DOI: 10.15439/2019F254
ISSN 2300-5963 ACSIS, Vol. 19

Delta Analyzer: Tool-based Evaluation of Modified Requirements for an Efficient Development Effort Estimation in the RFQ Process

Konstantin Zichler
Advanced Engineering Projects
HELLA GmbH & Co. KGaA
Lippstadt, Germany
Email: konstantin.zichler@hella.com

Felix Ritter, Aaron Schul
Department of Computer Science
University of Applied Sciences and
Arts, Dortmund, Germany
Email: {felix.ritter001, aaron.schul002}
@stud.fh-dortmund.de

Steffen Helke
Department of Electrical Engineering
and Information Technology
South Westphalia University of Applied
Sciences, Hagen, Germany
Email: helke.steffen@fh-swf.de

***Abstract*—Once an automotive OEM decides to source a new component, a Request for Quotation (RFQ) is send to potential suppliers. Among other documents the RFQ contains a Component Requirements Specification (CRS), which describes the properties of the desired component. As a next step, the supplier has to evaluate the requirements and other boundary conditions of the RFQ and to provide an offer to the OEM. In case the supplier already developed a similar component in the past, it is possible to compare the CRS of the predecessor product with the actual CRS, to estimate the additional development effort. This activity is known as the delta analysis. Since no sufficient tool support is offered, this activity is still a predominantly manual task. The main challenge arises from the fact, that specification documents within the RFQ are provided in different office formats, written by different authors and therefore cannot be compared automatically with the CRS from the predecessor product. In our previous work, we presented the Requirements to Boilerplates Converter (R2BC), which automatically converts random natural language requirements into a predefined syntax. The aim of the approach is to facilitate a subsequent tool-based delta analysis. Consequently, we hereby introduce our proprietary developed Delta Analyzer (DA). This tool is based on Natural Language Processing (NLP) and allows to compare automatically two random specification documents. Moreover, the DA prioritizes requirements deltas according to their impact on development effort. As an output of the DA requirements engineers receive a delta report, which outlines the major differences between the requirements of the two CRS. We validate our approach by experiments on real-life specification documents.**

Keywords: Requirements engineering, delta analysis, natural language processing, requirements delta prioritization

I. Introduction

SUPPLIER development projects are often triggered by a Request for Quotation (RFQ) of an OEM. Within an RFQ, OEMs provide information about a desired component, which shall be delivered by the supplier. Together with project related information like target vehicles, Start of Production (SOP) dates and product volumes, each RFQ contain a Component Requirements Specification (CRS). This specification describes all requirements for the desired component. Based on this document the supplier project team evaluates during the RFQ phase, whether to quote for the RFQ or not. A schematic illustration of a typical RFQ process is depicted in Fig. 1. According to this process, the project team analyses the CRS of the OEM. The target of this evaluation is to estimate the effort, which is necessary to develop the component. To this end, a delta analysis is performed. Such analysis is the activity of comparing two requirements specifications to determine the differences, namely the deltas, between the listed requirements. This procedure is conducted, in case a successor of an already available product is to be developed and the requirements specifications of both the successor and the predecessor products are available. Because the effort to fulfill the requirements of the predecessor component is already known, the project team now only has to estimate the additional effort to fulfill the requirements of the successor CRS [1]. In the next step the effort estimation of the project team is used to provide an offer for the requested product to the OEM.

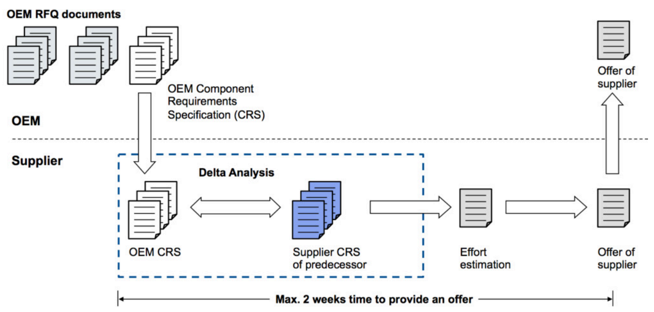

OEM RFQ documents
OEM Component Requirements Specification (CRS)
OEM
Supplier
Offer of supplier
Delta Analysis
OEM CRS
Supplier CRS of predecessor
Effort estimation
Offer of supplier
Max. 2 weeks time to provide an offer

Fig. 1. Schematic Illustration of the RFQ Process

Today, project teams are required to finish the RFQ phase within two weeks or even faster. To increase the efficiency of this process, project teams focus on the top ten requirements of a product. The top ten requirements describe the major characteristics of a product and have therefore a high impact on the development effort. The prioritization of these requirements and concentration on the evaluation of them, gives the project team the chance to quickly answer the question, whether the predecessor component is capable to fulfill the OEM requirements at all or a totally new development is necessary. Through the years, technical experts gathered knowledge on the effort necessary to adapt a given component according to changed customer requirements. By reading a CRS, experts are able to estimate whether a change in a parameter requires a new subcomponent and consequently the rework of the whole system. If for example, a given actuator with the specific characteristic of 5 Nm torque is required by the new OEM CRS to provide 10 Nm, experts can deduce that the fulfillment of this requirement would require a new motor. Since the motor is a major component of the actuator, further components like the gear may have to be adapted, as well. Following this type of consideration, technical experts can estimate the effort necessary to develop the changed component.

The delta analysis is still a predominantly manual task, which requires a lot of resources. Technical experts of the project team read up to several hundreds of pages and compare the predecessor CRS with the successor CRS. This work is tedious and time consuming. Currently, no sufficient tool support exists, that could support the project teams properly. Common tools, which are used by the industry allow the delta analysis for two states of the same document. To this end requirements, attributes and other information are stored as objects in a database. All changes made to these objects can be made visible by the database management system. The presented situation in the RFQ phase requires further functionality from a tool support. Time and again new requirements specifications are submitted to the supplier. These documents are written by various unknown authors and are provided in common office formats (e.g., PDF). Within the delta analysis these documents must be compared to the supplier specification, which was also written by another author. Therefore, a tool support is required, which can determine the requirements deltas between different documents.

In this work, we present our concept for an automated delta analysis and the architecture of the corresponding tool – the Delta Analyzer (DA). This tool allows the comparison of two completely different requirements documents. It uses Natural Language Processing (NLP) techniques, which is the basis of its flexibility. Also, we use our proprietary developed tool R2BC, which we presented in our previous work [1] as a prerequisite for the automated delta analysis. Based on NLP the R2BC converts random natural language requirements into predefined sentence structures, called boilerplates. Moreover, our concept for an automated delta analysis includes an algorithm for the prioritization of requirements deltas. This function prioritizes requirements deltas in the delta report in accordance to their impact on the development effort. All in all, our approach aims to decrease the amount of work needed during the RFQ phase.

The remainder of this work is structured as follows. Chapter II gives an overview of NLP techniques, which we apply in our approach. In Chapter III we present our concept for the automated delta analysis and the methodology for the prioritization of requirements deltas. We present the result of preliminary experiments with the DA in Chapter IV. In Chapter V, we give an overview of related work. Chapter VI summarizes the presented work and gives an outlook on our next research activities.

II. Fundamentals

A. NLP

Natural language processing is to be understood as the automated or semi-automated processing of natural language with the help of a computer. NLP connects various scientific fields. These are mainly linguistics and computer science, but there are also many links to psychology, philosophy, mathematics and logic [2].

The goal of NLP is usually the extraction of information out of a text through precise text analysis. This information is attached directly to the text via so-called annotations, which can later be used to define properties in program objects. By combining linguistics and computer science, NLP generally faces the challenge of having to solve both inherent problems of language and problems of automation. Linguistic issues include aspects such as ambiguity, sarcasm and slang or sayings. These can often not be explained solely by the given text, but are based on context and situation. In the implementation, it can then be hard to find consistent regularities as a basis for automation despite the linguistic features. But, if these challenges are understood correctly and are solved accordingly, NLP allows efficient text processing and information extraction, which can be used in a variety of applications, such as component requirements analyses.

B. GATE

One of the most commonly used tools for creating NLP applications is the *General Architecture for Text Engineering* (GATE). It allows for documents to be imported and then processed by a pipeline of different processing resources. Important processing resources used in the R2BC include:

1. Tokenizer: The tokenizer subdivides the text superficially into characters or strings categorized as numbers, words (case sensitive) and punctuation. This adds the token annotation to the affected areas, each containing type (number, word, etc.) and length in characters. The processing by the tokenizer should be as efficient as possible and only serve as a basis for future grammatical rules.

2. Gazetteer: A Gazetteer contains relevant proper names from certain areas. It is a nested list containing known proper names for each area (for example, cities, airports, football clubs). Gazetteers are used to recognize domain specific entities, as they are not found in a regular dictionary. The ready-made lists are therefore to be expanded or created individually depending on the document. If a list element is recognized in the text, it gets the lookup annotation by default. However, this can already be changed and determined depending on the list in the Gazetteer.

3. Sentence Splitter: The sentence splitter divides the entire text into individual sentences, which will wrap every sentence in a sentence annotation. This is needed later on for the Part-of-Speech Tagger. The splitter uses a gazetteer with abbreviations to distinguish punctuation, such as question marks, exclamation points, and points from other special characters, such as semicolons and colons.

4. POS-Tagger: The Part-of-Speech tagger annotates all words and symbols according to their grammatical function. No new annotation will be created for this, but the important category feature will be added to the token annotations. This is fundamental for grammatical analysis of sentences. It uses standard dictionaries and rules based on the corpus training of the Wall Street Journal.

Using these standard annotations, very customizable JAPE transducers can be implemented to find and attach the information needed for the specific application. These processing resources operate on the *Java Annotation Patterns Engine* (JAPE), which tests regular expressions of existing annotations and can create further annotations or executes java code according to the result.

JAPE transducers are implemented by JAPE rules consisting of a head (various parameters), a left hand side/LHS (regular expression) and a right hand side/RHS (result upon confirmation of LHS). Therewith, standard annotations the ANNIE system generates can be extended using customized rules in a JAPE transducer. The general composition of the processing pipeline consisting of both text annotation and JAPE transducers is given in [3].

III. Concept

In this chapter we present our concept and tool support for an automated delta analysis. As a first step, we give an overview of the main components of our tool chain consisting of the Requirements to Boilerplates Converter (R2BC) and the Delta Analyzer (DA) as depicted in Fig. 2. According to this concept the R2BC is used to convert random natural language requirements into predefined boilerplates. Once all requirements are available in the predefined syntax, the DA is used to perform an automated delta analysis. In our previous work [1], we present the concept of the R2BC together with the test results of the first implementation. In this chapter, we describe our concept for an automated delta analysis and the architecture of the corresponding tool the DA. Moreover, our concept for an automated delta analysis includes an algorithm for the prioritization of requirements deltas. The description of this algorithm constitutes the third part of this chapter.

A. Methodology of Prototypical Tool Chain

1. Requirements to Boilerplates Converter: The process starts once an OEM submits a CRS to the supplier. In a first step, natural language requirements are translated into predefined boilerplates by the R2BC. After processing, the specification of the OEM is available in a semi-formal language. The semi-formal format offers advantages over natural language. This version of the specifications can then be used for the development process. The aim of the R2BC is therefore to improve the readability or the potential of the requirements through formalization. On this basis, the content of the specifications can then be better understood by the developers at the supplier.

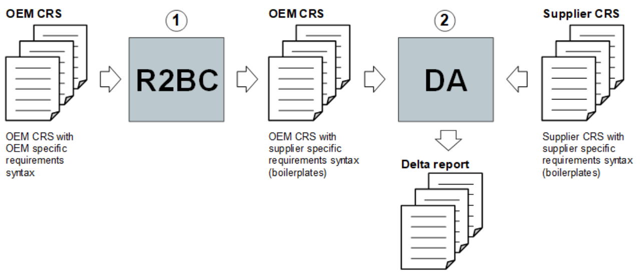

OEM CRS
1
R2BC
OEM CRS
2
DA
Supplier CRS
OEM CRS with OEM specific requirements syntax
OEM CRS with supplier specific requirements syntax (boilerplates)
Delta report
Supplier CRS with supplier specific requirements syntax (boilerplates)

Fig. 2. Methodology for an Automated Delta Analysis

2. Delta Analyzer: These formalized CRSs represent the fundamentals for the approach in this work, as they act as the input for our tool. The custom syntax of requirements allows to compare multiple specifications based on the structure of each requirement. These differences can be identified by the DA and be summarized in a delta report. Differences can then be output for a RE expert on screen or in a file. For this purpose, the program offers procedures for the evaluation of the deltas or metrics for the decision between whether requirements are matching or not. As we describe later, such a categorization can take place based on the annotations generated by the R2BC.

The methodology of the presented tools is shown as a prototypical tool chain in Fig. 2.

B. Delta Analyzer Concept

The Delta Analyzer concept features methods that can identify deltas between old and new CRS, which can then be rated and emphasized for the user. This processing is done for each requirement from the CRS. For an automated finding of suitable requirements in the DA, the requirements must be presented uniformly, which is why we use the boilerplates created by the R2BC (cf. our previous work [1]).

The functions described allow the design of a method for an automated comparison of two specifications in the DA. For this purpose, all requirements from the specification V2 (current, new OEM CRS) are compared with those from V1 (CRS of predecessor product). For each requirement in V2, V1 is traversed to find a corresponding statement there. This leads to the distinction of three categories: (1) identical, (2) comparable with deltas and (3) no comparable was found. In order to help the project members estimating the requirements, giving feedback and offer support to realize major issues to them is an important aspect of our concept. Especially, the impacts of deltas found in (2) has to be rated by our tool. Therefore, metrics were identified and bundled as part of the DA-algorithm to find the best possible match for each requirement. The following procedure was developed:

1. The individual requirements in boilerplates are loaded for specification V1 and V2.
2. For each boilerplate from CRS V2 it is checked whether identical, comparable (with deltas in the specification) or no requirements can be assigned from V1. These three categories are identified based on the GATE annotations from the R2BC. Using further similarity measurement metrics, we can calculate a total degree of correspondence as score.
3. According to the degree of correspondence found, the deltas are identified between the requirements:
(a) Identical requirement: There was an identical requirement found in both CRS. The request can be accepted without separate consideration.
(b) Comparable with deltas: Only a partially matching requirement was found. This is likely the case when requirements have changed or been revised. There is a delta between these requirements, that signals a change in the requirements and would therefore mean a change in product specifications. A fine granular breakdown of the distinguishable deltas will be described later.
(c) New requirement: No matching requirement from V1 could be found. CRS V2 could contain new requirements that were not specified in V1 in any way. This may concern, for example, properties that are defined later in the development process or no information was previously known about.
4. If there are more at least comparable requirements in V1, the best matching requirement is found.
5. The delta report for the user is generated by accessing the results from 4. when V2 was completely traversed.

If deltas have been found, the feasibility of these new requirements must always be checked separately. Different specifications in V2 might decrease the correctness of the requirements. The further correctness of V2 to V1 can not be guaranteed for many deltas, which makes it difficult to realize the requirements. There may be massive differences in specifications, which may result in a partial or complete redesign of the product. As a guideline, the report therefore also contains the overall degree of agreement of the CRS as an absolute and percentage value. The description of the functionality for our tool led to the concrete emergence of three core components. The tool contains components for the mapping of subject logic and GUI as shown in Fig. 3.

The components realize the two main functions 1 and 2 using GATE and Java resources. Based on this concept, the functionalities for each individual component are assigned as follows:

1. NL pre-processing: Since the DA represents the second stage of processing, it uses the results from the R2BC, i.e. the annotations generated there. These will be under access to the program read into the DA. Thus the requirements which are present in boilerplates after R2BC-conversion, are compared by their annotations.

2. Analyzer: The delta analyzer checks the various requirements of specification V2 for their compliance with the requirements V1. The program accesses the loaded annotations of both specifications and compares different requirements. Finally, a delta report is created that clearly displays all found deltas between the requirements.

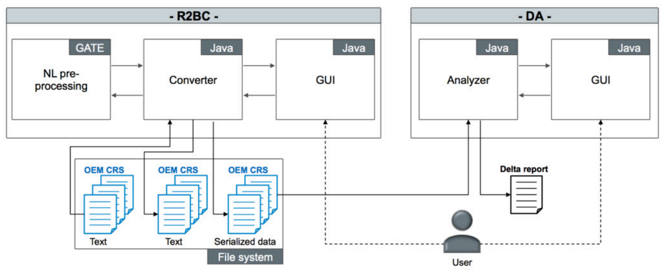


Fig. 3. Components of the Delta-Analyzer

3. GUI: The GUI allows the user to enter the two CRS to be compared. After processing, the user can export the delta report containing the compared requirements.

C. Agreement Level

Aiming to offer robust support in identifying deltas between two requirements, a fine-grained process analyzing each requirement match was designed to give a score to each of these matches. Each single requirement from V2 and V1 is represented as an object in our tool. Every match-object then consists of both found requirements and the agreement level the DA calculated.

The CRS are essentially treated as mathematical sets. For our concept, requirements from V1 and V2 can also be understood as set elements. A *boilerplate-match* then is a triple: $(r2, r1, a) : r2 \in V2, r1 \in V1$, agreement level a. The delta-analyzer aims to rate a match as close to category (a) as possible, searching for an identical requirement in V1 as long as no triple with that agreement level was constructed. The process of scoring the agreement of two requirements consists of two main steps. First, the boilerplate environment is used, analyzing boilerplate-type and then the features (text parts) of each boilerplate. In case of a different type, we assume the requirements as completely different (new requirement in V2, category c), as otherwise they would have shared the same boilerplate-category. If two compared requirements share the same boilerplate-features and category, they are at least comparable (b), if not even identical (a). The task then was to find a method rating several types of deltas all found as (b), as some requirements were partially identical while others were almost distinct.

After this rather rough analysis of R2BC-annotations, we use Levenshtein distance for a deeper, linguistic analysis. The Levenshtein distance calculates the difference between two given strings as the effort (operations) to transform string $s1$ into $s2$ [4]. This method is common when it comes to measuring similarity of two given strings or requirements in our case. After calculating the Levenshtein distance for each component of the requirements matched as (b), we can easily add it to the agreement level contained in the triple together with the requirements. As our tool searches for the best possible match, we can now simply search for the best score comparable requirements have reached.

D. Methodology for the Prioritization of Requirements Deltas

The determination of deltas is an important step to increase efficiency during the RFQ phase. The DA concept includes an additional function to speed-up the work progress during this phase. This function is called requirements delta prioritization. Its main purpose is to rank requirements deltas, which have the highest impact on the development effort, first. As a consequence, project team members can focus on prioritized requirements deltas to determine whether a predecessor component can fulfill the new requirements and if necessary, how much effort is needed to adapt the component. To this end, our requirements delta prioritization algorithm relies on weighting factors, which are defined prior to the delta analysis.

E. Determination of Weighting Factors for Product Features and Functions

Weighting factors indicate how much more development effort is required to adapt a certain feature or function of a product compared to another feature or function. They are a prerequisite for the requirements delta prioritization algorithm and therefore have to be determined first. Our concept for the determination of weighting factors is based on the Analytic Hierarchy Process (AHP) introduced by Thomas L. Saaty [5]. Within this methodology a pair-wise comparison matrix is used to determine which entry is more important than another. In our approach, this determination is executed by the project team within the DA GUI, as depicted in Fig. 4. As a first step, the project team selects a product, for which the weighting factors shall be determined (Fig. 4 Step 1). The selection of a product is important, since weighting factors for certain features and functions can be different for different products. A change in current draw can be of higher impact in a pump, than in an actuator. Second, the project team documents the top ten features and functions of the selected product (Fig. 4 Step 2). We suggest an amount of ten features and functions, but it is also possible to add more. It is sufficient to fill out the headings. The headings for the rows are filled out automatically. According to the application domain, the amount of the main features and functions may vary. The only one boundary condition is that the number of features and functions is finite. During the third step (Fig. 4 Step 3), project team members rate the impact for the development effort on a scale from 1-9, where 1 is equal effort, 2 is low effort, 3 is moderate effort, 4 is moderate plus, 5 is high effort, 6 is high effort plus, 7 is very high effort, 8 is very, very high effort and 9 is extreme effort. We use the scale as suggested by [5] and adjusted the definitions of the numbers, so that they fit our use case. To decrease the effort of filling out the matrix, only the white upper right corner of the matrix must be filled out by the user. The lower left corner is filled out automatically with the reciprocal value by the DA. Once all values have been entered, the DA automatically calculates the geometric mean per feature or function and show them in the respective column in the GUI. The geometric mean figures constitute the weighting factors, which at the same time describe a total order of all features and functions. As a result, the top ten features and functions can be ranked in order of development effort, which is necessary, if one of them shall be adapted in the selected product. Finally, the user can save the weighting factors by pressing "Save weighting factors" (Fig. 4 Step 4).

All weighting factors for a given product are saved in the repository of the DA and can hence be invoked by the requirements delta prioritization algorithm. Since over time the development effort for certain features or functions can change, it is possible to repeat this process. The DA will automatically calculate the mean for all adjustments of weighting factors per feature or function of a product. By this opportunity we aim to receive more accurate weighting factors over time.

F. Prioritization of Requirements Deltas

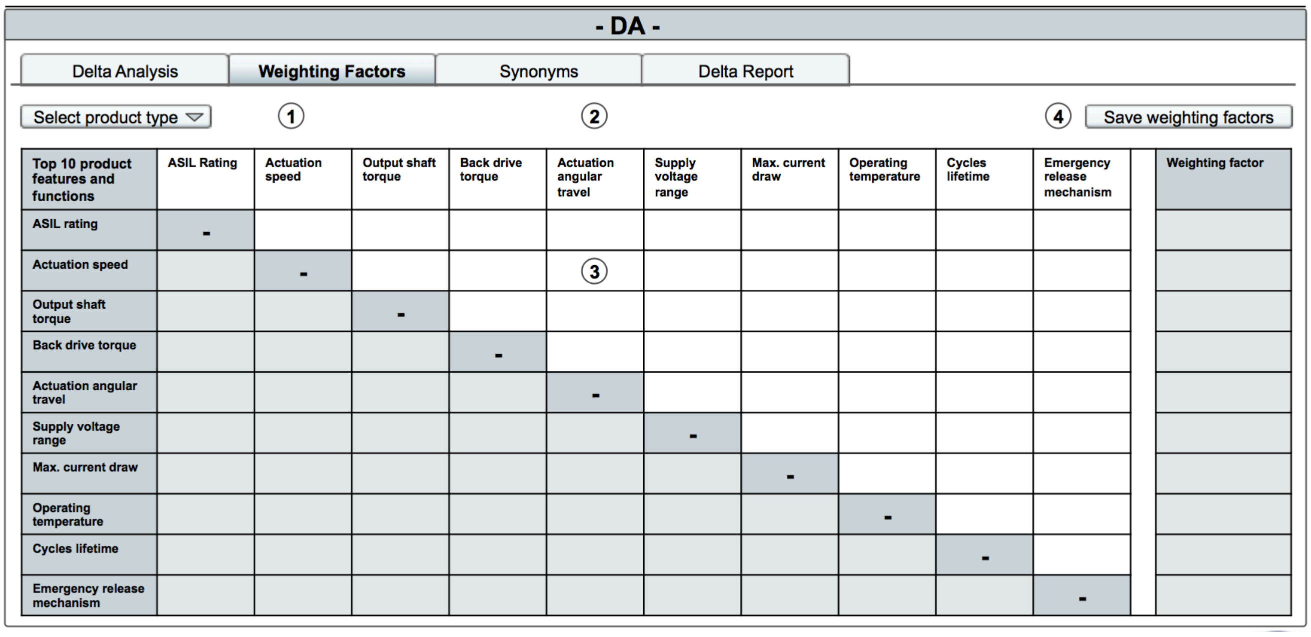
The general aim of the DA is to detect deltas between a supplier CRS and an OEM CRS by comparing requirements sentences with each other. For this purpose, a pairwise comparison of requirements is performed by the DA. Once two requirements address the same system and one of its features or functions, the DA allocates both requirements as a couple in the delta report. As an example, the supplier requirement A and the OEM requirement B constitute the following couple, since both of them address the torque feature of an actuator:

A: The actuator shall provide a stall torque of 50 Nm. B: The waste gate actuator shall provide a torque of 60 Nm.

Since not all requirements from the OEM CRS and the supplier CRS can be allocated as couples, single requirements will exist in the delta report. This is especially true for completely new OEM requirements, which previously were never addressed by the supplier CRS.

Once the delta analysis is finished, the requirements delta prioritization algorithm searches for keywords in all requirements, which are listed in the delta report. The product features and functions, which were documented during the determination of weighting factors, constitute these keywords. If for example the requirements delta prioritization algorithm finds the keyword "torque" in the requirements couple of supplier requirement A and OEM requirement B, this couple will automatically receive the previously defined weighting factor. To this end, the DA invokes the weighting factors from the repository. The DA assigns the same weighting factors to identified requirements couple as well as to single requirements, which could not be allocated to other requirements. To increase the success rate of the weighting factor assignment, the DA provides the user with the possibility to enter synonyms and other words or characters that may be a hint for a certain product feature or function.

Once the prioritization of requirements deltas is finished, the user can rank them by their weight in order to focus on the most heavy-weighted deltas first. However, in case the torque requirement of the OEM does not demand any changes in



Top 10 product features and functions	ASIL Rating	Actuation speed	Output shaft torque	Back drive torque	Actuation angular travel	Supply voltage range	Max. current draw	Operating temperature	Cycles lifetime	Emergency release mechanism	Weighting factor
ASIL rating	-										
Actuation speed		-			③						
Output shaft torque			-								
Back drive torque				-							
Actuation angular travel					-						
Supply voltage range						-					
Max. current draw							-				
Operating temperature								-			
Cycles lifetime									-		
Emergency release mechanism										-	

Fig. 4. Determination of weighting factors with the pairwise comparison matrix

the current degree of torque, the corresponding requirements couple should have a lower priority for the user, than those couples, which demand changes from the product. To this end, the DA provides a filter function. With this filter, the user can filter for the three categories of requirements couples: identical requirements (no changes of feature or function required), comparable with deltas (changes of feature or function required) new requirements (new feature or function required). With this filter, the user can focus on true requirements deltas with the highest weighting factor.

For those requirements couples, which did not receive a weighting factor, the user may add a factor according to his estimation. Then the prioritization of requirements couples can be repeated. The prioritization order is adapted by the DA automatically. This functionality allows to expand the number of weighting factors for new features and functions and assures that the new weighting factors are related properly to the already present ones. As a consequence, this methodology can be applied to different types of products and can be updated accordingly.

IV. Experiments and Discussion

In this chapter, we present the results of preliminary experiments with the DA prototype. As a first step we describe the target setting and the experimental setup. Second, we present figures from our experiments and discuss the results. Finally, based on the gathered data, we draw conclusions and suggest measures for improvement.

A. Target setting and experimental setup

The target of the experiments is to prove the general feasibility of the DA. The objective of the DA is to determine whether the requirements of the customer CRS are identical, comparable or completely new to the supplier CRS or an previous version of the customer CRS. To assess the performance of the DA, we calculate precision scores for the correct allocation of requirements to the respective categories: identical requirements, comparable with deltas and new requirements.

We conducted four experiments with five real-life CRS. All these specification documents have a common history. CRS 1 is the initial specification. It consists of fragmented sentences and is a rather short document. CRS 2 is a qualitatively improved version of CRS 1. CRS 3, 4 and 5 describe different product generations, which emerged over time. Each CRS contains additional requirements regarding new features and new sub-components. This is a common scenario in the automotive industry. Therefore, we decided to perform the delta analysis with the following couples: CRS 1 with CRS 2, CRS 2 with CRS 3, CRS 1 with CRS 4 and CRS 4 with CRS 5. All CRSs were provided to us as PDF documents.

According to our tool chain, the CRS must be processed by the R2BC first. The reason for this is, that the DA uses information created during the NL pre-processing applied and saved by the R2BC. As a preparatory step we created a new gazetteer list for the R2BC. We used the content of the glossary of terms of all five CRS for our gazetteer and performed the requirements to boilerplates conversion. Also, the JAPE rules for CRS 1, 4 and 5 required some adjustment. The results of the conversion showed a drop of the number of identified requirements for this CRS. We discovered that the requirements authors of these CRS have used “must” instead of “shall” or omitted it completely and applied “to be” or “is” to explain for example an action in their requirements. To this end, we substituted the word “shall” with “must” in our JAPE rules and added “to be” as an alternative explanation. This led

TABLE I
RESULTS OF PRELIMINARY EXPERIMENTS

	Category	Req. per cat.	Correct	Precision	Total req.
CRS 1	Identical requirements	0	0	-	
&	Comparable with deltas	35	33	94.3%	89
CRS 2	New requirements	23	-	-	
CRS 2	Identical requirements	27	25	92.6%	
&	Comparable with deltas	94	93	98.9%	180
CRS 3	New requirements	23	-	-	
CRS 1	Identical requirements	1	1	100%	
&	Comparable with deltas	23	23	100%	61
CRS 4	New requirements	8	-	-	
CRS 4	Identical requirements	3	3	100%	
&	Comparable with deltas	9	9	100%	69
CRS 5	New requirements	25	-	-	

to a high number of requirements conversions.

The gazetteers were mainly used to identify the system name. During preliminary experiments the R2BC converted all requirements sentences into boilerplates, whether the corresponding subjects were system names or other words for example, describing product features. Once the original requirement fit the boilerplate, it was considered useful for the experiments. We also did not correct the results of the R2BC. Some requirements had flaws as a result of the conversion. Since the number of proper requirements was high enough to prove the feasibility of the DA, we considered the incompletely converted requirements negligible.

In our previous work [1], we presented the results of preliminary experiments with the R2BC. In the remainder of this section we focus on the performance of the DA prototype. Within first experiments the DA achieved promising results, as can be seen in Table I.

Within the first experiment with CRS 1 and CRS 2, whereby CRS 2, the DA analyzed 89 requirements from both CRS in less than a second. As a result, the DA identified 33 comparable requirements couples with a precision of 94.3%. Two requirements contained not processable symbols, which lead to error outputs in the delta report. As consequence 33 of the 35 available couples were identified by the DA. All requirements couples, which were listed by the DA in the delta report in the "Comparable with delta" section, mention the same system name respectively. The latter parts of the requirements sentences differ from each other. As can be seen from the numbers, one requirement from CRS 2 can fit several requirements form CRS 1. The DA also found 23 new requirements in CRS 2, which were not listed in CRS 1 before. Yet, the novelty of these requirements is merely proven by their syntactical difference.

As a result of the delta analysis of CRS 2 and CRS 3, the DA identified 25 identical requirements couples with a precision of 92.6%. The precision was reduced by the fact, that two error outputs appeared in the delta report. These were caused by not processable symbols in the original CRS. In the category "Comparable with deltas", the DA found 93 comparable requirements. These include, seven requirements couples, which consisted in their original state of identical sentences per couple. As a matter of fact, these couples should actually be listed in the "Identical requirements" category. But the R2BC added additional words, which stem from the adjacent attribute column in the original CRS to the conversion results of these 14 requirements. Nevertheless, we consider the DA results correct, since the sentences provided by the R2BC differed indeed from each other. As previously mentioned, we defined an agreement level score to determine how equal two requirements are to allocate them as a couple. The observation of experiments with CRS 2 and CRS 3 showed, that requirements with an agreement level score of up to 1.02 differ in only one character. From a score of 1.06 they differ mostly in one word. Once the agreement level score surpasses the mark of 1.5, the requirements are clearly different except for the system name. Finally, the DA found 23 new requirements in CRS 3. Also, here the novelty of the requirements is caused by their syntactic character.

The delta analysis for CRS 1 and 4 resulted in one requirements couple consisting of identical requirements sentences and 23 requirements couples, which are comparable and contain deltas. For both categories the DA achieved 100% precision. For eight requirements from CRS 4, the DA could not identify a comparable requirement in CRS 1 based on the syntactical analysis. In the last experiment with CRS 4 and CRS 5, the DA identified three requirements couples with identical requirements. The identification of identical and comparable requirements worked both with a precision of 100%. In numbers, the DA identified nine requirements couples with comparable requirements and 25 new requirements. In total 399 requirements from five different CRS were analyzed during the experiments.

B. Discussion and Conclusion

Preliminary experiments with the DA prototype show sound results. The precision values range from 94.3% to 100%. Within seconds hundreds of requirements can be analyzed and deltas determined. From the number of requirements couples per category can be seen, that some CRS documents are more similar than others. The requirements from CRS 2 and CRS 3 seem to be similar. On the other hand, CRS 1 and CRS 2 seem to be very different.

The major benefit of a delta analysis is to find requirements couples, where both requirements slightly differ from each other. In this case, it is directly clear which feature or function of a component must be adapted. It is also an indication that the supplier already knows how to handle this kind of requirement, since it could be allocated to the supplier requirement or to the requirement of a previous version of the CRS. Especially in the results of the experiment with CRS 2 and CRS 3 this phenomenon can be observed. Requirements couples with an agreement level score of 1.02 differed in only one character. This kind of couples are beneficial, since they show the user of the DA the need for action in a clear matter. For instance, the necessary adjustment of the torque of an actuator could be detected by applying the DA. Scores higher than 1.06 represent requirements couples with requirements, which mostly differ in one word. Beyond the threshold of 1.5 only the system names are equal in both partners of the requirements couple.

Based on these findings we are planning to use the presented thresholds of agreement level scores to highlight requirements couples with slightly different requirements to the user. Furthermore, we plan to evaluate, whether the requirements which were identified as “New requirements” are really new. So far, their categorization was proven only by a syntactic analysis. This evaluation may lead to the understanding, whether a semantic delta analysis would lead to better results. Also, the presented results have shown that the R2BC is still a decisive component of our tool chain and that it has a huge effect on the outcome of the delta analysis. Therefore, we plan to further improve our prototype of the R2BC.

V. Related Work

Since the automated comparison of different data offers a high potential for reduction working hours, a large number of tools for document comparison have been developed. In general, all of these tools are designed to find differences between an original and a corresponding modified document and at the end to describe the identified changes in a third document [6]. However, scientific work on these tools is hard to come by because most of them are proprietary and little to nothing has been published about them. To the best of our knowledge, there is no work that implements an automatic delta analysis similar to our approach.

Schraps and Bosler present an approach to create a requirements ontology by extracting knowledge from software requirements and transferring this knowledge into the ontology. After annotating the requirements using NLP techniques, a pattern recognition algorithm searches for predefined grammatical patterns. Requirements or parts of them fitting the patterns are then integrated into the requirements ontology [7]. While this approach aims at detecting and solving inconsistencies between requirement specifications and software models, the goal of our approach is to find differences and mutualities between multiple requirement specifications.

The Module Comparison Wizard in IBM DOORS can be used to compare two different modules, possibly containing requirement specifications. The tool detects if objects have been added, deleted, edited or if heading numbers have changed. These deltas can be found automatically, they are however of structural or syntactic nature and show no relevance regarding the content similarity or the requirements [8].

The tool compareDocs from DocsCorp aims at comparing two versions of a document and showing differences in form of insertions, deletions and moves. It can be used on different types of documents and event hough it resembles a similar "delta-between-versions" approach as the tool described in this work, the detected deltas have no means of being evaluated regarding their semantic relevance [9].

VI. Summary and Outlook

In this work, we presented our concept for an automated delta analysis, which aims to support project teams in the limited time frame of an RFQ phase. Our proprietary developed prototype the Delta Analyzer (DA) analyses two different requirement specification documents provided in office format and determines the differences between the requirements – namely the deltas. The DA uses information generated by Natural Language Processing techniques applied by our tool R2BC [1]. Preliminary experiments with the DA on real-life requirements specifications yielded good results. The DA allocated requirements into the categories: identical requirements, comparable with deltas and new requirements with a precision of 94.3% to 100%. In addition to that, we introduced our methodology for the prioritization of requirements deltas. This technique is based on the Analytic Hierarchy Process (AHP) [5] and has the purpose of ordering requirements deltas in the delta report according to the development effort for their implementation. This functionality shall enable project teams to focus on the most important need for action first.

The objective of our upcoming research activity is to further improve the performance of the DA. To this end, we will refine the recognition of deltas in the category comparable with deltas. This will allow us to automatically distinguish between more high-level deltas at component level and low-level deltas that relate to specific features or functions of single components. We are currently in the process of implementing the algorithm for prioritizing requirements in our DA prototype. Once this functionality is available, we will test it with different project teams.

References

[1] K. Zichler and S. Helke. R2BC : Tool-Based Requirements Preparation for Delta Analyses by Conversion into Boilerplates. In *Proceedings Workshop on Automotive Software Engineering (ASE 2019)* CEUR-WS, **2308**, pages 45-52, 2019.

[2] R. Weischedel et al. White Paper on Natural Language Processing. In *Proceedings of the Workshop on Speech and Natural Language*. Association for Computational Linguistics, 1989.

[3] D. Thakker and T. Osman and P. Lakin. Gate Jape Grammar Tutorial. Nottingham Trent University, UK, Phil Lakin, UK, Version 1, 2009.

[4] L. Yujian and L. Bo. A Normalized Levenshtein Distance Metric. In *Proceedings IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29.6**, pages 1091-1095, 2007.

[5] Y. Wind and T.L. Saaty. Marketing Applications of the Analytic Hierarchy Process. Management Science **26.7**, pages 641-658, 1980.

[6] D. Massand. Systems and Methods for the Comparison of Annotations within Files U.S. Patent No. 8,732,181. 2014.

[7] M. Schraps and A. Bosler. Knowledge Extraction from German Automotive Software Requirements using NLP-Techniques and a Grammar-based Pattern Detection. In Proc. of the Int. Conf. on Pervasive Patterns and Applications, 2016.

[8] https://www.ibm.com/support/knowledgecenter/en/SSYQBZ_9.6.1/com.ibm.doors.requirements.doc/topics/c_modulecomparisonmarkup.html

[9] https://www.docscorp.com/products/comparedocs/softwaredevelopment-kit-SDK-API

[10] F. Ritter and A. Schul. Entwurf und Implementierung einer Werkzeugunterstützung zur sprachlichen Analyse und automatisierten Transformation von Projektlastenheften im Kontext der Automobilindustrie. Bachelor thesis, FH Dortmund, 2019.

[11] K. Zichler and S. Helke. Ontologiebasierte Abhängigkeitsanalyse im Projektlastenheft. In *Proceedings Automotive - Safety und Security (AUTOMOTIVE 2017)*, GI-LNI, **269**, 2017.