VERTO - A Visual Notation for Declarative Process Models

Lauren S. Ferro
Consorzio Interuniversitario Nazionale per l'Informatica
Rome, Italy
dr.lsferro@gmail.com

Andrea Marrella
Sapienza University of Rome
Rome, Italy
marrella@diag.uniroma1.it

ABSTRACT

Declarative approaches to business process modeling allow to represent loosely-structured (declarative) processes in flexible scenarios as a set of constraints on the allowed flow of activities. However, current graphical notations for declarative processes are difficult to interpret. As a consequence, this has affected widespread usage of such notations, by increasing the dependency on experts to understand their semantics. In this paper, we tackle this issue by introducing a novel visual declarative notation targeted to a more understandable modeling of declarative processes.

CCS CONCEPTS

• **Applied computing** → **Business process modeling**; • **Human-centered computing** → *Interaction design*; • **Software and its engineering** → *Visual languages*;

KEYWORDS

Business Process Modeling, Declarative Process, Visual Notation

ACM Reference Format:
Lauren S. Ferro and Andrea Marrella. 2018. VERTO - A Visual Notation for Declarative Process Models. In *Preprint of 2018 International Conference on Advanced Visual Interfaces, AVI '18, May 29-June 1, 2018, Castiglione della Pescaia, Italy.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3206505.3206594

1 INTRODUCTION

Language is an important method of communication, albeit verbal, visual, or kinetic. In the context of business processes, language is imperative communicate information about the flow of activities. To explain such processes, several graphical notations and languages exist (e.g., BPMN, YAWL, Petri-Nets, etc.), which are used by computer scientists and practitioners to graphically describe processes identifying the control flow of activities and the overall process structure. From here, this information needs to be communicated to business stakeholders. However, the issue with using such process orientated languages is their level of interpretability for persons who are not inherently from the field of computer science.

This issue is even more evident when processes are modeled using the declarative paradigm. Contrary to the commonly used imperative paradigm of process modeling, the declarative approach does not enforce a strict order of activities, but limits their behavior through the use of constraints [5]. State-of-the-art solutions, e.g., [3, 5, 8], struggle with effectively communicating explicit concepts of how to interpret a declarative process model, and results in existing literature suggest that a new notation, easing understandability, is needed, cf. [4]. In this direction, in this paper we present a novel graphical notation named VERTO. The development of this notation has been done with considerations of design principles and several parameters (e.g., use of shapes) to maintain a contextual fit.

1.1 An overview of existing notations

Declare is a declarative process modeling language originally introduced by van der Aalst et al. in [8], which describes a process as a set of temporal constraints that must be satisfied throughout the process execution. Technically, a Declare model $D = (A, \pi_D)$ consists of a set of possible activities A involved in a process and a collection of constraints π_D defined over such activities. Declare constraints are instantiations of well-defined templates [1, 2]. Each one has a graphical representation that should be understandable to the user, but also a precise semantics in different logics (e.g., LTL over finite traces [6]), making them verifiable and executable.

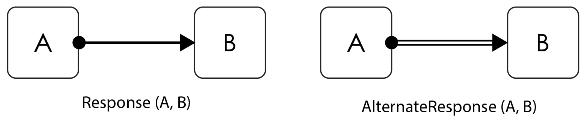

Figure 1: Examples of templates in Declare: (left) *Response(A,B)* and (right) *AlternateResponse(A,B)*.

Figure 1 shows two examples of templates with the Declare notation: on the left, *Response(A,B)* (if A occurs in the process instance, then B occurs after A), and on the right, *AlternateResponse(A,B)* (each time A occurs in the process instance, B occurs afterward, before A recurs). To date, the Declare notation includes around twenty templates. Researchers in the field of business process management have sought ways to present the Declare templates in a way that is interpretable by business users. The fact is that the fluency that a person can interpret these templates is crucial, having the potential to reduce the reliance on computer science expert consulting for their analysis.

Several attempts have been made to improve the interpretability of declarative models. For example, Di Ciccio et. al. [3] expand upon previous Declare notation, resulting with some improvement on clarity. For example, the change of symbols to communicate various templates such as a solid "X", double outlined boxes, and dotted edged boxes, cf. Figure 2.

On the other hand, Hanser [5] presents a notation that (in some ways) challenges the original notation of Declare with the use of "circles" as opposed to "squares" throughout the different templates. In this instance, one may argue that the change in shape is

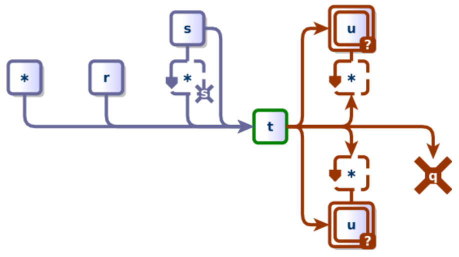

Figure 2: Examples from Di Ciccio et. al. [3]: (left) *respondedExistence(t,u)* and (right) *notResponse(t,q)*.

likely to cause confusion for those who are familiar with circles representing points of connections, squares representing process activities, rhombus for input/output, and so on. Therefore, while Hanser's model is an innovative approach as it challenges the design components, it is essentially an iterated version of the existing declarative templates presented in [8]. Figure 3 shows the same aforementioned templates for DECLARE in the notation by Hanser.

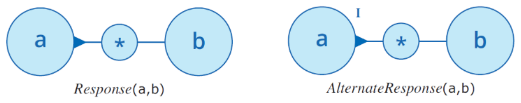

Figure 3: Examples in the notation by Hanser [5]: (left) *Response(a,b)* and (right) *Alternate Response(a,b).*

1.2 Design considerations for visual languages

The "anatomy" of a visual language consists of several components: graphical symbols, compositional rules, and so on which are bound by a syntax. Consequently, many schools of thought exist about the ability to learn, retain, utilize, and understand a language (e.g., prescriptive/descriptive theories). For example, based on prescriptive (learning) theory, Moody [7] expands upon nine key principles to consider for the development of visual notations in the context of engineering. As a result, parameters and guidelines exist to use as considerations for the design of visual languages.

2 METHODOLOGY

In this paper, we propose a new notation for modeling declarative processes, named VERTO. The current version of VERTO, which includes a restyling of the original DECLARE templates as presented in [8], has been obtained through a methodology made up by the following steps:

- check other existing process modelling languages (both imperative and declarative) to develop a foundation;
- establish a library of common graphical notations;
- identify areas where there is less clarity;
- define the design parameters for VERTO. We identified areas that may cause concern or confusion, such as the use of circles in place of squares, the size of arrows/arrow heads, ambiguous symbols, etc.;
- translate each DECLARE template into the VERTO notation (an overview of the notation is shown in Figure 4);

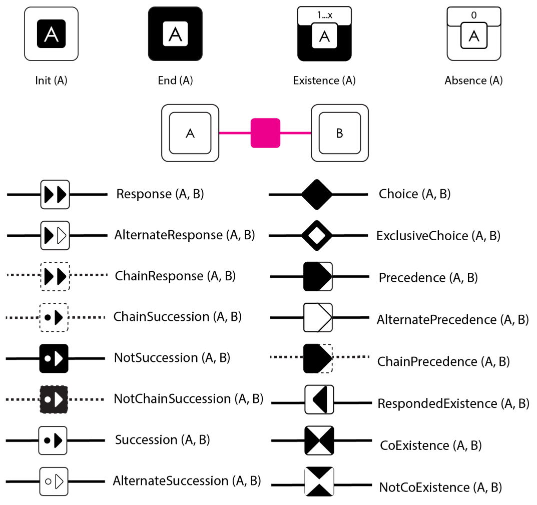

Figure 4: Overview of VERTO templates. The pink square and lines can be replaced with those following.

- informal testing and discussion of VERTO to determine if the templates could be understood.

3 PRELIMINARY RESULTS

Initial testing of the original version of DECLARE [8] revealed that the choice of "squares" and "circles" could remain the same as they are within the language of computer science. We performed a preliminary (informal) focus group to discuss the potential usefulness of VERTO. The criteria used to assess the effectiveness of the translation have been the following:

- Is the notation understandable by non-expert users?
- Does VERTO improve upon the DECLARE templates?
- Can VERTO templates be easily hand drawn?

The result of the focus group was that VERTO seems more "understandable" than previous declarative notations, even if we are aware that a robust evaluation with real business users must be performed to validate our thesis.

4 DISCUSSION

In this paper, we have presented VERTO, a novel graphical notation to model declarative processes. The choice of the name VERTO is due to its Latin meaning: "translate/interpret", making it appropriate for our purposes. The initial feedback from early testing of VERTO reveals that the notation is understandable but requires additional tests and refinements to make it a viable tool before widespread use. As a next step, we are working on defining several interactive testing with VERTO among university students who are using DECLARE for modeling declarative processes.

Acknowledgments. This work is partly supported by the Italian projects IT-SHIRT (Sapienza grant), Social Museum and Smart Tourism (CTN01_00034_23154), RoMA - Resilience of Metropolitan Areas (SCN_00064) and FilieraSicura.

REFERENCES

[1] Giuseppe De Giacomo, Fabrizio Maria Maggi, Andrea Marrella, and Fabio Patrizi. 2017. On the Disruptive Effectiveness of Automated Planning for LTLf-Based Trace Alignment. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI 2017.*

[2] Giuseppe De Giacomo, Fabrizio Maria Maggi, Andrea Marrella, and Sebastian Sardiña. 2016. Computing Trace Alignment against Declarative Process Models through Planning. In *Proceedings of the Twenty-Sixth International Conference on Automated Planning and Scheduling, ICAPS 2016.*

[3] Claudio Di Ciccio, Massimo Mecella, and Tiziana Catarci. 2011. Representing and Visualizing Mined Artful Processes in MailOfMine. In *Information Quality in e-Health*. Springer Berlin Heidelberg.

[4] Dirk Fahland, Daniel Lübke, Jan Mendling, Hajo Reijers, Barbara Weber, Matthias Weidlich, and Stefan Zugal. 2009. Declarative versus Imperative Process Modeling Languages: The Issue of Understandability. In *Enterprise, Business-Process and Information Systems Modeling*. Springer, Berlin, Heidelberg.

[5] Michael Hanser, Claudio Di Ciccio, and Jan Mendling. 2016. A new notational framework for declarative process modeling. *Softwaretechnik-Trends* 36, 2 (2016).

[6] Marco Montali, Maja Pesic, Wil M. P. van der Aalst, Federico Chesani, Paola Mello, and Sergio Storari. 2009. Declarative Specification and Verification of Service Choreographies. *ACM Trans. Web* 4, 1 (2009).

[7] D. Moody. 2009. The Physics of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering* 35, 6 (2009).

[8] W. M. P. van der Aalst, M. Pesic, and H. Schonenberg. 2009. Declarative workflows: Balancing between flexibility and support. *Computer Science - Research and Development* 2 (2009).