Declo: A Chatbot for User-friendly Specification of Declarative Process Models

Anti Alman[1], Karl Johannes Balder[1],
Fabrizio Maria Maggi[2], and Han van der Aa[3]

[1] University of Tartu, Estonia
anti.alman@ut.ee,Karl.Johannes.Balder@tudeng.ut.ee
[2] Free University of Bozen-Bolzano, Italy
maggi@inf.unibz.it
[3] University of Mannheim, Germany
han@informatik.uni-mannheim.de

Abstract. Proposed approaches for modeling knowledge-intensive processes include declarative, constraint-based solutions, which meet halfway between support and flexibility. A noteworthy example is the Declare framework, which is equipped with a graphical declarative language whose semantics can be expressed with multiple logic-based formalisms. So far, the practical use of Declare constraints has been mostly hampered by the difficulty of modeling them: the formal notation of Declare represents a steep learning curve for users lacking an understanding of temporal logic, while the graphical notation has proven to be unintuitive. This work presents Declo, a chatbot that allows users to easily define declarative constraints using natural language statements provided in the form of vocal or textual input. The supported constraints cover the entire Multi-Perspective extension of Declare (MP-Declare), complementing control-flow constraints with data and temporal perspectives.

1 Introduction

Knowledge-intensive processes are known to be complex and unpredictable [6]. These types of processes are difficult to model with standard procedural process modeling languages like BPMN (because these languages would produce complex models difficult to understand), but are more suitable to be represented with declarative process modeling languages like Declare [9].

However, Declare is known to be difficult to use in practice since its formal notation represents a steep learning curve for users lacking an understanding of temporal logic, while its graphical notation has proven to be unintuitive [7]. Declo is a chatbot supporting users in the specification of declarative process models using natural language. Users can define constraints providing them in the form of vocal or textual input without the need for previous knowledge about Declare. The constraints covered by Declo include the entire Multi-Perspective extension of Declare (MP-Declare) [5], complementing control-flow constraints with data and temporal perspectives.

2 Approach and Features

With Declo, the user can build declarative process models using natural language. A natural language statement can be provided using a chat, or a vocal input. In the latter case, the vocal input is recorded and interpreted using speech recognition. In both cases, to interpret the text, the tool first presented in [2] is used for the automatic extraction of constraints from natural language.[1]

Linguistic processing. During the first step of the constraint extraction, linguistic processing is performed to identify the semantic components from the input statement. Verbs are used to express actions in natural language texts, e.g., "to create". They can be effectively identified by using part-of-speech taggers. The subject of a verb specifies the entity performing the action (e.g., a manager creating a claim), while the object of a verb specifies the entity acted upon (e.g., a claim that is created). Subjects and objects are identified by computing dependency grammars using natural language processing techniques, which reveal the grammatical relationships among words using dependency relations.

The extracted verbs, objects, and subjects are further enhanced with specifiers, i.e., modal verbs, negations and prepositions. Modal verbs are auxiliary verbs indicating feasibility (mainly "can/could", "may/might", "must", "will/would", "shall/should") and are used to identify certain constraints, e.g., *response* (expressing the concept of obligation) and *precedence* (expressing the concept of permission). Negations are identified through the negative dependency relation and are used to identify negative constraints. Prepositions express relationships between nouns and other parts of a sentence; for example, the term "immediately" specifies the immediacy of the action (e.g., when specifying constraints of type *chain*).

Activity extraction. Activities can be either verb-based or noun-based. Verb-based activities are constructed from the semantic components extracted during the previous processing step, where the verb stands for the action, its subject for the actor and its object for the activity's business object. For example, the sentence "a manager processes a claim" translates to the activity *process claim*, with manager as the actor. Noun-based activities appear in a statement as noun phrases which correspond to process activities, e.g., "processing of a claim". In order to extract these activities, the tool focuses on identifying verbs which describe the process flow, i.e., temporal verbs such as "to start", "to precede", "to end", "to follow", "to happen". The object and/or the subject of the verb are used to generate the activity.

Data and time conditions. Declo supports adding data and time perspectives to declarative constraints, effectively turning Declare constraints into MP-Declare constraints. Three types of conditions are supported: activation, correlation and time conditions. Data and time conditions express more textual fragments rather than full sentences, unlike constraints; in addition, as they

[1] The tool can be found at https://github.com/hanvanderaa/declareextraction

are short statements, their variance is much lower than that of declarative constraints. As such, the tool uses pattern matching to extract conditions instead of grammatical parsing used during constraint generation.

Activation conditions define requirements that must be met for a constraint to be activated, e.g., the creation of a claim must be followed by a check only if the amount of the claim created is greater than a set value. Correlation conditions must be adhered to when the target of a constraint is being executed, expressing relations that must exist between attributes of the activation and the target. Declo supports two types of correlation conditions, i.e., either that the attribute values should be the same (e.g., "the amount is the same") or different (e.g., "the resource is different") for the activation and the target of a constraint. Complex conditions are supported by concatenating multiple logical statements: an input string is first split into substrings by detecting coordinating conjunctions (*and* and *or*) and splitting the string accordingly. Each substring is expected to match an expression. Expressions are then joined with logical operators. Pattern matching is supported on both numerical and categorical attributes.

Time conditions specify the time elapsed between the activation and the target of a constraint, e.g., that the target should occur between 2 to 6 days after the activation. Time conditions specified using seconds, minutes, hours and days are supported.

3 Pipeline and Implementation

Declo is integrated into the Rule Mining toolkit RuM[2] and is implemented using Java 11 and the speech recognition API provided by *Google Cloud Speech*.[3]

To interact with the chatbot, the user needs to access the MP-Declare editor of RuM, create a new MP-Declare model and press the "Start listening" button or the "Let's write" button in the chat. In the first case, Declo will listen for an input sentence from the microphone. Once the tool has recognized a sentence, it displays the sentence and adds the detected activities and constraints to the graphical model, to the list of activities and to the list of constraints. Then, Declo asks the user if the sentence was recognized correctly. If this is not the case, it is possible to edit the recognized sentence or discard the recognized constraints entirely and record again. If the user presses the "Let's write" button the sentence can be directly typed into the chat. Data and time conditions can also be added via speech recognition or the chat using the corresponding buttons: "Activation" for activation conditions, "Correlation" for correlation conditions, and "Time" for time conditions. Also these conditions can be modified or discarded. The user can repeat the steps above to create additional constraints.

Fig. 1 displays a screenshot of the MP-Declare editor in RuM. The top-left area of the screen shows the chat. The graphical notation of the constraints is shown in the top-right area of the screen, with the modifiable lists of activities and constraints in the bottom part of the screen. A screencast demonstrating the tool is available at https://www.youtube.com/watch?v=M_-kBu9Bqso.

[2] www.rulemining.org
[3] https://cloud.google.com/speech-to-text/

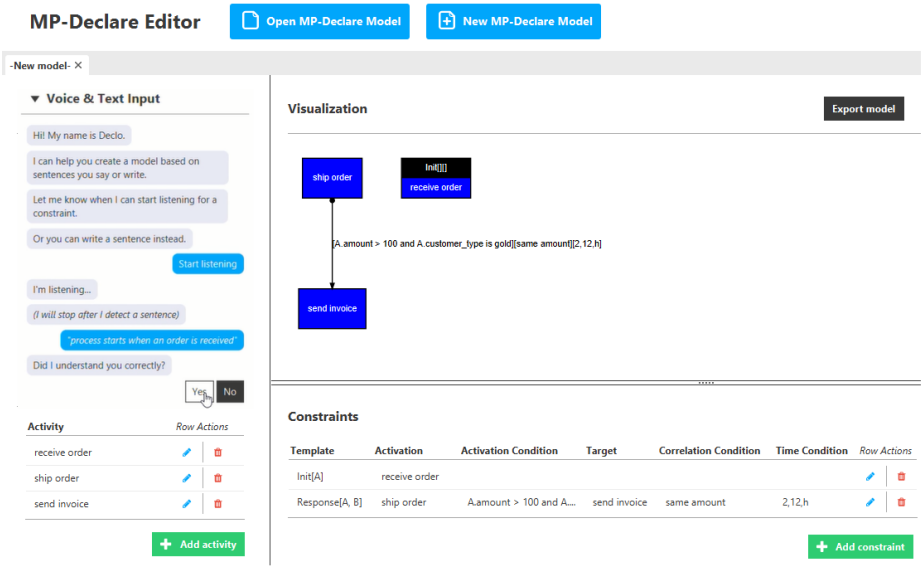

Fig. 1: A screenshot of the MP-Declare editor in RuM.

4 Maturity and Conclusion

A qualitative user study, presented in [1], was conducted to test the speech recognition component of Declo with the aim of (1) finding ways to improve the interface, (2) analyzing issues and needs of users from different backgrounds, and (3) discovering possible usage scenarios. For the study, eight participants were selected with different characteristics. Two participants were selected for their experience related to BPM (tier 1); two for their background in temporal logics (tier 2); two for having used Declare for modeling temporal constraints (tier 3); and two participants for having used different tools to model, analyze and execute Declare models (tier 4). The different backgrounds were used for differentiation in order to study how users with different levels of expertise perceive the tool and recognize possible issues and needs.

The study was conducted via Skype by a team consisting of a *facilitator* and an *observer*, with the facilitator guiding the participant and the observer serving in a supporting role. The participant shared her/his screen, whereas the facilitator started the video recording and began guiding the participant through a prepared test scenario, which consisted of constructing a declarative process model using speech recognition. At the end of the study, a follow-up interview was conducted, focusing on resolving questions about the issues the participants had encountered. The participants were asked what they liked about the tool, wished was different, under which circumstances they would use it. The study concluded with a post-survey questionnaire consisting of multi-point Likert scales including the System Usability Scale (SUS) [4] and scales covering satisfaction, expectation confirmation, continuation intention, and usefulness adapted from

Table 1: Questionnaire results represented as means across the different groups

	all	tier 1 (P6, P7)	tier 2 (P3, P8)	tier 3 (P2, P4)	tier 4 (P1, P5)
SUS	73.13	71.25	81.25	67.50	72.50
Satisfaction	3.46	3.67	3.67	3.00	3.50
Expectation	3.69	3.67	3.50	3.50	4.00
Future intentions	2.61	2.33	2.00	2.00	3.67
Usefulness	3.00	2.63	3.13	2.75	3.50

[3]. The collected data was analyzed by first creating an affinity diagram [8] based on the observations, follow-up interviews and video recordings by associating each reported issue with a single paper note. The notes were then clustered based on the appearing themes, resulting in 139 notes over 21 clusters. The data extracted from the questionnaires served as an additional qualitative data point (see Table 1).

Compared to the commonly reported average SUS score of 68 [4], the average score of 73.13 shows that using speech recognition for declarative process modeling is reasonably feasible, although, from the interviews, this approach resulted to be largely recognized to be especially suitable for mobile environments. For this reason, the most significant improvement applied to the tool presented in [1] was to integrate the speech recognition mechanism with the chat, which represents a valid alternative in the context of desktop applications.

Acknowledgments. This research is supported by the Estonian Research Council (project PRG887).

References

1. van der Aa, H., Balder, K.J., Maggi, F.M., Nolte, A.: Say It In Your Own Words: Defining Declarative Process Models Using Speech Recognition. In: BPM Forum. (2020)
2. van der Aa, H., Di Ciccio, C., Leopold, H., Reijers, H.A.: Extracting declarative process models from natural language. In: CAiSE. pp. 365–382. Springer (2019)
3. Bhattacherjee, A.: Understanding information systems continuance: an expectation-confirmation model. MIS quarterly pp. 351–370 (2001)
4. Brooke, J., et al.: SUS-a quick and dirty usability scale. Usability evaluation in industry 189(194), 4–7 (1996)
5. Burattin, A., Maggi, F.M., Sperduti, A.: Conformance checking based on multi-perspective declarative process models. Expert Syst. Appl. 65, 194–211 (2016)
6. Di Ciccio, C., Marrella, A., Russo, A.: Knowledge-intensive processes: characteristics, requirements and analysis of contemporary approaches. JoDS 4(1), 29–57 (2015)
7. Haisjackl, C., Barba, I., Zugal, S., Soffer, P., Hadar, I., Reichert, M., Pinggera, J., Weber, B.: Understanding Declare models: strategies, pitfalls, empirical results. Software & Systems Modeling 15(2), 325–352 (2016)
8. Holtzblatt, K., Wendell, J.B., Wood, S.: Rapid contextual design: a how-to guide to key techniques for user-centered design. Elsevier (2004)
9. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: Declare: Full support for loosely-structured processes. In: EDOC. pp. 287–300 (2007)