How to (re)design declarative process notations? A view from the lens of cognitive effectiveness frameworks

Hugo A. López[1,*], Vít Dexter Simon[2]

[1] Technical University of Denmark, DTU Compute. Richard Petersens Plads, 321, 2800 Kgs. Lyngby, Denmark
[1] University of Copenhagen, Computer Science Department. Universitetsparken 1, 2100 Copenhagen, Denmark

Abstract
Declarative process modelling notations are a family of approaches where events obey a set of constraints rather than specific flows. While declarative models can express in a few constructs a large set of process behaviours, there is little adoption of declarative approaches compared to their imperative counterparts. One possible reason is that while expressive, there has not been enough focus on the user-centred design of declarative notations. In this paper, we explore how cognitive effectiveness frameworks could improve the development of declarative notations that support novice and expert users. In this paper, we analyse one representative declarative notation (DCR graphs) against two cognitive effectiveness frameworks. Our analysis suggests thirteen areas of improvement. For notation developers, we provide theory-backed guidelines, while for researchers in process models we outline hypotheses in the understandability of process models for further empirical testing.

Keywords
Declarative Process Models, DCR graphs, Cognitive Effectiveness, Human-Computer Interaction, Physics of Notations

1. Introduction

Process models are a fundamental piece in the understanding, analysis, optimisation and implementation of business processes, providing a common understanding for business and technical users alike. While a plethora of notations supporting process models exists, it is commonly accepted that most notations fall between the imperative and the declarative modelling spectrum [1]. Imperative notations such as BPMN facilitate the description of sequential process flows, whereas declarative specifications like DECLARE [2], CMMN [3], or DCR graphs [4] describe cause-effect relations between events. While semantic aspects of declarative processes have been amply studied, there has been relatively less research on how to engineer declarative notations such that they support model understanding, and how to equip notations with features that allow for understanding when the complexity of models increases.

PoEM 2022 Forum, 15th IFIP Working Conference on the Practice of Enterprise Modeling 2022 (PoEM-Forum 2022), November 23-25, 2022, London, UK
*Corresponding author.
hulo@dtu.dk (H. A. López); pfs108@alumni.ku.dk (V. D. Simon)
http://lopezacosta.net/ (H. A. López)
0000-0001-5162-7936 (H. A. López)

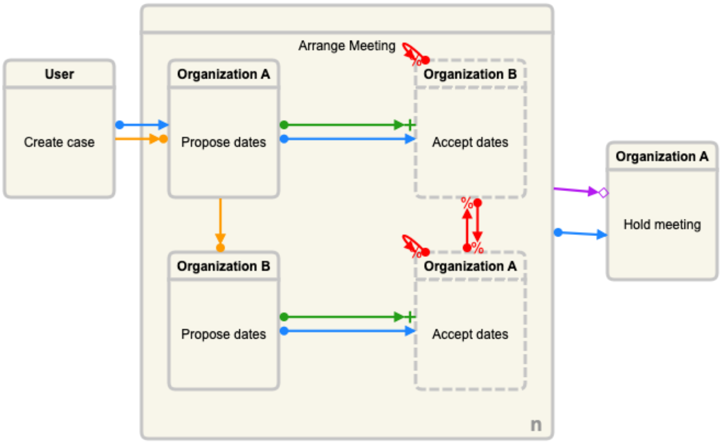

Figure 1: DCR example: the meeting booking process [6]

This paper studies the cognitive effectiveness of a declarative process modelling notation, the Dynamic Condition Response (DCR) graphs. DCR is a graph-based notation where events are constrained by behavioural constraints (similar to other formal notations, for example, DECLARE). The choice of notation is not incidental: compared to DECLARE, DCR is actively used by modellers in industry, and it is integrated into KMD Workzone, a case management solution provided to central government institutions in Denmark [5], Australia and Japan. Figure 1 introduces a simple DCR model. Intuitively, this process describes the interactions between different organisations to book a meeting. Events (rounded squares) denote what can happen in the process. Relations (arrows) impose constraints on when events can be executed. For instance, the condition relation (i.e., →•) says that the organization A cannot *propose dates* unless (a user) has executed *create case*. With a response relation (i.e., •→) we impose an obligation: once (organization A) has *proposed dates*, then (organization B) must accept them. Events can be placed in or out of the context: an event out of context (denoted with a dashed border) can be included using the inclusion relation (i.e., →+) and removed again using an exclusion relation (i.e., →%). Event collections (i.e., *arrange meeting*) simplify the visual notation by collapsing into one the multiple relations that enclosed events have to events from the outside. As with many other software notations, the notation has evolved over time to cover new business requirements. This has made the formalism and accompanying notation change and grow from its original presentation in 2011 [4]. We use its most recent iteration including all the extensions, with a total of 32 distinct visual elements. Also, industrial DCR models are orders of magnitude larger than Figure 1, with models that can reach the hundreds of constraints, events and collections[1]. While research has focused on building a language that is expressive and robust, how to represent concepts and control visual complexity has received far less attention, and its official language documentation[2] does not provide empirical evidence regarding the design rationale, nor about how the design best fits the potential users and the

[1]See, for example, the adoption consideration section in [7].
[2]https://documentation.dcr.design/

modelling tasks at hand.

This paper presents an initial investigation of the factors affecting the understandability of DCR graphs. In a purely empirical cycle, we focus on the observation stage, with the goal of building a set of hypotheses that can be further tested via user studies. We conduct a systematic analysis using two of the most widespread cognitive effectiveness frameworks: the Physics of Notations (PoN) [8] and SEQUAL [9]. They provide a set of general, notation-agnostic guidelines on how to maximise the perceptual properties of graphical notations and modelling tools that are backed by empirical evidence. Thus, the result of this paper is 1) a critical analysis of the current status of DCR according to cognitive effectiveness frameworks, and 2) a set of guidelines for improvement of the notation. The chosen frameworks provide us with an unbiased yardstick that can be used to compare against other notations and complements earlier work on tailor-made DCR quality frameworks from the perspective of experts [10].

The paper is structured as follows. Preliminaries for DCR, PoN and SEQUAL are introduced in Section 2. The analysis of DCR notation against PoN is presented in Section 3. The analysis using the SEQUAL framework is presented in Section 4. Suggestions for improvement of the notation are presented in Section 5. Related work is presented in Section 6. The paper concludes in Section 7.

2. Preliminaries

2.1. DCR graphs and its extensions

DCR graphs [4] is a modelling notation where processes are collections of events and constraints. We introduce DCR via a process model in execution (see Figure 2), using labels $A, B, C, \ldots$ to refer to specific events. We refer to [4] for the formal definition. Figure 4 in Appendix shows the metamodel of DCR and its extensions.

A DCR graph is a graph containing a set of events and event collections, a set of relations, and a marking. Graphs contain static and dynamic components. Starting with the static components, **events** can be atomic (e.g., A) or represent event collections (e.g., J). They can have 0, 1 or multiple roles assigned to them (e.g., A, T, S). An event may have a data payload (e.g., O) [11], execute computations (e.g., G), or determine their value via DMN tables (e.g., I) [12]. Finally, events can be linked to other DCR graphs using networks (e.g., R) [13]. **Relations** constraint events. DCR has ten types of relations. First, *precedence* relations such as condition (e.g., relation $A \rightarrow\!\bullet B$), milestone (e.g., relation $S \rightarrow\!\diamond C$), and precondition (e.g., relation $B \rightarrow\!\blacklozenge C$), allow the execution of the consequent once the precedent has been executed. Second, *obligations* such as response (e.g., relation $S \bullet\!\rightarrow B$) and no-response (e.g., relation $G \bullet\!\rightarrow\!\mathrm{x} C$) modify the pending state of the consequent depending on the execution of the antecedent. Finally, *alternative* relations such as include (e.g., relation $T \rightarrow\!+ B$), exclude (e.g., relation $K \rightarrow\!\% L$) and logical include (e.g., relation $G \rightarrow\!\pm E$) modify the inclusion state of each event depending on the execution of the antecedent. Relations can be applied to the same element (e.g., relation $B \rightarrow\!\% B$), or multiple relations can link a group of events (e.g., $B \rightarrow\!\bullet C$ and $B \bullet\!\rightarrow C$). In addition, relations can have guarded expressions (e.g., relation $G \xrightarrow{A>10}\!\pm E$), and conditions and responses may define how long an event will be delayed (e.g., relation $C \xrightarrow{PT1S}\!\bullet D$) or time boundaries when an event should be executed (e.g., relation $D \bullet\!\xrightarrow{PT5S} H$) [14].

The dynamic components include the execution state. Each event/collection has a **marking**. An event can be included (events using solid borders, e.g., A), excluded (events with dashed borders, e.g., B), pending (events with "!", e.g., C), executed (events with ✓, e.g., D) or be in a combination of states (e.g., F). It is the marking that defines whether an event can be executed or not: An arbitrary event is enabled if i) it is included, and ii) all the events with precedence relations towards the event have either been executed or are excluded. Also, part of the dynamic aspects involves access control (e.g., H), which binds the role of an activity according to the evaluation of an expression. **Event collections** include a stateless *nesting construction* (e.g., J) that enables modellers to use a single constraint to bind multiple events [6], a *subprocess construction* (e.g., M) that has the same marking as a regular event but whose state depends on the marking of its enclosing events and a *multi-instance subprocess* (e.g., U) that creates copies of each of the enclosed events and relations at runtime [15], bound to normal events via the spawn relation. Finally, multiple events can be grouped into one data form (e.g., Q) [11].

2.2. The Physics of Notations (PoN)

The Physics of Notations [8] is a scientific framework for the analysis of cognitive effectiveness in visual notations. It considers nine dimensions:

2.2.1. Semiotic Clarity

studies how symbols and concepts relate. We consider this principle satisfied only if symbols and concepts map 1:1. Misalignments can be categorised as: *symbol redundancy:* one concept is represented by multiple symbols, *symbol overload:* multiple concepts are visualised by the same symbol, *symbol excess:* at least one symbol has no semantics, and *symbol deficit:* no symbol matches one or more concepts.

2.2.2. Perceptual Discriminability

measures the ease at which a user can distinguish graphical symbols apart. Low discriminability renders diagrams less effective in conveying information and increases the cognitive load for novice users [8]. Visual variables represent quantifiable properties of visual objects. *Retinal variables* include shape, size, colour, brightness, orientation, texture, and *planar variables* include horizontal and vertical positioning in a canvas. Their combination allows for an estimation of the visual distance between symbols, as well as how easy is to differentiate them. This guideline is satisfied if the combination of visual variables distinguishes each concept apart.

2.2.3. Visual Expressiveness

analyses the use of visual variables and their diversity across the visual notation focusing on the visual distance and differences between individual symbols. The expression of a concept using a range of visual variables results in a richer representation that exploits multiple visual communication channels. We consider this guideline satisfied if there is more than one visual variable differentiating concepts.

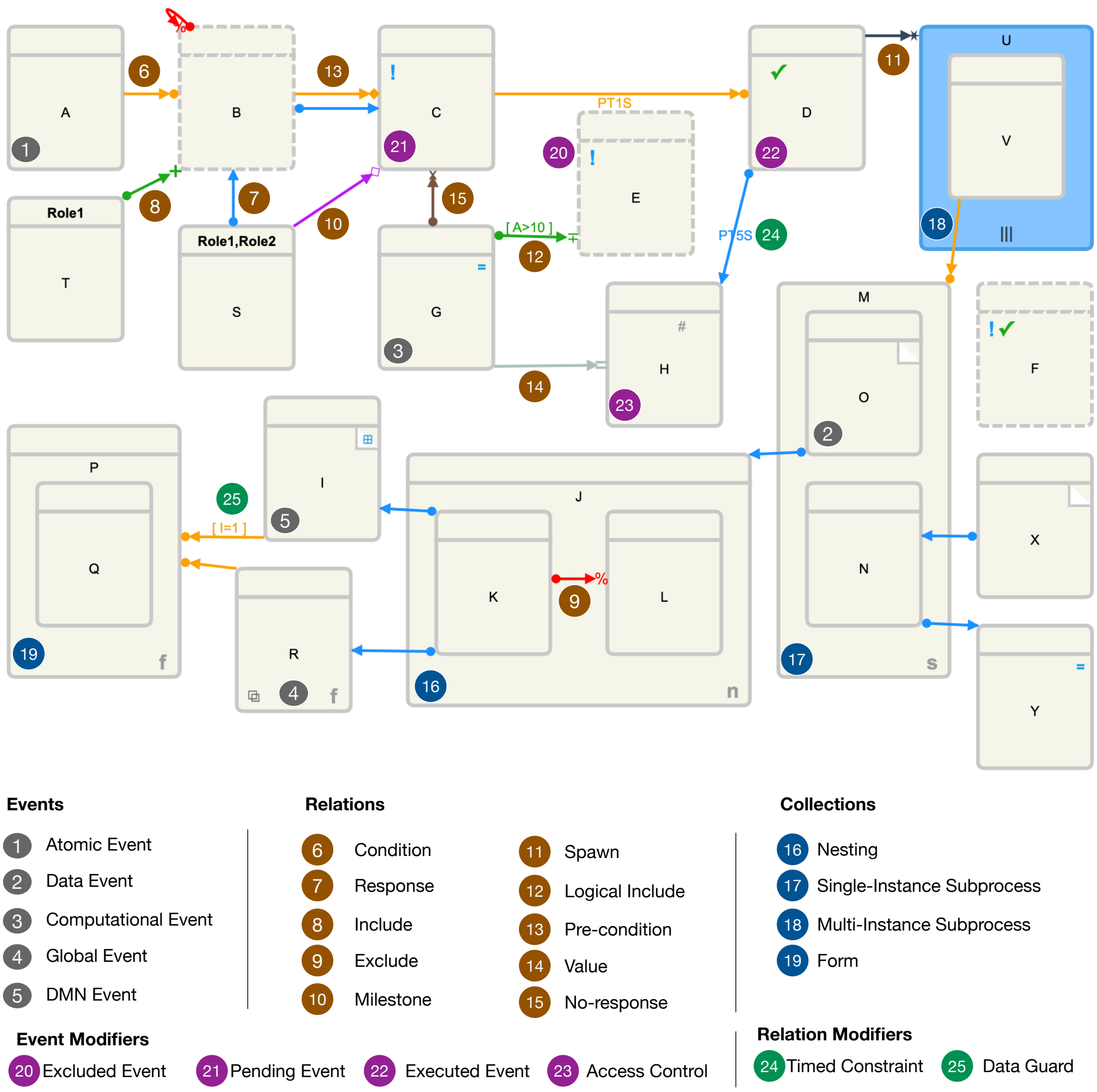

Figure 2: DCR graphs: notation example

2.2.4. Semantic Transparency

studies the intuitiveness of the notation. An intuitive symbol reduces the cognitive load placed on the reader [8], who can shift the attention to harder-to-decode elements. *Semantically perverse* symbols imply a different/opposite meaning than the concept. *Opaque* symbols have an arbitrary meaning. *Translucent* symbols provide hints to their concept but need initial explanations. Finally, *semantic immediate* symbols are linked with their concept without explanations. We consider this guideline satisfied if all symbols are semantically immediate.

2.2.5. Complexity Management

analyses the techniques used by a notation to manage large quantities of information without overwhelming users [8]. The limitations of human perception and comprehension render diagrams over a specific size less effective to use [16], in particular for novice users that do not have clustering and decoding capabilities to work with highly complex graphs [17]. Two techniques are considered: first, *hierarchy* allows systems to be represented at different levels of detail, allowing modellers to control the complexity at each level. Second, with *modularisation* large diagrams are divided into sub-systems, reducing the number of elements in scope and encapsulating sub-systems as single elements. This guideline will be fully satisfied if both techniques are present, partially satisfied if at least one technique is present, and not satisfied if none can be found.

2.2.6. Cognitive Integration

studies combinations of diagrams and their understandability. Homogeneous integration occurs when separate diagrams capture different areas of the system/different levels of abstraction. Heterogeneous integration presents varying views of an area of interest. Cognitive integration includes *conceptual integration* as techniques to link and contextualise diagrams, and *perceptual integration*, which describes navigation between diagrams and the user's ability to find their destination. This guideline will be satisfied if both techniques are present in the notation.

2.2.7. Dual Coding

studies specific use cases where textual labels objectively improve the effectiveness of the notation. PoN suggests that text should always be used as a supplement that provides further clarification to existing encodings realised by one of the eight visual variables [8]. In particular, PoN supports the use of texts in two cases: first, by adding *annotations* directly to diagrams, users can obtain detailed descriptions without referring to documentation or another external file. Second, in the usage of *hybrid coding* (text+graphics), users can reinforce and complement the meaning of symbols. This guideline is fully satisfied if both techniques are present in the notation, partially satisfied if one is implemented, and violated otherwise.

2.2.8. Graphic Economy

large symbol vocabularies are directly related to a decrease in notation effectiveness [18]. Moreover, large vocabularies are challenging for novice users who may be unable to understand the diagrams without constant interaction with a legend or documentation. Thus, shifting some aspects of the notation to textual descriptions should be considered. Strategies to reduce complexity include: 1) reducing semantic complexity, 2) introducing symbol deficit, and 3) increasing visual expressiveness. We consider this guideline partially satisfied if the symbol vocabulary is below Moody's threshold of 40 elements [8] and at least one graphic economy technique has been implemented and fully satisfied if all techniques are present.

2.2.9. Cognitive Fit

studies how audiences require different representations of information for different types of tasks. Novice users are susceptible to large vocabularies, whereas experts can cluster and parse larger numbers of information as one. Providing an extended vocabulary for novices/experts is not enough. Notation variants need to be chosen in a sensible way for each user type (e.g., using discriminated and transparent symbols in novice variants). Moreover, extending the vocabulary to cater for different types of users can reduce the effectiveness for each user type in isolation [8] This guideline is partially satisfied if the notation provides different representations depending on user types, and fully satisfied if the variants of the language are discriminate and implement design specific design principles for each user type.

2.3. SEQUAL

The SEQUAL framework [19] defines guidelines for the evaluation of modelling tools. We use SEQUAL's version for business process models [9], which considers seven dimensions. We focus on those concerning graphical notations:

2.3.1. Empirical Quality

collects the traits of visual or textual communication used in models and their empirical impact on understandability. SEQUAL highlights that extensive use of colour might confuse a large minority of subjects with colour vision deficiency. Therefore, its range should be limited to seven colours. Moreover, certain colours are designed to convey a specific type of information. Red, for instance, is considered to have a negative connotation. Finally, SEQUAL suggests a consistent use of a colour code to limit the association of different meanings to the same variable. Other considerations in this dimension include emphasis mechanisms such as changes in the size variable and the use of typographical emphasis in text labels.

2.3.2. Social Quality

this dimension studies the level of agreement on three dimensions: knowledge, interpretation, and model. It evaluates each dimension as an absolute or a relative agreement. We will focus on two aspects of social quality. First, the addition of *naming conventions* as an aid for users to identify objects thanks to familiar forms. Second, *language documentation* - clear documentation of language concepts, grammar and visual notation improves the users' understanding of the language and inherently increases the chances of agreement in individuals' comprehension of provided models.

3. Cognitive Effectiveness of DCR against PoN

We carried out a systematic analysis of DCR against the PoN and SEQUAL dimensions described in the previous section. The analysis included the definition of metrics for the analysis (that are not provided in the frameworks studied) followed by an independent analysis, that was revised until reaching an agreement with a second author, that has five years of experience teaching DCR graphs.

3.0.1. Semiotic Clarity

Events and collections are depicted using the same shape variable. The adherence to the 1:1 correspondence in the context of events is not met as the notation is overloaded to represent both the existence of events and the effects of both computations and data events. Figure 2 presents two examples: data events O and X have different types but are represented equally, and computational events G and Y compute different expressions. Continuing with relations, DCR contains ten relation types, and each type is depicted by a unique combination of a coloured arrow and an arrowhead (c.f. Figure 2). Note that while the semiotic clarity principle is observed, the symbols used to differentiate relations are very similar (c.f.: perceptual discriminability). Overall, we found that DCR notation presents a symbol deficit corresponding to the semantic concepts used. However, we consider this choice justified and supported by the graphic economy principle, restricting the vocabulary in favour of understandability.

3.0.2. Perceptual Discriminability

Events and relations are differentiable using the shape variable. However, a slight modification of the rectangular shape is used to differentiate events and (sub)processes (see legends 1, 18 and 19 in Figure 2). At the same time, events get their borders, icons and colours modified to denote a change in their type, their connection with other graphs, and their execution state (see legends 1, 2, 3, 4, 5, 18, & 20). While events, processes and sub-processes are identified by their shape and colour variables, colour is user-editable, removing it from the determinant variables. Focusing on shape only, all types of events use rounded squares, making them indistinguishable. Notice that while events may use text annotations to describe their semantics, this has zero visual distance according to PoN. Relations are depicted using shape and colour variables including a customised arrowhead. However, arrowheads are small and very similar in some cases (e.g., $+$ for include and $\pm$ for logical include). The small size of the symbols makes their effect on the user's pre-attentive processing of the elements debatable. Moreover, some occurrences of symbol overload are identified where a symbol (e.g., a pre-condition →⧫) has the same behaviour as the combination of others (e.g., a milestone →◇ and a condition →•). Colour is the second variable used to distinguish rules. In most cases, the used colours are far in their colour spectrum, aiding distinguishability. We can document two pairs of relations where colour variables are the same: logical include →± & include →+, and pre-condition →⧫ & condition →•. These rules share similarities from a semantics perspective which is likely the motivation behind their identical colour representation.

3.0.3. Semantic Transparency

DCR graphs is a conceptual modelling notation, thus, its notation does not include real-life objects. Entities are represented by geometric shapes with additional modifiers. The choice of modifiers ranges from translucent to perverse:

- Semantic transparent: an executed event is visualised with a green checkmark. This combination of shape and colour is attributed to successful execution.
- Semantic opaque or perverse: a data event is depicted with a “turning page” modifier, whose visual effect may be confusing as users may interpret it as an event having multiple steps or a document. Moreover, the use of the same notation for relations (the same arrow) may be perverse as users might associate the same meaning to all relations. Relations in DCR graphs denote constraints with different semantics, however, their representation as arrows might confuse novice users with a process modelling background, where arrows represent direct-follow-relations (e.g., BPMN [20]).

On several occasions, event modifiers use text in their visual notation. For instance, grouping types (forms, sub-processes, and nesting) are highlighted by the presence of the initial letter of its type. While PoN does not explicitly evaluate the use of text in semantic transparency, we consider the choice of letters as an example of semantically perverse objects when considering the cultural background. Initial letters may generate recall for native English speakers only and create confusion for other audiences.

Relations in DCR graphs describe behavioural constraints, and the choice of representation via arrows might be misleading as explained above. The other variable used in their representation is colour. The use of green and red colours for the include and exclude relations provides a hint of a positive or a negative effect of the rule; nonetheless, they can be classified as semantically translucent at best. Other colours, such as brown and violet, are clear examples of semantically opaque elements. Similarly, the choice of characters as arrowheads (see uses of “→%" in Figure 2) are classified as semantically opaque. Finally, relations can combine events and event collections (see Figure 3, left-hand side). They use a spatial arrangement to express the relationship between the child and the parent event. Using the principle of the spatial enclosure is known to be highly semantically transparent compared to connecting lines [8].

3.0.4. Complexity Management

We observe one complexity management mechanism: event nesting[3]. Nesting defines a visual hierarchy between a container and its enclosed events (see Figure 3). While nesting does not remove the essential complexity of the model, it does reduce accidental complexity (that is, the complexity originating from the graphical representation), and it can improve the clarity of the model.

3.0.5. Cognitive Integration

Homogeneous integration is present via the introduction of global events that synchronise the execution network of DCR models. While there is a formal execution engine capable of integrating networks of DCR graphs [13], the graphical notation provides only symbols to denote when an event is shared between graphs, lacking explicit constructs to denote networks of graphs. Moreover, DCR presents several remotely similar heterogeneous integration mechanisms. With

[3]The notation mentions an additional type “transactional subprocesses” but its usage could not be replicated in our tests

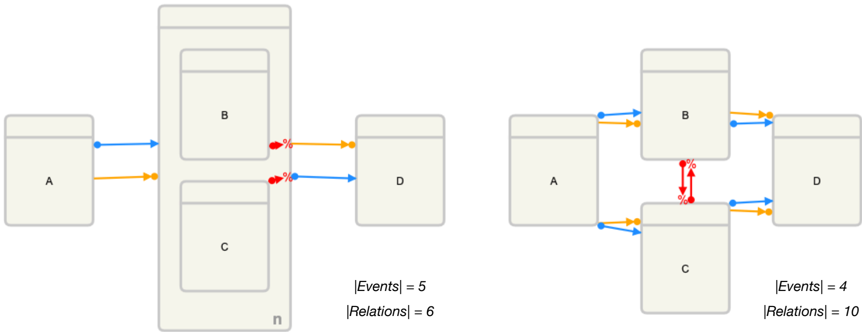

Figure 3: Complexity management: the use of the nesting construct (left-hand side) reduces the number of relations without affecting expressiveness

the Process Highlighter [21]. each processing element is linked with textual annotations. Other integrations include the link between simulation and DCR notation.

DCR networks would play a significant role in developing complexity management mechanisms. Developing a layer-based set of homogeneous diagrams with varying degrees of detail, similar to a process architecture diagram [22] will help both conceptual and perceptual integration. Notably, having a "root" diagram provides the user with a starting point and navigation capabilities to deeper layers of the process hierarchy [8]. Finally, we do not observe navigation capabilities for DCR graphs. While in imperative process notations there is a clear notion of start/end events, in declarative notations like DCR, the ability to execute events in DCR is entirely dependent on the marking and the constraints of a given event. This complicates the definition of layouts that could be fitted to users' reading behaviour, requiring users to perform a full graph exploration to orient themselves on what can be done and how to achieve their goals.

3.0.6. Visual Expressiveness

Shape, and in the case of relations, colour, is the sole information-carrying variable used by DCR. While colour should not be the only deciding variable, it is highly effective in reducing cognitive load. Other variables such as texture, horizontal and vertical position and size, are either used seldom (e.g., dashed texture for excluded events) or are free variables. The next variable used is text. However, PoN discourages the use of text as means to encode semantic elements: while textual elements allow to precisely label objects, they increase the cognitive load on the users' side and provide no value during pre-attentive processing. Moreover, labels are often crucial on the sentence level where they distinguish individual object instances. This cannot be achieved if the notation uses text labels to identify symbol types.

3.0.7. Dual Coding

The principle of dual coding is covered in annotations and symbol/text combinations. Annotations are not directly represented in the notation, and users need to shift focus onto different artefacts in order to integrate additional information. However, while the notation does not support annotations, tool support provides users with integrations with source information and event explanations, that have a positive impact on process model comprehension [23]. Regarding hybrid text/symbol combinations, the use of hybrid representations in DCR notation is limited to timed relations and guards, however, since both annotations are using the same visual variables (colour), it is possible that users find lower perceptual discriminability between them.

3.0.8. Graphic Economy

By March 2022, DCR contains a vocabulary of thirty-two distinct visual elements. This vocabulary is below PoN's threshold for excessive vocabulary. DCR introduces a symbol deficit to reduce some of the complexity in the graph (c.f. Section 3.0.1). No partition of semantic complexity is evidenced. As mentioned in Section 3.0.2 shape is the sole information-carrying variable in DCR and adding multiple visual variables to differentiate concepts will increase visual expressiveness, thus increasing perceptual discriminability.

3.0.9. Cognitive Fit

DCR presents a relation palette for novices and another for experts. Currently, there is no clear segmentation between the two modes, and the relations offered to novices add to the constraints offered to experts. When considering the representational medium, sketches of DCR models in whiteboards renounce the use of colour variables. This change is compensated by adding emphasis to the shape variable of constraints, for instance, by modifying the sizes of symbols in constraints to facilitate perceptual discriminability.

4. Cognitive Effectiveness of DCR against SEQUAL

4.0.1. Empirical Quality

We observe a mixed application of SEQUAL guidelines: relations use a fixed palette, and events and subprocesses have editable colours. Such liberty allows users to endow events with different meanings than the intended one (e.g., events can be coloured as subprocesses and vice-versa). The use of red-green colours to represent exclusion-inclusion constraints relates to SEQUAL guidelines to the transparent colour coding. Furthermore, we observe that the size variable is fixed for events, and there is little use of typographical emphasis tools. Bold typefaces are used in the description of roles. This differentiation likely helps users separate ownership of events from their content. Positioning of labels is used for guards and delays. However, positioning is likely to be insufficient once relations link events in a vertical plane (c.f. relation $D \bullet\!\!\rightarrow H$ in Fig 2), and, in some cases, perverse to the understanding of the graph (e.g., the colour label overlaps with a coloured relation). With respect to event positioning, DCR does not offer a guideline on how to layout them, but it is likely that experts define their own layout scheme following reading conventions [10].

Dimension	Summary of the analysis
Physics of Notations	
Semiotic clarity	Partially satisfied (symbol deficit)
Perceptual discriminability	Partially satisfied
Semantic transparency	*Transparent:* executed events, spatial enclosure in event collections. *Translucent:* colour choice for include/exclude relations. *Opaque or perverse:* complex events (subprocesses, nesting, transactional, data), colour code in other relations.
Complexity management	Partially satisfied: reduced accidental complexity but limited modularization support.
Cognitive integration	Partially satisfied: homogeneous integration is absent. Heterogeneous integration is evidenced. Conceptual & perceptual integration capabilities are absent.
Visual expressiveness	Partially satisfied. The shape is the only primary variable, but the colour variable is bound in some cases.
Dual coding	Partially satisfied (text annotation is not supported, hybrid coding is supported in special relations).
Graphic economy	Partially satisfied. Visual vocabulary is not excessive and symbol deficit reduction techniques are. A partition of semantic complexity is absent.
Cognitive fit	Partially satisfied (novice/expert relation palette is suggested, but notation variants are not discriminable).
SEQUAL	
Empirical quality	Partially satisfied. Manageable colour spectrum, but inconsistent colour usage. Colour choices are not recommended for colour vision deficiency users. Limited use of typographical emphasis.
Social quality	Not satisfied.

Table 1
Cognitive dimensions in DCR: summary of findings in Sections 3 and 4.

4.0.2. Social Quality

Naming conventions aid users in identifying objects thanks to familiar forms (e.g., a linguistic pattern "*verb+object*" signifies an activity performed in a process). We noticed that such a pattern is not always used in DCR. Events are placeholders for both actions that participants perform, and external sources of change (e.g., a deadline is reached, a message is received [10]). The differentiation of controllable and uncontrollable events has been studied in literature [24] and their graphical representations are different in other process modelling notations [20]. This overload of the notation for two different concepts might likely affect users' understandability. The application of linguistic patterns will probably help if there is a clarification of what constitutes an event in DCR. Regarding language documentation, the only source of documentation comes from a company website[4], and there is no standardised documentation. The volatility and likely change of the sources inhibit the generation of a common understanding and limit the creation of other tools that can support the notation.

[4]https://documentation.dcr.design/

Nr.	Improvement suggestion	Concerning dimensions
1	Relations between events should use more visual variables than shape and colour.	Perceptual discriminability, visual expressiveness, empirical quality
2	Other symbols different than arrows should be used in order to facilitate discriminability and understanding of relations.	Semantic transparency, perceptual discriminability
3	Adding symbols to denote delays, timeouts and data guards will facilitate their identification and differentiation.	Perceptual discriminability, semantic transparency
4	Adding additional symbols to denote controllable and uncontrollable events will support model understanding.	Visual expressiveness, graphic economy, social quality
5	A colour-neutral version of DCR graphs will support colour-blinded minorities and relate to analogue versions of DCR models	Empirical quality, social quality
6	Colours and shapes should not be reused to denote the different types of relations.	Perceptual discriminability
7	Different types of events (e.g., nesting, form, computation, subprocess) should be identifiable via a change in the retinal variables	Perceptual discriminability
8	Events and event collections (nesting, and the different types of subprocesses) should differ in the choices of the shape variable.	Perceptual discriminability
9	Novice/expert modes need to be clearly separated, thus reducing the number of constraints shown at each mode.	Cognitive fit
10	High-level, non-executable representations of DCR graphs might help users to navigate over complex process architectures.	Complexity management
11	Adding the principle of cognitive integration for networks of graphs may help in the contextualisation of process architectures	Cognitive integration
12	Users should be able to complement the graphical notation with their own comments & annotations.	Dual coding
13	Provide a standardised reference model for the graphical notation.	Social quality

Table 2
Improvement suggestions according to cognitive dimensions

5. Implications for Research and Practice

We suggest a roadmap for the improvement of DCR notation based on the analysis presented. It suggests that notation extensions should make an effort in designing perceptually differentiable elements by novice users, using colours that are inclusive to visually impaired subjects. Moreover, it suggests the importance of supporting novices in their learning process by allowing them to annotate their models (dual coding), navigate complex models (complexity management) and even provide model transformation techniques between expert and novice views (cognitive fit). These suggestions, as well as the suggestions in Table 2, can be operationalised by tool developers, or used as research hypotheses to be further tested via user studies. Moreover, the analysis executed in perceptual discriminability, semantic transparency and social quality can be transposed to the DECLARE process modelling notation [2], as it uses similar symbols to denote events and relations as DCR graphs.

6. Related Work

While previous works on the cognitive effects of *imperative* process modelling notations using PoN and SEQUAL exist [25, 9], to the best of our knowledge, this paper presents the first systematic analysis of a declarative process modelling notation using general cognitive effectiveness frameworks. Other works use PoN partially. PoN has been used to study the semiotic clarity of CMMN [26]. Figl et al [27] conduct empirical experiments on the semantic transparency of DECLARE models. Their assessment suggesting that the representation of constraints as arrows is semantically perverse to users is in line with our analysis in Sec. 3.0.3. PoN's graphic economy, cognitive integration, complexity management and semiotic clarity dimensions have been used to drive novel graphical representations of DECLARE [28]. With respect to SEQUAL, the role of layout and edges in the understandability of imperative process models [29]. Regarding notations for declarative processes, Colombo Tossato et al. [30] proposed a visual notation for norms, conditional and defeasible rules including a larger set of visual variables (texture and shape). The representations of relations might be especially useful for DCR as the semantic concepts are similar. Finally, Andaloussi et al. [10] use personal construct psychology to define a quality framework from experts in DCR graphs.

7. Conclusion

We provided a systematic analysis of the cognitive effects of DCR graphs in the context of two general frameworks for the understandability of notations and provided a set of 13 guidelines that could improve the cognitive effectiveness of the notation. We believe that the implementation of such guidelines will benefit DCR graphs, as PoN principles have been demonstrated to positively influence a notation's perceived usefulness [31]. It is worth mentioning that even if PoN and SEQUAL provide guidelines for evaluation, they gave no operational support [32]. Even when our paper describes the analysis metric used, a purely unbiased and objective observation is not possible using the current setup unless users are involved. Moreover, some of the guidelines may have counterarguments and further empirical analysis needs to be conducted. This is the case for modularization: the application of modularization might require an additional mental effort due to the need to switch between sub-processes and integrate information [33]. In future work, we would like to complement this work with larger user studies and extend it to other declarative notations, such as DECLARE and CMMN.

Acknowledgements We would like to thank Amine Abbad Andaloussi and Barbara Weber for their comments in earlier versions of this paper.

References

[1] D. Fahland, D. Lübke, J. Mendling, H. Reijers, B. Weber, M. Weidlich, S. Zugal, Declarative versus imperative process modeling languages: The issue of understandability, in: EMMSAD, Springer, 2009, pp. 353–366.

[2] M. Pesic, H. Schonenberg, W. M. Van der Aalst, Declare: Full support for loosely-structured processes, in: EDOC, IEEE, 2007, pp. 287–287.

[3] OMG, Case management model and notation, version 1.1, 2016.

[4] T. T. Hildebrandt, R. R. Mukkamala, Declarative event-based workflow as distributed dynamic condition response graphs, arXiv preprint arXiv:1110.4161 (2011).

[5] L. H. Norgaard, J. B. Andreasen, M. Marquard, S. Debois, F. S. Larsen, V. Jeppesen, Declarative process models in government centric case and document management, in: BPM (Industry Track), volume 1985 of *CEUR*, CEUR-WS.org, 2017, pp. 38–51.
[6] T. Hildebrandt, R. R. Mukkamala, T. Slaats, Nested dynamic condition response graphs, in: FSEN, Springer, 2011, pp. 343–350.
[7] H. A. López, S. Debois, T. Slaats, T. T. Hildebrandt, Business process compliance using reference models of law, in: FASE, volume 12076 of *LNCS*, Springer, 2020, pp. 378–399.
[8] D. L. Moody, The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering, IEEE TSEN 35 (2009) 24.
[9] J. Krogstie, Quality of business process models, in: Quality in Business Process Modeling, Springer, 2016, pp. 53–102.
[10] A. A. Andaloussi, C. J. Davis, A. Burattin, H. A. López, T. Slaats, B. Weber, Understanding quality in declarative process modeling through the mental models of experts, in: BPM, volume 12168 of *LNCS*, Springer, 2020, pp. 417–434.
[11] R. Strømsted, H. A. López, S. Debois, M. Marquard, Dynamic evaluation forms using declarative modeling, in: BPM (Dissertation/Demos/Industry), volume 2196 of *CEUR*, CEUR-WS.org, 2018, pp. 172–179.
[12] T. T. Hildebrandt, H. Normann, M. Marquard, S. Debois, T. Slaats, Decision modelling in timed dynamic condition response graphs with data, in: BPM Workshops, Springer, Cham, 2022, pp. 362–374.
[13] S. Debois, H. A. López, T. Slaats, A. A. Andaloussi, T. T. Hildebrandt, Chain of events: modular process models for the law, in: iFM, Springer, 2020, pp. 368–386.
[14] D. Basin, S. Debois, T. T. Hildebrandt, In the nick of time: Proactive prevention of obligation violations, in: CSF, IEEE, 2016, pp. 120–134.
[15] S. Debois, T. T. Hildebrandt, T. Slaats, Replication, refinement & reachability: complexity in dynamic condition-response graphs, Acta Informatica 55 (2018) 489–520.
[16] J. Sweller, Cognitive load theory, in: Psychology of learning and motivation, volume 55, Elsevier, 2011, pp. 37–76.
[17] G. A. Miller, The magical number seven, plus or minus two: Some limits on our capacity for processing information., Psychological review 63 (1956) 81.
[18] A. v. Klopp Lemon, O. v. Klopp Lemon, Constraint matching for diagram design: Qualitative visual languages, in: International Conference on Theory and Application of Diagrams, Springer, 2000, pp. 74–88.
[19] O. I. Lindland, G. Sindre, A. Solvberg, Understanding quality in conceptual modeling, IEEE software 11 (1994) 42–49.
[20] OMG, Business Process Model and Notation (BPMN), Version 2.0, 2011.
[21] H. A. López, S. Debois, T. T. Hildebrandt, M. Marquard, The process highlighter: From texts to declarative processes and back, in: BPM (Dissertation/Demos/Industry), volume 2196 of *CEUR*, CEUR-WS.org, 2018, pp. 66–70.
[22] M. Dumas, M. La Rosa, J. Mendling, H. A. Reijers, et al., Fundamentals of business process management, volume 1, Springer, 2013.
[23] B. Aysolmaz, F. N. Cayhani, H. A. Reijers, Narration as a technique to improve process model comprehension: Tell me what I cannot see, in: CAiSE, volume 13295 of *LNCS*, Springer, 2022, pp. 407–422.

[24] M. Broy, M. Wirsing, On the algebraic specification of nondeterministic programming languages, in: CAAP, volume 112 of *LNCS*, Springer, Berlin, Heidelberg, 1981, pp. 162–179. doi:10.1007/3-540-10828-9_61.
[25] N. Genon, P. Heymans, D. Amyot, Analysing the cognitive effectiveness of the bpmn 2.0 visual notation, in: SLE, Springer, 2010, pp. 377–396.
[26] M. K. Bule, G. Polančič, J. Huber, G. Jošt, Semiotic clarity of case management model and notation (CMMN), Computer Standards & Interfaces 66 (2019) 103354.
[27] K. Figl, C. D. Ciccio, H. A. Reijers, Do declarative process models help to reduce cognitive biases related to business rules?, in: ER, volume 12400 of *LNCS*, Springer, 2020, pp. 119–133.
[28] M. Hanser, C. D. Ciccio, J. Mendling, A novel framework for visualizing declarative process models, in: ZEUS, volume 1562 of *CEUR*, CEUR-WS.org, 2016, pp. 5–12.
[29] A. Burattin, V. Bernstein, M. Neurauter, P. Soffer, B. Weber, Detection and quantification of flow consistency in business process models, SoSym 17 (2018) 633–654.
[30] S. Colombo Tosatto, G. Boella, L. v. d. Torre, S. Villata, Visualizing normative systems: An abstract approach, in: International Conference on Deontic Logic in Computer Science, Springer, 2012, pp. 16–30.
[31] K. Figl, M. Derntl, The impact of perceived cognitive effectiveness on perceived usefulness of visual conceptual modeling languages, in: ER, Springer, 2011, pp. 78–91.
[32] D. v. d. Linden, A. Zamansky, I. Hadar, How cognitively effective is a visual notation? on the inherent difficulty of operationalizing the physics of notations, in: Enterprise, Business-Process and Information Systems Modeling, Springer, 2016, pp. 448–462.
[33] S. Zugal, Applying cognitive psychology for improving the creation, understanding and maintenance of business process models, Ph.D. thesis, University of Innsbruck, 2013.

A. DCR graphs metamodel

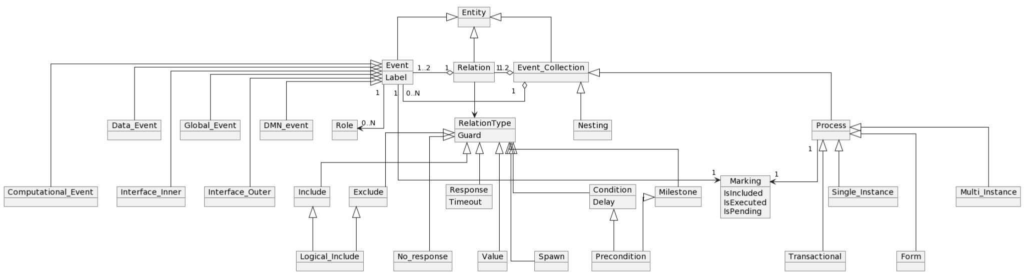

Figure 4: DCR graphs metamodel