Say It In Your Own Words: Defining Declarative Process Models Using Speech Recognition

Han van der Aa[1], Karl Johannes Balder[2],
Fabrizio Maria Maggi[2], and Alexander Nolte[2]

[1] Humboldt-Universität zu Berlin, Germany
han.van.der.aa@hu-berlin.de
[2] University of Tartu, Estonia
{balder,f.m.maggi,nolte}@ut.ee

Abstract.

1 Introduction

Process models are an important means to capture information on organizational processes, serving as a basis for communication and often as the starting point for analysis and improvement [5]. For processes that are relatively structured, *imperative* process modeling notations, such as the Business Process Model and Notation (BPMN), are most commonly employed. However, other processes, in particular knowledge-intensive ones, are more flexible and, therefore, less structured. An important characteristic of such processes is that it is typically infeasible to specify the entire spectrum of allowed execution orders in advance [4], which severely limits the applicability of the imperial process modeling paradigm. Therefore, such processes are better captured using *declarative* process models. These models do not require an explicit definition of all allowed execution orders, but rather use constraints to define the boundaries of the permissible process behavior [3].

Although their benefits are apparent, establishing process models is known to be difficult, especially for domain experts that lack modeling expertise [6,10], and can be highly time-consuming [7]. Due to these barriers to model creation, processes are often documented using natural language instead []. Recognizing this, several works have been developed that automatically extract process models from natural language texts (cf., [2,6,8,9]). The vast majority of these focus on the generation of imperative process descriptions, whereas only one preliminary approach [1] targets the extraction of declarative process models.

In this work, we aim to support the elicitation of declarative process models based on natural language input. In particular, we present an interactive approach that employs speech recognition and is integrated in a declarative modeling and analysis toolkit. In this manner, we are able to provide the following contributions in comparison to the state of the art [1]:

1. greatly increase coverage of declare constraint types covered
2. handling more flexible input

3. no one-shot approach
4. additional expressiveness by incorporating conditions from data, time, and resource perspectives
5. constraints are now linked to each other, enabling the definition of actual declarative process models, rather than just individual constraints
6. directly use the generated process models as input for, e.g., conformance checking, event log generation, and process monitoring

describe evaluation results

describe remainder of paper

2 Background

2.1 Declarative Process Modeling

Assigned to Fabrizio.

overview of declare constraints, including MP declare

2.2 From Natural Language to Declarative Models

The extraction of declarative constraints from text is highly challenging due to the inherent flexibility of natural language. This manifests itself in the sense that, on the one hand, the same declarative constraint can be expressed in a wide variety of textual descriptions, whereas on the other hand, subtle textual differences can completely change the meaning of the described constraint. These two complimentary challenges can be illustrated as follows:

Table 1: Different descriptions of the same constraint

Variability of textual descriptions. As shown in Table 1, the same declarative constraint can be described in a broad range of manners. Key types of differences that may occur are:
– **Synonymous terms**
– **Grammatical structure.**
– **Reverse order.** **create examples and illustration for each of these**

Subtle differences leading to different constraints.
– Negation
– Order indicators
– Propositions
– Mandatory versus optional.

- state-of-the-art extraction approach
which challenges it handles and which constraints it covers
- our delta w.r.t. state of the art

Table 2: Subtle differences between constraint descriptions

3 Interactive Elicitation of Declarative Models

Figure 1 provides an overview of the main components of the Speech2RuM approach. As shown, the user provides input through speech as well as interaction with the Graphical User Interface (GUI). Based on this input, users are able to construct a declarative process model through three main functions: (1) describing constraints in natural language using speech, (2) augmenting constraints with data- and time-perspectives, and (3) optionally editing and connecting the model's constraints. In the remainder of this section, we will describe each of these three components in detail.

do we need to mention how to handle speech or do we put implementation details somewhere else?

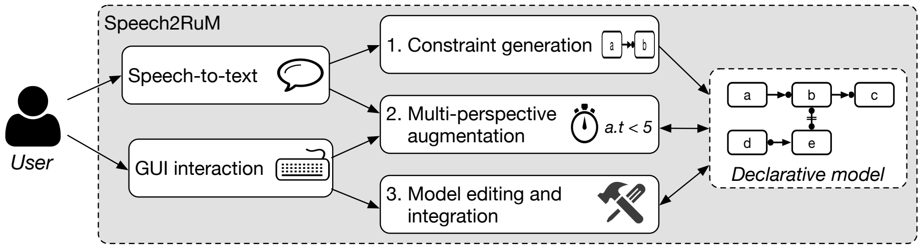

Fig. 1: Overview of the Speech2RuM approach

3.1 Constraint generation

In this component, our approach aims to turn a recorded sentence (transformed from speech to text using a standard software library [?]) into one or more declarative constraints. For this task, we enhanced the existing state-of-the-art approach for the extraction of declarative constraints from natural language text [1].

In particular, we take the result of the parsing step of that existing approach as input to the constraint generation approach. Given a sentence s, parsing yields a list A_s of actions described in the sentence and the inter-relations that exist between the actions, i.e., a mapping $rel_s : A_s \times A_s \rightarrow type$, with $type \in \{xor, and, dep\}$. As depicted in Figure 2, an action $a \in A_s$ consists of a verb and optional subjects and objects, which may, furthermore, all be augmented with additional information, such as whether or not an action is negated.

By building on this parsing step and the existing constraint-generation approach, we have added support to handle the additional constraint types presented in Table 3. Based on a set of activities A_s and a relation rel_s extracted for a sentence s, these additional types are handled as follows:

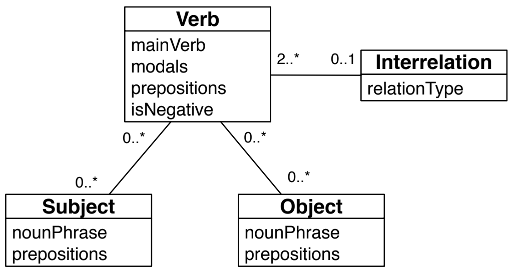

Fig. 2: Semantic components returned in the parsing step of [1]

- ***Participation* and *Absence.*** If A_s contains only one action, either an existence or absence constraint is established for the action, depending on whether it is negative or not.
- ***AtMostOne.*** If a *Participation* constraint is originally recognized, we subsequently check if s specifies a bound on its number of executions, i.e., by checking for phrases such as *at most once*, *not more than once*, *one time.*
- ***CoExistence* .** When a sentence has two actions in A_s that are in a relation of type *and* (and without any *dep* relations) are transformed into either a *CoExistence* or a *NotCoExistence* constraint, depending on whether there is a negation present.
- ***NotCoExistence* .**
- ***RespondedExistence.*** Responded existence constraints are similar to the originally extracted *Response*Their extraction occurs when two actions are in a dependency relation, i.e., a_1 *dep* a_2 for which it holds that the target action, a_2, is indicated to be mandatory, e.g., using *must.* The key difference between *Response* and *RespondedExistence* is that the former includes a notion of order, i.e., a_1 precedes a_2, whereas no such order is specified for *RespondedExistence* constraints.
- ***ChainPrecedence* and *ChainResponse.*** These constraint types are specializations of *Precedence Response* constraints. Chain constraints are recognized by the presence of a temporal preposition that indicates immediacy. Generally, such a preposition is associated with the verb of either an action a_1 or a_2 in a relation a_1 *dep* a_2. For this, we consider the preposition *immediately* and several of its synonyms, i.e., *instantly*, *directly*, and *promptly.*

Note that if a sentence does not yield constraints for these additional types, constraint generation proceeds to extract the six constraint types covered by the original approach.

Constraint	**Example**
Participation	An invoice must be created
Absence?	Dogs are not allowed in the restaurant.
AtMostOne	An invoice should be paid at most once.
CoExistence	Order shipment and invoice payment should occur together.
NotCoExistence	If an application is accepted, it cannot be rejected.
RespondedExistence	If a product is produced, it must be tested.
ChainPrecedence	After an order is received, it may immediately be refused.
ChainResponse	After an order is received, it must directly be checked.

Table 3: Additional constraint types in Speech2RuM

3.2 Multi-Perspective Augmentation

3.3 Model Editing and Integration

4 User Evaluation

Aim of the user study was to... Find improvements, identify who could use it, find usage scenarios

4.1 Setting

... qualitative case study... support material

4.2 Data sources

Tier 1: BPM but not declare (P6, P6)
Tier 2: Formal background about temporal logic (P3,P8)
Tier 3: Knows Declare but not the tool (P2,P4)
Tier 4: Declare super expert (P1,P5)

Participant selection (criteria) Tasks Observation / video, interview, questionnaire scales (check times)

4.3 Analysis procedure

aim, setup (scenario), data gathering, analysis

4.4 Findings

findings (minor changes, who can / should use it, when should people use it?) link to scenario

who uses the visual model / who uses text? who uses natural sentences? how useful is it for newcomers? usage scenarios adding features

5 Related work

6 Conclusion

References

1. van der Aa, H., Di Ciccio, C., Leopold, H., Reijers, H.A.: Extracting declarative process models from natural language. In: International Conference on Advanced Information Systems Engineering. pp. 365–382. Springer (2019)
2. van der Aa, H., Leopold, H., Reijers, H.A.: Checking process compliance against natural language specifications using behavioral spaces. Information Systems 78, 83–95 (2018)
3. van Der Aalst, W.M., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science-Research and Development 23(2), 99–113 (2009)
4. Di Ciccio, C., Marrella, A., Russo, A.: Knowledge-intensive processes: characteristics, requirements and analysis of contemporary approaches. Journal on Data Semantics 4(1), 29–57 (2015)
5. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A., et al.: Fundamentals of business process management, vol. 1. Springer (2013)
6. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: International Conference on Advanced Information Systems Engineering. pp. 482–496. Springer (2011)
7. Herbst, J., Karagiannis, D.: An inductive approach to the acquisition and adaptation of workflow models. In: Proceedings of the IJCAI. vol. 99, pp. 52–57. Citeseer (1999)
8. de Oliveira, J.P.M., Avila, D.T., dos Santos, R.I., Fantinato, M.: Assisting process modeling by identifying business process elements in natural language texts. In: Advances in Conceptual Modeling: ER 2017 Workshops AHA, MoBiD, MREBA, OntoCom, and QMMQ, Valencia, Spain, November 6–9, 2017, Proceedings. vol. 10651, p. 154. Springer (2017)
9. Schumacher, P., Minor, M., Schulte-Zurhausen, E.: Extracting and enriching workflows from text. In: 2013 IEEE 14th International Conference on Information Reuse & Integration (IRI). pp. 285–292. IEEE (2013)
10. Selway, M., Grossmann, G., Mayer, W., Stumptner, M.: Formalising natural language specifications using a cognitive linguistic/configuration based approach. Information Systems 54, 191–208 (2015)