International Journal on Advanced Science Engineering Information Technology

Vol.10 (2020) No. 3
ISSN: 2088-5334

An Ontology Framework for Generating Requirements Specification

Amarilis Putri Yanuarifiani[a,1], Fang-Fang Chua[a,2], Gaik-Yee Chan[a,3]

[a] *Faculty of Computing and Informatics, Multimedia University, Cyberjaya, Malaysia*
E-mail: [1]1161401040@student.mmu.edu.my; [2]ffchua@mmu.edu.my; [3]gychan@mmu.edu.my

***Abstract*— Requirements engineering is the initial process of software development that critically determines the overall software process quality. However, this process is error-prone. This is generally related to the factors of communication, knowledge, and documentation. With the lack of business knowledge, it is complicated for (technical) engineers to define customer needs. Also, modeling and documenting requirements need much time and effort to ensure that the requirements are valid, and nothing is missed. The current approach of the requirements modeling process mostly focuses on the Unified Modeling Notation (UML) use case, that not provide enough information for a stakeholder to define system requirements. The generated SRS is still lack of detail development guideline that increase risk of development error. The purpose of this research is to guide for the elicitation process to avoid missing and mismatch requirements, and to make the modeling and documentation process more effective and efficient. We propose an ontology framework for generating requirements specification. This framework, namely the Rule-Based Ontology Framework (ROF) consists of two main processes: First, requirements elicitation. This step provides a guideline for the stakeholder to define system requirements based on the problem of the current system and business process enhancement. Based on this final requirement list, the requirements ontology is generated. Second, the auto-generation of the requirements specification document. The document consists of semi-formal modeling and natural language. In this research, we use Business Process Modeling Notation (BPMN) is a modeling language. For natural language documents, we use IEEE for the SRS template.**

***Keywords*— requirements elicitation; requirements engineering; ontology; Business Process Modeling Notation (BPMN).**

I. INTRODUCTION

One of the reasons for the failure of software projects is poor management of requirements. Among the most common issues are endless changes, incompleteness, ambiguous, unrealistic requirements, and up to inappropriate modeling process. Particularly for Requirements Engineering (RE), which is the first process of software development that defines requirements and produces Requirements Specification (RS) as an output, mismatch requirements lead to significant losses to the project.

Commonly, requirements specification is built only based on general requirements which explicitly written by the customer. This document usually contains only initial requirements and is not the complete version. With the lack of business knowledge, it is challenging for engineers to define customer needs. On the other hand, modeling and documenting requirements need much time and effort to ensure that the requirements are valid, and nothing is missed. Errors in this phase could be transferred in the subsequent phases of the software development and compromise the overall process or increase the cost of the development [1]. The use of an ontology helps to formalize the structure of requirements specification in a way that system can be understood. Study in [2] focuses on developing requirements ontologies that define requirements in general, their main types, how they are documented, and also how they are using requirements during software execution.

Research question raised are: 1) How to elicit all customer needs and avoid missing and mismatch requirements? 2) How to improve process efficiency in building software requirements specification by reducing the manual writing process. This paper is proposing a framework to build the requirements specification document using the ontology approach. The framework consists of two main processes: First, Requirements elicitation process. Second, a process to build requirements specification documents automatically, which includes Business Process Modeling Notation (BPMN) and natural language template. By using the framework, it is expected to avoid missing and mismatched requirements. Automation of process modeling and documentation requirements can make the process more effective and efficient.

Requirements Engineering is a process to help identify, clarify, and agree upon the actual requirements with the customer [3]. The activities start with eliciting the requirements from stakeholders, document it in a certain template, continued by validating the criteria to make sure its completeness, and manage the requirements changing. This research only focusses on the first two processes, which are:

requirements elicitation and requirements documentation. In contrast, requirements validation and negotiation, and requirements management will be considered as the next stage in future works.

II. Materials and Method

A. Requirements Elicitation

Requirements elicitation is the process of in-depth and thorough information finding from all stakeholders associated with the built software [4]. Table 1 shows a summary of the classification of the traditional requirements elicitation techniques

TABLE I
Traditional Requirements Elicitation Techniques

Technique	Category [5]
Classic/Traditional	Interview, Questionnaires survey, task/domain analysis, introspection
Contextual	Observation/Social analysis
Modern and Group	Prototyping workshops, brainstorming, group work, use cases, JAD/RAD
Cognitive	Card sorting, laddering CRC, Repertory Grids

Problems arising from the implementation of traditional techniques of the elicitation process are [6]–[8]: 1. The effectiveness and efficiency of the techniques depend on the scope of the project 2. Stakeholders difficult to define the system according to their needs 3. Limited time and resources 4. Limited knowledge related to the company's business processes. Regarding these issues, several methods have been proposed with different approaches (Table 2).

TABLE II
Approach for Elicitation Process

Method	Review
Goad-oriented [9]	Focuses on managing final requirements from the stakeholder. It does not explain how the stakeholder defines requirements from scratch.
Conceptual model [10]	Works well only if stakeholders have enough knowledge to define the needs of the system.
User-centered design approach [11]	Runs effectively on commercial software, where the users of the services in a very large number.

In respect of these previous research [9]–[11], none of them have detailed explanations to assist stakeholders in defining their needs from scratch. This research applied traditional techniques, such as interviews and brainstorming, for setting the list of the initial requirements and continued with improving the Analytics Method [6] by proposing Gap Identification Process. It guides both stakeholders and engineers to identify current system problems and build system to solve those problems. This proposed method also ensure system enhancement is aligned with business process enhancement.

Once the requirements are collected, the process will be continued by determining the priorities of each requirement. To choose the appropriate technique, it is essential to consider several attributes such as time, effort, and project properties. In many projects, simple ad hoc prioritization techniques are well suited. Compared to other techniques that address customer needs analysis, the Kano model has been applied and proven widely in various industries. [12].

Kano model is a technique that classifies customer desires by assessing how important a product or functionality can meet customer needs. In many cases, it is complicated to see the customer's needs clearly; therefore, the Kano model provides positive and negative questionnaires to ensure each functionality matches the customer's expectations. The results of the questionnaire classify functionality into several areas, which will then determine whether the functionality needs to be implemented or even to be eliminated. (as shown in fig 1).

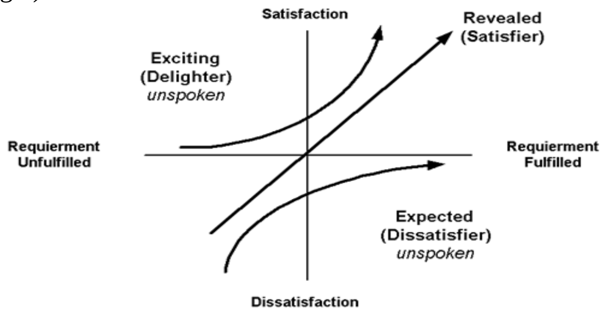


Fig. 1 The Kano diagram

Revealed requirements represent what customers want by explicitly doing the interview. Expected requirements represent a critical requirement that has to be implemented; otherwise, the system will be considered to fail. Those are often basic, which stakeholders may fail to describe. Exciting requirements is hard to describe, which even stakeholder might not even consider it. The proposed framework used the Kano model as the validation method to finalize and prioritize the requirements list. By answering Kano questionnaires, stakeholders define the priority of requirements and confirm if any of them should be eliminated.

B. Requirements Documentation

Requirements documentation [13] is a process to write the requirement in a certain way that represents system needs. Documentation could be classified based on the degree of formality:

- Informal Method: the most natural way of expressing requirements, even it considers lacking the need for quality. i.e., IEEE SRS template.
- Semi-Formal Method: Graphical representation of a system with specific symbols. i.e., Unified Modelling Language (UML) or Business Process Modelling Notation (BPMN)
- Formal Method: Minimize ambiguity and support full automatic test case generation, but it costs much for software performance, i.e., methods are Z Notation.

Natural language strictly needs to be built to gain stakeholder understanding. Nevertheless, a Semi-formal modeling approach is needed as a bridging language between the technical and business domain. Building documentation in natural language with minimum errors in syntax and semantic perspectives is challenging. UML use-case diagram is a good way to represent top-level requirements, but UML does not completely represent important features associated with the systems [14]. When compared to other business process modeling languages, emphasize that BPMN may be considered as the de facto standard for business process modeling [15]. This research focuses on documenting requirements specification through informal method (IEEE SRS Template) and semi-formal way (BPMN).

BPMN is developed by Object Management Group (OMG), which is dedicated to being easily read by a user with both business and technical perspectives. The basic categories of elements are [16]: Flow object, Connecting objects, Swim lanes, and Artifacts.

Auto-generating requirements specification propose a better way of documenting requirements by minimizing human error and efficiently reduce time to build specification. It needs the knowledge domain that can store the interrelationship between the functions on the requirements list. An ontology represents requirements in standard terminology, and stores its knowledge using concepts, attributes, and relations among concept instances. Therefore, ontology is used as the knowledge domain of the requirements list.

There are two main benefits [17] for using ontology as domain knowledge. First, relate the semantic processing of requirements. To build requirement specification effectively, processing of requirement description is necessary. With ontology representation, it is easier to detect basic semantic properties. By applying the rules, we can specify the level of completeness for the requirements description. Secondly, we can use the (semi) automated technique to build an ontology. Existing works [18] propose concepts to build ontology from the middle to large text documents. These concepts can help to reduce the effort of building domain ontologies from the scratch.

Previous research [19], [20] using ontology to build requirements specification document (Table 3) in natural language. Most of these tools generate documents based on one specific template, i.e., IEEE template.

TABLE III
REQUIREMENTS DOCUMENTATION METHOD

Method	Review
GUITAR's project [19]	Ontology-based tools to automatically transform the natural language specification into a structured specification.
Ontology approach [20]	Using ontology engineering to store and include additional information to build SRS in the IEEE format.

Ontology is used as domain knowledge in the proposed framework to auto-generate requirements specification document which consists of semi-formal modelling (BPMN) and natural language template.

III. RESULTS AND DISCUSSION

This research proposes a framework to build requirements specification document using ontology approach. The input, process and output of every stages shown in Fig 2.

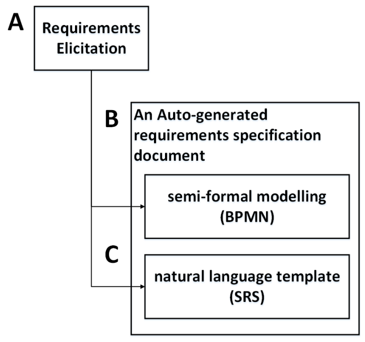

A
Requirements Elicitation
B
An Auto-generated requirements specification document
semi-formal modelling (BPMN)
C
natural language template (SRS)

Fig. 2 Rule-Based Ontology Framework (ROF)

- Stage A is a process of gathering all related documents and stakeholder information to elicit company needs. Requirements list will be transformed into ontology form.
- Stage B is a process of generate requirements specification in modelling language by mapping the instance of the ontology into the elements of BPMN. Output of this stage is BPMN diagram which is generated from the ontology.
- Stage C is a process of generate requirements specification in natural language by mapping requirements ontology to SRS template (IEEE). The output of this stage is structured requirements specification document in natural language.

A. *Requirement Elicitation*

Fig 3 shows the requirements elicitation process which consists of four main processes: Gathering initial requirements, identifying gap, prioritizing requirements list using Kano model, and building requirements ontology. The purpose of this process is to define functional requirements in ontology form.

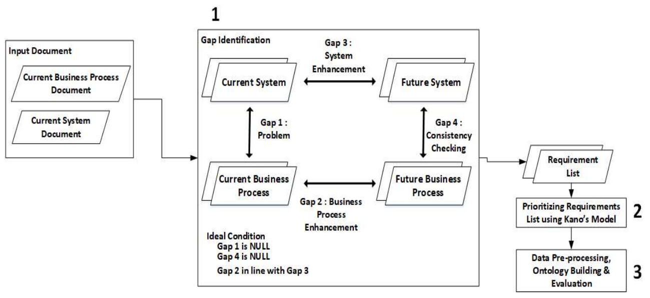


Fig. 3 Elicitation process

1) Gap Identification: It is the process to identify the gap between business and system enhancement to ensure final requirements solving company issues through discussion and brainstorming with stakeholder. Main input of this process is current business process document and Current System document. Fig 3 shows four information classification in gap identification stage which consists of: current system, current business process, future system, and future business process. Gap 1 is problem. Problems occur due to the business process that is not yet supported by the current system. In some cases, even if it has been supported, the process still faces some difficulties. By finding the problem, we can define the solution which will be transformed as requirement list. Errors or missing information in source documents often occur that can make the gap identification process ineffective. Therefore, it must be ensured that all documents have been validated. Gap 2 is Business Process Enhancement. Changes in business processes become one of the biggest reasons of project failure. It is necessary to identify the gap between current and future business process to ensure that the system can be used effectively. Gap 3 is System Enhancement. When problem and business process enhancement is defined, by comparing functional of current system and requirements list, it can be seen how much the system will be changed, and how much effort to implement the requirements.

TABLE IV
GAP 4 CONSISTENCY CHECKING

No	Process Name (Future)	Future System Type	Requirement List (Rev)
1	Build workload rubric template	User interaction	The system should provide university HR administrator with the ability to manage workload rubric template.
2	Distribute workload rubric template	User interaction	The system should provide university HR administrator with the ability to activate workload rubric template with workload template form as input.
3	Review workload rubric template	User interaction	The system should provide faculty HR administrator with the ability to access workload rubric template

Gap 4 is Consistency Checking. This process ensures all future business processes are supported by new system. Gap is identified by checking if there is any process which is not supported in requirements list. Requirements will be revised until no gap is identified. Table 4 shows example of consistency checking which produce requirements list as output.

TABLE V
KANO QUESTIONNAIRE BASED ON REQUIREMENTS LIST

If the system provide functionality to check registration progress, how do you feel?	Like Must Do not care Can't live with it Dislike
If the system does not provide functionality to check registration progress, how do you feel?	Like Must Do not care Can't live with it Dislike

1) *Prioritizing Requirements List using Kano Model:* Based on Kano model, all requirements list will be generated to a questionnaire which consists of positive and negative questions (Table 5). There will be five possible answers for each process. Questionnaires will be distributed to the stakeholders. Results of the questionnaire are calculated according to Kano formula [21] and classified with following order: Must-Be, One-Dimensional, Attractive, Indifferent, and Reverse requirements (Fig 4). It is stakeholder's decision to skip or postpone in filtering of any of those requirements.

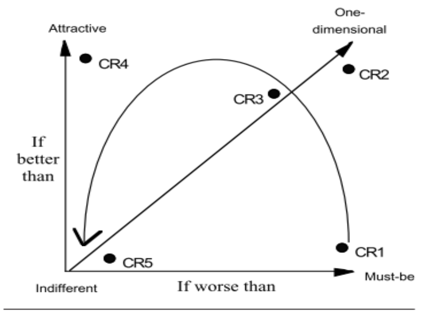


Fig. 4 Order of decreasing importance

2) *Data Pre-processing, Ontology Building and Evaluation:* This process includes defining the class, instance, and relationship of ontology from requirements list. By using wordnet, we can get Part-of-Speech (POS) of every sentence [22] and rule-based approach is used to build ontology instance and relationship. The Output of this process is Functional Requirements (ontology form). Fig 5 shows the concept of ontology.

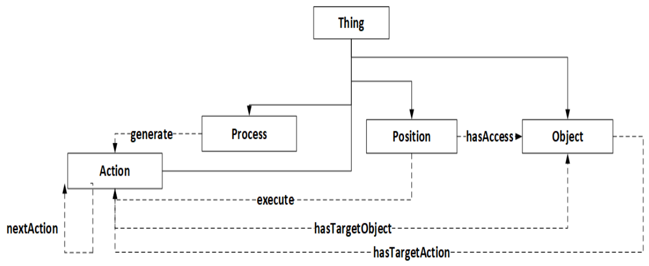


Fig. 5 Ontology Concept

B. *Requirement Documentation*

1) *Auto-Generate Requirements Specification Document –Modelling Language in Semi-Formal Method (BPMN):* Fig 6 shows auto-generate requirements model process which consists of two processes: Map the instances of requirements ontology to the BPMN basic elements and generate structure file in XML format. Purpose of this process is generating requirements model in semi-formal method.

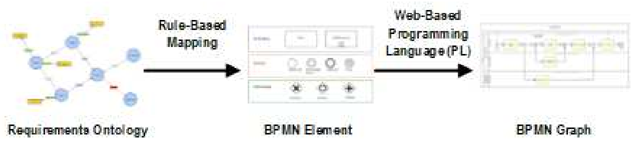


Fig. 6 Process of modelling language in semi-formal method (BPMN)

In this process, the final representation of the requirements specification is the BPMN chart. Rule-based method is a process to map the requirements ontology instance into BPMN elements (Fig 7). The BPMN elements are implemented in Web-based programming language to visualize the diagram. We use .xml file as data type and java as programming language.

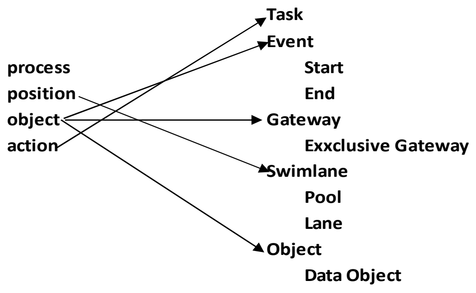


Fig. 7 Ontology Mapping to BPMN Elements

2) Auto-Generate Requirements Specification Document – Natural Language Template: Fig 8 shows auto-generate requirements specification process in natural language which consists of two main processes: Map the instance of Requirements Ontology to IEEE Section and Generate file in .docx format. In this process, the final representation of the requirements specification is natural language in IEEE SRS template.

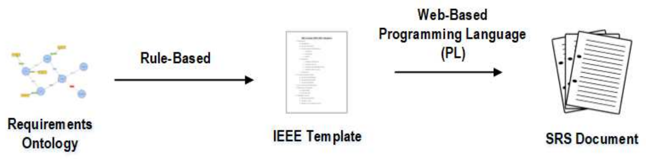
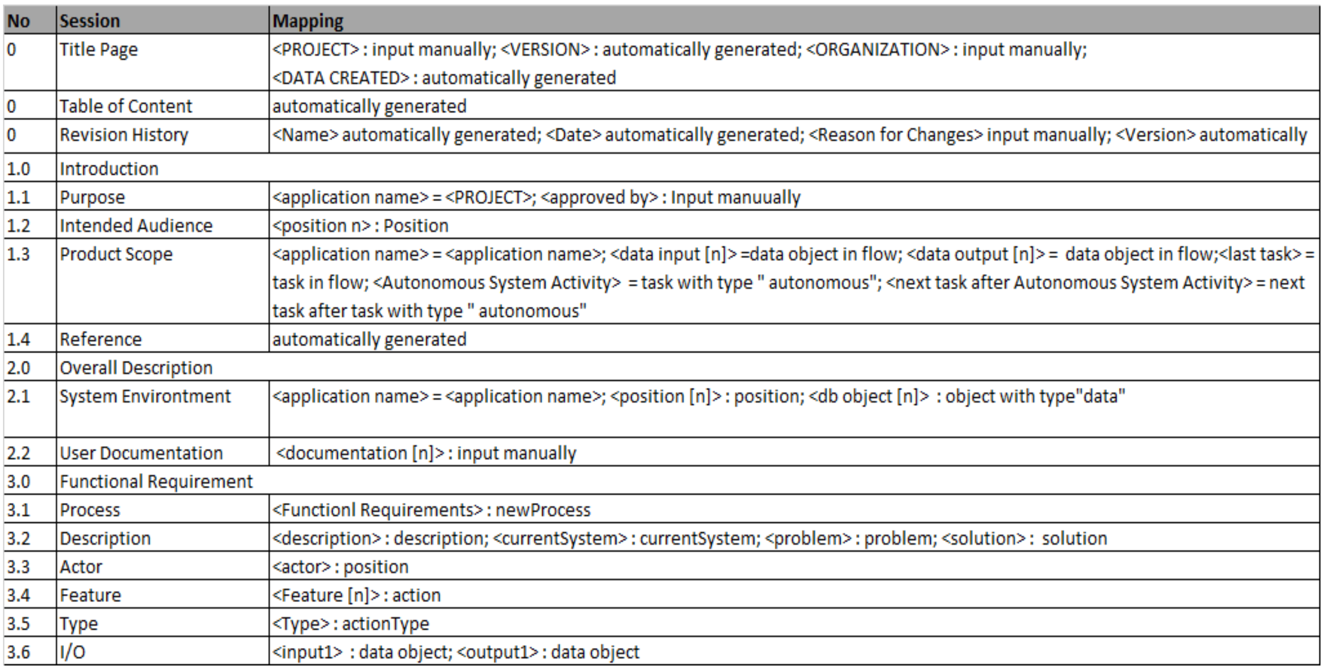


Fig. 8 Process of building document in natural language template

This process includes defining specific template of SRS. In this case, we use IEEE SRS template with eliminating non-functional section (Figure 6). The ontology instances are mapped into SRS template. Using Web based programming language, word document is generated. We provide interface for user to add some additional information in GUI to complete the SRS such as: name of project and revision history.



No	Session	Mapping
0	Title Page	<PROJECT> : input manually; <VERSION> : automatically generated; <ORGANIZATION> : input manually; <DATA CREATED> : automatically generated
0	Table of Content	automatically generated
0	Revision History	<Name> automatically generated; <Date> automatically generated; <Reason for Changes> input manually; <Version> automatically
1.0	Introduction	
1.1	Purpose	<application name> = <PROJECT>; <approved by> : Input manuually
1.2	Intended Audience	<position n> : Position
1.3	Product Scope	<application name> = <application name>; <data input [n]> =data object in flow; <data output [n]> = data object in flow;<last task> = task in flow; <Autonomous System Activity> = task with type " autonomous"; <next task after Autonomous System Activity> = next task after task with type " autonomous"
1.4	Reference	automatically generated
2.0	Overall Description	
2.1	System Environtment	<application name> = <application name>; <position [n]> : position; <db object [n]> : object with type"data"
2.2	User Documentation	<documentation [n]> : input manually
3.0	Functional Requirement	
3.1	Process	<Functionl Requirements> : newProcess
3.2	Description	<description> : description; <currentSystem> : currentSystem; <problem> : problem; <solution> : solution
3.3	Actor	<actor> : position
3.4	Feature	<Feature [n]> : action
3.5	Type	<Type> : actionType
3.6	I/O	<input1> : data object; <output1> : data object

Fig. 9 IEEE template mapping

By using this framework, it is expected that requirements engineering process performed in a more efficient way. Evaluation of proposed framework performance will be done by implementing a case study in Information system unit at Telkom University, Bandung, Indonesia. The case study consists of two module programs which are: Workload Application. Evaluation is conducted by validating requirements ontology and requirements specification document. This implementation using similarity metrics for validating generated BPMN and checklist for validating generated SRS [23].

Quality aspect of requirements specification [13] that will be used are:

- Quality aspect "content": completeness and correctness
- Quality aspect "documentation": unambiguity and Conformity to document format and to documentation structures
- Quality aspect "agreement": agreed and agreed after changes

IV. Conclusion

Requirements engineering is the initial process in software development that critically determines the success of the next process. This paper focuses on reducing missing and mismatched requirements in elicitation process and increase effectiveness and efficiency in modelling and documentation process. We propose an ontology framework for generating requirements specification which consists of two main processes. The requirements elicitation process and auto-generation of requirements specification document which consists of BPMN and natural language template.

This framework may encourage other researchers to elicit initial requirement in different methods. The proposed gap identification process contributes in processing resource document to gain semantic information. Building ontology can be improved automatically since requirements list has been defined in a structured template.

REFERENCES

[1] P. Spoletini, "Requirements Elicitation: A Look at the Future through the Lenses of the Past," IEEE 25th Int. Requir. Eng. Conf., 2017.

[2] B. Borlini, A. Luiz, D. C. Leal, R. De Almeida, and G. Guizzardi, "Ontological Foundations for Software Requirements with a Focus on Requirements at Runtime," Appl. Ontol., vol. 13, pp. 1–3, 2018.

[3] R. Chitchyan, C. Becker, and B. Penzenstadler, "Sustainability Design in Requirements Engineering: State of Practice," Proc. 38th Int. Conf. Softw. Eng. Companion, pp. 533–542, 2016.

[4] M. A. Ramdhani, D. Sa, A. S. Amin, and H. Aulawi, "Requirements Elicitation in Software Engineering," Int. J. Eng. Technol., vol. 7, pp. 772–775, 2018.

[5] R. N. Rehman T., Naeem M., "Analysis of Requirement Engineering Processes, Tools / Techniques and Methodologies," I.J. Inf. Technol. Comput. Sci., vol. 3, pp. 40–48, 2013.

[6] O. Isam, A. Mrayat, N. Norwawi, and N. Basir, "Requirements Elicitation Techniques: Comparative Study," Int. J. Recent Dev. Eng. Technol., vol. 1, pp. 1–10, 2013.

[7] S. Khan, A. B. Dulloo, and M. Verma, "Systematic Review of Requirement Elicitation Techniques," Int. J. Inf. Comput. Technol., vol. 4, pp. 133–138, 2014.

[8] S. Sharma, "Requirements Elicitation: Issues and Challenges," Comput. Sustain. Glob. Dev. (INDIACom), 2014 Int. Conf., pp. 151–155, 2014.

[9] F. Adikara, B. Hendradjaya, and B. Sitohang, "Organization Goal-Oriented Requirements Elicitation Process to Enhance Information System," Int. J. Electr. Comput. Eng., vol. 6, no. 6, p. 12802, 2016.

[10] J. Melegati, A. Goldman, and S. Paulo, "Requirements Engineering in Software Startups: a Grounded Theory approach," IEEE Int. Technol. Manag. Conf., pp. 57–65, 2016.

[11] P. Jakkaew, "Requirements Elicitation to Develop Mobile Application for Elderly," 2017 Int. Conf. Digit. Arts, Media Technol., no. March, pp. 464–467, 2019.

[12] Q. Xu, R. J. Jiao, X. Yang, M. Helander, H. M. Khalid, and A. Opperud, "An analytical Kano model for customer need analysis," Des. Stud., vol. 30, pp. 87–110, 2009.

[13] K. Pohl, Requirements Engineering: Fundamentals, Principles, and Techniques, vol. 1. 2010.

[14] E. Pereira, A. Rettberg, and S. Soares, "Model-based requirements specification of real-time systems with UML, SysML and MARTE," Softw. Syst. Model., vol. 17, pp. 343–361, 2018.

[15] R. S. K.Deyb, Alaaeddine Yousfi, Christine Bauer, "uBPMN: A BPMN extension for modeling ubiquitous business processes," Inf. Softw. Technol., vol. 24, pp. 55–68, 2016.

[16] M. Geiger, S. Harrer, J. Lenhard, and G. Wirtz, "BPMN 2.0: The state of support and implementation Matthias," 2016.

[17] H. Kaiya, "Using Domain Ontology as Domain Knowledge for Requirements Elicitation," Requir. Eng. 14th IEEE Int. Conf., vol. 14, pp. 189–198, 2006.

[18] M. El Ghosh, H. Naja, H. Abdulrab, and M. Khalil, "Ontology Learning Process as a Bottom-up Strategy for Building Domain-specific Ontology from Legal Texts Ontology Learning Process as a Bottom-up Strategy for Building Domain-specific Ontology from Legal Texts," J. Intell. Manuf., vol. 27, pp. 263–282, 2016.

[19] T. H. Nguyen, J. Grundy, and M. Almorsy, "GUITAR: An ontology-based automated requirements analysis tool," 2014 IEEE 22nd Int. Requir. Eng. Conf. RE 2014 - Proc., vol. 22, pp. 315–316, 2014.

[20] R. a. Elliott, Sr. and E. B. Allen, "A Methodology for Creating an IEEE Standard 830-1998 Software Requirements Specification Document," J. Comput. Sci. Coll., vol. 29, pp. 123–131, 2013.

[21] I. W. Taifa and D. A. Desai, "Student-Defined Quality by Kano Model: A Case Study of Engineering Students in India," Int. J. Qual. Res., vol. 10, no. 3, pp. 569–582, 2016.

[22] M. Straka and J. Strakov, "Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2. 0 with UDPipe," Proc. CoNLL 2017 Shar. Task Multiling. Parsing from Raw Text to Univers. Depend., vol. 2, pp. 88–99, 2017.

[23] M. De Laat and M. Daneva, "Empirical Validation of a Software Requirements Specification Checklist," CEUR Workshop Proc., vol. 2075, 2018.