Towards A Systematic Derivation Of BPMN Model From Business Process Textual Description

Wiem Khlif[1], Nourchène Elleuch Ben Ayed[2], Faten chihi[1]
[1]*Mir@cl Laboratory, University of Sfax, Sfax, Tunisia*
[2]Higher Colleges of Technology, ADW, U.A.E
wiem.khlif@gmail.com, nbenayed@hct.ac.ae, chihi.faten.95@gmail.com

***Abstract*—Deriving the Business Process (BP) model from its Textual Description (TD) is crucial to its consistent analysis, especially by making process information accessible to various stakeholders. However, establishing or maintaining the TD-BP alignment is not trivial when the enterprise develops a BP. In fact, there is a clear risk that model and text become misaligned when changes are not applied to both descriptions consistently. This paper proposes a new transformation methodology that helps business analyst to build BP model, which is aligned to its textual description. It is based on the use of the business concept's template that is enriched by linguistic-based business rules. Compared to existing methods, our methodology provides more comprehensive alignment, which encompass all BPMN elements. We examined the performance of the transformations through the calculation of recall and precision rates.**

Keywords- Textual Description, BPMN Model, Deriving, Transformation, Business Concept's Template

I. INTRODUCTION

Business processes capture organizational operations and involve numerous actors with various roles [1] [2]. To provide them with the information that they need, organizations have recognized the value of capturing process descriptions in model-based as well as text-based representations [1]. In this context, several methods are proposed to automate the transformation of a given representation into other one: Form model to text and from text to model.

Regarding the model-to-text transformation, [3], the authors have generated textual descriptions of a set of process models using manual and automatic approaches. In Leopold et al. [4], the authors proposed an approach that transforms textual as well as model-based process descriptions into a unified data format to automatically detect inconsistencies between them. [5] define a semi-automated approach that consists of a process model-based procedure for capturing execution-related data in requirements models and an algorithm that takes these models as input for generating natural language requirements.

Apropos of text-to-model transformation, [6] presented an approach to generate BPMN models from natural language text where they faced the complexity of natural language. In [1], the authors presented the first automated approach for the extraction of declarative process models from natural language.

In this paper, we focus on the works related to the generation of BPMN model from its textual description. Nevertheless, the existent works do not cover all BPMN elements. In addition, few works derive automatically the BPMN model. To overcome this gap, we propose a methodology called MONET (a syste**M**atic derivati**O**n of a bpm**N** mod**E**l from business process Textual description) that allows generating a BPMN model from the textual description of a business process. To achieve this, the BP description is split into business concepts that achieve a specific business goal. Then, each business concept is specified by using an enriched template [7] that encapsulates the semantic information pertinent to the business logic. Since there are many templates' format, we use the Task and Task & Support Descriptions [8] for the requirements specification to document the business concept. This template is enriched by business rules based on linguistic patterns to support the derivation of all BPMN elements. To evaluate our methodology, we examined the performance of the transformations experimentally through the calculation of recall and precision rates.

The remainder of the paper is organized as follows: Section II introduces the proposed methodology MONET and discusses the transformation definition strategy. Section III describes the BPMN model derivation phase that covers the pre-processing step and the transformation rules, which allow the generation of a BPMN model from the business concept template. Section IV evaluates the quality of the generated BPMN model by considering the recall and precision rates. Section V illustrates our tool MONET that implements the transformation rules and the ontology to generate the BPMN model. Section VI enumerates the threads to validity of our methodology. Section VII discusses the related work and identifies the research gap interest. Finally, Section VIII summarizes the research results and draws the future works.

II. OVERVIEW OF MONET

MONET (a syste**M**atic derivati**O**n of a bpm**N** mod**E**l from business process Textual description) is a methodology that derives the BPMN model from a given textual description. Its novelty resides in the production of a BPMN model that is aligned to the input business concepts. More specifically, we propose the business concept as a mean to define the textual description of the business process. Each business concept is enhanced by business rules that transform each linguistic patterns to its corresponding BPMN elements. Fig. 1 depicts our methodology for deriving the BPMN model from a textual description. MONET followed two major phases: BPMN model derivation phase and BPMN model evaluation phase.

The activities of the BPMN model derivation phase are organized essentially in three steps: *A pre-processing step*

DOI reference number: 10.18293/SEKE2020-043

during which the Business Analyst receives a textual description of a BPMN model in a natural language.

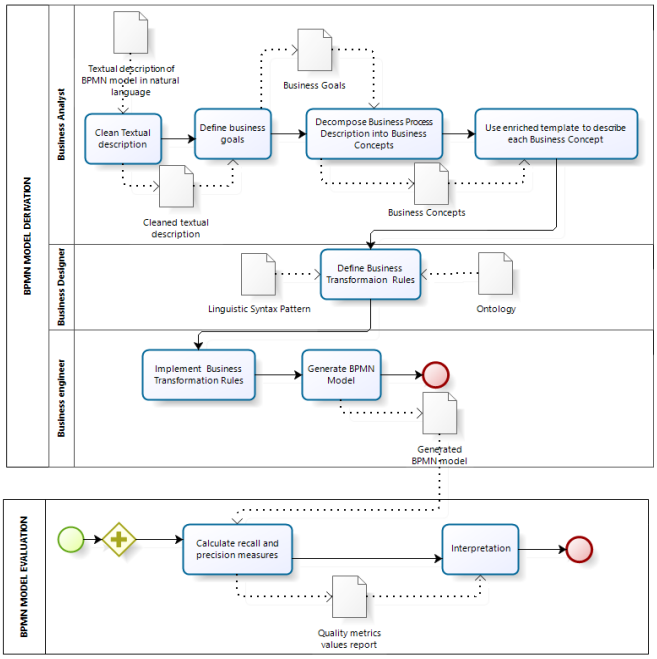

Figure 1. Conceptual process of MONET.

The description is cleaned based on a simple NLP technique (Stanford CoreNLP tool) [9]. Then, the Business Analyst uses the output to identify the business goals that are used to divide the business process description into different business concepts. For each business concept, the Business Analyst prepares its textual description according to a specific template. To handle this requirement, we rely on the use of the enriched template presented in [7] (See Section III). Based on this template, the Business Designer defines an ontology in the *transformation-definition step*. The ontology and the linguistic syntactic patterns are used to define the business transformation rules (See Section III). The Business Engineer formalizes/implements the transformation rules in the *transformation-implementation step* which provides for the automated generation of the BPMN model.

The evaluation phase (See Section IV) of our methodology is based on calculating the recall and precision rates in order to assess the performance of the transformations experimentally. Once the calculation is done, a quality report is generated, which is used by the quality interpretation activity.

III. BPMN MODEL DERIVATION PHASE

A. Natural Langage Pré-processing

We use natural language processing concepts that are syntax parsing and semantic analysis. The syntax parsing consists on obtaining a structured representation of the business knowledge. Therefore, the business analyst has first to clean the textual description by using the Stanford CoreNLP tool [9], and second to organize it according to a specific template's structure. Stanford CoreNLP tool is used to obtain a more manageable and readable text. The tool relies on the following methods:

- *Tokenization* is the task of breaking a character sequence up into pieces (words/phrases) called tokens, and perhaps at the same time throw away certain characters such as punctuation marks [10].
- *Filtering* aims to remove some stop words from the text. Words, which have no significant relevance and can be removed from the documents [11].
- *Lemmatization* considers the morphological analysis of the words
- *Stemming* aims to obtain the root of derived words [12].
- *Part of Speech Tagging* tags for each word (whether the word is a noun, verb, adjective, etc), then finds the most likely parse tree for a piece of text.

The cleaned file is then used to identify the business goals of the business process. By business goal, we mean a collection of business activities that are related to describe a functional process of the BPMN model. Each goal will correspond to a business concept.

To guide and improve the generation of a BPMN model, the business analyst associates to each business concept a template that is described by a set of linguistic patterns. The template covers the semantic and organizational information related to the business logic. It is composed of three blocks (Fig. 2).

Overview
ID of the business concept: *<Unique ID assigned to a business concept>*
Name of the business concept: *<Unique name assigned to a business concept>*
Purpose of the business concept: *<Summary of the business concept purpose>*
Trigger/Precondition for execution: *<List of conditions that must be true to initialize the business concept>*
Participants: *<Actors and/or units involved in the business concept execution>*
Relationships between BC and its successors:<Sequence flow or control structure>

Main Scenario
Begin at
<Steps of the scenario of the trigger to goal>
[<Pre-condition>] <$Task_i$> < Task Description > <Task Type>
….
[<Pre-condition>] <$Task_{i+k}$> < Task Description > <Task Type>
Relationships between each task and its direct successors:<Sequence flow or control structure>
End

Alternative Scenario
Begin at < $Task_i$ >
[<Pre-condition>] < $Task_j$> < Task Description > <Task Type >
….
[<Pre-condition>] <$Task_{j+l}$> < Task Description > <Task Type>
Relationships between each task and its direct successors:<Sequence flow or control structure>
End

Error Scenario
Begin at < $Task_i$>
The main scenario fails
End

Post-condition *:*<A list of business objects as result of BC execution>

Figure 2. Detailed description of a business concept.

The first block gives an executive summary of the business concept in terms of its id, name, purpose, pre-conditions, participants involved in its execution, and its relationships with business concept's successors. We defined a specific structure for the triggers, which is **[<Pre-condition>] <Event Description> [< Event Type:{timer | Message | Signal | Conditional}>]**. The event type can be explicitly specified or implicitly extracted from its description. In addition, to formalize the relations among participants, we created a WordNet, which is a lexical database for all business words. It defines a set of synonyms of a participant called Synsets and records the relations among them such as hypernym (Type of), meronym (part of), and antonym (opposite word). The relationships that a business concept has with its successors follows the linguistic pattern: **[<Pre-condition>] <Current Business Concept ID> is related <sequentially | exclusively |**

parallel | inclusively>to<Business Concept ID>, where the **<Precondition>** construct respects this structure **<if> condition <then>**.

The second block describes the main, alternative, and error scenarios. These scenarios respect this pattern **[<Pre-condition>] <Task#><Task descriptions> <Task Type >:**

- *Task Description*: We defined a linguistic syntax pattern to describe the tasks: **ActionVerb | CommunicationVerb + BusinessObject | NominalGroup + [[to ReceiverName] | [from SenderName]]** to label the tasks. We mean by *BusinessObject* any entity that describes the business logic. The *NominalGroup* is a set of pre/post-modifiers, which are centered on a Headword that constitutes the *BusinessObject*. The pre-modifiers (respectively post-modifiers) can be a noun, an adjective, or an ed/ing-participle (respectively, a noun, an adjective, or adverb). The *VerbalGroup* indicates the relationship type between *BusinessObjects*. We note that the expression between brackets is optional.

- *Task Type:* The task type can be "ActiveREQ", "ActiveREP", "ActiveRET", "ActivePER" or "Passive" representing respectively "Entry", "eXit", "Read", "Write" or "data manipulation". "ActiveREQ" corresponds to a task representing the act of asking for something. "ActiveREP" corresponds to a reply sent after asking for something.

  "ActiveRET" corresponds to a task allowing data retrieval. "ActivePER" corresponds to a task allowing the data record. "Passive" task does not lead to an exchange of data.

The third block illustrates the list of business objects as result of the execution of the business concept.

For the semantic analysis of the business concepts' template, we propose to create an ontology (See Fig. 3).

For the semantic analysis of the business concepts' template, we propose to create an ontology. It is designed to describe the entities related to the BPMN metamodel. The annotation process is based on the result of the preprocessing task and the defined template. It takes business concept templates of the business process model and define the similarities (the links) between concepts. We use the concept names to produce an expanded list of equivalent or related terms. Each term of the input textual description may be associated with one or more entities from the ontology. To find the similarities, we used the following matching techniques:

- Exact matching identifies the identical entities (String) in the text and in the ontology;
- Morphological matching identifies the entities with a morphological correspondence;
- Syntactical similarities using Levenshtein measure [13];
- Semantic matching identifies the synonyms relations with WordNet ontology.

- Semantic matching identifies the synonyms relations with WordNet ontology.

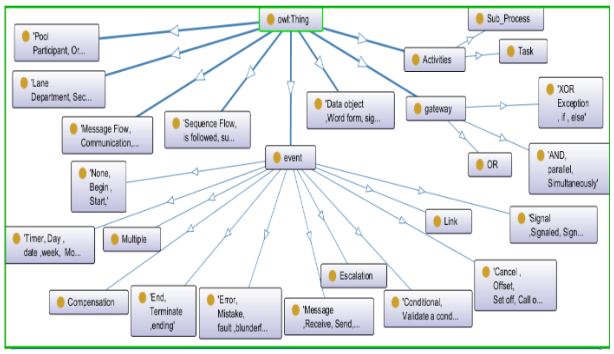

Figure 3. Meta-model ontology.

B. From Textual Description to BPMN Model

We defined eighteen transformation rules. Each transformation rule operates on the different components of the template.

R1. Each trigger is transformed into an event that will be linked to the first element of the current business concept. Based on the trigger type, we add the corresponding event.

R1.1: If the trigger type describes the time, so add a Timer Event.

R1.2: If the trigger type describes a certain condition that must be satisfied to start a process, so add a Conditional Event.

R1.3: If the trigger type describes any action that refers to a specific addressee and represents or contains information for the addressee, so add a Message Event.

R1.4: If the trigger type describes any action that refers to anyone and represents or contains information for anyone, so add a Signal Event.

R2. Each participant is transformed into pool or lane depending on its type.

R2.1: If all participants are business workers, then add a pool that has the same name of the business worker. We note that a business worker represents an abstraction of a human that acts within the business to realize a service.

R2.2: If one of the participants is a metonymy of "department", "unit", "division" or "organizational unit", then add a pool that has the same name of the participant and transform others participants to lanes. Based on the ontology result, if the relation between the participant which is represented by a pool and the other one that is represented by a lane is metonym (part of), so add a lane.

R3. Each relationship between the business concept and its successors respects the linguistic pattern: **[<Pre-condition>] <Current Business Concept ID> is related <sequentially | exclusively | parallel | inclusively>to<Business Concept ID>**.

R3.1: If the relationship is <sequentially>, then add a sequence flow if the last element of the current business concept and the first element of its successor are in the same pool. Otherwise, add a message flow.

R3.2: If the relationship is <parallel>, then add a parallel gateway between the last element of the current business concept and the first element of its successor.

R3.3: If the relationship is <exclusively> and there is a precondition, then add an exclusive gateway between the last element of the current business concept and the first element of its successor. The precondition expression is associated with the gateway outgoing sequence flow.

R3.4: If the relationship is <inclusively> and there is a precondition, then add an inclusively gateway between the last element of the current business concept and the first element of its successor. The precondition expression is associated with the gateway outgoing sequence flow.

R4. For each step of a BC's scenario respecting the linguistic pattern: **[<Pre-condition>] <Task#> < Task Description > <Task Type >**, then add the following:

R4.1: If the task description is « Action verb + BusinessObject », then add a service task that has the same name of the pattern and a data object.

R4.2: If the task description is « Action verb + NominalGroup », then add a service task that has the same name of the pattern. If the pre/post-modifier is a noun that merely represents a pure value, so there is no data object to add. Otherwise, if the pre/post-modifier is a complex noun (an entity) then add a data object.

R4.3: If the task description is « CommunicationVerb+ BusinessObject|NominalGroup + [[to ReceiverName(s)] | [from SenderName]] », then add send or receive task that has the same name of the pattern and a data object. Then, call R4.4, R4.5, and/or R4.6.

R4.4: If the task type is ActivePER, then add an outgoing object flow between the task and its data object/store.

R4.5: If the task type is ActiveRET, then add an ingoing object flow between the task and its data object/store.

R4.6: If the task type is ActiveREP, then add a message event and an outgoing message flow between the task and message event. The message event name is the concatenation of the Business Object extracted from the task and the past participle of Receive.

R5. Each relationship between the task and its successors respects the linguistic pattern: **[<Pre-condition>] <Current Task ID> is related <sequentially | exclusively | parallel | inclusively>to<Task ID>**.

R5.1: If the relationship is <sequentially>, then add a sequence flow if the current task and its direct successor are in the same pool. Otherwise, add a message flow.

R5.2: If the relationship is <parallel>, then add a parallel gateway between the current task and its direct successor.

R5.3: If the relationship is <exclusively> and there is a precondition, then add an exclusive gateway, if it does not exist, between the current task and its direct successor. The precondition expression is associated with the gateway outgoing sequence flow.

R5.4: If the relationship is <inclusively> and there is a precondition, then add an inclusively gateway, if it does not exist, between the current task and its direct successor. The precondition is associated with the gateway outgoing sequence flow.

R5.5: If the relationship is <sequentially>, and there is a <complete> construct related to a task, then add an end event.

IV. BPMN MODEL EVALUATION PHASE

The evaluation of our methodology is based on the experimental comparison activity that calculates for each element type of the BPMN model, the recall and precision rates according the following equations:

$$\text{Precision} = TP/(TP+FP) \quad (1)$$

$$\text{Recall} = TP/(TP+FN) \quad (2)$$

Where:

- True positive (TP) is the number of existing real elements generated by our transformation;
- False Positive (FP) is the number of not existing real elements generated by our transformation;
- False Negative (FN) is the number of existing real elements not generated by our transformation.

V. MONET TOOL

To facilitate the application of our methodology, we developed a tool for deriving the BPMN model from a given textual description, named MONET Tool. Our tool is implemented as an EclipseTM plug-in [14]. It is composed of three main modules : Parser, generator, and evaluator.

The pre-processing engine uses as input the textual description of a BPMN model written in a natural language. It cleaned the file using the Stanford CoreNLP tool. The cleaned file is used by the business analyst to define manually business goals. Then, the latter associates each business goal to its corresponding BC. The business analyst creates the enriched template corresponding to each BC. Fig. 4 shows the BC4's template.

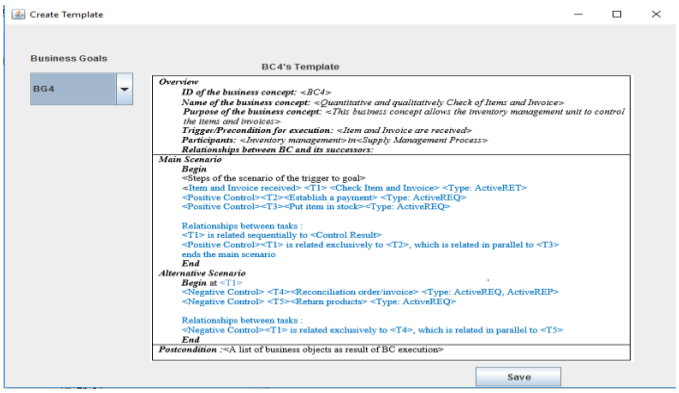

Figure 4. BC4's Enhanced template.

Next, the analyst selects one or more BCs. If he selects one BC, the corresponding fragment is generated. Else, the business analyst can select all business concepts to transform.

The generator engine uses the ontology and applies the transformation rules to derive the BPMN model. Fig. 5 illustrates the generated BPMN model: "Supply Management Process".

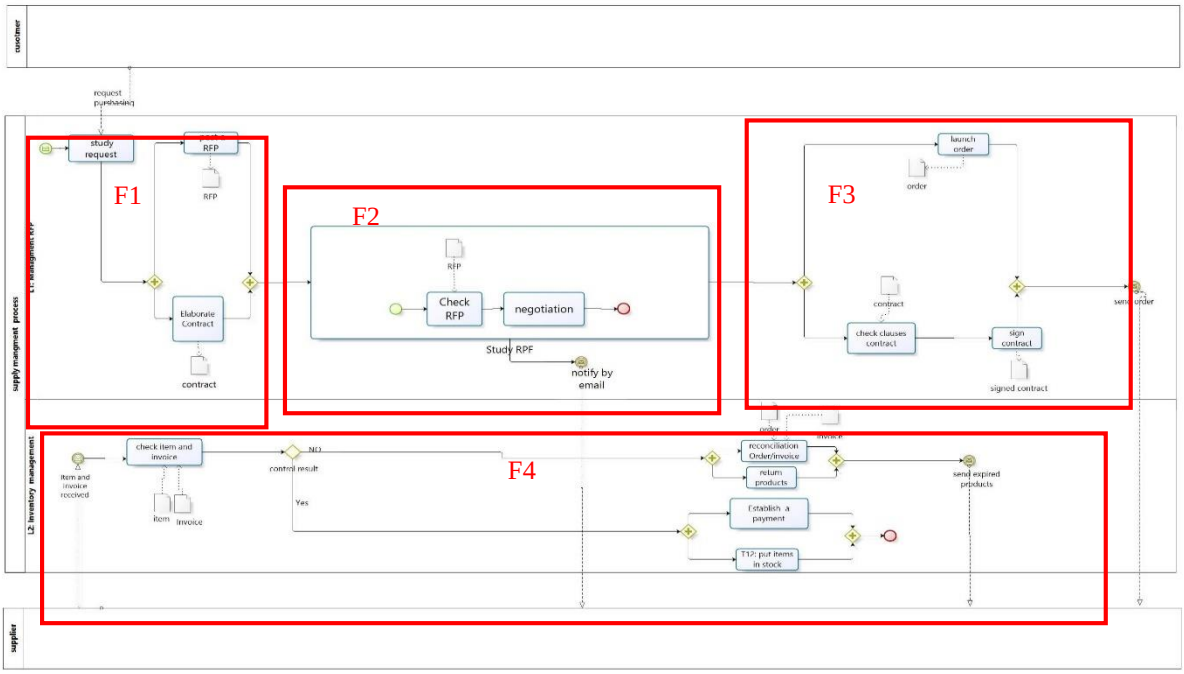

Figure 5. The generated BPMN model: "Supply Management Process".

The obtained model is generated as follows: First, by applying **R2.2**, we add a lane "PurchaseDepartmentSystem" inside the pool "Supply Management Process". Second, by applying **R1**, we add the message event "Item and Invoice are received" in the corresponding lane.

The transformation of the main scenario calls **R4.2** and **R4.5** that generate a task labelled *"Check Item and Invoice"*, two data objects labelled invoice and item, and add an ingoing object flow between the task and its data objects. Then, **R4.2** produces a task labelled Establish a payment (respectively, Put item in stock).

By applying **R5,** we linked the business task "*check invoice and item*" to the exclusive gateway labelled "*control result*". Then, we applied **R3.3** and added the precondition related to the default outgoing flow expressing the main scenario. By applying **R5**, a parallel gateway is created between establish payment and put item in stock.

Then, by selecting the "Check alignment" button, the generator displays each element in all the business concepts and their corresponding BPMN elements (See Fig. 6).

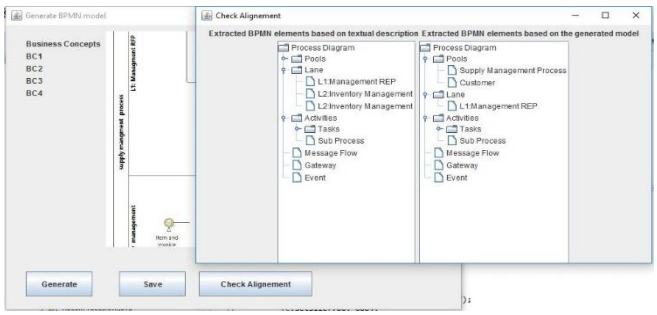

Figure 6. The generated BPMN model: "Supply Management Process".

If each element has its correspondence in the BPMN model, then we can deduce that the textual description is aligned to its model.

The BPMN quality evaluator evaluates experimentally the BPMN model through the calculation of recall and precision rates.

Precision==0.86
Recall =0.95

The high scores for both ratios mean that the generated BPMN model covers the whole domain precisely in accordance with the experts' perspective (See Fig. 7). We can deduce that the performance of our methodology approaches the human performance.

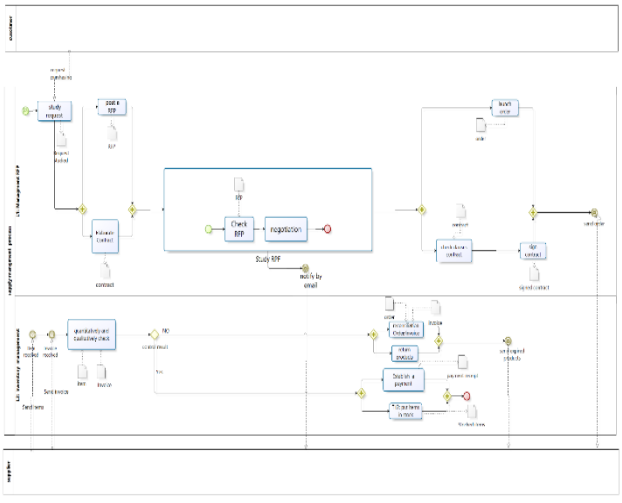

Figure 7. The elaborated BPMN model by the expert

VI. THREATS TO VALIDITY

In our study, threats to validity are relevant to internal validity and external validity [15].

The internal validity threats are related to four issues: The first threat to validity focus on who write the textual description expressing the functional requirements and which template have been used to describe the functional requirements. Expert should well write these requirements based on a particular style to generate a high quality of a BPMN model. The Second problem is addressed when there is a diversity of description of the requirements. In this case, which one can be used to describe the functional requirements? The third issue is related to the impact of an error-prone generation of a BPMN model. This case may lead to misalignment and inconsistency between the textual description and business process model.

The external validity threats deal with the possibility to generalize this study results to other case studies. The limited number of case studies used to illustrate the proposed methodology could not generalize the results. Automation of our methodology needs to be considered even it is easy to use manually given its simplicity.

VII. RELATED WORK

Several methods explicitly emphasis on the generation of process models from different types of text documents. The authors of [6] presented an automatic approach to generate BPMN models from natural language text, where they faced the complexity of natural language.

In [1], the authors present an automated approach for the extraction of declarative process models from natural language. They developed a tailored Natural Language Processing (NLP) techniques that identify activities and their inter-relations from textual constraint descriptions. By considering the semantics of these extracted components, the authors generate declarative constraints aiming to capture the logic defined in the textual description.

In [16] the author uses natural language processing with a focus on the verb semantics, and creates a novel unsupervised technique *TextProcessMiner* that discovers process instance.

In summary, many researchers studied the alignment between BPMN model and textual description. However, they don't cover all BPMN elements.

VIII. CONCLUSION

This paper proposed a transformation-based approach to generate a business process model from its textual description. It provides for the generation of a BPMN model that is aligned to the input business concepts. Compared to existing works, our methodology has the merit of accounting for all BPMN elements and their relationships. To do so, our methodology used the enriched template as the starting point for deriving BPMN model. Then, it defines transformation rules that transform each linguistic patterns to its corresponding BPMN elements. The methodology has been implemented. An evaluation of a business process model shows that our methodology approaches the expert performance and generates BPMN models respecting the quality measurements. Although the current results are very promising, our technique still requires further empirical tests.

We intend to generalize the methodology in order to derive BPEL from the textual description as well as the information system's design models from the textual description and check the alignment between all generated models: BPMN model and information system design models.

REFERENCES

[1] H. Van der Aa, C.D.Ciccio, H. Leopold, H.A. Reijers, "Extracting Declarative Process Models from Natural Language", 31st Conf. Advaneced Information Systems Engineering, Italy pp. 365-338, 2019.

[2] M. Dumas, M. La Rosa, J. Mendling, H.A. Reijers, "Fundamentals of Business Process Management". Springer, ISBN, pp. 1-527, 2018.

[3] S. Zaheer, K. Shahzad, R. M. A Nawab, "Comparing Manual- and Auto-Generated Textual Descriptions of Business Process Models", 6th Conf. on Innovative Computing Technology, Ireland, August. 2016.

[4] Leopold, H., H. van der Aa, F. Pittke, M. Raffel, J. Mendling, H.A. Reijers, "Searching textual and model-based process descriptions based on a unified data format", International Journal of Software and system Modeling. 18, No.2, pp. 1179-1194, 2019.

[5] B. Aysolmaz, , H. Leopold, H.A. Reijers, O. Demirörs, "A semi-automated approach for generating natural language requirements documents based on business process models", International Journal of Information & Software Technology, Vol 93, pp. 14-29, 2018.

[6] F. Friedrich, J. Mendling, F. Puhlmann, F., "Process model generation from Natural Language Text", 23th International Conference on Advanced Information Systems Engineering, LNCS in Computer Science book series, Vol. 6741, London, June, pp. 482–496, 2011.

[7] W. Khlif, A. Sallemi, M. Haoues, H. Ben-Abdallah, "Using COSMIC FSM Method to Analyze the Impact of Functional Changes in Business Process Models", 13th International Conference on Evaluation of Novel approaches to software engineering, Portugal, March, 2018.

[8] S. Lauesen, "Software Requirements: Styles and Techniques", Addison-Wesley, London, 2002.

[9] CD. Manning, M. Surdeanu,, *J.* Bauer, *J.*, Jenny Rose J.R. Finkel, *S.*Bethard, *D.* McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit", The 52nd Annual Meeting of the Association for Computational Linguistics, June 22-27, pp.*55-60. 2014.*

[10] J. Webster, C. Kit, "Tokenization as the initial phase in nlp", 14th conference on Computational linguistics, Association for Computational Linguistics, Vol.4, pp. 1106-1110, 1992.

[11] Saif, H., Fernandez, M., He, Y., Alani, H. , "On stopwords, filtering and data sparsity for sentiment analysis of twitter", the 9 th Inter. Confe. on Language Resources and Evaluation, European Language Resources Association, Iceland, May 26-31, pp. 810-817, 2014.

[12] J.B.Lovins, "Development of a stemming algorithm", Mechanical Translation and Computational Linguistics, Vol 11, No.1-2, june, pp. 22-31, 1968.

[13] V.I.Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals", *journal of Soviet physics doklady*, Vol. 10, pp. 707-710, 1966.

[14] Eclipse Specification. (2011), Available from: http://www.eclipse.org/

[15] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, "Experimentation in Software Engineering: An Introduction", 2000.

[16] E.V. Epure, P. Martín-Rodilla, C. Hug, R. Deneckère, C. Salinesi, 2015. "Automatic process model discovery from textual methodologies", 9th IEEE International Conference on Research Challenges in Information Science, Greece, May 13-15, 2015.