From Text to Visual BPMN Process Models: Design and Evaluation

Ana Ivanchikj
Software Institute, USI
Lugano, Switzerland
ana.ivanchikj@usi.ch

Souhaila Serbout
Software Institute, USI
Lugano, Switzerland
souhaila.serbout@usi.ch

Cesare Pautasso
Software Institute, USI
Lugano, Switzerland
c.pautasso@ieee.org

ABSTRACT

Most existing Business Process Model and Notation (BPMN) editing tools are graphical, and as such based on explicit modeling, requiring good knowledge of the notation and its semantics, as well as the ability to analyze and abstract business requirements and capture them by correctly using the notation. As a consequence, their use can be cumbersome for live modeling during interviews and design workshops, where participants should not only provide input but also give feedback on how it has been represented in a model. To overcome this, in this paper we present the design and evaluation of BPMN Sketch Miner, a tool which combines notes taking in constrained natural language with process mining to automatically produce BPMN diagrams in real-time as interview participants describe them with stories. In this work we discuss the design decisions regarding the trade-off between using mining vs. modelling in order to: 1) support a larger number of BPMN constructs in the textual language; 2) target both BPMN beginners and business analysts, in addition to the process participants themselves. The evaluation of the new version of the tool in terms of how it balances the expressiveness and learnability of its DSL with the usability of the text-to-visual sketching environment shows encouraging results. Namely, while BPMN beginners could model a non-trivial process with the tool in a relatively short time and with good accuracy, business analysts appreciated the usability of the tool and the expressiveness of the language in terms of supported BPMN constructs.

CCS CONCEPTS

• **Software and its engineering → Domain specific languages**.

KEYWORDS

Textual Domain Specific Language, Prototype Design, BPMN, Modeling with Mining, Usability Evaluation, User Study

ACM Reference Format:
Ana Ivanchikj, Souhaila Serbout, and Cesare Pautasso. 2020. From Text to Visual BPMN Process Models: Design and Evaluation. In *ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems (MODELS '20), October 18–23, 2020, Montreal, QC, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3365438.3410990

MODELS '20, October 18–23, 2020, Montreal, QC, Canada

ACM ISBN 978-1-4503-7019-6/20/10…$15.00
https://doi.org/10.1145/3365438.3410990

1 INTRODUCTION

The Business Process Model and Notation (BPMN) standard [25] facilitates the communication and knowledge sharing between domain experts, process participants and business analysts due to the standardised language [40] and its graphical visual notation [32] for modelling business processes [38]. Most BPMN graphical editors support the design of business process models by dragging, dropping and connecting visual elements. Following existing text-based modeling tools (e.g., PlantBPMN [13], PlantUML[1] and its PlantText[2] environment, ZenUML[3], WebsequenceDiagrams[4], Textografo[5]), which generate visual models from textual descriptions, in this paper we present the design and evaluation of the BPMN Sketch Miner[6], a proof-of-concept tool for rapidly generating BPMN models from descriptions specified with a textual Domain Specific Language (DSL).

The aim of the tool is to speed up the iteration cycles of requirements gathering leading to the initial sketch of a process model, with a language mimicking structured notes taken during participant interviews [34] and design workshops [7, 28]. To accomplish the goal the tool augments textual modelling with process mining, thus reducing the complexity and the number of keywords of the textual DSL. Its modelling environment replaces the complex stencil palette usually found in graphical editors with a simple text editor. Textual descriptions are transformed into diagrams, which correctly use the BPMN visual syntax, simultaneously while the modeler is typing them. The main design challenge consists of the trade-off between usability, learnability, and expressiveness of the textual DSL. While BPMN is a rich notation with hundreds of constructs, our DSL focuses on a subset of the notation [41] with the intention to support an iterative model refinement process, where the initial model is obtained quickly from its constrained natural language description. Basic type annotation keywords need to be learned only if it becomes necessary to classify model elements during a second refinement step. At this point, the model can also be exported so that its refinement can continue using traditional standard-compliant BPMN editors [11]. In other words, we see text-based modeling as complementary to graphical editors, and particularly useful to quickly get started with an initial sketch.

[1] https://plantuml.com/
[2] https://www.planttext.com/
[3] https://app.zenuml.com/
[4] https://www.websequencediagrams.com/
[5] https://textografo.com/
[6] https://www.bpmn-sketch-miner.ai

The evaluation of the tool has been conducted with a user study and surveys [24] involving students having no prior knowledge of the BPMN language, as well as industry analysts, with advanced BPMN knowledge, who typically enjoy working with 'quick modeling' shortcuts as they need to, as mentioned by one of them, "represent the content of their processes without having to fiddle with the graphical layout". The evaluation focused on two research questions. The research question we would like to answer with the user study is: "RQ1: Does BPMN Sketch Miner help BPMN non-experts to produce accurate BPMN models in a relatively short time?". To answer "RQ2: What is the perceived usability of the tool and its features?", in the survey with both students and industry participants, we included, among other questions, also the standard System Usability Scale (SUS) survey.

The main contributions of this paper are: 1) the design of a tool which needs to balance the trade-off between the use of textual modelling and process mining for quickly creating valid visual BPMN models, and 2) evaluating the usability and learnability of the tool with BPMN novices and business analysts.

The rest of this paper is organized as follows: In Sec. 2, we describe the design of the BPMN Sketch Miner and its textual DSL while in Sec. 3, we discuss the evaluation methodology and the results from the conducted surveys and the user study. In Sec. 4 we discuss the results of the evaluation as well as the lessons learned during the design and evaluation of the tool, providing an overview of the related work in Sec. 5, while drawing conclusions in Sec. 6.

2 BPMN SKETCH MINER DESIGN

The design of BPMN Sketch Miner (Fig. 1) is driven by the attempt to trade off the expressiveness against the learnability of the textual DSL, without sacrificing the efficiency with which the tool can be used to produce BPMN diagrams.

2.1 Live Modeling Environment

We have identified two different contexts where the BPMN Sketch Miner can be used: 1) during co-located or remote requirements elicitation meetings between domain experts and business analysts, and 2) in classrooms while teaching BPMN. In the first context, there can be two usage scenarios: the more common one, where the business analysts interview the domain experts while creating the BPMN model, or the one where domain experts are empowered to participate in the creation of the BPMN model themselves. That said, the goal of BPMN Sketch Miner is to streamline the rapid model creation for obtaining feedback from domain experts and process participants, and to facilitate learning BPMN and process modelling. Studies have shown that starting with textual input (e.g., activity tables, written use-case scenarios) and then abstracting the information using visual models improves process understanding for people who are not process modelling experts [5, 10, 33]. Thus, given the goal of our tool, and the contexts in which its use is envisioned, our first design decision was to use textual input for the creation of visual models to be represented in BPMN. The second design decision was to design a live modeling environment, where the BPMN model is produced in real-time as the textual description is typed. The third design decision was to deliver it as a Web-based tool, thus avoiding the need for software installation to get started with a modeling session.

The main flow of interaction with the tool involves the following steps: (1) Open a Web link to go directly into the modeling environment (no user registration or authentication is required). (2) To edit an existing model, use the provided link which embeds its textual description (it can be easily shared in an email message or chat room). (3) As the textual description is edited, the BPMN diagram is immediately updated (no need to click a submit or refresh button) (4) At the end of the session, export the generated BPMN model in SVG, PNG, or XML formats so that it can be displayed or further refined in any compatible tool.

2.2 BPMN as a Textual Domain-Specific Language

The main feature of the BPMN Sketch Miner is its textual DSL, by means of which the user can enter a textual description of a process. The purpose of the textual language is to provide a simple notation for easy and rapid [20] representation of a process by enumerating execution traces of its instances (e.g., while taking notes describing concrete examples during an interview) from which valid BPMN sketches (i.e., non-executable models [3]) can be obtained.

The main design constraint for the DSL is that it should reflect the largest possible subset of BPMN (a rather large and complex visual notation [41]) while using a limited number of textual constructs, which should be easy to learn and remember. Unlike the graphical syntax, the textual one is characterized by its monodimensional structure [19]. This constraint makes it an adequate choice for representing sequential business processes but makes it challenging to use plain text to represent control flow graphs of arbitrary structures like the ones which can be visualized in BPMN. Like first proposed in [23], we address this challenge by using process mining [36] to reconstruct a model of the process control flow graph from a set of sequential execution traces, which can be easily written in plain text. This also makes it possible to target domain experts being interviewed during requirements elicitation who have no or limited BPMN knowledge. Following Karsai et al. [26]'s advice, the textual DSL used in the tool represents ordered lists of tasks and events that are envisioned to be written as the process participants enumerate their activities, or as BPMN students read a given process description they are supposed to model. These lists are then used as input traces by a process mining algorithm.

2.2.1 Design Decisions. Based on continuous formative evaluation with test users, including BPMN consultants, experts, and trainers, we have extended the textual DSL, which now supports the BPMN constructs shown in Tab. 1. The most important design decisions involve:

- annotating the name of the role followed by ":", to speed up annotating tasks or events with their roles. The role is applied to all tasks following the annotation until another role is declared. The roles are automatically mapped to swimlanes or pools depending on the presence or absence of handovers and message exchange between them.
- events are distinguished from tasks because they are entered in round parenthesis referring to their round visual shape. The mining algorithm determines automatically whether they are *start*,

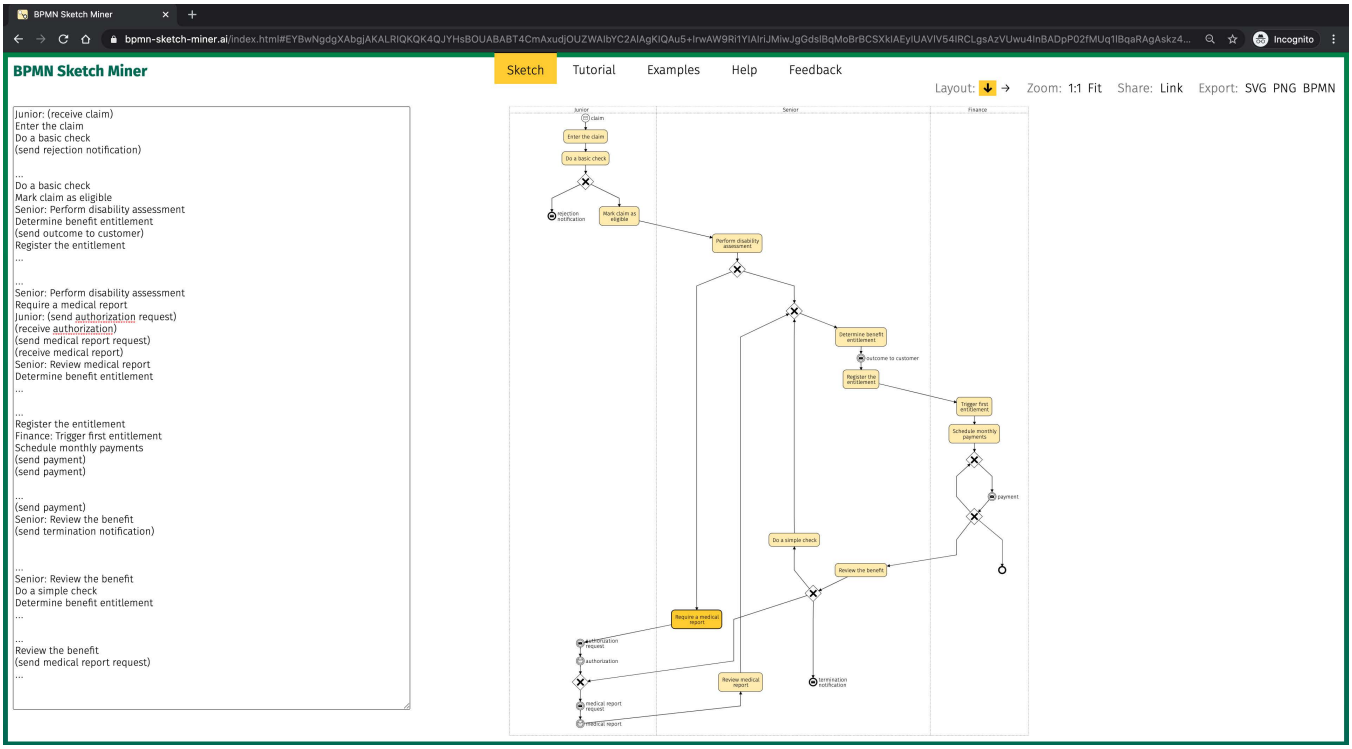


Figure 1: BPMN Sketch Miner with the textual and visual BPMN representation of the process modeled during the user study

Table 1: BPMN Constructs supported in different versions of the BPMN Sketch Miner

BPMN Construct	Derived from mining	Explicitly modeled	DSL construct
tasks		✓	state each task name in a new line
task type		✓	use the "manual", "user", "script", "rule", "service", "send", "receive" keywords
events	✓	✓	use "( )" to indicate an event. Whether it is a start, intermediate or end event is automatically inferred.
event type		✓	use the "time", "error", "receive", "send", "notify", "publish", "escalate", "terminate" keywords
link events	✓		derived from the lack of a matching fragment
exclusive gateway split/merge	✓		repeat the last common task before the split
labeled exclusive gateway split		✓	use "?" after the name of the label
loops	✓		repeat the tasks in the loop
sequence flow	✓		state tasks in the order in which they happen
conditional flow	✓		state the condition following the gateway label
message flow	✓		use matching names for the throw and catch events
event-based gateway	✓		the last common task before the split has to be followed by events
parallel AND gateway		✓	separate the parallel tasks with "|"
lanes		✓	state the name of the lane followed by ":" and a task
pools	✓		lanes become pools in case of message exchange
text annotation		✓	use "//" before the task to which the text annotation is to be attached

intermediate, or *end* events, depending on whether there are stated tasks preceding/following the event in question.

- tasks and events can be annotated by prefixing them with a type. This addresses the requirement of business analysts who need to refine their model after the initial sketch. Stating event types and task types requires the use of keywords, which attempt to match how those constructs are named in BPMN.
- exclusive split gateways can be annotated with a label, specified as a question (i.e., a line ending with ?) in the text. As exclusive gateways denote decisions, the line after the question represents the chosen alternative and becomes the expression associated with the outgoing conditional flow of the gateway.
- we intentionally decided not to mine parallel gateways as it would require manually entering multiple traces including different

permutations of the same set of tasks, which early adopters considered too much effort. Instead, parallel tasks are simply declared on the same line by separating them with "|".

The design of the BPMN Sketch Miner's textual DSL attempts to draw the line between the usefulness of mining algorithms to infer BPMN constructs implicitly embedded into the textual description vs. explicitly stating a construct, as evident in Tab. 1. Our goal is, whenever possible, to reduce the cognitive effort of the users [40] by not requiring them to explicitly state BPMN constructs. For example, as event-based gateways can be deduced easily by analyzing traces in which the same task is followed by different events, we do not require the user to explicitly include such gateways in the textual description. The same applies to message flows which can be inferred by detecting a matching message name for a pair of send and receive tasks or message events. Likewise, lanes are automatically clustered into pools based on the presence of message flow or sequence flow handovers.

2.2.2 Language Syntax and Control Flow Pattern Examples. The textual DSL concrete syntax is meant to be closer to natural language than to common textual programming languages, which are based on control structures, blocks, and the use of keywords. The textual DSL's context-free grammar is expressed in the following EBNF specification:

⟨*text*⟩ ::= (⟨*trace*⟩ 'EOL')$^+$ 'EOF';

⟨*trace*⟩ ::= ⟨*dots*⟩ (⟨*line*⟩ 'EOL' ⟨*dots*⟩)$^+$;

⟨*dots*⟩ ::= ['...' 'EOL'] ;

⟨*line*⟩ ::= ⟨*parallel*⟩ | ⟨*annotation*⟩ | ⟨*comment*⟩ ;

⟨*comment*⟩ ::= '///' (any character until EOL)* ;

⟨*annotation*⟩ ::= '//' ⟨*label*⟩ ;

⟨*parallel*⟩ ::= ⟨*element*⟩ ('|' ⟨*element*⟩)*;

⟨*element*⟩ ::= [⟨*role*⟩] (⟨*task*⟩ | ⟨*event*⟩ | ⟨*XOR label*⟩);

⟨*role*⟩ ::= ⟨*label*⟩ ':' ;

⟨*XOR label*⟩ ::= ⟨*label*⟩ '?' ;

⟨*task*⟩ ::= [⟨*ttype*⟩] ⟨*label*⟩;

⟨*event*⟩ ::= '(' [⟨*etype*⟩] ⟨*label*⟩ ')';

⟨*ttype*⟩ ::= ('send' | 'receive' | 'user' | 'manual' | 'service' | 'script' | 'rule') ;

⟨*etype*⟩ ::= ('start' | 'finish' | 'timer'| 'send' | 'receive' | 'publish' | 'notify' | 'error' | 'escalate' | 'terminate') ;

⟨*label*⟩ ::= any valid BPMN element label;

While BPMN support for workflow patterns is well-known [17, 27, 39], in Tab. 2 we show a few abstract usage examples to illustrate to which extent the BPMN Sketch Miner textual DSL supports a number of important control-flow patterns. It showcases how these frequently used patterns can be modelled in the tool.

2.3 Model Generation Pipeline

Using textual input to generate visual BPMN models requires multiple model generation and transformation steps (Fig. 2). The initial control flow model is generated using a process mining algorithm

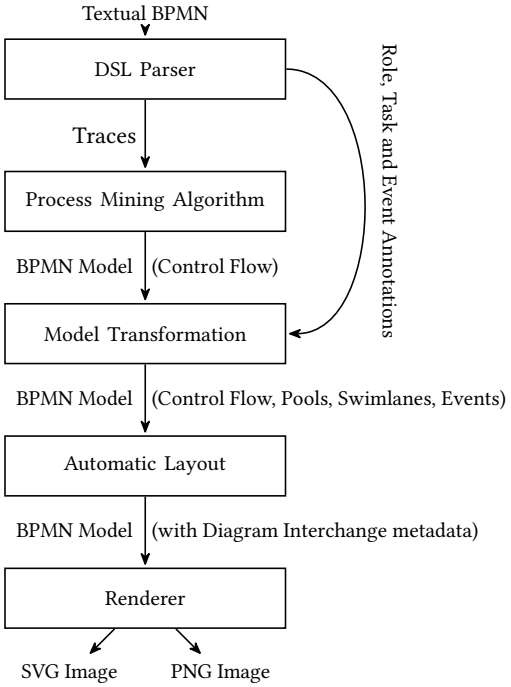


Figure 2: Model Generation and Transformation Pipeline

applied to traces that are extracted from the textual input. The textual annotation of roles, task and event types is part of the textual modelling and thus is not fed into the mining algorithm, but used during a second stage as follows: first, the nodes of the control flow graph are transformed into tasks or events (according to the type annotations found in the original description). Then, the role annotations are used to place the tasks and events in the corresponding swimlanes. While this would already be sufficient for obtaining a valid BPMN process model, in order to make the result visible, the model needs to be further augmented with additional diagram interchange metadata. This is produced by a hierarchical layout algorithm [14, 16], which has been tailored to consider idioms of the notation and can produce both vertical and horizontal layouts. The result can be exported as standard BPMN2.0/XML files, but also rendered as vector or bitmap images, using the dagre-d3 library[7] to display the result.

3 EVALUATION

3.1 Methods, Metrics and Setup

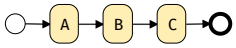
Industry Survey. We have posted a survey on the BPMN Sketch Miner's website in February 2020 and shared it in different Business Analyst social network groups and through our personal network of contacts. The survey consists of 4 parts, which although interconnected with links to direct respondents to the next part, can be answered independently and thus have a different number of answers. The first part (12 answers) refers to the evaluation of the tool and the DSL features with a 5 levels scale from "Useless" to "Very Useful". The second part allows respondents to vote for new

[7] https://github.com/dagrejs/dagre-d3

Table 2: Control Flow Patterns [37] supported by the BPMN Sketch Miner DSL

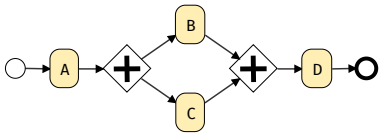
WCP-1: Sequence

```
A
B
C
```



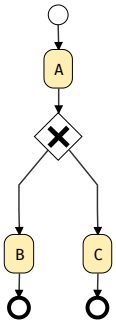
WCP-2,3 : Parallel split and synchronization

```
A
B|C
D
```



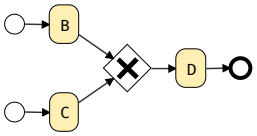
WCP-4: Exclusive choice

```
A
B

A
C
```



WCP-5,8: Simple and Multi- Merge

```
B
D

C
D
```

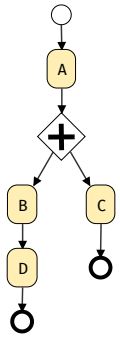


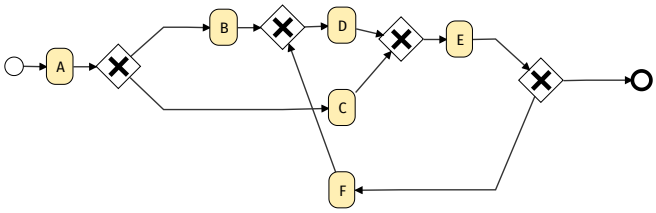
WCP-10: Arbitrary Cycles

```
A
B
D
E

A
C
E
...

...
E
F
D
...
```

WCP-11: Implicit termination

```
A
B|C
...

...
B
D

...
C
```



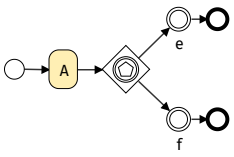


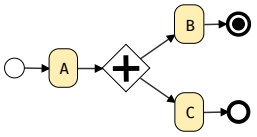
WCP-43: Explicit termination

```
A
B|C
...

...
B
(terminate)

...
C
```

WCP-16: Deferred Choice

```
A
(e)

A
(f)
```





features (16 votes). The third part (9 answers) is a lengthier survey intended to test user satisfaction. It is the only part that gathers background information on the respondent in terms of job position and process modelling experience. Seven of the respondents have an IT background, and the two other respondents have respectively a business background and a business and informatics background. 44% of them have more than 10 years of experience in working with business processes. This part of the survey tackles questions referring to the use of the tool, and different usability characteristics [22], such as satisfaction, effectiveness, efficiency, and learnability. The last part (12 respondents) consists of the traditional System Usability Scale (SUS [4]), composed of 10 statements which alternate between positive and negative and are scored on a 5-point Likert scale. The answers are combined into a final score (0-100). In addition to the survey, we have also logged events about the use of the tool (visits to tutorials and examples, time spent modeling, etc.), as well as metrics about the language constructs found within the exported models. The data collection period is 01.02.–24.05.2020.

User Study: Participants, Setup and Task. To evaluate the usability and learnability of the BPMN Sketch Miner with beginners we have conducted a study with 21 USI MSc students (70% with IT background). The students were not given any BPMN training beyond an introduction covering the abstract notions of business processes (e.g., "who does what when"), activities vs. events and control flow. On the day of the user study, they took a 45 minutes hands-on, which included a short warm-up tutorial where they used the tool to model a simple process with two process instances, one send message and one receive message, one loop and one control flow branch. After the warm-up, they were provided a natural language description of a non-trivial health insurance claim management process. The BPMN model and the corresponding textual model in BPMN Sketch Miner are provided in Fig. 1. The students were asked to model the process using the tool. We measured their usage of different language constructs, the accuracy of their models, as well as the time it took them to prepare the models. After creating the model, we asked the study participants to take a survey (a

subset of the industry survey described above) adding some additional questions targeting the specific group of users. As this survey is placed at a different link we were able to track the answers from the students separately from the answers from the industry.

3.2 DSL Expressiveness

In addition to analysing the workflow patterns support shown in Tab. 2, to evaluate whether the expressiveness of the DSL is sufficient, and how it can be improved, we surveyed our target users from industry. The students were not included in this part of the evaluation due to their limited knowledge of BPMN. Out of 9 industry respondents who answered the question "Do you feel you had sufficient BPMN constructs to model your business processes?", 56% answered that they were sufficient, 44% replied that some constructs were missing, but they were mostly sufficient, and no respondents missed a lot of crucial constructs. When asked which BPMN constructs they would like to see added in the next version of the BPMN Sketch Miner, out of 16 industry respondents, 67% voted for adding subprocesses, 47% for exclusive event-based gateways, 40% for the data store, data objects and cancel events, 33% for error events, 27% for conditional events and inclusive gateways, 20% for activity markers, and parallel event-based gateways, 13% for multiple events, parallel multiple events and call activities, and 7% for complex gateways. Nobody voted for the compensation events.

DSL Construct Usage. To depict the size and complexity of the process the students had to model, as well as to analyse the type of constructs that they used, in Fig. 3 (left) we plot the frequency of use of specific constructs (e.g., roles, tasks, events etc.) relative to the total number of used constructs and (right) the absolute usage numbers. In Fig. 4 we show the boxplot of the number of unique model elements and the number and length of the sequences used to describe the models. As a reference, we also include statistics about the same metrics applied to a collection of models exported by business analysts testing the tool. The purpose of these graphs is to show the size and complexity of the processes modelled with the BPMN Sketch Miner by the students (who all modelled the same process) vs the business analysts who modelled arbitrary models.

The most frequently used construct by the students is the task, with a median number of unique tasks in each model of 19.5 (see Fig. 4). These tasks were enumerated across 31 out of 56 lines of text (also median values shown in Fig. 3). Regarding roles, students used between 3 and 5 unique roles, which were listed a median of 10 times in their textual models (the language requires to state the role each time there is a hand-over of tasks between the roles). All, but one study participant, were able to make the distinction between tasks and events, with a median of 7.5 unique events per model. An unexpected observation is that, 48% of participants used questions to describe up to 3 unique XOR gateway labels, even though we did not explain this feature during the warm-up.

Regarding the usage of fragments, although everyone did use fragments (with a median of 8 dotted lines and 5 out of 7 fragment sequences per model as per Fig. 4), 20% of the participants used the "..." only at the beginning of a sequence, repeating redundant tasks at the end of the sequence. In the survey we conducted after the task was completed, 3 participants stated that they found the use of the "..." syntax somewhat complex. Nonetheless, 67% found it a useful feature of the textual DSL, with 14% finding it very useful.

3.3 Usability and Learnability Study Results

Model Understanding and Accuracy. The accuracy of the models was evaluated based on the completeness of the described process instances, as well as the participants' ability to identify the roles in the process, distinguish tasks from events, use appropriate task granularity etc. The evaluation was done systematically across the different solutions and discussed between the authors. The evaluation scale is available in the replication package together with the accuracy scores. The results have shown that students were able to produce an accurate BPMN representation of the model, with an average correctness grade of 87% (Figs. 5 and 6). All study participants finished the task in less than an hour and a half, with most of them taking between half an hour and an hour. We have plotted the types and the frequency of the constructs used by individual students in relation to the correctness of their model in Fig. 5. The plot shows similarity in the structure of the used constructs among the students with model accuracy of above 95%. These are students who have also understood well the use of the DSL and its functionalities of avoiding repetition by using fragments, and the correct assignment of roles.

The results indicate that the study participants have grasped well how to enumerate different process instances by stating them as lists of tasks. After the study, we have asked some control questions regarding the process to verify student's understanding, and 69% of the students answered them correctly with 95% of them stating that they have used the visual model to answer the questions. So while they were not taught how to draw BPMN constructs in order to model the process, they found the automatically generated visual model beneficial for their understanding. As one student stated: "the graphical representation is easier to be checked than the textual description". When asked how they would rate the concept behind BPMN Sketch Miner of using mining to derive visual process models from textual descriptions, 62% of them found it promising and 19% found it very powerful. Furthermore, all students agreed or strongly agreed that the generated visual model helped them understand the flow of the process better.

Modeling Strategies. As there can be different strategies to model with the textual DSL, we have annotated each model with the order in which the 5 process instances representing the described process were listed (Fig. 6). While there is no correlation between the accuracy of students' models and the order of the modelled process instances (correlation = -0.27), we have discovered interesting modelling strategies. For instance, although the claim rejection process instance was the first to be described, in 30% of the solutions it was the last to be modelled in the textual DSL (see the plot of instances in Fig. 6 marked with squares). This means that the students were gradually expanding their solution by inserting new tasks and fragments in their initial solution as they moved along the process description. In fact, only 38% of the students enumerated the instances in the order in which they appeared in the process description (see the plot of instance 12345 in Fig. 6), while the rest of the students mostly used different permutations of the last three process instances. No strategy for modelling the process

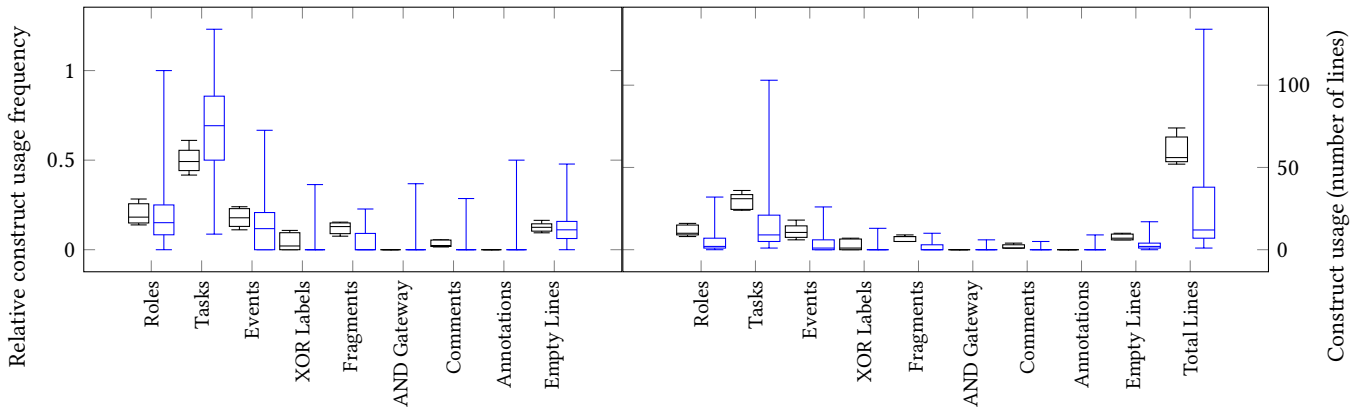


Figure 3: Relative and Absolute Construct Usage (Students and Analysts)

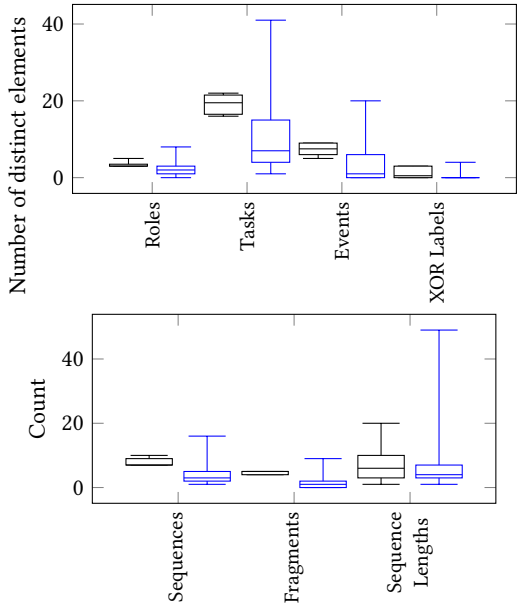


Figure 4: Unique Model Elements and Sequences (Students and Analysts)

instances in a certain order appears to outperform the others in terms of generating an accurate model or generating it faster.

System Usability Scale. We had 30 participants in the SUS survey, a mix of students that participated in the user study and of industry practitioners. The combined result of the students and industry (Fig. 7) reveals a usability score of 69 as an average of all the answers, with a 95% confidence interval of 64 to 73. Although in general higher scores indicate better usability, the thresholds used to interpret the score differ in different studies. In [31] they state that a score below 64 is unacceptable, above 85 is excellent and everything in between is acceptable. In [2] it is suggested that a SUS score of above 70 should be considered passable, with a score of above 60 meaning high marginal usability of the tested tool.

If we consider the students (18 respondents) and industry practitioners (12 respondents) as separate groups, we can see a notable difference in the usability score, which is 64 ± 5 for students and 76 ± 7 for industry practitioners with 95% confidence interval. The lowest individual score is 45 given by one student, while the highest individual score is 95, given by two respondents from the industry. To improve the usability of the system in the future, and get users more confident with using it, we will increase the number of tutorials and examples available on the website of the tool.

Efficiency and Learnability Survey. Among the SUS responses shown in Fig. 7, the statement with which the respondents tend to agree the most refers to the learnability of the system where only two of the respondents disagreed that it can be learned quickly. From the negative statements on the other hand the one with which the respondents disagreed the most refers to the need of support of a technical person to use the system.

In Fig. 8, we depict the answers of 9 analysts and 21 students for a set of questions about the learnability and the efficiency of the tool. We can see that a high majority of the analysts agree with the fact that using the tool does not require much cognitive effort, while it seems that some few students struggled with it. None of the students disagreed with the fact that the generated visual model was helpful for them to understand the flow. Moreover, the majority of both groups of users agrees with the fact that they can type faster than they can model the process with a graphical editor.

User Satisfaction In addition to the conducted usability study, we tracked also the users' satisfaction by means of surveys for analysts and students. The goal is to reflect how likely it is that these target users will use the tool in the future, either in their work or their studies, as well as to know about their general impression after using the tool, and, when it comes to the analysts, to know their opinion on whether the tool can make their job easier and faster. Overall, the answers of the 30 respondents can be summarized as follows: 1) Highly unsatisfied (0%), 2) Unsatisfied (3%), 3) Neutral (27%), 4) Satisfied (57%), 5) Highly satisfied (13%).

On-boarding, Engagement and Retention. To help with the on-boarding and engagement of the users of the tool, in addition to the DSL documentation, we have also added tutorials, to get the users started, and examples to show-case different modelling challenges. The industry survey respondents found the tutorials and examples either very or extremely helpful. Not all the 295

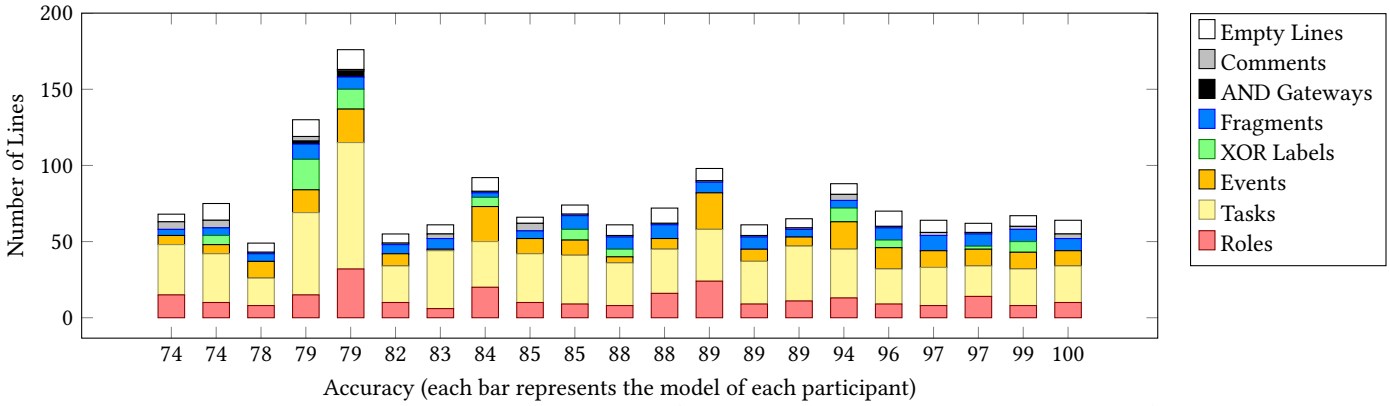


Figure 5: Construct usage vs. Accuracy (multiple constructs can share the same line).

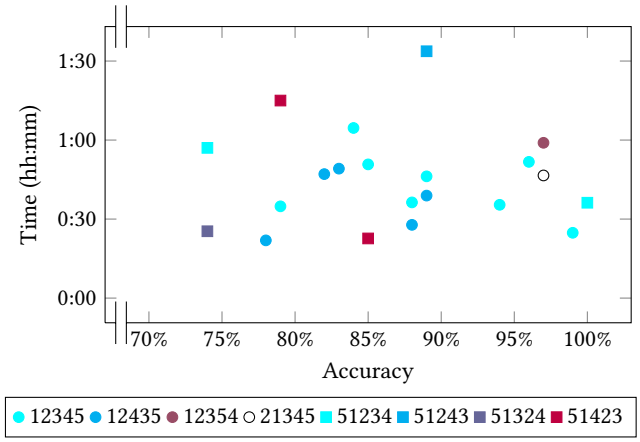


Figure 6: Model Accuracy vs. Time for Different Process Instance Orderings

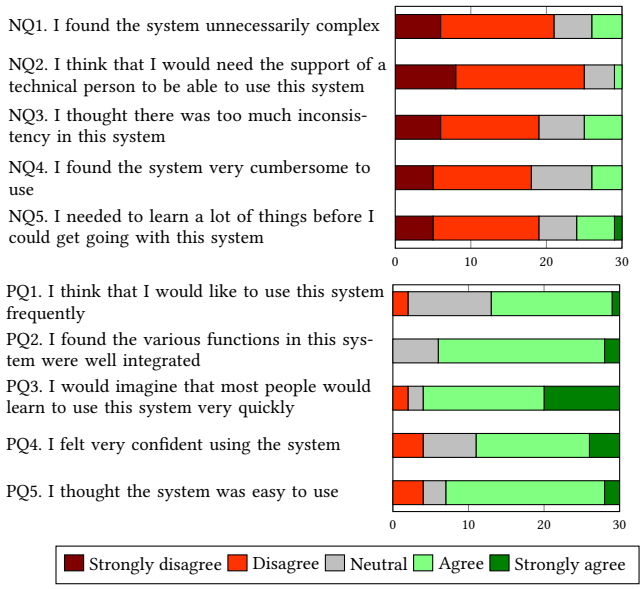


Figure 7: System Usability Scale (SUS) Survey Results

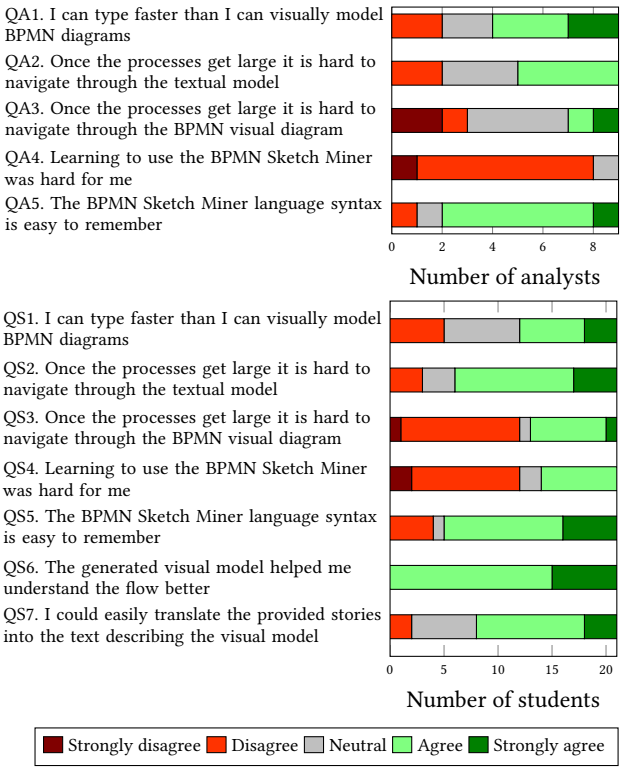


Figure 8: Efficiency & Learnability Assessment Survey

users who tried to sketch at least one model replied to our survey. However, based on our logs, many of them read the tutorials and examples and 40% of the users did return to the site at least once during the observed period, with 14 users returning more than 5 times. In fact, people who visited more of the tutorials and examples, and thus spent more time modeling with the tool, tend to come back more often with a correlation of 0.6 regarding the visits and 0.73 regarding the time. This shows that the tool and the language learning curve is not steep, and users who do invest the time to overcome it with the help of the provided tutorials, examples, and reference documentation tend to come back to use the tool.

4 DISCUSSION

4.1 Research Questions

The goal of the different evaluation methodologies we used is to tackle different aspects of the usability of the BPMN Sketch Miner and answer different research questions. Regarding RQ1: "Does BPMN Sketch Miner help BPMN non-experts to produce accurate BPMN models in a relatively short time?", the fact that all students participating in the user study managed to model a non-trivial business process with an accuracy of over 74% in less than one and a half hour provides positive evidence for the usefulness of the tool for introducing BPMN to novices. If we can judge their understanding of which process instances should be modeled based on the number of sequences they have listed (see Fig. 4), we can state that they have chosen both a complete, and in some cases, a minimal set of task sequences. 76% of them stated a full process instance sequence for the management and payment of the short term claims, which meant that they had to select and connect multiple separate pieces of the given natural language process description together. As one student stated BPMN Sketch Miner "is a really useful tool for learning to create the model without taking care of positioning. Also it is easy to learn the basic concepts". Fig. 5 shows that in addition to getting started with the study warm-up tutorial, some students could also benefit from the onboarding examples to learn by themselves how to use XOR label gateways or AND gateways.

Regarding RQ2: "What is the perceived usability of the tool and its features?", the results from the SUS survey, with an overall score of 69 ± 5 and industry score of 76 ± 7 show that, although there is always room for improvement, the tool is on the right track. The notably lower SUS score of 64 ± 5 based on students' answers could be due to the students involved having no experience neither in business process modeling, nor in using tools that support business process modeling. In fact, 7 of the students agreed that learning to use the BPMN Sketch Miner was hard for them (see Fig. 8) and we did see them struggle to embrace abstraction and process thinking during the user study. Nevertheless, most of the students did find the DSL easy to remember. Thus, the perceived usability of the tool by the students based on the survey follows the same trend as for the SUS score. Industry practitioners, on the other hand, tend to agree with the learnability both of the DSL and the tool. As one of the SUS respondents from industry commented: "It is a very intuitive tool that I will use from now on and recommend to my work colleagues!". Another one stated: "Fantastic tool, having daily use of it. Link feature is impressive". All but two industry respondents have agreed that the BPMN Sketch Miner would enable them to accomplish their tasks more quickly.

4.2 Threats to Validity

The number of responses we have obtained for the user study (21 participants) and for the diverse surveys (from 9 to 16 responses) is limited, and as such cast a shadow over the validity of our conclusions. Furthermore, the diverse background and experience of the respondents can bias their responses. To mitigate such risks, in addition to the survey and user study, we have analysed the tool usage logs, which provide a bigger sample size of 295 early adopters. When it comes to the user study, another threat comes from the inherent subjectiveness of the grading used to determine the accuracy of the solutions of the study participants. To mitigate this risk we created a systematic grading scale which we documented in the replication package and systematically applied on all solutions.

4.3 Lessons Learned

Setting the main goal of a tool to rapid sketching of business processes in BPMN comes with constraints relative both to the goal itself and to the targeted users. Using mining of task lists expressed in natural language has proved to be a good approach for BPMN non-experts as it helps them to grasp how a model is automatically obtained from enumerating different process instances. It removes the need of memorising many BPMN constructs and worrying about producing a syntactically valid model, thus keeping the focus of the novices on representing "who does what when". What we have learned from our evaluation is that novices initially struggle with the concept of process fragments (using ... within traces to avoid the repetition of previously stated tasks). Using fragments correctly requires modelers to learn how to abstract away common traces, which in turns requires first to state all traces fully, then look for repetitions, and finally convert redundant traces into fragments. While fragments are an important language feature to reduce the amount of code duplication, after the evaluation we have learned that the concept should be introduced only as soon as users notice or complain about having to enter redundant traces. After understanding the relationship between a process model and its corresponding execution traces, students are ready to start capturing directly the model without the use of the miner. From our experience, some students prefer to do so using the graphical editor as they can directly manipulate the visual elements, while others continued using the BPMN Sketch Miner to create the initial sketch and then used the graphical editor to further refine their model.

Business analysts who are accustomed to using graphical editors will need to discover how they can represent the visual model that they have in mind starting from its textual description. As one business analyst stated in the survey "While at this moment I think I would be still way faster using an editor, I reckon that an experienced user with a good text editor could really speed up the process of writing BPMN files by using this method". Such speed up is due to the automatic layout combined with the typing speed being faster than dragging and dropping visual elements. Another important component is the use of the process mining to automatically introduce some control flow gateways, as well as message flow and start/end events, which do not have to be explicitly specified in the textual input. We learned that for other BPMN constructs (e.g., parallel gateways), the amount of textual input required to infer their presence would require a greater effort than simply describing them explicitly. The same applies to the limited precision with which tasks and events can be automatically distinguished based on their labels expressed in natural language.

Overall, we have learned that there should be a sweet spot in determining the extent to which a process model can be obtained by mining process instance examples and how many modeling annotations need to be explicitly provided. The perfect mix may depend on the level of BPMN expertise of the modelers.

5 RELATED WORK

In [23] we have introduced the idea of obtaining BPMN models by means of mining traces described using a textual DSL. We presented an environment for analysts and process participants where they can rapidly sketch business process models as they discuss them using natural language during structured interviews. The approach supported only a very small number of BPMN constructs (start/end event, XOR gateway, task, and sequence flow BPMN constructs) and lacked the modeling annotations necessary to distinguish different types of tasks and events, pools, message flow, etc. The tool presented in this paper is still based on process mining, and as such provides access to the core constructs of the BPMN notation without requiring an explicit description of the process control flow, as, e.g., when using the PlantUML textual syntax for activity diagrams. On the other hand, it provides a much more expressive DSL compared to the tool we presented in [23] by combining the process mining with textual modeling.

Using textual DSL for process modelling is not a new idea. Starting from the same motivations as ours, the authors of [21] proposed a textual DSL for describing S-BPM [12] processes. However, unlike the BPMN Sketch Miner's textual DSL, the S-BPM is designed for explicitly declaring the whole model's structure textually. T-Square [35] is another declarative textual DSL for rapid processes description by specifying the tasks and the branching conditions. This DSL is incorporated in NOVA [30] which is an Eclipse-based editor which enables modelling workflows graphically based on the Compensable Workflow Modeling Language (CWML). While the goal of our textual DSL is generating a visual BPMN compliant artefact that can be exported in different formats, including the BPMN XML format, the goal of T-Square is to generate executable workflows from textual specifications, by means of a model transformation using Xtend. BPMN Sketch Miner's goal of speeding up the initial model construction phase is also shared with the Rapid Business Process Discovery (R-BPD) tool [18], whose solution combines both text-to-model and model-to-model transformations, which can extract models from any textual resource discovered in an enterprise repository, and also foster the reuse of existing models. However, none of the above stated textual DSLs targets BPMN as a modelling language. To the best of our knowledge, the only existing work on textual modelling for BPMN is PlantBPMN [13], which is a textual DSL created using the Xtext [9] framework, supported by an Eclipse-based editing tool. A model transformation between the metamodels created by the BPMN textual DSL and the BPMN XML Schema is implemented. The DSL presented in this paper is more abstract compared to PlantBPMN, as BPMN Sketch Miner uses process mining to infer the presence of many constructs (e.g., exclusive vs. event-based gateways, pools vs. swimlanes, start vs. intermediate vs. end events etc. - see Tab. 1 for details) which need to be explicitly detailed in PlantBPMN.

While our text to visual model transformation is motivated by streamlining the process modeling workshop feedback cycle, there are many tools targeting the reconstruction of process models starting from pre-existing documentation written in natural language. For example, text highlighting [1] augmented with Natural Language Processing (NLP) [29] has been proposed as a technique to guide the transition from informal text-based process descriptions to formal declarative process models. Friedrich et al. [15] combine a set of NLP techniques, which help to obtain a model which contains all the information needed for a BPMN representation. This approach has been successfully applied to reconstruct processes within specific domains (e.g., archeological excavation methods [8]). However, when written process documentation is not available, our approach supports business analysts during requirements gathering interviews where the process is being described for the first time or its initial sketch needs to be agreed upon with process participants.

The need of "instant feedback and shared understanding" of a business process between the business analyst and the domain experts has also been recognized by Grosskopf et al. [28]. Contrary to our approach, they use tangible BPMN elements (t.BPM) to be moved around by domain experts while physically building the visual business process model on a table. They have found that t.BPM allows domain experts to identify the need of model corrections faster, due to the gradual building up of the visual model, and that it motivates them to think more about their process. Dixit et al. [6] take a different approach to include the domain expert in the process discovery by providing suggestions for next possible constructs to be added in the model based on the probabilities discovered with process mining algorithms. This approach is limited for processes which lack event logs.

6 CONCLUSION

This paper presents the design and evaluation of the BPMN Sketch Miner with respect to its expressiveness and usability. The design of the textual BPMN implemented in the tool attempts to combine together modeling (prescribing what the process does) and mining (describing what the process does) by using constrained natural language. To validate our design decisions we have used different evaluation methodologies, including both the internal analysis of the expressiveness of the textual DSL for BPMN modeling used in the tool and external analysis of the usability and learnability of the BPMN Sketch Miner in form of a user study and surveys. The results from our evaluation show that the trade-off between expressiveness and usability is well balanced. Namely, a SUS score from industry practitioners of 76 ± 7 reflects the good usability of the tool and the survey respondents mostly found the DSL expressiveness as sufficient. On the other hand, the perceived usability of the tool by the students participating in the user study is lower with SUS score of 64 ± 5, as they struggled with some optional language constructs such as the use of fragments. None the less, the accuracy of their models of a non-trivial process with an average score of 87% shows that the combination of textual input and visual model output supported by the tool has facilitated their understanding of process modelling.

In the next version of the tool we plan to improve the editor with the most voted concepts by the survey respondents, such as allowing for manual adjustments in the layout, renaming tasks, and providing model verification. We also plan to provide support of the most requested BPMN constructs such as subprocesses, data stores and objects, and to conduct experiments with more complex processes to prove the effectiveness of the tool in rendering business analysts more efficient in their modelling efforts compared to the use of graphical tools.

Reproducibility and Data Availability

The user study task description and the datasets collected during the evaluation presented in this paper are available at: https://zenodo.org/record/3842020. The datasets include the answers from the user study and survey done with BPMN students, as well as the answers from the surveys taken by the industry practitioners.

Acknowledgements

The authors would like to thank Mathias Weske, the BPM class of 2020, the anonymous survey participants and reviewers for their invaluable feedback. The work is partially funded by the SNF, with the API-ACE project nr. 184692.

REFERENCES

[1] Amine Abbad Andaloussi, Jon Buch-Lorentsen, Hugo A. López, Tijs Slaats, and Barbara Weber. 2019. Exploring the Modeling of Declarative Processes Using a Hybrid Approach. In *Proc. Conceptual Modeling (ER 2019)*. Springer, Cham, 162–170.

[2] Aaron Bangor, Philip T Kortum, and James T Miller. 2008. An empirical evaluation of the system usability scale. *Intl. Journal of Human–Computer Interaction* 24, 6 (2008), 574–594.

[3] Marco Brambilla, Jordi Cabot, and Sara Comai. 2007. Automatic Generation of Workflow-Extended Domain Models. In *Proceedings of the 10th International Conference on Model Driven Engineering Languages and Systems (MODELS'07)*. Springer-Verlag, Berlin, Heidelberg, 375–389.

[4] John Brooke et al. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry* 189, 194 (1996), 4–7.

[5] Nadja Damij. 2007. Business process modelling using diagrammatic and tabular techniques. *Business process management journal* 13, 1 (2007), 70–90.

[6] PM Dixit, HMW Verbeek, JCAM Buijs, and WMP van der Aalst. 2018. Interactive data-driven process model construction. In *International Conference on Conceptual Modeling*. Springer, 251–265.

[7] Sebastian Döweling, Tarik Tahiri, Benedikt Schmidt, Alexander Nolte, and Mohammadreza Khalilbeigi. 2013. Collaborative Business Process Modeling On Interactive Tabletops. In *ECIS*.

[8] E. V. Epure et al. 2015. Automatic process model discovery from textual methodologies. In *Proc. 9th International Conference on Research Challenges in Information Science (RCIS)*. IEEE, 19–30. https://doi.org/10.1109/RCIS.2015.7128860

[9] Moritz Eysholdt and Heiko Behrens. 2010. Xtext: Implement Your Language Faster than the Quick and Dirty Way. In *Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion* (Reno/Tahoe, Nevada, USA) *(OOPSLA '10)*. ACM, 307–309. https://doi.org/10.1145/1869542.1869625

[10] Dirk Fahland and Matthias Weidlich. 2010. Scenario-based process modeling with Greta. In *Proceedings of the Business Process Management 2010 Demonstration Track (CEUR Workshop Proceedings)*, Marcello La Rosa (Ed.), Vol. 615. CEUR-WS.org, 52–57. http://ceur-ws.org/Vol-615/paper16.pdf

[11] Vincenzo Ferme, Jörg Lenhard, Simon Harrer, Matthias Geiger, and Cesare Pautasso. 2017. Workflow management systems benchmarking: unfulfilled expectations and lessons learned. In *Proceedings of the 39th International Conference on Software Engineering, ICSE 2017, Buenos Aires, Argentina, May 20-28, 2017 - Companion Volume*. IEEE Computer Society, 379–381. https://doi.org/10.1109/ICSE-C.2017.126

[12] Albert Fleischmann. 2010. What Is S-BPM?. In *S-BPM ONE – Setting the Stage for Subject-Oriented Business Process Management*, Hagen Buchwald, Albert Fleischmann, Detlef Seese, and Christian Stary (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 85–106.

[13] Nicole Freund. 2018. *Development of a Text-Based Representation of BPMN Models*. Master's thesis. Leibniz Universität Hannover, Hannover, Germany.

[14] Carsten Friedrich and Falk Schreiber. 2004. Flexible layering in hierarchical drawings with nodes of arbitrary size. In *Proceedings of the 27th Australasian conference on Computer science-Volume 26*. Australian Computer Society, Inc., P.O. Box 319 Darlinghurst, NSW 2010Australia, 369–376.

[15] Fabian Friedrich, Jan Mendling, and Frank Puhlmann. 2011. Process Model Generation from Natural Language Text. In *Proc. CAiSE*. Springer, 482–496.

[16] Emden R Gansner and Stephen C North. 2000. An open graph visualization system and its applications to software engineering. *Software: practice and experience* 30, 11 (2000), 1203–1233.

[17] Matthias Geiger, Simon Harrer, Jörg Lenhard, and Guido Wirtz. 2018. BPMN 2.0: The state of support and implementation. *Future Generation Computer Systems* 80 (2018), 250–262.

[18] A. Ghose, G. Koliadis, and A. Chueng. 2007. Process Discovery from Model and Text Artefacts. In *2007 IEEE Congress on Services*. IEEE Computer Society, Los Alamitos, CA, USA, 167–174. https://doi.org/10.1109/SERVICES.2007.52

[19] Thomas Goldschmidt, Steffen Becker, and Axel Uhl. 2008. Classification of Concrete Textual Syntax Mapping Approaches. In *Model Driven Architecture – Foundations and Applications*, Ina Schieferdecker and Alan Hartman (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 169–184.

[20] Hans Grönninger, Holger Krahn, et al. 2007. Textbased modeling. In *Proc. of the 4th International Workshop on Software Language Engineering*. Springer-Verlag.

[21] Kai Michael Höver, Stephan Borgert, and Max Mühlhäuser. 2013. A Domain Specific Language for Describing S-BPM Processes. In *S-BPM ONE - Running Processes*, Herbert Fischer and Josef Schneeberger (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 72–90.

[22] ISO/IEC 25010:2011(en) 2011. *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*. Standard. International Organization for Standardization.

[23] Ana Ivanchikj and Cesare Pautasso. 2019. Sketching process models by mining participant stories. In *Proc. BPM Forum*. Springer, 3–19.

[24] Sven Jannaber, Dennis M Riehle, Patrick Delfmann, Oliver Thomas, and Jörg Becker. 2017. Designing a framework for the development of domain-specific process modelling languages. In *International Conference on Design Science Research in Information System and Technology*. Springer, 39–54.

[25] Diane Jordan and John Evdemon. 2011. Business Process Model And Notation Version 2.0. OMG. http://www.omg.org/spec/BPMN/2.0/.

[26] Gabor Karsai, Holger Krahn, Claas Pinkernell, Bernhard Rumpe, Martin Schindler, and Steven Völkel. 2009. Design guidelines for domain specific languages. In *Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling (DSM' 09)*. Association for Computing Machinery, New York, NY, USA.

[27] Katja Kous. 2010. Comparative analysis versions of BPMN and its support with Control-flow patterns. In *The 33rd International Convention MIPRO*. IEEE, 315–319.

[28] Alexander Luebbe and Mathias Weske. 2012. Determining the effect of tangible business process modeling. In *Design Thinking Research*. Springer, 241–257.

[29] H. A. López, M. Marquard, L. Muttenthaler, and R. Strømsted. 2019. Assisted Declarative Process Creation from Natural Language Descriptions. In *Proc. 23rd International Enterprise Distributed Object Computing Workshop (EDOCW 2019)*. IEEE Computer Society, Los Alamitos, CA, USA, 96–99.

[30] Wendy MacCaull and Fazle Rabbi. 2012. NOVA Workflow: A Workflow Management Tool Targeting Health Services Delivery. In *Foundations of Health Informatics Engineering and Systems*, Zhiming Liu and Alan Wassyng (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 75–92.

[31] Sam McLellan, Andrew Muddimer, and S Camille Peres. 2012. The effect of experience on System Usability Scale ratings. *Journal of usability studies* 7, 2 (2012), 56–67.

[32] Yousra Odeh. 2017. BPMN in engineering software requirements: an introductory brief guide. In *Proceedings of the 9th International Conference on Information Management and Engineering*. Association for Computing Machinery, New YorkNYUnited States, 11–16.

[33] Avner Ottensooser, Alan Fekete, et al. 2012. Making sense of business process descriptions: An experimental comparison of graphical and textual notations. *Journal of Systems and Software* 85, 3 (2012), 596–606.

[34] Carla Pacheco et al. 2018. Requirements elicitation techniques: a systematic literature review based on the maturity of the techniques. *IET Software* 12, 4 (2018), 365–378.

[35] Fazle Rabbi and Wendy MacCaull. 2012. T-Square: A Domain Specific Language for Rapid Workflow Development. In *Proceedings of the 15th International Conference on Model Driven Engineering Languages and Systems* (Innsbruck, Austria) *(MODELS'12)*. Springer-Verlag, Berlin, Heidelberg, 36–52. https://doi.org/10.1007/978-3-642-33666-9_4

[36] Wil van der Aalst. 2011. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer.

[37] Wil MP van Der Aalst, Arthur HM Ter Hofstede, Bartek Kiepuszewski, and Alistair P Barros. 2003. Workflow patterns. *Distributed and parallel databases* 14, 1 (2003), 5–51.

[38] Mathias Weske. 2012. *Business Process Management: Concepts, Languages, and Architectures* (2nd ed.). Springer.

[39] Petia Wohed, Marlon Dumas, Arthur HM Ter Hofstede, and Nick Russell. 2006. Pattern-based Analysis of BPMN-An extensive evaluation of the Control-flow, the Data and the Resource Perspectives. (2006).

[40] Michael Zimoch, Rüdiger Pryss, Thomas Probst, Winfried Schlee, and Manfred Reichert. 2018. The Repercussions of Business Process Modeling Notations on Mental Load and Mental Effort. In *Proc. BPM*. Springer, 133–145.

[41] Michael Zur Muehlen and Jan Recker. 2008. How much language is enough? Theoretical and practical use of the business process modeling notation. In *Proc. CAiSE*. Springer, 465–479.