I'm correct you have created the model car yourself?
- Why was the choice made to build a model car and not a simulation?
- Which components have been tested on the the model car?
- Were there any problems or barriers with testing components on the model car that could not be fixed?
- When looking back at the process, what would you have done different?

3. Model car and the simulation tool. (near and far future, wants and needs, lessons learned) This phase will focus what useful information can be learned to use in the simulation.

- If both a simulation and a model car were available and you could do a recommendation, what would you test on the model car and what would you test in the simulation?
- Could you see a sensible cooperation between the model car and a simulation? For instance first test components in the simulation and then on the model car or vice versa.
- What would you like the simulation tool to be able to do this year?
- And what would you like to see on the long term (longer than a year)?

4. Feedback and conclusion. The feedback phase is for eliciting whether other stakeholders are of importance to be interviewed and to receive feedback on the process of the interview. The following questions will be asked:

- Is there anything you would like to add?
- If this interview were a group meeting to discuss the project, who, besides us, do you think should participate to the meeting?
- Did the interview went to your expectations? What would you have liked to see different?
- Do you have any concerns regarding the process that has been or will come?

Then the interview will be concluded. The interviewer will give a short recap in his own words using his memory and the notes he made during the interview. He will ask for confirmation regarding this recap. Any items that may need follow-up will be discussed. Finally, the interviewee will be thanked for their time.

C - Additional interview files

S4.1.3 Teams are free to use whatever simulation software they choose. Development of their own sensor models, environments and vehicle models will be rewarded. Simulating more than one dynamic event will be rewarded. Vehicle performance (ie, laptime) won't be directly rewarded or compared.

Topics judges would like to see covered include, but are not limited to:

- Digital twin environments and environmental factors (ie weather)
- Perception and localisation algorithm development
- Vehicle models and dynamics analysis
- FS-AI Mission Control implementation
- Further software stack development, including path planning and vehicle controls (driver models)
- Debugging and visualisation tools
- Integration of the vehicle interface with simulated vehicle actuator controller(s)
- Correlation and validation methodologies (noise factors, etc)
- Data analysis methodologies
- How does simulation development inform real-world testing?

Figure 30: S4.1.3 from FS-AI rulebook

D - Greenfield Case User Stories

Table 31: I1 User Stories

ID	User Story
I1.1	As a user, I want to be able to add noise to the cone detection, so that the cone detection is more realistic.
I1.2	As a user, I want the system to automatically add noise to the cone detection when I select the add noise option, so that the cone detection is less accurate and more realistic
I1.3	As a user, I want to be able to limit the maximum steering angle the car can take, so that the simulation is more realistic.
I1.4	As a user, I want to be able to limit the maximum acceleration for the car, so that I can model the simulation after the real car.
I1.5	As a user, I want to be able to see the output of sensors, so that my software can interact with these.
I1.6	As a user, I want to be able to see LIDAR sensor output, so that my software can interact with the LIDAR sensor output.
I1.7	As a user, I want to be able to see the IMU sensor output, so that my software can interact with the IMU sensor output.
I1.8	As a user, I want to be able to see the GPS sensor output, so that my software can interact with the GPS sensor output.
I1.9	As a user, I want to be able to simulate the camera output, so that my software can interact with the camera output.
I1.10	As a user, I want to see a visualisation of the track and car driving on it, so that I can see how and where the car crashes.
I1.11	As a user, I want to see a visualisation of the track and car driving on it, so that I can see how the car performs on the track.
I1.12	As a user, I want to see a visualisation of the track and car driving on it, so that I can use the visual output for debugging purposes.
I1.13	As a user, I want to be able to see the history of run simulations and its results, so that I can infer conclusions from past runs.
I1.14	As a user, I want to be able to store different car configurations, so that loading a configuration that I have already used becomes less time costly.

continues on next page

I1.15	As a user, I want to be able to run different simulations multiple times over night, so that I can come back the next day and multiple simulations have been run.
I1.16	As a user I want to be able to run the simulation tool on Linux, so that I can use my preferred system.
I1.17	As a user, I want the generated tracks to comply with the Formula Students Rule Set, so that all generated tracks are legal.
I1.18	As a user, I want to be able to select different race missions, so that I can test my software in the different missions.
I1.19	As a user, I want to be able to use the simulation tool during development, so that I can test my software while developing it.
I1.20	As a user, I want to be able to use the simulation tool before a race, so that I can make last minute adjustments to my software.
I1.21	As a user, I want to be able to use the simulation tool on a laptop, so that I can use the tool on every location that I need it.
I1.22	As a new developer, I want good documentation, so that I can start improving the tool quicker.
I1.23	As a new developer, I want clean code, so that it is easy to understand.
I1.24	As a new developer, I want the simulation tool software to be tested, so that I can start with a well working tool.
I1.25	As a user, I want to be able to report bugs and crashes on GitHub, so that the tool can be improved.
I1.26	As a user, I want the simulation tool to support the English language, so that everybody from the Formula Student Team can use it.
I1.27	As a user, I want to be able to use the simulation tool while it is being developed, so that I don't have to wait until it is finished.

Table 32: I2 User Stories

ID	**User Story**
I2.1	As a simulation tool developer, I want the simulation tool to be written in ROSS, so that I can continue on the previously build versions.
I2.2	As a mechanic, I want the simulation tool to be able to test mechanical parts, so that I don't have to build a testing site for testing the drag and airflows.

continues on next page

I2.3	As a user, I want the simulation tool to be able to handle C++ files, so that I can code my algorithms in my preferred coding language.
I2.4	As a user, I want the simulation tool to be able to handle Python files, so that I can code my algorithms in my preferred coding language.
I2.5	As a path planning developer, I want to be a able to test my path planning algorithms in the simulation tool, so that I don't have use real resources such as model cars or the actual car.
I2.6	As a user, I want to be able to create custom maps, so that I can test in multiple ways and on multiple different tracks.
I2.7	As a user, I want to be able to insert maps in CSV form, so that I can use maps created by other people/teams.
I2.8	As a user, I want the simulation tool to simulate slipping and sliding, so that I can get a realistic view of the car behaviour.

Table 33: I3 User Stories

ID	User Story
I3.1	As the Formulate Student Team, we want to be able to use the simulation tool for sponsoring purposes, so that we can show what we are working on.
I3.2	As a sponsor, I want to steer the simulated car in the simulation tool with an actual steering wheel, so that I can have the feeling I'm driving an actual racing car.
I3.3	As the leader of the drivers team, I want to be able to upload the schematic of the actual car to a simulation tool, so that I can test the drivers on an as close to reality simulation as possible.
I3.4	As a perception engineer I want to be able to test my perception algorithms in the simulation tool, so that I can save time and materials.
I3.1	As a perception engineer I want to be able to test the prediction of the perception in the simulation tool, so that I can save time and materials.
I3.5	As a perception engineer I want to be able to run the simulation N number of times, so that I can use that data as training data for my perception algorithms.
I3.6	As a user, I want the simulation tool to be able to run the Skidpad event, so that I can test the algorithms used in the Skidpad event.

continues on next page

I3.7	As a user, I want the simulation tool to be able to run the Acceleration event, so that I can test the algorithms used in the Acceleration event.
I3.8	As a user, I want the simulation tool to be able to run the Autocross event, so that I can test the algorithms used in the Autocross event.
I3.9	As a user, I want the simulation tool to be able to run the Endurance and Efficiency event, so that I can test the algorithms used in the Endurance and Efficiency event.
I3.10	As a user, I want the simulation tool to be able to run the Trackdrive event, so that I can test the algorithms used in the Trackdrive event.
I3.11	As the Formula Student Team, we want to be able to put all our code into the simulation tool at the same time, so that we can test the complete car.
I3.12	As an ECU developer, I want to be able to run the ECU algorithm, so that I can test the ECU algorithm.
I3.13	As a perception engineer, I want the simulation tool to be able to simulate all the sensors, so that I can test my perception algorithms.
I3.14	As a perception engineer, I want the simulation tool to be able to simulate the LIDAR, so that I can test my perception algorithms.
I3.15	As a perception engineer, I want the simulation tool to be able to simulate the GPS, so that I can test my perception algorithms.
I3.16	As a perception engineer, I want the simulation tool to be able to simulate the optical sensor, so that I can test my perception algorithms.
I3.17	As a perception engineer, I want the simulation tool to be able to simulate the ground speed sensor, so that I can test my perception algorithms.
I3.18	As a user, I want to be able to simulate torque vectoring, so that I can test the how the Board Computer and the ECU work together.
I3.19	As the Formula Student Team, we want the simulation tool to simulate digital twin environments and environmental factors (ie weather), so that we get extra points in the static event.
I3.20	As the Formula Student Team, we want the simulation tool to simulate perception and localisation algorithm development, so that we get extra points in the static event.

continues on next page

I3.21	As the Formula Student Team, we want the simulation tool to simulate vehicle models and dynamics analysis, so that we get extra points in the static event.
I3.22	As the Formula Student Team, we want the simulation tool to simulate FS-AI mission control implementation, so that we get extra points in the static event.
I3.23	As the Formula Student Team, we want the simulation tool to simulate path planning, so that we get extra points in the static event.
I3.24	As the Formula Student Team, we want the simulation tool to simulate vehicle controls, so that we get extra points in the static event.
I3.25	As the Formula Student Team, we want the simulation tool to include debugging and visualisation tools, so that we get extra points in the static event.
I3.26	As the Formula Student Team, we want the simulation tool to include an integration of the vehicle interface with simulated vehicle actuator controller(s), so that we get extra points in the static event.
I3.27	As the Formula Student Team, we want the simulation tool to include correlation and validation methodologies (noise factors, etc), so that we get extra points in the static event.
I3.28	As the Formula Student Team, we want the simulation tool to include data analysis methodologies, so that we get extra points in the static event.
I3.29	As a user, I want the simulation tool to have more flexibility, so that I can try different algorithms and different coding languages on the simulator.
I3.30	As a user, I want the simulation tool to have more compatibility, so that I can try more different algorithms next to each other.
I3.31	As a user, I want the simulation tool to be able to run for the amount of time/laps that the event takes, so that I can simulate performance of the whole of the event.
I3.32	As a user I want the simulation tool to automatically run the event multiple times, so that I can leave and come back later to receive data of many runs.
I3.33	As a path planning developer, I want to be able to run multiple different path planning algorithms, so that I can gather data and compare the algorithms.

continues on next page

I3.34	As a car mechanic, I want the simulation tool to have the correct physics, so that I can test how different simulated vehicle models may work.
I3.35	As a car mechanic, I want the simulation tool to have the right car dynamics, so that I can test how the car works in a simulation.
I3.36	As a car mechanic, I want the simulation tool to be able simulate aerodynamics, so that I can test the airflows around my car.
I3.37	As a user, I want the simulation tool to be able to handle a real driver, so that I can see myself sitting in the car and actually have the feeling I'm riding a car.
I3.38	As a user, I want to be able to adjust the hardness and granularity of the ground, so that I can how the car slides and slips on different sorts of underground.
I3.39	As a user, I want the simulation tool to have the correct kinematics, so that my simulated car works in a realistic way.
I3.40	As simulation tool developer, I want the simulation tool to be testable, so that I can make sure it correlates with reality and that reality does not give wildly different results than that the simulation gave.
I3.41	As a simulation tool developer, I want to be able to use real-world data to test the simulation tool, so that I can make sure it correlates with reality and that reality does not give wildly different results than that the simulation gave.
I3.42	As a map builder, I want to be able to convert LIDAR maps to maps for the simulation, so that I can easily upload real-life maps to the simulation.

Table 34: I4 User Stories

ID	**User Story**
I4.1	As an ECU engineer, I want the simulation tool to be able to couple with the ECU, so that I can test if the ECU works properly.
I4.2	As an ECU engineer, I want the simulation tool to be able to simulate the starting sequence for the ECU, so that I can test if the ECU starts up well and safe.

continues on next page

I4.3	As an ECU engineer, I want to be able to simulate shock up detections, so that I can see how the ECU responses in case of a shock up.
I4.4	As a user, I want the simulation to simulate the CAN connections, so that I can see if everything connects together the way it should.
I4.5	As an ECU engineer, I want to be able to simulate the ECU, the inverter, the accumulator and the drive train together, so that I don't have to disconnect all those from the real car in the real world.
I4.6	As an ECU engineer, I want to be able to feed the simulated ECU with simulated data from the accumulator, the inverter and the engine, so that I can test the ECU as a closed system.
I4.7	As a user, I want to have dynamic drive data that is based on real world situations, so that I can test the motor, the accumulator, the inverter and the ECU in the most realistic way possible.
I4.8	As a user, I want the simulation to be able to simulate every part and program individually and modularly, so that I can easily switch a part and see how it interacts.
I4.9	As an accumulator engineer, I want to be able to push the accumulator to its limits and then send false data, so that I can see how it handles under stressful situations.
I4.10	As an ECU engineer, I want the simulation tool to be able to handle Simulink files, so that I can test the ECU in the simulation.
I4.11	As an ECU engineer, I want the simulation tool to be able to handle MATLAB files, so that I can test the ECU in the simulation.
I4.12	As an ECU engineer, I want to be able to simulate the ECU and the Board computer and their connection, so that I can test how the ECU reacts on the board computer.
I4.13	As an ECU engineer, I want to be able to read what the ECU is doing at any moment, so that I can get a better understanding of how it responds to different inputs.
I4.14	As an ECU engineer, I want to be able to simulate system errors or failures, so that I can see how the ECU responds.
I4.15	As an ECU engineer, I want to be able to simulate every possible scenario and simulate 10.000 hours, so that I see if the system stays within the lowest safety category (1 failure in 10.000 hours).
I4.16	As a user, I want to be able to simulate the throttle, so that I can see what effect it has on the rest of the system.

continues on next page

I4.17	As a user, I want to be able to test for short circuits, so that I can see if everything is safe.

Table 35: I5 User Stories

ID	User Story
I5.1	As a perception engineer, I want to be able to simulate the data from the sensors, so that I can verify the algorithms.
I5.2	As a perception engineer, I want to be able to have data generation with noise, so that I can do more elaborate verification of the software.
I5.3	As a perception engineer, I want to have a fixed test framework, so that I can retest and make sure I don't break older stuff.
I5.4	As a perception engineer, I want to be able to simulate the camera, so that I can test the perception software.
I5.5	As a perception engineer, I want to be able to simulate the LIDAR, so that I can test the perception software.
I5.6	As a perception engineer, I want to be able to simulate when a cone is not upright, so that I can test how the perception works on fallen cones.
I5.7	As a perception engineer, I want to be able to simulate different kinds of illumination, so that I can test how the perception works at different light levels.
I5.8	As a perception engineer, I want to be able to simulate different weather conditions, so that I can test how the perception works in different types of weather.
I5.9	As a perception engineer, I want to be able to simulate different types to track, so that I can test how the perception works on different types of track.
I5.10	As a perception engineer, I want to be able to simulate the CPU capacity, so that I can test if I can run all my software along with the other kinds of software that use the CPU.
I5.11	As a perception engineer, I want to be able to simulate the network traffic, so that I can test if there is room on the traffic for my perception software.
I5.12	As a system engineer, I want to be able to simulate the network traffic, so that I can test if there is enough space on the network for all the different software components.

Table 36: I6 User Stories

ID	User Story
I6.1	As an autopilot engineer, I want to be able simulate the dynamic model of the car, so that I can simulate my software with an accurate representation of the real car.
I6.2	As an autopilot engineer, I want to be able to do automated testing, so that I can run a multitude of 100 test runs with my software.
I6.3	As an autopilot engineer, I want to have an accurate representation of the car in the simulation, so that I can test the path planning and control.
I6.4	As an autopilot engineer, I want the automated testing to have a log, so that I can automatically run different tracks and see which tracks ran well and which tracks failed.
I6.5	As an autopilot engineer, I want to be able to easily load the different types of missions, so that I can test my software on these different types of missions.
I6.6	As an autopilot engineer, I want to be able to simulate raw data from the sensors, so that I can test for errors in the autopilot estimations.
I6.7	As an autopilot engineer, I want to be able to simulate different weather conditions, so that I can test how the autopilot deals with the influences of the weather on the interaction between the wheels and the ground.
I6.8	As an autopilot engineer, I want to be able to change the slip ratio, so that I can test how the autopilot deals with the influences of more slipperiness on the interaction between the wheels and the ground.

Table 37: S1 User Stories

ID	User Story
S1.1	As a user I want to be able to use the simulation tool for real time validation, so that I can test my software while developing it.
S1.2	As the Formulate Student Team, we want the development of the simulation tool to have no costs, so that we can spend our budget on other things to improve the car.
S1.3	As a user, I want the simulation tool to check the configurations for rule conformity, so that I am prevented from doing illegal tests.

continues on next page

S1.4	As a user, I want the simulation tool to check for optimization on the car, so that I can infer how improve the car from doing simulations.
S1.5	As a user, I want the simulation tool to have an Acceleration mission mode, so that I can test my software on the Acceleration mission.
S1.6	As a user, I want the simulation tool to have a Skidpad mission mode, so that I can test my software on the Skidpad mission.
S1.7	As a user, I want the simulation tool to have an Autocross mission mode, so that I can test my software on the Autocross mission.
S1.8	As a user, I want the simulation tool to have a Trackdrive mission mode, so that I can test my software on the Trackdrive mission.
S1.9	As a user, I want the simulation tool to have an EBS mode, so that I can test my software on the EBS moment.
S1.10	As a user, I want the simulation tool to have an inspection mode, so that I can test my software on the inspection moment.
S1.11	As a user, I want the simulation tool to have a manual mode, so that I can test my software while driving the simulated car manually.
S1.12	As a user, I want the simulation tool to support account management, so that I can track who ran what simulation.
S1.13	As a user, I want the simulation tool to make backups, so that I do not have to be afraid that I lose data in case of a crash.
S1.14	As a user, I want the simulation to remove any data generated during a simulation tool system crash, so that I don't create redundant data.

E - Customization Case User Stories

Table 38: A1 User Stories

ID	User Story
A1.1	As an IFA chief information officer, I want to monitor the financial balance of the football teams, so that I can see if everything is managed properly.
A1.2	As an IFA chief information officer, I want to specify a policy in the system, so the system can alert of certain violation.
A1.3	As an IFA chief information officer, I want to set regulation rules for audits, so that these can be set as policies
A1.4	As an IFA administrative, I want to see the various transactions, so that I can monitor them.
A1.5	As an IFA administrative, I want to access the budgeting system, so that I can monitor the team budgets.
A1.6	As a team, I want to access the budgeting system, so that I can work with the team budget.
A1.7	As a referee, I want to insert the events that happen during a game into the system, so that information about the game can be updated in real time.
A1.8	As an IFA information officer, I want that only the referee or side referee can insert events into the system, so that the information is as reliable as possible.
A1.9	As an IFA information officer, I want each event that happens during the game to have a timestamp, so that it is as reliable as possible.
A1.10	As a referee, I want to add extra information about an event into the system, so that the event is detailed.
A1.11	As an IFA chief information officer, I want everybody to be able to see all the data about games and results, so that there is complete transparency.
A1.12	As a user, I want to be able to perform analysis of the data, so that I can get deeper insights.
A1.13	As an IFA football team, I want that my team is the only one with the ability to alter budget data about my team, so that it is secured.

continues on next page

A1.14	As an IFA scheduler, I want to make policies for scheduling games, so that the scheduling algorithm abides by these constraints when making the schedule.
A1.15	As an IFA scheduler, I want to make policies for scheduling referees, so that scheduling referees does not have to be done manually.
A1.16	As a referee, I want to enter my preferences for scheduling in the system, so that the system takes my preferences into account when making the referee schedule.
A1.17	As an IFA chief information officer, I want that the season schedules cannot be changed, so that it is clear what everybody is up to once the schedule is done.
A1.18	As an IFA administrator, I want to override the schedule if there are any problems, so that I can adjust the schedule to fix these problems.
A1.19	As an IFA administrator, I want another IFA representative to check and approve any overrides I made, so that bias is prevented and transparency ensured.
A1.20	As a fan, I want all information in one place called the fan portal, so that I can find information about the games, teams and my favourite players in that fan portal.
A1.21	As a fan, I want to register to a game, so that I get notified about all the information on that game.
A1.22	As a fan, I want to register to a team, so that I get notified about all the information on that team.
A1.23	As a fan, I want to register to a player, so that I get notified about all the information on that player.
A1.24	As a fan, I want to see the live game updates that a referee posts, so that I can get reliable information about a game.
A1.25	As a player, I want to have my own page, so that I can post updates and interact with the fans.
A1.26	As a user, I want to access the system through a mobile app, so that I can access the system from my mobile phone.
A1.27	As a user, I want to access the system through a website, so that I can access the system from my computer.
A1.28	As a fan, I want to set specific notifications, so that I can be notified when specific events or changes occur.

continues on next page

A1.29	As an IFA administrator, I want the system to respond within a second, so that I don't have to wait long for response from the system.
A1.30	As a fan, I want the system to respond within a second, so that I don't have to wait long for response from the system.
A1.31	As an IFA football team, I want the system to respond within a second, so that I don't have to wait long for response from the system.
A1.32	As a referee, I want the system to respond real fast, so that I can upload real time events to the system.
A1.33	As a referee, I want my reports to have the highest priority, so that I can send updates in real time.
A1.34	As a side-referee, I want to have a live recording feature, so that the game updates can be done more rapidly.
A1.35	As an IFA chief information officer, I want to completely outsource development of the system, so that I can focus on my main tasks.
A1.36	As an IFA chief information officer, I want to completely outsource maintenance of the system, so that I can focus on my main tasks.
A1.37	As an IFA chief information officer, I want the system to comply with all the laws of the country a fan is using the system from, so that there are no legal problems.
A1.38	As a fan, I want the system to be in my language, so that I can understand it.
A1.39	As a fan, I want the system to be in my timezone, so that it shows the correct times for my location.
A1.40	As a fan, I want the system to show my currencies, so that I can see the correct pricing.
A1.41	As an IFA chief information officer, I want the budget data to be encrypted, so that it doesn't fall into the wrong hands.
A1.42	As a fan, I want to make an account, so that I can register to the parts of the system I like.
A1.43	As a guest, I want to be able to see all the information in the system except for budgeting, so that I can have complete transparency.
A1.44	As an IFA chief information officer, I want all data to be stored, so that it can be used for analysis.

Table 39: A2 User Stories

ID	User Story
A2.1	As an IFA chief information officer, I want the system to define the leagues, so that I am supported in my decision making.
A2.2	As an IFA chief information officer, I want the system to define the teams, so that the teams easily get added to the system.
A2.3	As an IFA chief information officer, I want to define different seasons for different leagues, so that these are easily set.
A2.4	As an IFA chief information officer, I want a budget control over the teams, so that I can check their finances.
A2.5	As a fan, I want to query the system, so that I get reliable and comprehensive data.
A2.6	As a fan, I want to query for statistics on a team, so that I can base decisions on that.
A2.7	As a fan, I want to query for statistics on a player, so that I can base decisions on that.
A2.8	As a player, I want to manage my own page on the network, so that I can update, share articles, show new statistics and provide news.
A2.9	As a local IFA, I want to manage my own countries league, so that I can adjust it to my countries needs.
A2.10	As a local IFA, I want to mange my own countries seasons, so that I can adjust it to my countries needs.
A2.11	As a local IFA, I want to set the system to my local currency, so that it fits my country.
A2.12	As a local IFA, I want to set the system to my local language, so that it fits my country.
A2.13	As an IFA chief information officer, I want all transactions made a team to be reported within the system, so that there is transparency on the teams budget.
A2.14	As an IFA chief information officer, I want all transactions made a team to be archived within the system, so that there is transparency on the teams budget.
A2.15	As an IFA chief information officer, I want transparency on events that happen during a game, so that everyone can see what happens during a game.

continues on next page

A2.16	As an IFA chief information officer, I want the events that happen during a game to be updated in real time, so that everyone can see in real time what happens during a game.
A2.17	As a referee, I want to update on events during the game, so that it's from a reliable source.
A2.18	As a referee, I want to update the events after the game, so that I don't have to do everything during the game.
A2.19	As a referee, I want add comments to the events after the game, so that I don't have to do everything during the game.
A2.20	As an IFA chief information officer, I want the teams to only use the system for their financial activity, so that they don't have to use their own systems next to it.
A2.21	As an IFA chief information officer, I want the system to automatically generate a financial report for a team, so that they don't have to do this manually anymore.
A2.22	As an IFA chief information officer, I want to determine policies for scheduling the games, so that I can set certain rules for how the games should be scheduled.
A2.23	As an IFA chief information officer, I want to determine policies for scheduling the referees, so that I can set certain rules for how the referees should be scheduled.
A2.24	As a referee, I want to set my preferences for scheduling, so that I only get scheduled on days that I'm available and locations where I'm available.
A2.25	As an IFA chief information officer, I want that overrides in the schedule are detailed and approved by another IFA representative, so that biases are avoided.
A2.26	As an IFA chief information officer, I want the scheduling of the games and the scheduling of the referees to happen at the same time, so that biases are avoided.
A2.27	As a fan, I want to subscribe to a team, so that I can receive notifications on news regarding my favourite team.
A2.28	As a fan, I want to subscribe to a game, so that I can receive notifications on changes and other events for that game.
A2.29	As a fan, I want to subscribe to a player, so that I can receive notifications regarding news on that player.

continues on next page

A2.30	As an IFA chief information officer, I want the data to stored as fine grained as possible, so that I can get any statistic that I want.
A2.31	As an IFA chief information officer, I want only the referees to be able to insert data into the system, so that bias is prevented.
A2.32	As a referee, I want direct access to the system, so that the information is more reliable.
A2.33	As a user, I want to receive notifications through an app on my phone, so that I'm informed on what I subscribed to.
A2.34	As a user, I want to access the system through an app on my phone, so that I can access the system everywhere.
A2.35	As a user, I want to access the system through a web based application, so that I can enter data.
A2.36	As an IFA chief information officer, I want a mechanism for controlling the access of other people, so that different stakeholders can not access the whole system.
A2.37	As a local IFA, I want to be able to set rules for the finances of teams, so that teams are controlled in their budget.
A2.38	As the IFA, I want to approve new teams in the system, so that the teams can complete their registration.

Table 40: A3 User Stories

ID	User Story
A3.1	As an IFA information officer, I want each transaction by a team done within the system, so that I can receive immediate updates.
A3.2	As an IFA information officer, I want each transaction by a team done within the system, so that I can have continuous control.
A3.3	As an IFA information officer, I want each transaction by a team done within the system, so that I can be alerted of deviation from the rules of budgetary control.
A3.4	As an IFA information officer, I want to incorporate budget rules, so that I can use these to control the team budgets.
A3.5	As a local IFA, I want to determine my own rules, so that they fit withing the local context.

continues on next page

A3.6	As an IFA information officer, I want to determine policies for scheduling the games, so that these policies are applied when automatically scheduling games.
A3.7	As a local IFA, I want to determine the rules of how to schedule the games, so that the games are scheduled according to local rules.
A3.8	As a local IFA, I want the system to suggest a schedule, so that I don't have to do this manually.
A3.9	As a local IFA I want to approve the suggested schedule, so that I can check if it is correct or needs a change.
A3.10	As a local IFA, I want to explicate the reason for changing the schedule and overriding the system, so that bias is avoided and transparency is allowed.
A3.11	As a local IFA, I want two IFA representatives to confirm the schedule, so that bias is prevented.
A3.12	As a fan, I want to query for any information that is saved in the system, so that I can do my analysis with this data.
A3.13	As a fan, I want to register to certain information that I require, so that I get updates of information that I find interesting.
A3.14	As a team, I want to have a social network page within the system, so that I can send updates about the team.
A3.15	As a fan, I want to register to a certain game, so that I get notified about updates on that game.
A3.16	As a referee, I want to insert data into the system, so that I can immediately register events that happen during the game.
A3.17	As an IFA information officer, I want to see exactly who is responsible for the data inserted during games, so that transparency is ensured.
A3.18	As an IFA information officer, I want the system to make statistics based on the fine grained data saved, so that all statistics can be traced back to the original data.
A3.19	As an IFA information officer, I want all data except the budgetary data to be available to everybody, so that transparency is ensured.
A3.20	As an IFA information officer, I want all communication to go through the system, so that communication becomes more efficient.

continues on next page

A3.21	As an IFA information officer, I want all important notifications to be distributed automatically, so that everybody who needs information is ensured to receive it.
A3.22	As an IFA information officer, I want the system to make the management of the entire operation easy for the various stakeholders, so that workload is reduced for the IFA.
A3.23	As a team, I want to be able to enter all information about my team, so that workload is reduced for the IFA administratives.
A3.24	As a team CEO, I want to be able to insert players into the system, so that they are a registered IFA player.
A3.25	As a referee, I want to insert my preferences for the scheduling, so that I only get scheduled on the days that I'm available.
A3.26	As a referee, I want to have mobile application, so that I can use it during a game.
A3.27	As a referee, I want to have a list of possible events that occur during a game, so that I can quickly insert it.
A3.28	As a fan, I want to have an account, so that I can get notified about certain events.
A3.29	As a guest, I want to query the system, so that I can get the information I need.
A3.30	As a player, I want to have my own social network page, so that I can kind of communicate with my fans.
A3.31	As a gambler, I want to query statistical information from the system, so that I can make better bets.
A3.32	As a gambler, I want to get recommendations for bets on teams based on previous played games, so that I can make better bets.
A3.33	As a user, I want to use the system on my desktop through a web based application, so that I can conveniently walk through the system.
A3.34	As an IFA information officer, I want 50000 fans to be able to use the system at the same time, so that it doesn't freeze.
A3.35	As a referee I want to have priority when adding data to the system, so that I can give real time updates.
A3.36	As a local IFA, I want the system to change to my local preferences, so that I can use the system in my own language, currency and timezone.

continues on next page

A3.37	As an IFA information officer, I want the system to be of low maintenance, so that we can focus on our core tasks.

Table 41: A4 User Stories

ID	User Story
A4.1	As an IFA information officer, I want every transaction to be reported within the system, so that the granularity of the information becomes much higher and finer.
A4.2	As an IFA information officer, I want to aggregate the budget data, so that I can check the teams budget when needed.
A4.3	As an IFA information officer, I want to set budget rules, so that a team is penalized when these are violated.
A4.4	As an IFA information officer, I want to set policies for scheduling, so that that personal judgement is prevented in making the schedule.
A4.5	As an IFA administrative, I want to be able to override the automatic scheduling with a specific reason, so that transparency is ensured by providing this reason.
A4.6	As an IFA administrative, I want someone else to approve my override of the automatic schedule, so that bias is prevented and scheduling is ethical.
A4.7	As an IFA administrative, I want to only be able to change policies at the beginning of the season, so that no changes are made during the season.
A4.8	As a user, I want to query information the database, so that I can get information on the teams, the players, the coaches and the games.
A4.9	As a fan, I want to follow social network pages, so that I can get information about the teams, the players and the games.
A4.10	As a fan, I want to register to information that I'm interested in, so that I get a notification whenever there's something new on that information.
A4.11	As an IFA chief information officer, I want that every event that happens during a game is recorded, so that the data is at the lowest granularity as possible.
A4.12	As a referee, I want record events that happen during the game, so that it can be saved within the system.

continues on next page

A4.13	As an IFA chief information officer, I want that all stakeholders are automatically notified when something changes in the system, so I don't have to do extra efforts in order to notify about the changes.
A4.14	As a user, I want to receive notifications through the application on my phone, so that I don't get SMS or email notifications.
A4.15	As an IFA administrative, I want to define leagues, so that teams can enlist for this league.
A4.16	As an IFA administrative, I want to initiate a season, so that it starts on a set date.
A4.17	As an IFA administrative, I want to assign referees specifically, so that the referees get assigned to certain games.
A4.18	As an IFA administrative, I want to determine how to calculate the place of the group by points, so that every team has a score on the scoreboard.
A4.19	As an IFA administrative, I want to activate automatic scheduling, so that I don't have to make the schedule manually.
A4.20	As an IFA administrative, I want to set rules for the budget control, so that I can check the budgets of the teams.
A4.21	As a team, I want to manage my own social network page, so that I can be in touch with my fans.
A4.22	As a team, I want to mange my own resources, so that I can manage players, coaches, schedules training etc.
A4.23	As a team, I want to manage my finances, so that it can be controlled by the IFA.
A4.24	As a referee, I want to see what games I'm scheduled for, so that I can anticipate on that.
A4.25	As a main referee, I want to upload a final report after the game ends, so that the game is finalized.
A4.26	As a referee, I want to insert constraints regarding the schedule, so that I only get scheduled on days that I'm available.
A4.27	As a fan, I want to comment on a social network page, so that I can interact.
A4.28	As a player, I want to manage my own social network page, so that I can be in touch with my fans.
A4.29	As a coach, I want to manage my own social network page, so that I can be in touch with my fans.

continues on next page

A4.30	As a social network page manager, I want to manage the pages I'm assigned to, so that I can do my job.
A4.31	As a social network page manager, I want to upload things to the pages I'm assigned to, so that I can post new information on the page.
A4.32	As an IFA chief information officer, I want the system to be able to handle 50,000 people at the same time, so that it never gets overloaded.
A4.33	As a referee, I want to get high priority during a game, so that I can quickly report on events that happen.
A4.34	As a fan, I want the system to respond reasonably quick, so that I don't get annoyed when waiting for the system.
A4.35	As a team, I want to register to the system by sending a request to the IFA, so that the IFA can approve my request.
A4.36	As a user, I want the system to be adjusted to my preferences, so that I can view it in my own language, currency and time zone.

Table 42: B1 User Stories

ID	User Story
B1.1	As a head of urban planning, I want to reduce the time of decision making, so that I can act quicker on problems.
B1.2	As a head of urban planning, I want to make suggestions based on real data, so that I can convince my stakeholders that the suggestion is rooted in a realistic simulation.
B1.3	As a head of urban planning, I want the system to simulate traffic based on real data, so that the simulation is realistic.
B1.4	As a head of urban planning, I want the system to allow me to compare historical data, so that I can anticipate days where the situation is different than the current data shows.
B1.5	As a head of urban planning, I want to store the data on our servers, so that it can be used as a historical reference.
B1.6	As a head of urban planning, I want to recognize patterns in the data, so that I can anticipate traffic situations.
B1.7	As a head of urban planning, I want to see what impact my changes have, so that I don't build unnecessary roads.

continues on next page

B1.8	As a head of urban planning, I want to use the sensors placed in the city, so that the data from these sensors can be used in the simulation.
B1.9	As a head of urban planning, I want the simulation to run on google maps, so that I can get an overlay of the sensors on google maps.
B1.10	As a head of urban planning, I want the simulation to take environmental factors into concern, so that environmentally the roads optimized.
B1.11	As a head of urban planning, I want the system to be usable by everybody, so that I don't need technical people to run it.
B1.12	As a head of urban planning, I want a user friendly interface, so that everybody in the urban planning department can use it.
B1.13	As a head of urban planning, I want the system to visualize what is going on at the moment, so that I can see an indication of cars moving in real time.
B1.14	As a head of urban planning, I want to compare the current situation to one in the past, so that I see the differences in situations.
B1.15	As a head of urban planning, I want the system to automatically tell me of any major differences, so that my work is accelerated.
B1.16	As a head of urban planning, I want to implement standard types of analysis, so that I can quickly run these.
B1.17	As a head of urban planning, I want to be able to define a customized workflow, so that I can analysis special situations.
B1.18	As a head of urban planning, I want to see the major alternatives to a special situation, so that I can compare the situations.
B1.19	As a head of urban planning, I want to do a repeatable analysis, so that I can apply for an ISO certification.
B1.20	As a head of urban planning, I want to have a standard way of analysing a problem, so that I can apply for an ISO certification.
B1.21	As a head of urban planning, I want to see the co2 sensors data overlayed on the map, so that I can see the environmental impact.
B1.22	As a head of urban planning, I want to see the noise pollution sensors data overlayed on the map, so that I can see the environmental impact.

continues on next page

B1.23	As a head of urban planning, I want the simulation to calculate environmental impacts, so that I can get an indication of the environmental impacts where there are no sensors.
B1.24	As a head of urban planning, I want to have an indication of the credibility of calculations done by the simulation, so that we can cross check the for the credibility of the simulation.
B1.25	As a head of urban planning, I want to add new data sources, so that the system can grow over time.
B1.26	As a head of urban planning, I want the system to be available 24/7, so that I can always react on emergencies.
B1.27	As a head of urban planning, I want the system to be available from home, so that I can always access it if necessary.
B1.28	As a head of urban planning, I want the start up time to be under 2 minutes, so that I can quickly react to emergency situations.

Table 43: B2 User Stories

ID	User Story
B2.1	As a head of urban planning, I want to reduce time spend on reporting, so that I can focus on other things.
B2.2	As a head of urban planning, I want to show the media that we have concrete ways to analyse our solutions, so that our credibility is increased.
B2.3	As a head of urban planning, I want to determine what the effects of changes that I make are going to be, so that I can save on effort and costs.
B2.4	As a head of urban planning, I want to increase fairness, so that everybody is happy.
B2.5	As a head of urban planning, I want to test a solution before it is implemented, so that I can check the impact of the changes I propose.
B2.6	As a head of urban planning, I want to validate a solution before it is implemented, so that I can check the impact of the changes I propose.
B2.7	As a head of urban planning, I want to show evidence that our solution will work, so that the consultancy company will believe me.

continues on next page

B2.8	As a head of urban planning, I want to convince everybody that we rely on real data, so that I show that we didn't just invent how many cars are moving around.
B2.9	As a head of urban planning, I want the data to be stored on premise, so that privacy is ensured.
B2.10	As a head of urban planning, I want to store the data for six months, so that I can look back at old data for comparisons.
B2.11	As a head of urban planning, I want to take snapshots of the data of special days, so that I can use it for analysis of these special days.
B2.12	As a head of urban planning, I want to simulate special situations, so that I can anticipate on these.
B2.13	As a head of urban planning, I want to be shown possible solutions, so that I can take these into account.
B2.14	As a head of urban planning, I want the simulation to take pollution levels into account, so that solutions are environmental friendly.
B2.15	As a head of urban planning, I want to create the reports as automatically as possible, so that the time spend on them can be reduced.
B2.16	As an operational urban planner, I want the system to be usable for operational planning, so that I can give advice to the police on how to handle situations.
B2.17	As a head of urban planning, I want the system to support with emergency operations, so that I can give advice to the police on how to handle situations.
B2.18	As an operational urban planner, I want to be able to see the current situation, so that I can compare it against older data.

Table 44: C1 User Stories

ID	**User Story**
C1.1	As an enterprise architect, I want a new hospital management system, so more than 32 different systems are integrated into one.
C1.2	As an enterprise architect, I want to ensure that patient safety is a top priority, so that the patients are taken care of.
C1.3	As a doctor, I want a prescription written by me to be automatically transferred to a pharmacy information system, so that I don't have to do that manually.

continues on next page

C1.4	As an enterprise architect, I want the system to be integrated with external parties, so that everything becomes automated.
C1.5	As a patient, I want to have access to the system, so that I am integrated into the system.
C1.6	As an enterprise architect, I want to have one global database that is accessible by all the stakeholders, so that I have the most updated data available.
C1.7	As an enterprise architect, I want to have one global database that is accessible by all the stakeholders, so that I don't have discrepancies between data.
C1.8	As an enterprise architect, I want to have one global database that is accessible by all the stakeholders, so that work and costs are reduced.
C1.9	As an enterprise architect, I want the electronic medical record to be integrated into the system, so that it is accessible by all who need it.
C1.10	As an enterprise architect, I want the appointment scheduling to be integrated into the system, so that all appointments are handled in one system.
C1.11	As an enterprise architect, I want mobile devices to be integrated into the system, so that gathered data by these devices is automatically transferred into the medical record.
C1.12	As an enterprise architect, I want the prescription management system to be integrated into the system, so that a patient can request a new prescription through the system.
C1.13	As an enterprise architect, I want the business analytics component to be integrated into the system, so that I can do predictive and prescriptive analytics.
C1.14	As an enterprise architect, I want bed occupancy rate data to be analysed, so that I can get a better understanding of how to schedule our employees.
C1.15	As an enterprise architect, I want room scheduling data to be analysed, so that I can get a better understanding of how to schedule our employees.
C1.16	As an enterprise architect, I want all analysed data to be anonymized, so that privacy is ensured.
C1.17	As an enterprise architect, I want to have a good security for the system, so that we follow the GDPR standards.

continues on next page

C1.18	As a user, I want to have my own account that I can use to access the system, so that I can access everything that I need to access while only using one account.
C1.19	As a patient, I want to schedule an appointment online, so that I don't have to call.
C1.20	As a patient, I want to change my personal data online, so that I don't have to call or come in person when I needs change some of my personal data.
C1.21	As a patient, I want access to my electronic medical record, so that I can check all my test results and other findings.
C1.22	As a patient, I want to have access to the prescription management system, so that I can request a new prescription if needed.
C1.23	As a patient, I want to use my mobile device to access the system, so that I don't have to use my stationary computer.
C1.24	As an enterprise architect, I want the system to integrate all mobile medical devices, so that data gathered by these machines is automatically incorporated.
C1.25	As a doctor, I want to use my mobile device to access the system, so that I can access the system from anywhere in the hospital.
C1.26	As an enterprise architect, I want to collect the start and end dates from appointments, so that I can analyse the average duration of an appointment.

Table 45: C2 User Stories

ID	User Story
C2.1	As a system administrator, I want assign people to a role, so that I can create accounts.
C2.2	As a system administrator, I want to override restrictions, so that I can create special cases for the people that need that.
C2.3	As a system administrator, I want to give rights to accounts, so that I can limit what people can access.
C2.4	As a doctor, I want to only see my own patients and their records, so that information stays clear and private.
C2.5	As an enterprise architect, I want the system to be able to handle 7600 people at once, so that the system doesn't freeze.

continues on next page

C2.6	As an enterprise architect, I want to connect the medical record to all other systems, so that data entries don't have to be made separately for each system.
C2.7	As an enterprise architect, I want one central database, so that all the data is stored in one place.
C2.8	As an enterprise architect, I want to restrict who can access what data, so that users can only access what they are allowed to.
C2.9	As a patient, I want to have a personal ID, so that all my information is linked together with that ID.
C2.10	As a patient, I want to change my personal information in only one place, so that my personal information is updated everywhere in the system.
C2.11	As a patient, I want to view my personal information, so that I can check if everything is correct.
C2.12	As a patient, I want to see laboratory results, so that I can see the results of the tests that have been done on me.
C2.13	As a patient, I want to get a new prescription through the system, so that I don't have to go through the doctor for that.
C2.14	As a patient, I want to connect any band that I wear to the system, so that the system can check my lifestyle and how healthy I am through the data that it receives from these bands.
C2.15	As a patient, I want to determine who can access my data, so that no unauthorised people can see my data.
C2.16	As an enterprise architect, I want all data to be secure, so that all privacy issues are considered.
C2.17	As an enterprise architect, I want that every time someone leaves their computer, tablet or phone that person is locked out, so that security of the system is ensured.
C2.18	As an enterprise architect, I want every user to have a unique login, so that privacy is ensured.
C2.19	As an enterprise architect, I want every user to log in with a password or some form of security, so that privacy is ensured.
C2.20	As a laboratory technician, I want to access the patients medical record, so that I can see what kind of test the doctor ordered.
C2.21	As a laboratory technician, I want to access the patients medical record, so that I can see if the test is urgent.

continues on next page

C2.22	As a laboratory technician, I want to access the patients medical record, so that I can insert test results.
C2.23	As a laboratory technician, I want to add a note, so that I can add additional information I want to share.
C2.24	As a doctor, I want to access the patients medical record, so that I can see the test results.
C2.25	As a doctor, I want to access the patients medical record, so that I can see the progress on a test.
C2.26	As an enterprise architect, I want to add new data sources to the patient medical record if needed, so that I can add more data fields on the patient.
C2.27	As a doctor, I want to access the patient medical file, so that if I perform a test I can enter that information.
C2.28	As a doctor, I want to access the patient medical file, so that if I perform an examination I can enter that information.
C2.29	As an enterprise architect, I want mobile devices to input data into the medical record, so that if the patient does a CT scan the results are automatically put into the record.
C2.30	As an enterprise architect, I want all data to be available 24/7, so that everybody can access the data all the time.
C2.31	As an enterprise architect, I want the data to be backed up, so that nothing gets lost in case of a crash.
C2.32	As an enterprise architect, I want a solution that does not share information with any company, so that privacy is ensured.
C2.33	As an enterprise architect, I want the system to respond relatively fast, so that in an emergency situation I don't have to wait for the system to load.
C2.34	As a doctor, I want to add information to any test results, so that I can add notes to the results.
C2.35	As a patient, I want my electronical medical record to have tabs, so that the doctor can easily add and see results.
C2.36	As a doctor, I want to select options for special cases, so that I can add if the patient was in a hurry when taking their blood pressure.
C2.37	As a receptionist, I want to have access to the typical schedules of the doctors and nurses, so that I can schedule on availability.

continues on next page

C2.38	As a receptionist, I want to know what rooms are available, so that I can schedule on available rooms.
C2.39	As a doctor, I want to only access the medical files of patients that are assigned to me, so that privacy is ensured.
C2.40	As a doctor, I want to access the patients file even if the patient is not there, so that I can check the laboratory results.
C2.41	As a patient, I want to give my preference for a doctor, so that I can pick the doctor I feel most comfortable with.
C2.42	As a patient, I want to have multiple doctors assigned, so that multiple people can help me.

Table 46: C3 User Stories

ID	User Story
C3.1	As an enterprise architect, I want to integrate electronic medical records, so that they are easily available.
C3.2	As a doctor, I want the system to support mobile devices, so that I can access it from my tablet.
C3.3	As an enterprise architect, I want the system to support stationary devices, so that I can access it from my pc.
C3.4	As an enterprise architect, I want the system to support mobile devices, so that I can access it from my phone.
C3.5	As a patient, I want to be integrated with the system, so that I can access it from home.
C3.6	As an enterprise architect, I want the system to be web based, so that everyone can access it through their browser.
C3.7	As a patient, I want to collect data by bands and other mobile devices, so that that data can contribute to my healthcare.
C3.8	As an enterprise architect, I want to integrate the data transmission with pharmacies, so that prescriptions can be send over.
C3.9	As an enterprise architect, I want measurement devices to be integrated into the system, so that medical data is automatically entered into the medical record.
C3.10	As an administrator, I want to change the privacy settings, so that I can adjust for privacy regulations.

continues on next page

C3.11	As a patient, I want to change the privacy settings, so that I can further restrict data exchange.
C3.12	As a user, I want to login with two step authentication, so that my data is secure.
C3.13	As a user, I want to have unique identifier number, so that I can be identified by my unique number.
C3.14	As a user, I want to login with a password, so that my data is secure.
C3.15	As a doctor, I want the login to be very user friendly, so that when I get logged out I can quickly log back in.
C3.16	As an enterprise architect, I want a user to be automatically logged out when they are not using the system anymore, so that nobody can access someone else their account.
C3.17	As a doctor, I want to access the medical record, so that I can view my patients data.
C3.18	As a doctor, I want to access the appointment scheduling, so that I can see who's my next patient.
C3.19	As a doctor, I want to be connected to the laboratory, so that I can request a test.
C3.20	As a doctor, I want to be connected to the pharmacy, so that I can send prescriptions.
C3.21	As a doctor, I want to be connected to the pharmacy, so that I can give information about the intake.
C3.22	As a pharmacist, I want to access the unique identification of a patient, so that if the patient comes I can identify that person and access all the important information such as the dose intake, descriptions and the name of medicine.
C3.23	As an enterprise architect, I want to have a part of the database for pharmacies created, so that we can save certain information such as name, address and maybe contact number.
C3.24	As a receptionist, I want to access the appointment scheduling, so that I can schedule everyone.
C3.25	As a receptionist, I want to access room scheduling, so that I can schedule where an appointment takes place.
C3.26	As a receptionist, I want have access to the schedule of the doctors, so that I can match doctors and patients.

continues on next page

C3.27	As a nurse, I want to access the medical record, so that I can enter information.
C3.28	As a nurse, I want to access the medical record, so that I can prepare the room accordingly.
C3.29	As a nurse, I want to be restricted in my access, so that privacy is ensured.
C3.30	As a system administrator, I want to set access rights, so that people get restricted in their access to the system.
C3.31	As a janitor, I want to have my own login for the system, so that I can see what needs to be fixed.
C3.32	As a doctor, I want to enter information into the system about what tests I want to be done, so that I don't have to go to the laboratory.
C3.33	As a laboratory technician, I want to receive a notification, so that I can see when a test is requested.
C3.34	As a laboratory technician, I want to give status updates on tests, so that the doctor can see the progress.
C3.35	As a laboratory technician, I want to add notes, so that I can give additional information.
C3.36	As a doctor, I want to add notes, so that I can give additional information.
C3.37	As a laboratory technician, I want the messages to be automatic, so that I only have to enter the result.
C3.38	As a patient, I want to see progress updates on tests, so that I can see how long it is going to take.
C3.39	As a patient, I want to see test results, so that I don't have to ask the doctor.
C3.40	As a patient, I want to access the appointment scheduling, so that I can schedule my own appointment.
C3.41	As a patient, I want to access the medical record, so that I can see my medical information and test results.
C3.42	As a patient, I want request prescriptions, so that I can get a refill.
C3.43	As an enterprise architect, I want the system to work 24/7, so that patients and surgeries don't have to wait.

Table 47: C4 User Stories

ID	User Story
C4.1	As an enterprise architect, I want to access the patients medical record, so that I can view the patients information.
C4.2	As an enterprise architect, I want to enter information into the medical record, so that I can add new information on a patient.
C4.3	As an enterprise architect, I want to have bed room management, so that I can see who is assigned to what room.
C4.4	As an enterprise architect, I want to have bed room management, so that I can see what beds are free.
C4.5	As an enterprise architect, I want to have inventory management, so that I can see how much supplies we have.
C4.6	As an enterprise architect, I want a whole new system, so that all old components are replaced.
C4.7	As an enterprise architect, I want to have an overview of the beds we have, so that I can do bed room management.
C4.8	As an enterprise architect, I want to see the occupancy rate of the beds, so that I can see how much each bed is used.
C4.9	As an enterprise architect, I want to see the most updated data, so that patient safety is ensured.
C4.10	As an enterprise architect, I want to access all the data all the time, so that fatal accidents are prevented.
C4.11	As an enterprise architect, I want no downtime, so that data is always available in emergency situations.
C4.12	As an enterprise architect, I want the data to be secure, so that patient privacy is ensured.
C4.13	As a doctor, I want the data to be secure, so that I am not liable.
C4.14	As an administrative staff member, I want data to be updated everywhere, so that I don't have to spend time on duplicating data.
C4.15	As an enterprise architect, I want all data stored one system, so that there is a clear boundary on how the data is restricted.
C4.16	As a patient, I want to have access to the system, so that I can more involved.
C4.17	As an administrative staff member, I want access to inventory management, so that I can check our supplies.

continues on next page

C4.18	As an administrative staff member, I want access to room scheduling, so that I can assign the correct rooms to the correct people.
C4.19	As a nurse, I want to see which patient I have to go to, so that I can take care of that patient.
C4.20	As a nurse, I want to see which bed the patient has to go in, so that the patient ends up in the correct bed.
C4.21	As a nurse, I want to see where the doctor is that I'm working with today, so that I can find them.
C4.22	As a facility manager, I want to see what broken things I need to fix, so that I can go fix them.
C4.23	As an enterprise architect, I want to integrate medical devices in the system, so that data from these devices is automatically entered into the medical record.
C4.24	As an enterprise architect, I want what happens during an appointment to be automatically entered into the medical record, so that this doesn't has to happen manually.
C4.25	As an enterprise architect, I want the system to be available on mobile devices, so that I can use it on my phone or tablet.
C4.26	As a patient, I want the system to be available on mobile devices, so that I can use it on my phone.
C4.27	As a patient, I want to access my medical record, so that I can see my test results.
C4.28	As a patient, I want to access my medical record, so that I can see my medical history.
C4.29	As a patient, I want access to the appointment scheduling, so that I can schedule my own appointments.
C4.30	As a patient, I want to edit my own data, so that I can enter new personal information.
C4.31	As a user, I want to have an unique ID, so that I can only access information linked to my ID.
C4.32	As a patient, I want access to the appointment scheduling, so that I can give the reason for my appointment.
C4.33	As a patient, I want access to the appointment scheduling, so that I can choose my own doctor.
C4.34	As a patient, I want access to the appointment scheduling, so that I reschedule my appointment.

continues on next page

C4.35	As a patient, I want to edit my own information, so that I can change it if something changes.
C4.36	As a receptionist, I want access to the appointment scheduling, so that I can cancel an appointment if needed.
C4.37	As a receptionist, I want access to the appointment scheduling, so that I can edit information for a patient.
C4.38	As a receptionist, I want access to the appointment scheduling, so that I can schedule an appointment for someone.
C4.39	As a doctor, I want access to the appointment scheduling, so that I can see who my next patient is.
C4.40	As a doctor, I want access to the appointment scheduling, so that I can see why my next patient is here.
C4.41	As a nurse, I want access to the appointment scheduling, so that I can see who my next patient is.
C4.42	As a nurse, I want access to the appointment scheduling, so that I can see why my next patient is here.
C4.43	As an enterprise architect, I want the room and bed scheduling to be connected to the appointment scheduling, so that I can easily manage everything from one system.
C4.44	As a patient, I want to access the data that is produced and collected on me, so that I am more empowered.
C4.45	As a facility manager, I want to be included in the bed management, so that I can get an alert if a bed dysfunctions.
C4.46	As a nurse, I want to be alerted if a bed needs to be cleaned, so that I can do that.

F - MLC metric scores per file, divided over Role, Goal Benefit

The tables have been copied from Excel to show the color scale formatting. Darker green means a higher score. The "DIV/0!" errors show because the divisor when calculating the F1 score is a zero.

File	Fragment	Accuracy:	Precision:	Recall:	F1:
I1	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.148	0.190	0.148	0.167
	Benefit	0.000	0.000	0.000	#DIV/0!
I2	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.000	0.000	0.000	#DIV/0!
	Benefit	0.000	0.000	0.000	#DIV/0!
I3	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.119	0.093	0.119	0.104
	Benefit	0.000	0.000	0.024	0.000
I4	Role	0.059	0.026	0.059	0.036
	Goal	0.059	0.026	0.059	0.036
	Benefit	0.000	0.000	0.059	0.000
I5	Role	0.083	0.045	0.083	0.059
	Goal	0.083	0.045	0.167	0.071
	Benefit	0.000	0.000	0.000	#DIV/0!
I6	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.250	0.061	0.250	0.098
	Benefit	0.000	0.000	0.000	#DIV/0!
S1	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.214	0.158	0.214	0.182
	Benefit	0.000	0.000	0.000	#DIV/0!

Figure 31: MLC metric scores with color scale formatting for the Greenfield Case, divided over Role, Goal and Benefit.

File	Fragment	Accuracy:	Precision:	Recall:	F1:
A1	Role	0.023	0.032	0.023	0.027
	Goal	0.136	0.194	0.182	0.188
	Benefit	0.000	0.000	0.000	#DIV/0!
A2	Role	0.105	0.089	0.105	0.096
	Goal	0.184	0.156	0.211	0.179
	Benefit	0.000	0.000	0.000	#DIV/0!
A3	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.054	0.056	0.054	0.055
	Benefit	0.054	0.056	0.081	0.066
A4	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.194	0.143	0.278	0.189
	Benefit	0.028	0.020	0.083	0.033
B1	Role	0.071	0.043	0.071	0.053
	Goal	0.179	0.106	0.179	0.133
	Benefit	0.000	0.000	0.000	#DIV/0!
B2	Role	0.056	0.050	0.056	0.053
	Goal	0.111	0.100	0.222	0.138
	Benefit	0.000	0.000	0.000	#DIV/0!
C1	Role	0.115	0.176	0.115	0.140
	Goal	0.269	0.412	0.269	0.326
	Benefit	0.000	0.000	0.000	#DIV/0!
C2	Role	0.071	0.057	0.071	0.063
	Goal	0.143	0.113	0.214	0.148
	Benefit	0.024	0.019	0.095	0.031
C3	Role	0.093	0.118	0.093	0.104
	Goal	0.233	0.294	0.326	0.309
	Benefit	0.047	0.059	0.047	0.052
C4	Role	0.022	0.059	0.022	0.032
	Goal	0.065	0.176	0.109	0.135
	Benefit	0.000	0.000	0.000	#DIV/0!

Figure 32: MLC metric scores with color scale formatting for the Customization Case, divided over Role, Goal and Benefit.

G - LC metric scores per file, divided over Role, Goal and Benefit

The tables have been copied from Excel to show the color scale formatting. Darker green means a higher score. The "DIV/0!" errors show because the divisor when calculating the F1 score is a zero.

File	Fragment	Accuracy:	Precision:	Recall:	F1:
I1	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.148	0.190	0.148	0.167
	Benefit	0.000	0.000	0.000	#DIV/0!
I2	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.000	0.000	0.000	#DIV/0!
	Benefit	0.000	0.000	0.000	#DIV/0!
I3	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.119	0.093	0.119	0.104
	Benefit	0.000	0.000	0.024	0.000
I4	Role	0.059	0.026	0.059	0.036
	Goal	0.059	0.026	0.059	0.036
	Benefit	0.000	0.000	0.059	0.000
I5	Role	0.083	0.045	0.083	0.059
	Goal	0.083	0.045	0.167	0.071
	Benefit	0.000	0.000	0.000	#DIV/0!
I6	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.250	0.061	0.250	0.098
	Benefit	0.000	0.000	0.000	#DIV/0!
S1	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.214	0.158	0.214	0.182
	Benefit	0.000	0.000	0.000	#DIV/0!

Figure 33: LC metric scores with color scale formatting for the Greenfield Case, divided over Role, Goal and Benefit.

File	Fragment	Accuracy:	Precision:	Recall:	F1:
A1	Role	0.023	0.032	0.023	0.027
	Goal	0.136	0.194	0.182	0.188
	Benefit	0.000	0.000	0.000	#DIV/0!
A2	Role	0.105	0.089	0.105	0.096
	Goal	0.184	0.156	0.211	0.179
	Benefit	0.000	0.000	0.000	#DIV/0!
A3	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.054	0.056	0.054	0.055
	Benefit	0.054	0.056	0.081	0.066
A4	Role	0.000	0.000	0.000	#DIV/0!
	Goal	0.194	0.143	0.278	0.189
	Benefit	0.028	0.020	0.083	0.033
B1	Role	0.071	0.043	0.071	0.053
	Goal	0.179	0.106	0.179	0.133
	Benefit	0.000	0.000	0.000	#DIV/0!
B2	Role	0.056	0.050	0.056	0.053
	Goal	0.111	0.100	0.222	0.138
	Benefit	0.000	0.000	0.000	#DIV/0!
C1	Role	0.115	0.176	0.115	0.140
	Goal	0.269	0.412	0.269	0.326
	Benefit	0.000	0.000	0.000	#DIV/0!
C2	Role	0.071	0.057	0.071	0.063
	Goal	0.143	0.113	0.214	0.148
	Benefit	0.024	0.019	0.095	0.031
C3	Role	0.093	0.118	0.093	0.104
	Goal	0.233	0.294	0.326	0.309
	Benefit	0.047	0.059	0.047	0.052
C4	Role	0.022	0.059	0.022	0.032
	Goal	0.065	0.176	0.109	0.135
	Benefit	0.000	0.000	0.000	#DIV/0!

Figure 34: LC metric scores with color scale formatting for the Customization Case, divided over Role, Goal and Benefit.

H - REFSQ 2023 Paper Draft

Please find the draft on the next page.

Designing a Pipeline to extract User Stories from Requirements Engineering Meeting Transcripts

Hans van Tuin[1], Marcela Ruiz[2], Tjerk Spijkman[3], Fabiano Dalpiaz[1], and Sjaak Brinkkemper[1]

[1] Dept. of Information and Computing Sciences, Utrecht University. Utrecht, the Netherlands
[2] ZHAW. Zurich, Switzerland
[3] Fizor. Utrecht, the Netherlands

Abstract. **[Context Motivation]** The quality of Requirements Engineering meetings and the resulting requirements correlate with the quality of a software product. These requirements can be written in the standardized form of a User Story. **[Question/Problem]** It takes time to manually extract these requirements from the meeting transcripts and write these down as User Stories. The thesis goal is to explore how to automatically extract User Stories from software Requirements Engineering (RE) meeting transcripts. **[Principal ideas/results]** A Transcript to User Story Pipeline (TUSP) is designed. This pipeline combines already existing Requirement Specification Algorithms (RSAs). The TUSP also uses a novel RSA, specifically designed for this thesis, called the Fit-Gap Searcher. The TUSP consists of 2 configurations: a Machine Learning Configuration and a Lexical Configuration. These two configurations are validated qualitatively and quantitatively on two real-world test cases: a Greenfield test case and a Customization test case. Both configurations predict a couple of good quality User Story fragments, but no whole User Story of good quality is found. The configurations are assessed on the precision, accuracy, recall and F1 metrics, all of which score no higher than 0.1 for both of the configurations. **[Contribution]** This thesis presents a novel way of automatically extracting User Stories from RE meeting transcripts. The future research section provides suggestions on improving the RSAs and a future view for combining the TUSP with issue-tracking and project management systems.

Keywords: Requirements Engineering · User Stories · Transcripts · Pipeline

1 Introduction

Every product has certain requirements that are to be met. This goes for software products as well. Sessions for extracting *what* a software product needs to contain, so called Requirements Engineering (RE) sessions, are not to be underestimated, as Mund, Fernandez, Femmer and Eckhardt show that the quality

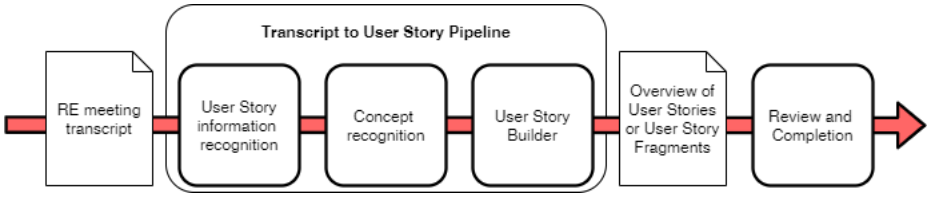


Fig. 1. High-level view of the TUSP.

of RE sessions correlates with the quality of the resulting software product [3]. However, as Ruiz and Hasselman state, these RE sessions take time, and this time could be reduced through automatization [4] .

This paper presents a novel approach for automatically extracting User Story (US) fragments from RE meeting transcripts. This is done through a pipeline of components called the Transcript to User Story Pipeline (TUSP). The pipeline will take the transcript from an RE meeting or interview as input and output a list of User Stories (USs) and US fragments. The TUSP will consist of multiple segments and is thus called a pipeline. Figure 1 depicts a high level view of the TUSP. It shows how the TUSP takes a transcript from an RE meeting as input and uses multiple pipeline segments to output an overview of USs. The pipeline needs to recognize which parts of the transcript contain information on USs. The pipeline also needs to be able to recognize certain concepts which can help build the USs. All the available information is then used by the User Story builder to build USs or fragments of USs. The pipeline will not achieve a 100% precision and recall and thus the overview USs and US fragments will have to be reviewed for completeness and be complemented manually. As Abualhaija, Arora, Sabetzadeh, Briand and Traynor describe, is the cost of misclassification not symmetrical when classifying requirements [1]. It is theorized that the cost of removing a false positive (something that is not a requirement but is classified one) is much lower than missing a false negative (not classifying a requirement as a requirement). Thus a high recall combined with an acceptable precision is sought after [1, 7].

2 Design of the Transcript to User Story Pipeline

All lines and thus turns in every transcript are coded with a unique ID to ensure traceability. This ID consists of the file name, the abbreviation of the role of the actor whose turn it is and the line number in the whole transcript. The different parts of the ID are divided by a "/". An example of the traceability IDs can be seen in figure 2. The abbreviation for the role is included as a backup for when the Word Classifier cannot find an actor in a specific turn. These abbreviations consist of two letters and are case sensitive. The abbreviations are saved in an excel (.xlsx) worksheet.

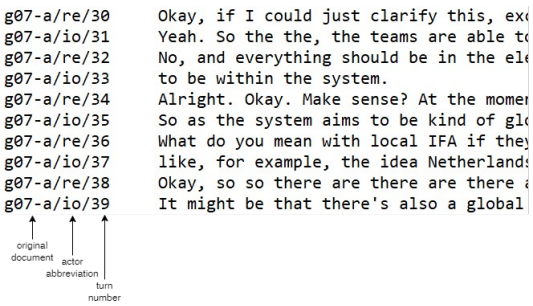


Fig. 2. Example of the traceability IDs. re stands for Requirements Officer, io stands for IFA information officer.

A Deep Learning (DL) and a Machine Learning (ML) classifier have been compared on a combination of generated text and real interviews. The DL classifier performed better than the ML classifier on the recall metric and thus has the choice been made to include the DL classifier in the TUSP. Furthermore are a Domain Concept Extractor, a Requirement Word Classifier, a Fit-Gap Searcher and a User Story builder are included in the TUSP design.

Since the different components of the TUSP influence one another, it is envisioned that a user may not want to use all the components. This can be dependent on the result that the user is looking for. Therefore two TUSP configurations have been made, the Machine Learning Configuration (section 2 and the Lexical Configuration (section 2.

The Machine Learning Configuration (MLC) The Machine Learning Configuration uses the Sentence Classifier, the Concept Extractor, the Word Classifier and the User Story Builder components. It is called the Machine Learning Configuration because it uses both a Deep Learning and a Machine Learning component. The transcript is first processed by the Sentence Classifier, which extracts all the turns that are supposed to have information on a User Story. These turns are then processed by the Concept Extractor, which extracts the known and unknown concepts. Then, the Word Classifier indicates which words in a turn belong in which User Story part. If nothing of interest is found then a User Story part is left blank.

The Lexical Configuration (LC) The Lexical Configuration uses the Concept Extractor, the Fit-Gap Searcher, the Word Classifier and the User Story Builder components. The transcription is first processed by the Concept Extractor, which extracts the known and unknown concepts. Then, the Fit-Gap Searcher extracts the relevant turn sections. If the Fit-Gap Searcher does not find anything, then the Word Classifier will be used to try and fill in the gaps. If the Word Classifier also does not result in anything then that part of the User

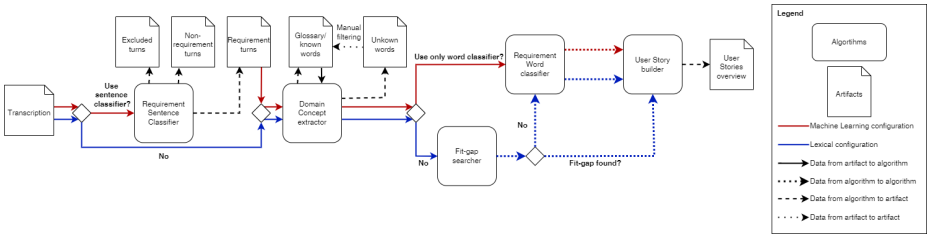


Fig. 3. Overview of the TUSP and its two configurations.

Story is left blank.

The whole TUSP and its two configurations can be seen in figure 3. This figure shows that there are four ways that data can flow: from algorithm to algorithm, from artifact to artifact, from algorithm to artifact and vice versa. All of data flows from artifact to algorithm and from artifact to artifact in the figure have to be done manually at the moment. All of the data flows from algorithm to artifact and from algorithm to algorithm are automated and do not require manual interference.

3 Validation

The TUSP has been validated on two use cases, named the Greenfield case (GC) and the Customization case (CC). Both cases consisted of transcribed interviews from which USs have been extracted, which we used as a ground truth to compare the TUSP results against.

The MLC starts with running a transcript through the Sentence Classifier where all turns that do not contain User Story information should be filtered out. The metric scores have been calculated on the Customization Case and the Greenfield Case separately, which can be seen in table 1.

Table 1: Metric scores of the Sentence Classifier on the two use cases

Metric	**GC**	**CC**
Accuracy	0.51	0.50
Precision	0.36	0.18
Recall	0.70	0.40
F1	0.47	0.25

These tables show that the Sentence Classifier performed worse on the GC than on the CC for the precision, recall and F1 metrics.

The turns that were predicted by the Sentence Classifier to contain User Story information were then used as input for the Word Classifier. The output is analysed both qualitatively and quantitatively in the following sections.

MLC - Qualitative analysis. The results of the MLC configuration are analysed on User Story completeness, User Story quality and traceability.

User Story Completeness. In both Cases no predicted User Story exactly matched a complete Ground Truth User Story. Only fragments were predicted, ranging from 0 to 2 US fragments in a predicted US. In file B2 not a single Ground Truth fragment is predicted.

User Story Quality. The US quality is analysed based on the Quality User Story Framework [2]. In the Customization Case, four User Story candidates were counted which included at least a role and a goal (*well-formed*). One of these expresses a requirement for exactly one feature (*atomic*). The other three describe more than one feature. In the end, there was one US with the best US qualities, from file C3:

> *NoTurnIndex As a doctor [and the doctor], I am able to see if the test is in progress*

This User Story candidate is considered *minimal* meaning it contains nothing more than role, goal and reason (even though it contains the same role twice). The goal fragment of the candidate expresses a feature (*conceptually sound*) and is *problem-oriented*. It is also *conflict-free*, *scalable*, *unique* and *independent*. However, it is not *unambiguous* as some doubt could arise about what kind of test is meant. It is also not a *full sentence* in the sense that it is well-formed, the role contains a repetition of the fact that it concerns the doctor.

Regarding all the User Story candidates it can be said that they are not *complete* and do not account for *explicit dependencies* and most of all, are not *uniform*.

Traceability. To ensure traceability should every US candidate should have a traceability ID that refers back to the original turn or turns that the candidate was based on. However, not every US candidate outputted by the MLC configuration has a traceability ID. This goes for both the Customization Case and the Greenfield Case.

MLC - Quantitative analysis. The resulting files with predicted User Stories were loaded into Nvivo12 so they could be coded and thus quantified. Both the Ground Truth User Stories and the predicted User Stories were split into the separate User Story fragments: *role, goal and reason*. If a predicted fragment matched a Ground Truth fragment it would be counted as a True Positive. To give an example: if a predicted User Story contained a goal that matched with

a goal of a Ground Truth User Story, this would be counted as a True Positive. Since the Word Classifier only predicts positives, only the True Positives and False Positives were counted. To calculate the recall the total number of Ground Truth User Story fragments that appeared in the predicted User Stories were counted as the sum of the True Positives and all the fragments that did appear in a prediction but as a wrong fragment. For instance, if a fragment was predicted as a role fragment, but in the Ground Truth, it appears as a goal fragment. This sum is then divided by the number of relevant fragments in the Ground Truth. Table 2 shows the total number of Ground Truth User Story (GT US) fragments per case and the total number of predicted User Story (P US) fragments per case. Table 3 and the accuracy, precision, recall and F1 metrics.

Table 2: Number of Ground Truth User Story fragments against the number of predicted User Story fragments that resulted from the MLC configuration

Case	**Ground Truth US**	**Predicted US**
Customization	1074	571
Greenfield	384	362

Table 3: Quantitative metrics of the MLC configuration

Case	**Accuracy**	**Precision**	**Recall**	**F1**
Customization	0.012	0.023	0.034	0.027
Greenfield	0.013	0.014	0.031	0.019

Using a two-sided t-test with unequal variances and an alpha of 0.05 shows no significant difference in the accuracy, precision, recall and F1 numbers of the two cases.

Results of the Lexical Configuration

LC - Qualitative analysis. The results of the LC configuration are analysed on User Story completeness, User Story quality and traceability.

User Story Completeness. In both Cases no predicted User Story exactly matched a complete Ground Truth User Story. File I2 produced no correct predictions for any Ground Truth fragment. However, in some cases, the predicted US contained all the elements of the matching Ground Truth US but were these elements not placed in their correct respective US fragment. An example is the predicted US B2/hu/41:

B2/hu/41 As a head of urban planning, I want to tell people I need to be able to convince everyone that we rely on real data I cannot tell them we just invented how many cars are moving around , so that I need

Which matches Ground Truth US B2.8:

B2.8 As a head of urban planning, I want to convince everybody that we rely on real data, so that I show that we didn't just invent how many cars are moving around.

In this case, the role was correctly predicted, the predicted goal contains both the Ground Truth goal fragment and the Ground Truth reason fragment and the predicted reason contains nonsense. However, concerning a high recall and filtering all the relevant information from a whole transcript, B2/hu/41 can be seen as a good prediction.

User Story Quality. The US quality is analysed based on the Quality User Story Framework [2]. For the Customization Case were 12 predicted User Stories *well-formed*. 8 of these 12 predicted User Stories only contain a requirement for 1 feature and are thus *atomic*. The description for a *minimal* US is that the US contains nothing more than a role, goal and reason. But since there are no User Stories that contain all three of these elements correctly the choice has been made to filter the predicted User Stories on whether the role and the goal are minimal. This leaves 4 of the 8 predicted User Stories that are *well-formed*, *atomic* and *minimal*, from which one example is given:

C4/ea/54 As an enterprise architect, I want to the system to be available on mobile devices , so that has kind of two sides So once we need

None of these four is *conceptually sound* as all four contain a reason fragment that describes something else than a rationale. C4/ea/54 can be seen as *problem-oriented*. The candidates is also *conflict-free*, *unique* and *independent*. However, the candidates is not a *full sentence* as it contains grammatical errors and loose words or uncompleted sentences.

Traceability. In both of the cases do all of the US candidates contain a traceability ID.

LC - Quantitative analysis. The metrics were calculated in the same way as for the MLC configuration. Table 4 shows the total number of Ground Truth User Story (GT US) fragments per case and the total number of predicted User Story (P US) fragments per case. Table 5 shows the accuracy, precision, recall and F1 metrics.

Table 4: Number of Ground Truth User Story fragments against the number of predicted User Story fragments that resulted from the LC configuration

Case	Ground Truth US	Predicted US
Customization	1074	1047
Greenfield	384	735

Table 5: Quantitative metrics of the LC configuration

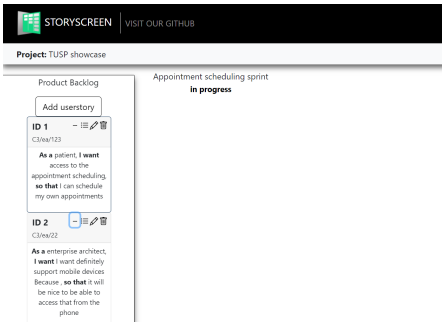
Case	Accuracy	Precision	Recall	F1
Customization	0.074	0.076	0.096	0.085
Greenfield	0.047	0.024	0.055	0.034

Using a two-sided t-test with unequal variances and an alpha of 0.05 shows no significant difference in the accuracy, precision, recall and F1 numbers of the two cases. The table above shows that for the Customization Case almost as many US fragments were predicted as there were Ground Truth fragments. For the Greenfield Case, however, almost double the fragments were predicted than were possible. A possible explanation is that even though actors were not talking about requirements, there were possibly still using a lot of trigger words on which the Fit-Gap searcher had a hit.

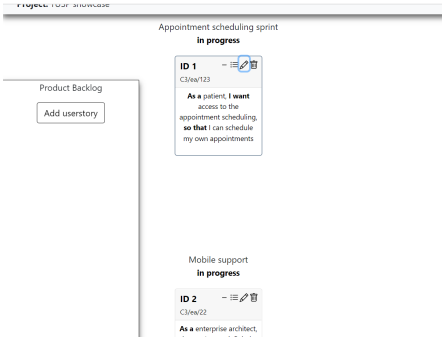
4 Research Vision

The TUSP has been designed to function as a tool to aid the Requirements Engineer in the extraction of USs from transcripts. However, the USs and US fragments produced by the TUSP are not the finished product. The output from the TUSP can be used in issue-tracking systems and project management systems such as Jira and Backlog or perhaps a self-developed system such as Storyscreen, which is made by students from the ZHAW Zurich specifically for managing USs.[4][5][6] For instance, an idea can be that the TUSP gets hooked up to the back-end of the Storyscreen. Thus, when a transcript gets uploaded, all US candidates would automatically be extracted and uploaded to the Storyscreen where they would appear in the backlog as can be seen in figure 4.

[4] https://www.atlassian.com/nl/software/jira
[5] https://www.backlog.com/
[6] https://www.storyscreen.ch/

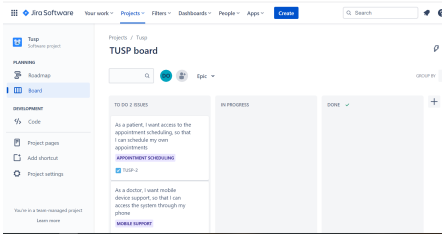


Example of TUSP results loaded into Storyscreen

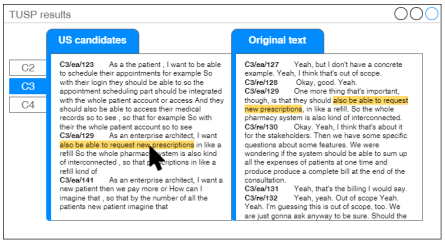


Example of TUSP results used in Storyscreen

Once all the US candidates have been loaded into the backlog these can be filtered and adjusted and finally they can be dragged into the appropriate sprint column as can be seen in figure 4. Another example is being able to automatically load the US candidates into Jira. An example can be seen in figure 4 where the candidates appeared in the To-Do list. Ultimately it should be possible to automatically add the Epics (in this example 'appointment scheduling' and 'mobile support') to which the US candidates belong.



Example of TUSP results loaded into Jira



TUSP mock-up.

Regarding traceability, it is envisioned that the TUSP will output a set of US candidates, which can be compared against the original transcript to show where the US or US fragment originates from. Figure 4 shows a mock-up of a program where the TUSP results can be shown on the left and the original text is shown on the right. If the user then hovers with their mouse over a US fragment it is shown by highlight where the fragment originates from.

Finally, there is a vision to track changes in USs over multiple interviews. This would mean being able to recall and combine with US candidates from previous interviews when analysing a new interview. The example above would then show the traces to multiple transcripts and show how the US has evolved.

5 Future recommendations

The following list contains points that can be used as a way of improving the qualitative and quantitative performance of the TUSP. The main points are listed below:

- **Research on other Requirements Engineering methods for eliciting requirements with multiple stakeholders:** The used elicitation methods for this thesis, interviews and meetings, have proven to be very unstructured. Therefore, it can be worthwhile to research elicitation methods which follow a structured question-answer format. Or perhaps a method which follows a more strict role-goal-reason format. This could help ML classifiers with training in a more structured manner or lexical classifiers with finding more precise hits.
- **Use external databases for giving semantic information on words:** There are readily available online databases such as Wordnet for lexical information and Framenet for semantic information.[7][8] Using lexical and semantic information could lead to the algorithms better understanding of the words used in a transcript and the relationships between these words. Because as of right now do words not have any value or meaning for an algorithm. Research into combining these databases with the TUSP or TUSP components could show if these databases could lead to improvement.
- **Take environmental impacts into account:** Training a neural network for NLP can be very computationally intensive and thus cost a lot of energy, as [6] show in their paper. As the ELMO model from the Word Classifier cannot be saved and thus has to be trained every time the TUSP runs, it is not hard to imagine that over time an environmental impact is created which has to be taken seriously. Rule-based models, such as the Fit-Gap Searcher do not require any form of training and can therefore be a solution when one wants to look for an alternative instead of machine-learning-based models.
- **Focus on unsupervised models:** If one chooses to ignore the previous point due to various reasons, it is recommended to focus on unsupervised learning models.
- **Allow the Concept Extractor usage of .owl format files:** Currently can the concept extractor only read known concepts from a .txt format file. The .owl format is the most often used type for writing down ontologies. If the concept extractor is modified to use .owl format files it could be used as it was intended: as an ontology crawler which can get an ontology as input and output all the matching concepts.
- **Process multiple turns simultaneously:** Whereas the MLC configuration can create US candidates from multiple turns, is the Fit-Gap Searcher currently not able to. Being able to process multiple turns is for instance important when the RE engineer rewords a requirement or gives an example and asks for a confirmation. Whether the rewording or example can then be seen as a new requirement then depends on the answer of the client.
- **Process multiple requirements in one turn:** In opposition to the previous point refers this point to when a turn contains multiple requirements. For instance, if a client lists multiple small features or if the client lists the most important features of the system to-be.

[7] https://wordnet.princeton.edu/
[8] https://framenet.icsi.berkeley.edu/fndrupal/

– **Research on Fit-Gap trigger words:** The current Fit-Gap Searcher is based on the research by [5]. For this research, the authors analysed 12 hours of RE meetings at the company of one of the authors. This means that the fit-gap trigger words extracted are context-dependent on the company that these words were extracted from. Thus, more research on other RE meetings in other contexts could perhaps lead to more trigger words which can in turn be used to improve the Fit-Gap Searcher.

6 Discussion

This section will handle possible threats to the validity of the research done. There was some ambiguity when labelling the test files for the comparison of the two sentence classifiers. There were no guidelines written down by Ruiz and Hasselman and thus it was assessed by the author himself if a turn contained no requirement information, functional requirement information or non-functional requirement information [4]. This also poses the next challenge which is that the definition of a non-functional requirement still varies depending on what literature is checked. The training data for both the Deep Learning Sentence Classifier and the Machine Learning Sentence Classifier was already labelled, while some of the validation data had to be relabelled to fit the for needed to be used as input. This could have led to cases with both of the classifiers where wrong classifications are the result of a different way of labelling instead of how well the classifier performs. This is thus a threat to internal validity. The biggest threat to internal validity is how the RE interviews were conducted. There is a possibility that if the interviews were held differently or with different questions, the results would have been different. Therefore it is impossible to say if the results are causally produced by the TUSP. The second threat to internal validity is that the USs that were used as a ground truth have not been checked by an independent source. Since the USs were made after the construction of the TUSP, these are susceptible to bias. In the same spirit has the labelling of the US candidates not been checked by an independent source. This makes the labelling susceptible to confirmation bias. For instance, the author can label a fragment as a correct prediction of a Ground Truth US fragment, while in reality, this was not what the actor talking during the turn meant.

The biggest threat to external validity is that the TUSP has been validated in only 2 contexts. To get a better idea of how generalizable the TUSP is, it is recommended to apply the TUSP to more contexts. Another threat to external validity is that the TUSP has not been compared with other tools. However, since the TUSP is so innovative and at the moment there do not exist any other tools that encompass the whole process from transcript to US it is not possible to compare the TUSP with other tools at the moment.

7 Conclusion

In conclusion, there is a lot that can be improved on the TUSP. However, it has been shown that the use of a pipeline with multiple components to extract USs from software Requirements Engineering meeting transcripts can result in multiple correct US fragments which can assist in the RE process.

References

1. Abualhaija, S., Arora, C., Sabetzadeh, M., Briand, L.C., Traynor, M.: Automated demarcation of requirements in textual specifications: a machine learning-based approach. Empirical Software Engineering **25**(6), 5454–5497 (2020)
2. Lucassen, G., Dalpiaz, F., Van Der Werf, J.M.E., Brinkkemper, S.: Forging high-quality user stories: towards a discipline for agile requirements. In: 2015 IEEE 23rd international requirements engineering conference (RE). pp. 126–135. IEEE (2015)
3. Mund, J., Fernandez, D.M., Femmer, H., Eckhardt, J.: Does quality of requirements specifications matter? combined results of two empirical studies. In: 2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). pp. 1–10. IEEE (2015)
4. Ruiz, M., Hasselman, B.: Can we design software as we talk? In: Enterprise, Business-Process and Information Systems Modeling, pp. 327–334. Springer (2020)
5. Spijkman, T., Dalpiaz, F., Brinkkemper, S.: Requirements elicitation via fit-gap analysis: A view through the grounded theory lens (2021)
6. Strubell, E., Ganesh, A., McCallum, A.: Energy and policy considerations for deep learning in nlp. arXiv preprint arXiv:1906.02243 (2019)
7. Winkler, J.P., Grönberg, J., Vogelsang, A.: Optimizing for recall in automatic requirements classification: An empirical study. In: 2019 IEEE 27th International Requirements Engineering Conference (RE). pp. 40–50. IEEE (2019)