Natural Language Processing To Assess Structure and Complexity of System Requirements

Dr. Maximilian Vierlboeck [1] | **Dr. Roshanak Nilchiani** [1] | **Dr. Mark Blackburn** [1,2]

Abstract — **The development process of a system is shaped by numerous variables that influence its progress and outcome. As a result, complexity can increase throughout the development process, potentially leading to negative consequences, which makes the management of complexity critical. Most development processes begin with the definition of needs and requirements, and in this paper, the authors present a novel approach that enables the automated extraction of structure from requirements specifications. The approach uses Natural Language Processing to elicit three structural layers from a set of requirements, which are subsequently analyzed using metrics to assess complexity. In a case study, the approach is demonstrated using a set of 79 requirements, within which 246 individual entities are identified. These entities and the requirements are structured and analyzed using network density and spectral entropy. The metrics allow for interpretation and insight generation, such as identifying an increase in the number of potentially problematic loops. The approach achieved a detection and structural accuracy of over 98 percent in the given case study and is planned to be expanded with future cases.**

Keywords — ***systems development, requirements engineering, systems structure, natural language processing, complexity, quantification***

1. INTRODUCTION

During the process of systems development, countless factors and variables must be considered. As a result, from inception to the eventual roll-out of the system, complexity grows, which is further exacerbated by the increasing demand for higher performance [1]. Complexity can lead to issues and even severe problems in the long term. As projects grow in complexity, they also become more challenging, have higher failure rates, and are subject to cost and schedule overruns. For instance, the increased system complexity and higher development effort in realizing fixed-wing aircraft has been assessed by the RAND Corporation [2] with striking conclusions regarding cost increases over time due to complexity. Such circumstances have also been shown for small-scale applications [3]. The worst-case outcome is that system development projects fail to achieve their planned schedule and performance [4]. In order to address these issues and risks, the ability to characterize, quantify, and subsequently effectively manage and hedge complexity is crucial [5]. The urgency of this is further underlined by shrinking cycle times and the strong correlation between system development and complexity [5].

Due to the aforementioned circumstances, managing complexity and its undesirable consequences for the systems and development process is an acute issue. Yet, two general problems make addressing this issue difficult: (1) the inability to effectively characterize, quantify, and manage complexity [5] and (2) the necessary balance of performance, complexity, and development effort [6]. Sinha and de Weck highlight the timely necessity for quantifiable and comprehensive complexity measurement frameworks [6] that allow for tracking and trade-off management in the development and design process. Such quantifiable metrics to assess complexity eventually enable the identification and hedging of undesirable aspects of the system and development process, which can be leveraged to find advantageous options and even allow multi-objective optimization [7].

Affiliations: [1] Stevens Institute of Technology [2] Systems Engineering Research Center (SERC)

The complexity management described applies to the entire development process and should be considered from the first steps onwards. Traditionally, most system development models begin with the definition of stakeholder needs and subsequently requirements elicitation [8-11]. Thus, the consideration of complexity has to start at these early steps to fully leverage the above-mentioned advantages. However, most complexity quantification approaches and frameworks, as further demonstrated in the second section, rely on a sufficiently developed system architecture or design elements of the system. Such assessments thus rely on a development process that has already progressed into at least the preliminary design phase, which then only considers the effect of requirements indirectly. While assessing and quantifying complexity based on a sufficiently developed system architecture and/or design is useful, we argue that the impact of requirements complexity must not be ignored. Requirements introduce complexity to the process as well, for instance by increasing effort or load, as described by various metrics [12, 13]. While determining causal relationships and/or correlations between different complexity metrics can be challenging, these metrics all contribute to the overall system and development complexity, as outlined by Salado and Nilchiani [14]. Additionally, it is important to note that while architectures can be identified early in the process, they evolve in fidelity over time; in the early stages, they usually do not include enough connections and relationships to enable quantifiable complexity analyses and comparisons.

To address the gap described above, this paper presents a novel approach to assess the structure and quantify the complexity created during the requirement phase and the continued effects throughout the development. To enable this approach and application, Natural Language Processing (NLP) is used to dissect and evaluate the requirement body. The result of this evaluation is then assessed with quantification metrics. The presented assessment can be used from the early development phases onward, which is demonstrated through a case study involving an unmanned aerial vehicle requirements specification currently under research at the Systems Engineering Research Center (SERC) [15].

It has to be noted that the focus of this paper and research is not the detection or causal definition of the mechanisms that are the root of complexity. Since the development of a causal model for complexity for systems development is a substantial research undertaking by itself, as also outlined by Sinha and de Weck [6], the presented research focuses on the automated elicitation of structure from a requirements specification and a quantitative analysis based on the results thereof. This foundation then enables the analysis of causality in future work.

The paper at hand is divided into the following six parts: Part 1 introduces the problem and research gap to be addressed. Part 2 provides the background of the topic and a literature review to understand the current state of the art. Based on the identified gap, Part 3 describes the developed approach, which is then applied to the case study in Part 4, including a presentation of the achieved results. Part 5 interprets and discussed these results, and Part 6 concludes the paper.

2. BACKGROUND & LITERATURE

Since the approach presented in this paper consists of an NLP component and a complexity assessment component, Part 2 is divided into two sections. The first section addresses the state of the art regarding NLP with a focus on structural extraction from text sources in Requirements Engineering, whereas the second section summarizes existing research on system and requirements complexity. This division facilitates the specific explanation of the opportunities that exist in each field separately.

2.1. NLP for Requirements Engineering

Natural Language Processing for Requirements Engineering (NLP4RE) and related topics have been extensively researched by computer science scholars and have also made their way into adjacent domains [16]. To survey and summarize existing approaches and possibilities regarding the field and application of NLP to Requirements Engineering in the context of systems engineering and development, the authors conducted a survey of 133 NLP tools [17] to assess their usefulness for eliciting structure from a requirements specification. All tools were analyzed with a set of criteria to allow for both quantitative and qualitative analysis.

The result of the review and survey was that no approach currently meets all the criteria necessary to elicit sufficient structure for a subsequent assessment of complexity. The contenders that fulfilled most criteria only partially

targeted the extraction of structure and thus require further adaptation to be considered useful. Other approaches and tools that did target the elicitation of structure primarily turned out to be not openly accessible as far as their code basis is concerned, which makes them only conceptually useful and consequently requires reconstruction of the actual tools.

Despite the lack of existing solutions, which led to the development of the approach described in Part 3, the four publications most closely related to the presented work are discussed here to provide a better understanding of the contribution of the presented research.

The first related research project by Ferrari, Gnesi, and Tolomei [18] used an algorithm called “Sliding Head-Tail Component” (S-HTC) to cluster requirements based on two factors: lexical relation and contiguous order in the document. The algorithm emulates the process of reading a requirements document with the inclusion of relations. The authors claim that by applying the S-HTC, a hidden structure of the document can be elicited that can even be refined with additional runs. Despite their ability to structure a document, the clustering within the approach does not enable a specification-wide structure definition without consideration of the original setup of the document.

The second related publication was published by Hamza and Walker [19]. In their work, they outline their “Feature and Feature Relationship Extraction” (FFRE) approach, which allows for the decomposition of a requirement set into features and their respective structure. An algorithm takes into account semantic similarities to form connections and ultimately yields a feature model. While such a model can be seen as a structure of the set, it is limited to the features and thus leaves other connections out that are not useful for the feature model.

Third, Tahvili et al. [20] published an NLP approach that enables the functional dependency detection between integration test cases. By analyzing internal communication between modules, the approach allows for the detection of dependencies of requirements and test cases, respectively. These dependencies are then used to build a structure of the respective requirements in addition to the modules. While this approach does elicit structure, it only links the requirements to models and does not assess the structure inherent to the requirements.

Lastly, Arora and Zimmer outlined an approach using NLP to extract domain name models from natural language requirements [21]. While their approach specifically targets the extraction of models and not structure, part of it elicits connections from requirement statements that link subjects and objects sequentially, which aligns with the presented approach. Although Arora and Zimmer used similar NLP tools, their structural extraction has to be considered partial since it was not the main focus. Nevertheless, due to their connection to the work at hand, the correctness achieved by Arora and Zimmer supports the results of the case study presented in Part 4.

All in all, the literature review mentioned above [17] and the presented related works do not address the structural elicitation from textual requirements nor the complexity, which is what the presented approach provides. Due to the multitude of existing complexity metrics, the next section provides an overview of the field.

2.2. Complexity and Metrics

The general literature regarding system complexity is characterized by many publications from different scholars. With this diversity comes the fact that currently, there is no standardized model to characterize the complexity of a system or network, for instance [22]. Furthermore, some scholars describe inherent problems and difficulties that come with the nature of complexity, leading to inability to quantify or even manage the resulting dynamics [5, 6]. As a result, when looking for complexity measures and metrics in a general sense, a multitude of options can be found that are associated with various scientific fields [23]. The most popular and closely related ones are discussed below.

Please note that complexity in the context of this research, and specifically regarding requirements, is classified as the identification of the attributes and characteristics of complex systems/constructs, and pertains to the concepts of multi-dependency dynamics, uncertainty, and emergence, caused by the behavior and interaction of known components [37-40]. This definition also aligns with the works cited earlier in this chapter.

One of the most, if not the most famous metric for complexity is Shannon’s entropy [24]. By using the probability of a specific variable, the metric assesses the information content in the form of weighted averages. The metric was used to measure complexity in a multitude of fields and applications and is still applied as a measure of graph complexity [25]. Furthermore, Shannon’s metric has inspired the development of various other information-based complexity metrics [26], which also include metrics such as Gell-Mann’s effective complexity [27, 28]. Other metrics

that have gained popularity over time are McCabe's Cyclomatic Complexity [12] and Halstead's approach [29], both for software code.

Since the output of the NLP, as shown in Part 3 and 4, is a structural network representation in different forms, graph theoretical and spectral metrics for complexity are applicable as well. Looking into these metrics shows that approaches such as Gutman's graph energy [30, 31], which was developed based on chemistry approaches from the 1940s, can quantify complexity based on the spectrum (set of eigenvalues) of a graph and its matrix. More recently, Sinha and de Weck [1] developed a comprehensive structural complexity metric that uses graph energy/entropy to assess topological aspects of a system. Similar approaches with a different basis have been shown by Nikiforov [32] using singular values instead. Also, Wu et al. proposed a metric that uses natural connectivity [33]. Many of these approaches also consider and incorporate the use of network/graph density [33, 34]. More recently, Lei, Liu, and Wei [22] proposed an approach that claims to resolve the network scaling dependence of existing approaches by combining structural entropy with the absolute density of a graph.

Looking specifically for requirements complexity, Salado and Nilchiani's [14] concept of problem complexity incorporates the conflicts and physical properties within requirements to add a factor to other complexity dimensions. Also, Sharma and Kushwaha [35] proposed a metric including NLP for software requirements that relies on classification and structure inference based on past projects in the context of knowledge-based NLP. Lastly, Purawinata, Ariadi, and Abbas [36] recently proposed a neural network-based algorithm to predict software complexity by calculating requirements complexity based on the number of categories.

Overall, the literature regarding complexity of requirements shows that current approaches either rely on manual processing or pre-existing data for training. Thus, none of these approaches can be considered fully automated due to these drawbacks. A similar conclusion was also found by the authors regarding NLP4RE in general [17]. Therefore, the goal of the presented research was the development of an automated requirements processing approach that elicits structure and allows for quantitative complexity assessment.

3. METHODOLOGY AND SETUP

As mentioned earlier, the research presented enables the assessment of the complexity that is inherent to and introduced by the requirements. One of the limitations of the approaches discussed in the previous section is that they depend on an existing system structure for their implementation, which requirements do not necessarily contain. While some requirements specifications do contain a hierarchy or levels, their structure does not have the same kind of lateral and vertical connections as the system itself, and therefore, system structure cannot be inferred. Thus, for the presented research, an NLP framework was designed that allows for the identification and organization of all entities within a requirements specification. By applying the NLP process presented in Figure 1, two types of structure can be deduced from a set of requirements (in addition to a hierarchy structure, if available): (1) the structure based on the terms and entities in the requirement text, and (2) the connections of the requirements based on the entities within them. All of the outlined steps were achieved by using the spaCy [41] library, which is an open-source NLP library that provides the necessary functions shown below and described on the next page.

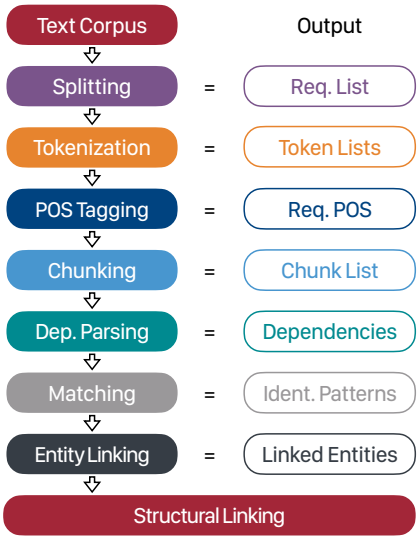

Figure 1 - NLP Flowchart and Process

As depicted, in a first step, an imported requirement text is split into the different requirements. Depending on the input, this can be based on a table or directly on the content. This splitting yields the **Requirement List.** With this list, the NLP is used to identify the tokens in each requirement, which yields a **Token List**. Based on the Token List, the parts of speech can be tagged, which identifies the different roles/categories of the text in accordance with Universal POS tags [42]. This tagging yields the **Requirement Part-of-Speech** list for each requirement and tokens therein. Adding to the POS tagging, chunking is applied to reveal the noun chunks within the sentences, which results in a **Chunk List**. With the chunks and parts of speech, their dependencies are analyzed, which produces the role that each token plays in a sentence as a **Dependency** overview. A pattern matcher is used to identify structural patterns, such as lists and concatenations between the noun chunks, which then allows for the **linking of the entities** that are found within the noun chunks. Lastly, this entity linking then can be combined to build the structure of the requirements for the **structural linking**. The last step connects the entities based on their relationships in the text, such as the nouns with the objects, and concatenations/lists.

To provide an example, the following requirement is divided into the entities listed below.

"The landing gear shall be designed for a service life equal to that of the air vehicle airframe structure.":

(A) landing gear
(B) service life
(C) air vehicle airframe structure

Herein, the following connections indicate the structure: (A) is connected to (B) and (B) is connected to (C).

The result of the process described above is a network of terms/entities that are connected based on their sentence structure and text relationships. In addition, since the different entities are part of individual requirements, these requirements can also be linked based on their content, which enables the creation of another dimension with additional insights. Thus, overall, three different layers and structural aspects of the requirements can be elicited, as shown in Figure 2: the hierarchy of the requirements (if existent), the structural network of the requirements based on the contained entities, and the structural network of the entities separately. The networks were managed using the open-source library NetworkX [43], which also enables the computation of the metrics in the next paragraph.

With the results produced by the NLP, analyses can b fe performed for each of the layers. To represent the potential of the created approach and provide some useful initial insights for the application, the metric of **network density** was chosen to be demonstrated for the **Requirement Structure,** and the **spectral entropy** [30, 31] was used to analyze the **NLP Structure**, which represents the entity connections. These metrics can be used to quantify complexity overall and have been used by the authors and researchers cited in the previous chapter as well.

Figure 2 - Extracted Dimensions and Relations

4. CASE STUDY, RESULTS, AND VALIDITY

With the process and possibilities outlined above, a case study was conducted to address two main objectives: (1) the validation and proof of concept for the NLP approach, and (2) the application and test of different metrics. To achieve this, a case study from the Systems Engineering Research Center (SERC) was used. The case study was based on an ongoing model-based systems engineering (MBSE) research project, which uses an unmanned aircraft system (UAS) as an experimental subject. The project includes multiple efforts such as physical factors and cost modeling [15]. As part of these efforts, the project includes a set of requirements for the UAS landing gear in text form, which were used as a foundation for the presented study.

For an initial assessment of the requirements, the developed NLP approach was applied to detect requirements that contain potential spelling errors, incorrect wordings, and term ambiguities by reviewing the results of the identified entities/terms and checking them for consistency and errors. For instance, if the NLP algorithm detected two different forms for the same entity, such as “airplane” and “plane,” one of the two could be replaced with the other to ensure consistency and that the entities would be identified as the same network nodes. Another example were appearances where terms were used as acronyms in some requirements and spelled out in others. Thus, using the detection of the NLP algorithm allowed for the identification and subsequent removal of clarity issues. This initial application resulted in a specification that included 79 individual requirements in tabular form. While these requirements were written with standards such as ISO 9001 in mind, they were not all in accordance with the standard. The robustness/error handling of the approach was also tested in the case study.

The 79 requirements were processed according to the steps shown in Figure 1. The result was a catalog of 389 identified entities within the 79 requirements that contained 246 unique individual entities. With the structural linking then, the connections between those terms were identified, which yielded the network of entities that the requirement structure is derived from. Figure 3 shows an excerpt of the adjacency matrix of the entities, with a legend provided below.

	A	B	C	D	E	F	G	H	I	J	…
A	0	1	0	0	0	0	0	0	0	0	…
B	1	0	1	0	0	0	0	0	0	0	…
C	0	1	0	0	0	0	0	0	0	0	…
D	0	0	0	0	1	0	0	0	0	0	…
E	0	0	0	1	0	1	0	0	0	0	…
F	0	0	0	0	1	0	1	0	0	0	…
G	0	0	0	0	0	1	0	1	0	0	…
H	0	0	0	0	0	0	1	0	0	0	…
I	0	0	0	0	0	0	0	0	0	1	…
J	0	0	0	0	0	0	0	0	1	0	…
…	…	…	…	…	…	…	…	…	…	…	…

```
A = landing gear structure
B = service life
C = air vehicle airframe structure
D = reversal
E = landing gear command
F = actuation
G = landing gear
H = last position
I = alternate extension system
J = capability
```

Figure 3 - Structural Entities Adjacency Matrix

The full matrix in Figure 3 has a size of 246 by 246 since it includes all the unique individual entities and their connections. Furthermore, a 79 by 79 adjacency matrix of the individual requirements was created to represent the network of the requirements, rather than the entities they contain. Based on these adjacency matrices, the structures can also be represented as an interactive network, which was realized using the pyvis library [44]. The result is a fully interactive network that allows for the assessment of connections and constellations, as shown in Figure 4.

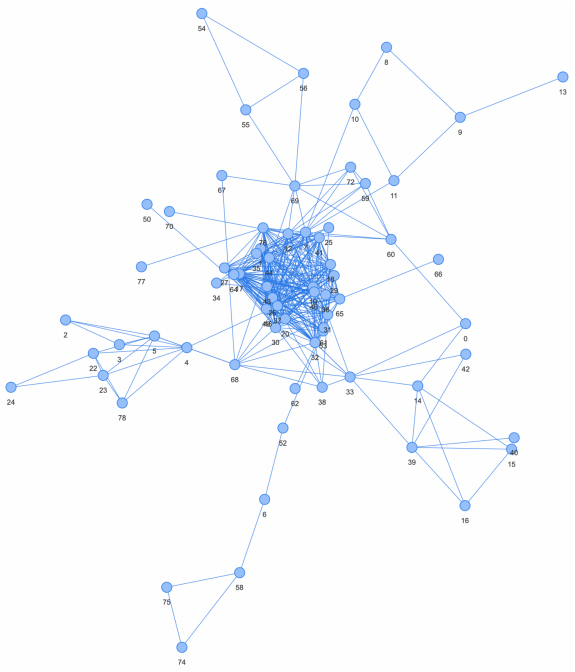

Figure 4 - Requirements Structure Network

With the produced results, the actual analysis was conducted to apply the metrics to the output. Thus, the density of the requirements network was calculated, and the entropy of the term/entity network, since these two metrics are applicable and, as shown in Section 2.2, form a good foundation for the development of further metrics. Since a single calculation of the two factors as a snapshot of the final state would provide only a limited view, their evolution through the requirements definition process was analyzed. The requirements specification was processed sequentially, and the two metrics were recorded for each additional requirement, allowing the progression over time to be observed.

The two charts shown in Figure 5 and Figure 6 illustrate the outputs discussed in Part 5.

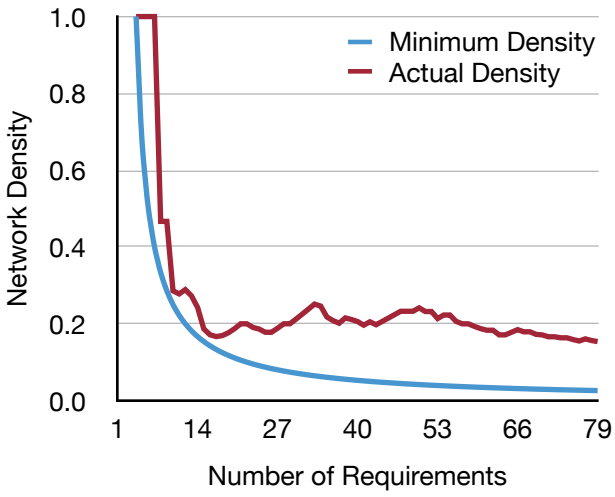

Figure 5 - Requirement Network Density - Progressive

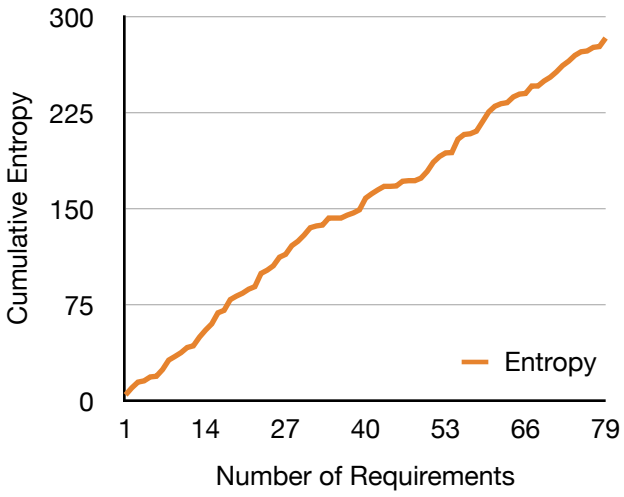

Figure 6 - Entity Network Entropy - Cumulative

Lastly, to test the validity of the results regarding the NLP output that was used to produce the calculated metrics above, a blind test was conducted to assess the correctness and validity of the results similar to Aurora et al. as well as Falessi and Layman [21, 45]. A specific metric was established to evaluate the quality of the results. To guide this evaluation, the accuracy and quality measures detailed by Derczynski [46], as well as Nakache, Metais, and Timsit [47], were used. Consequently, the precision value was calculated as a metric to measure correctness. Precision is the ratio of all true positives to the sum of all true positives and false positives. For the blind test, a human participant was given the requirements specification for the presented case and instructed to manually apply the rules and elicitation approach. The list of entities and the requirements network structure produced by the human participant were then cross-checked against the NLP approach's output, allowing for the precision to be calculated using the human results as a benchmark. The developed NLP approach was 99.74 percent correct regarding the identification of the entities within the requirements, which means that 388 out of the 389 entities were correctly identified. Additionally, the structure produced by the NLP approach was 98.71 percent correct, which equates to 310 out of the 314 non-duplicate connections in the network being correctly linked according to the defined rules and sequence of the algorithm.

5. DISCUSSION & INTERPRETATION

The results produced by the developed approach provide a range of interpretations and insights, even without a compound metric such as [1]. Thus, the paragraphs below will discuss the obtained results and how they can be analyzed, before discussing the current limitations and potential improvement points for the approach.

First, looking at Figure 5–the density measure of the requirements network–shows that the density trends downwards, which is to be expected in a growing network that is not fragmented and significantly disconnected. Yet, in contrast to the minimum density in the same graph, the actual density shows that it veers further away from the possible minimum. The minimum was calculated with the following formula:

$$d_{min} = \frac{(n-1)}{\frac{(n \cdot (n-1))}{2}} \quad \text{where } n \text{ is the number of nodes} \tag{1}$$

The formula above expresses the minimum number of connections necessary in a network (*n-1*) to connect all nodes with at least one edge, without creating fragments. This also implies that no loops are created, since a loop, regardless of its size, would necessitate an additional edge, and the resulting edge count would exceed the minimum density.

The progress of the density in Figure 5 shows that over time, the increasing distance from the minimum density means that more and more loops are introduced, which can eventually also be clearly seen in the visual representation in Figure 4. These loops are potentially problematic, as they can turn into reinforcing loops (negative as well as positive), which can make satisfaction of a requirement and change management [48] difficult. While the opposite is possible–the creation of balancing loops–the mere existence of loops increases tracking/tracing effort.

In addition to the issues created by loops, the growth or distance from the minimum density increases the number of connections overall, and thus the edges to track. In the case of humans and even machines working with the resulting structure, we argue that this increase leads to higher effort and cognitive difficulties that could negatively affect the development of the system in various ways, which metrics such as McCabe [12] corroborate. Validating these circumstances is currently being done in a follow-up case study and is planned to be published soon.

Lastly, the density graph also shows different segments with stronger and less strong implications. For instance, the requirements after number 31 increase the density disproportionately, which means that in this section, an unusually high number of connections is added to the system with respectively fewer nodes. Such insights can point to requirements of interest that merit scrutiny and could be assessed by a human in the loop regarding their role to either lessen their impact or consider their position.

Regarding Figure 6 and the entropy within the entity network, we can see a steadily increasing trend, which also was to be expected due to the growing size of the network throughout the requirements specification. Yet, similar insights to those gained from the density above can be deduced: the impact each requirement has on the overall entropy is not uniform. Some requirements introduce higher amounts of entropy than others, potentially due to a higher number of added nodes or connections. To investigate this, the number of entities per requirement was plotted versus the entropy impact of the respective requirement. The impact was obtained by removing the respective individual requirement and recalculating the entropy difference of the remaining structure. The results are shown in Figure 7 below.

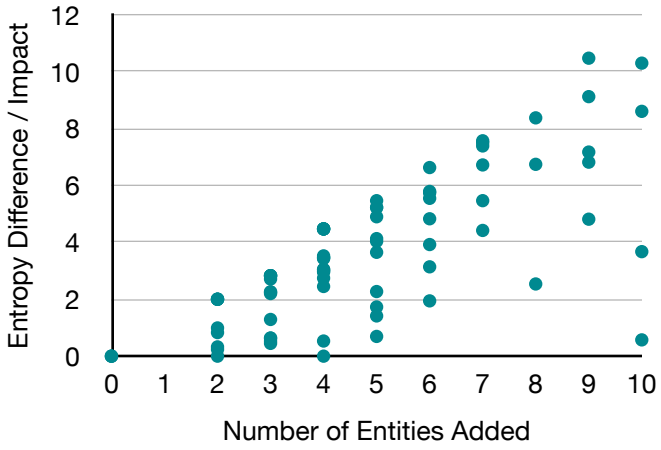

Figure 7 - Entropy Impact vs. Added Entity Number

Calculating the R and R^2 value of the scatter in Figure 7 reveals a strong correlation (R=0.75) between the number of entities added by a requirement and the entropy impact it has. The R^2 value is computed to be 0.5565. Thus, the number of entities identified in a requirement does correlate with the entropy it introduces, which allows for the argument that shorter requirements with fewer factors and entities add less entropy on a per requirement basis and are easier to process and understand.

Despite the insightful possibilities and analyses enabled by the metrics above, certain drawbacks have to be mentioned, which also motivated the future work described in the conclusion section. First and foremost, the metrics presented in this case study show a dependence on the overall size of the network and, consequently, the size of the requirements specification. Such issues have been noted by other researchers [22] who attempted to decouple their metrics from the size and scope of the network, but in doing so, introduced other potential drawbacks. Therefore, this issue is considered critically important for the future case studies currently underway for our approach.

Second, the presented assessment has some dependence on the sequence of the requirements catalog/specification. While the assumption can be made that the sequence of the requirements from beginning to end is of importance, this significance diminishes as the specification grows larger and once modularization or segmentation of specific requirement aspects, such as functional and non-functional requirements, for example, are reached. As a result, for these cases, the inclusion and reliance on sequence must be reassessed. This also applies to the same entities possibly detected in different modules, where a connection might not be relevant or accurate.

Yet, despite these considerations, the approach presented offers valuable insights and will be further improved.

Lastly, to discuss the limits of the current state of the research, three limitations/scope constraints should be mentioned. First, due to the researcher's knowledge of the case study, the created approach and its included rule set may be subject to overfitting to the specifics of the study. The implemented rules address potential problems and, for instance, significant deviations from standards, like the mentioned ISO 9001. However, this potential overfitting and associated restrictions do not constitute a hard limit of the developed approach, since the rules within the algorithm can be adjusted and amended flexibly. Therefore, we are confident that the developed approach can be transferred

to a variety of cases and problems without loss of functionality, provided that emerging issues and errors are addressed iteratively. This also means that other standards or semi-structured content are possible, albeit not functional yet, which is also subject of the next case study.

Second, existing limitations of NLP have to be mentioned. For instance, the current iteration of the algorithm does not include any analysis and treats different entities necessarily as unique. This could potentially lead to issues stemming from working ambiguities and variances. For instance, looking at Figure 3, we see that there are different variations that could all be interpreted as "landing gear," yet they are separated due to their different terminology. While this might be correct from a semantic perspective, general correctness cannot be assumed overall, and thus, such unification issues have to be considered moving forward. One possible solution would be a human-in-the-loop approach that allows for on-the-fly correction and adjustment of possible synonyms. The same issue applies even more to pronouns such as "that," for instance, which need to be replaced or explained to fulfill their correct role as a substitute.

Finally, the meaningfulness of the presented metrics has to be discussed. Since the analysis addresses one case study, the results cannot easily be transferred or compared to other cases without calibration. Due to the nature of language, deviations from standards and inherent randomness within text can have different implications for requirement sets and, consequently, for the metrics. These can and are planned to be addressed with a compound metric that allows for calibration.

The developed approach, with the correctness and validity numbers above, demonstrates its functionality and potential. Additionally, the application has revealed further benefits, such as the error detection possibilities and improvement recommendations for the requirements. While these benefits were not the main focus of approach development, they reveal an additional and arguably different direction for adaptation. Overall, given the successful application, expansion, and adaptation opportunities, we argue that there is great potential in this research.

6. CONCLUSION

The paper at hand presented a new automated system analysis approach based on requirements specifications in natural language text form. By using a sequence of NLP tools, an algorithm was developed that extracts three structural layers from a requirements document: the hierarchy structure of the requirements, the structure of the requirements based on their content, and the structure of the specific terms/entities within the requirements. With these results, the complexity of the requirement set can be assessed on each layer using different metrics. For the presented case study—an UAS—the developed approach achieved a correctness of 99.74 percent for the entities within the requirements, and the correctness of the produced structure was 98.71 percent.

With the obtained result, the case study was evaluated regarding the density of the requirements network and the spectral entropy of the entity network. The analysis showed an increasing trend for both metrics, indicating a growing complexity of the requirements throughout the specification process and an increasing potential for disadvantageous feedback loops in the network. However, the current metrics show a dependency on the size of the network, the size of the requirements specification, and the number of terms introduced by each requirement. The entropy impact of each requirement was correlated with the number of contained entities with an R-value of 0.75.

Current limitations of the approach are a potential overfitting to the presented case study. However, this can be mitigated by the flexible and adjustable rule set within the algorithm, which allows for transfer to other cases. Additionally, the use of NLP introduces certain limitations, such as potentially incorrect semantic links and different scaling challenges, which are to be addressed with future iterations and case studies.

Overall, the developed approach poses a new and useful tool to address the need for quantifiable complexity assessment, as outlined in the introduction. By making complexity measurable and quantifiable based on the requirements, the approach enables analyses that can be used to prevent unnoticed increases or jumps in complexity (complexity creep), disproportionate complexity levels compared to other and/or previous projects, and clusters that contribute significantly and excessively to the overall complexity. By monitoring and analyzing such metrics, projects can be better controlled, helping to reduce risk by providing a way to identify and address potential problems early before the consequences become visible and possibly irrevocable. Early identification and assessment are especially crucial for requirements, as they significantly affect nearly all other development phases.

Additionally, complexity–being characterized by emergent behavior–can also be quantified with the presented approach, which allows for potential minimization and or optimization as well.
Through the quantification and measurements enabled by the approach, the complexity introduced by requirements adds another factor to the sources of complexity in systems development, as outlined by Salado and Nilchiani [14]. Requirements complexity thus is an additional aspect of the overall development complexity sum and must be considered alongside other factors to provide a holistic overview for systems development. Furthermore, since requirements complexity is not always directly related to system components and parts, it offers an evaluation of complexity on a meta level, which also relates to human effort and project management factors.
In line with these benefits and opportunities, the authors are currently working on a follow-up case study to not only transfer the approach but also develop a compound metric that addresses the limitations and concerns of those presented in this paper. Furthermore, in preparation for future case studies, the approach has also been applied to a less structured requirement set that contained different input formats and partial tables. This set is publicly available as part of [49]. In initial tests, the approach demonstrated great potential for accuracy, and an expanded set of correctness metrics is currently being developed as well.

BIBLIOGRAPHY

[1] K. Sinha and O. L. d. Weck, "A network-based structural complexity metric for engineered complex systems," in 2013 IEEE International Systems Conference (SysCon), 15-18 April 2013 2013, pp. 426-430, doi: 10.1109/SysCon.2013.6549917.

[2] M. V. Arena, O. Younossi, K. Brancato, I. Blickstein, and C. A. Grammich, Why Has the Cost of Fixed-Wing Aircraft Risen? A Macroscopic Examination of the Trends in U.S. Military Aircraft Costs over the Past Several Decades. Santa Monica, CA: RAND Corporation, 2008.

[3] D. Nagar, A. Furman, and G. Nitschke, "The Cost of Complexity in Robot Bodies," in 2019 IEEE Congress on Evolutionary Computation (CEC), 10-13 June 2019 2019, pp. 2713-2720, doi: 10.1109/CEC.2019.8790084.

[4] H. A. Bashir and V. Thomson, "Estimating Design Complexity," Journal of Engineering Design, vol. 10, no. 3, pp. 247-257, 1999/09/01 1999, doi: 10.1080/095448299261317.

[5] H. A. Bashir and V. Thomson, "Models for estimating design effort and time," Design Studies, vol. 22, no. 2, pp. 141-155, 2001/03/01/ 2001, doi: 10.1016/S0142-694X(00)00014-4.

[6] K. Sinha and O. L. de Weck, "Empirical Validation of Structural Complexity Metric and Complexity Management for Engineering Systems," Systems Engineering, vol. 19, no. 3, pp. 193-206, 2016, doi: https://doi.org/10.1002/sys.21356.

[7] K. Miettinen, Nonlinear Multiobjective Optimization. Boston, MA: Springer US, 1998.

[8] W. W. Royce, "Managing the Development of Large Software Systems," presented at the IEEE WESCON, Los Angeles, CA, August 25-28, 1970.

[9] B. W. Boehm, "Guidelines for Verifying and Validating Software Requirements and Design Specifications," presented at the Euro IFIP 79, 1979.

[10] B. W. Boehm, Software Engineering Economics. Englewood Cliffs, NJ: Prentice Hall, 1981.

[11] R. G. Cooper, "Stage-gate systems: A new tool for managing new products," Business Horizons, vol. 33, no. 3, pp. 44-54, 1990/05/01/ 1990, doi: 10.1016/0007-6813(90)90040-I.

[12] T. J. McCabe, "A Complexity Measure," IEEE Transactions on Software Engineering, vol. SE-2, no. 4, pp. 308-320, 1976, doi: 10.1109/TSE.1976.233837.

[13] M. H. Halstead, Elements of Software Science (Operating and programming systems series). Elsevier Science Inc., 1977.

[14] A. Salado and R. Nilchiani, "The Concept of Problem Complexity," Procedia Computer Science, vol. 28, pp. 539-546, 2014/01/01/ 2014, doi: 10.1016/j.procs.2014.03.066.

[15] M. R. Blackburn, T. McDermott, B. Kruse, J. Dzielski, and T. Hagedorn, "Digital Engineering Measures Correlated to Digital Engineering Lessons Learned from Systems Engineering Transformation Pilot," INSIGHT, vol. 25, no. 1, pp. 61-64, 2022, doi: https://doi.org/10.1002/inst.12375.

[16] L. Zhao et al., "Natural Language Processing for Requirements Engineering: A Systematic Mapping Study," ACM Comput. Surv., vol. 54, no. 3, p. Article 55, 2021, doi: 10.1145/3444689.

[17] M. Vierlboeck, C. Lipizzi, and R. Nilchiani, "Natural Language in Requirements Engineering for Structure Inference--An Integrative Review," arXiv preprint arXiv:2202.05065, 2022.

[18] A. Ferrari, S. Gnesi, and G. Tolomei, "Using Clustering to Improve the Structure of Natural Language Requirements Documents," Berlin, Heidelberg, 2013: Springer Berlin Heidelberg, in Requirements Engineering: Foundation for Software Quality, pp. 34-49.

[19] M. Hamza and R. J. Walker, "Recommending Features and Feature Relationships from Requirements Documents for Software Product Lines," in 2015 IEEE/ACM 4th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering, 17-17 May 2015 2015, pp. 25-31, doi: 10.1109/RAISE.2015.12.

[20] S. Tahvili et al., "Functional Dependency Detection for Integration Test Cases," in 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), 16-20 July 2018 2018, pp. 207-214, doi: 10.1109/QRS-C.2018.00047.

[21] C. Arora, M. Sabetzadeh, L. Briand, and F. Zimmer, "Extracting domain models from natural-language requirements: approach and industrial evaluation," presented at the Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, Saint-malo, France, 2016.
[22] M. Lei, L. Liu, and D. Wei, "An Improved Method for Measuring the Complexity in Complex Networks Based on Structure Entropy," IEEE Access, vol. 7, pp. 159190-159198, 2019, doi: 10.1109/ACCESS.2019.2950691.
[23] M. Vierlboeck and R. R. Nilchiani, "Requirement Engineering in the Age of System and Product Complexity - A Literature Review," in 2021 IEEE International Symposium on Systems Engineering (ISSE), 13 Sept.-13 Oct. 2021 2021, pp. 1-8, doi: 10.1109/ISSE51541.2021.9582439.
[24] C. E. Shannon, "A mathematical theory of communication," The Bell System Technical Journal, vol. 27, no. 3, pp. 379-423, 1948, doi: 10.1002/j.1538-7305.1948.tb01338.x.
[25] S. A. Akundi, Information entropy measures applied to hierarchial complex technical and soci-technical systems. The University of Texas at El Paso, TX, 2016.
[26] J. Fischi and R. Nichiani, "Complexity Based Risk Evaluation in Engineered Systems," Procedia Computer Science, vol. 44, pp. 31-41, 2015/01/01/ 2015, doi: 10.1016/j.procs.2015.03.044.
[27] M. Gell-Mann, "What is complexity? Remarks on simplicity and complexity by the Nobel Prize-winning author of The Quark and the Jaguar," Complexity, vol. 1, no. 1, pp. 16-19, 1995, doi: 10.1002/cplx.6130010105.
[28] M. Gell-Mann and S. Lloyd, "Information measures, effective complexity, and total information," Complexity, vol. 2, no. 1, pp. 44-52, 1996, doi: 10.1002/(SICI)1099-0526(199609/10)2:1<44::AID-CPLX10>3.0.CO;2-X.
[29] M. H. Halstead, Elements of Software Science (Operating and Programming Systems Series). Elsevier, 1977.
[30] I. Gutman and B. Zhou, "Laplacian energy of a graph," Linear Algebra and its Applications, vol. 414, no. 1, pp. 29-37, 2006/04/01/ 2006, doi: 10.1016/j.laa.2005.09.008.
[31] I. Gutman, "Hyperenergetic and hypoenergetic graphs," Selected Topics on Applications of Graph Spectra, Math. Inst., Belgrade, 2011.
[32] V. Nikiforov, "The energy of graphs and matrices," Journal of Mathematical Analysis and Applications, vol. 326, no. 2, pp. 1472-1475, 2007/02/15/ 2007, doi: 10.1016/j.jmaa.2006.03.072.
[33] W. Jun, M. Barahona, T. Yue-Jin, and D. Hong-Zhong, "Natural Connectivity of Complex Networks," Chinese Physics Letters, vol. 27, no. 7, p. 078902, 2010/07 2010, doi: 10.1088/0256-307x/27/7/078902.
[34] K. Sinha, "Structural Complexity and its Implications for Design of Cyber-Physical Systems," Doctor of Philosophy, Engineering Systems Division, Massachusetts Institute of Technology, 2014.
[35] A. Sharma and D. S. Kushwaha, "Natural language based component extraction from requirement engineering document and its complexity analysis," SIGSOFT Softw. Eng. Notes, vol. 36, no. 1, pp. 1-14, 2011, doi: 10.1145/1921532.1921547.
[36] W. M. Purawinata, F. L. Gaol, A. Nugroho, and B. S. Abbas, "The prediction of software complexity based on complexity requirement using artificial neural network," in 2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), 20-22 Nov. 2017 2017, pp. 73-78, doi: 10.1109/CYBERNETICSCOM.2017.8311687.
[37] S. E. Phelan, "What Is Complexity Science, Really?," Emergence, vol. 3, no. 1, pp. 120-136, 2001/04/01 2001, doi: 10.1207/S15327000EM0301_08.
[38] S. H. Strogatz, SYNC: the emerging science of spontaneous order. New York, NY: Hyperion, 2004.
[39] J. K. DeRosa, A. M. Grisogono, A. J. Ryan, and D. O. Norman, "A Research Agenda for the Engineering of Complex Systems," in 2008 2nd Annual IEEE Systems Conference, 7-10 April 2008 2008, pp. 1-8, doi: 10.1109/SYSTEMS.2008.4518982.
[40] J. Cotler, N. Hunter-Jones, J. Liu, and B. Yoshida, "Chaos, complexity, and random matrices," Journal of High Energy Physics, vol. 2017, no. 11, p. 48, 2017/11/09 2017, doi: 10.1007/JHEP11(2017)048.
[41] "spaCy." https://spacy.io (accessed June 8, 2022).
[42] "Universal POS tags." Universal Dependencies. https://universaldependencies.org/u/pos/ (accessed September 1, 2022).
[43] "NetworkX." https://networkx.org (accessed June 3, 2024).
[44] "pyvis." https://pyvis.readthedocs.io/en/latest/ (accessed June 8, 2022).
[45] D. Falessi and L. Layman, "Automated classification of NASA anomalies using natural language processing techniques," in 2013 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), 4-7 Nov. 2013 2013, pp. 5-6, doi: 10.1109/ISSREW.2013.6688849.
[46] L. Derczynski, "Complementarity, F-score, and NLP Evaluation," Portorož, Slovenia, May 2016: European Language Resources Association (ELRA), in Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), pp. 261-266. [Online]. Available: https://aclanthology.org/L16-1040. [Online]. Available: https://aclanthology.org/L16-1040
[47] D. Nakache, E. Metais, and J. F. Timsit, "Evaluation and NLP," Berlin, Heidelberg, 2005: Springer Berlin Heidelberg, in Database and Expert Systems Applications, pp. 626-632.
[48] U. Lindemann and R. Reichwald, Integriertes Änderungsmanagement. Berlin Heidelberg, Germany: Springer-Verlag (in German), 1998.
[49] "ARCHITECTING SPACECRAFT WITH SYSML." http://sysml-models.com/spacecraft/models.html (accessed September 1, 2022).