Systematic Analysis and Evaluation of Visual Conceptual Modeling Language Notations

Dominik Bork and Dimitris Karagiannis and Benedikt Pittl

Published in:

2018 12th International Conference on Research Challenges in Information Science (RCIS), IEEE, pp. 1-11, 2018

The final version is available online here:
https://ieeexplore.ieee.org/abstract/document/8406652/

OMiLAB®
www.omilab.org

Systematic Analysis and Evaluation of Visual Conceptual Modeling Language Notations

Dominik Bork
Research Group Knowledge Engineering
University of Vienna
Waehringer Street 29
1090 Vienna, Austria
Email: dominik.bork@univie.ac.at

Dimitris Karagiannis
Research Group Knowledge Engineering
University of Vienna
Waehringer Street 29
1090 Vienna, Austria
Email: dk@dke.univie.ac.at

Benedikt Pittl
Research Group Knowledge Engineering
University of Vienna
Waehringer Street 29
1090 Vienna, Austria
Email: benedikt.pittl@univie.ac.at

***Abstract*—In systems analysis and design it is common to refer to some widely used de-facto industry standards like Unified Modeling Language (UML) and Business Process Model and Notation (BPMN). Albeit the wide adoption of such standard modeling languages, only limited research focuses on the techniques in which these standards are specified and the quality they provide. Most research focuses on case studies of applying standards, ways of extending standards to domain-specific requirements, e.g., by means of profiling, or evaluations of single modeling languages, e.g., using questionnaires or semiotic theories. By contrast, this paper critically reflects on the current state of modeling standards with a focus on their graphical representation (notation). The contribution of this paper is threefold: First, a systematic analysis is performed thereby investigating how different modeling standards specify notational aspects. Second, an evaluation is performed by applying Moody's Physics of Notation theory to the identified standards. Third, based on the findings, recommendations are given to improve modeling standard specifications in the future w.r.t. their notational aspects.**

I. Introduction

Visual modeling languages such as the Business Process Modeling and Notation (BPMN) and the Unified Modeling Language (UML) are widely used in academia and industry. Enterprises usually make use of thousands of visual models [1] for systems analysis and design. Most visual modeling languages are introduced in overarching specification documents. These specifications are therefore vital for: (i) modelers, interested in understanding and applying a modeling language, (ii) researchers, aiming to evaluate and adapt modeling languages, e.g., to domain-specific purposes [2], and (iii) tool vendors, interested in developing a modeling tool for a modeling language.

When it comes to the adoption of a modeling language by a modeler and the expressive power of the created models - i.e., the understanding of conceptual models by human beings - notational aspects play an important role. While the importance of modeling language specifications is obvious, to the best of our knowledge, notation specification techniques were never systematically analyzed. Most modeling language specifications concentrate on the syntactic aspects (also referred to as **abstract syntax** in computer science modeling languages [3]), i.e., the definition of the meta concepts, their semantics, and the allowed relationships between them. While the syntax is vital for language specifications, especially when considering visual modeling languages, other aspects such as the language's notation (also referred to as **concrete syntax** in computer science modeling languages [3]) are also important. However, support for those responsible for creating meaningful visualizations is scarcely given [4]. "Although the current language specification recognizes the importance of defining a visual notation carefully, it does so by relying on common sense, intuition and emulation of common practices, rather than by adopting a rigorous scientific approach" [5]. Notational aspects also play an important role when thinking about possibilities of extending a modeling language to reflect domain-specific requirements (cf. [6]). Recent research investigated for example the applicability of the ArchiMate modeling language in a case study of modeling enterprise ecosystems [7]. The authors conclude that the given standard notation lacks in expressiveness and adequacy. Research presented in [8] proposed the usage of crowd-sourcing as a means to improve the notation of domain-specific conceptual modeling languages.

This paper targets three objectives: First, a systematic analysis of how modeling language specifications introduce notational aspects is performed, giving an overview of different specification techniques. Second, an evaluation of the visual expressiveness of the notation is performed by applying Moody's Physics of Notation theory [10]. Based on the findings, third, recommendations for how to improve future modeling language specifications with respect to notational aspects are given. The paper will provide concrete guidelines on *what* notational aspects to specify, and *how* to specify them most appropriately. This paper will also critically reflect to what extend existing knowledge about semiotics and the Physics of Notation are respected in standard visual modeling languages and to which extent Bertin's Physics of Notation are still valid for today's conceptual modeling languages.

The contribution of this paper is of benefit for three kinds of people: *researchers*, aiming to create a specification for a visual modeling language; *maintaining institutions*, intending to improve existing specifications; and *modelers*, interested in learning how to comprehend relevant information of conceptual models.

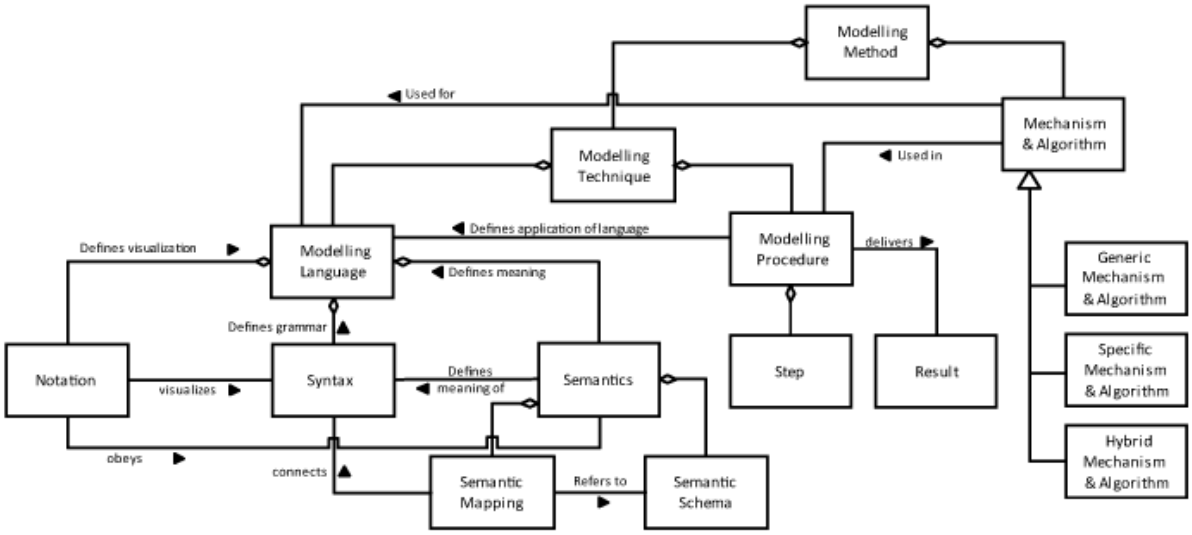


Fig. 1: Components of modeling methods (adapted from [9])

The remainder of this paper is structured as follows: In Section II foundations of modeling languages and visual expressiveness are introduced. Moreover, relevant related work is presented. Section III summarizes the research questions and the applied research methodology. The results of the analysis are then presented in Section IV. A discussion of the key findings and recommendations for improving modeling method specifications are given in Section V. The paper closes with concluding remarks and some future research directions in Section VI.

II. Background and Related Work

A. Terminological Foundation

As the scientific community uses different terms to refer to elements of modeling languages, a terminology which will be used in the following is first established. The most important terms are introduced by the generic components of conceptual modeling methods [9] as visualized in Fig. 1. Accordingly, modeling methods are composed of a *modeling language*, a *modeling procedure*, and *mechanisms & algorithms*. Modeling languages constitute a prerequisite for the latter two. Consequently, most specifications focus on the modeling language part, more precisely on the syntax of a modeling language - often referred in UML/OMG-related literature as abstract syntax [3]. The syntax of a modeling language can be specified on different levels of formality [11]. One of the most used techniques to formally specify the syntax is by using visual metamodels [11], [12]. Such metamodels usually comprise constructs and relationships between these constructs. In addition, a modeling language specification comprises notation - often referred in UML/OMG-related literature as concrete syntax [3] - and semantics. Notation refers to the graphical representation of syntactic elements of the modeling language whereas semantics specify the meaning of them. Besides the modeling language, a comprehensive modeling method also comprises a modeling procedure and mechanisms & algorithms. The modeling procedure specifies the way how modelers apply a modeling language (steps) in order to create valid models (results). Finally, mechanisms & algorithms are used by in the modeling procedure to work on the created models. Model transformations, validity checks, simulations, or queries are examples for such mechanisms & algorithms.

Visual expressiveness, a metric introduced in [10], is heavily used in the scientific community for the evaluation of the notation of modeling languages (see e.g., [5]). The metric itself is build upon Bertin's visual alphabet [13], visualized in Fig. 2. The alphabet comprises eight variables which can be used to design a notation. The variables are partitioned into *planar variables* and *retinal variables*. The former addresses the spatial dimensions (horizontal and vertical) whereas the latter is concerned with the retinal image [13]. The variables are self-describing to a certain degree - for more details please see [10], [13]. Moody [10] introduces a set of nine principles for designing cognitively effective visual notations. The survey at hand focuses on an analysis of the notation using only the principle *visual expressiveness*. This is because of two reasons: (i) the visual expressiveness is non-ambiguous, i.e. it is free of subjective perceptions of the evaluator, and (ii) most of the specifications do not have strict notation guidelines, hindering a detailed analysis based on the specification. Indeed, the visual alphabet focuses on objects, not on connectors. Consequently, this survey focuses on analyzing the notation of objects/constructs.

The visual expressiveness represents the number of variables of Bertin's visual alphabet which are respected in a modeling language [10]. The maximal visual expressiveness is 8, the lowest visual expressiveness is 0. Usually, the higher the visual expressiveness, the better is the perception of a diagram by a human being.

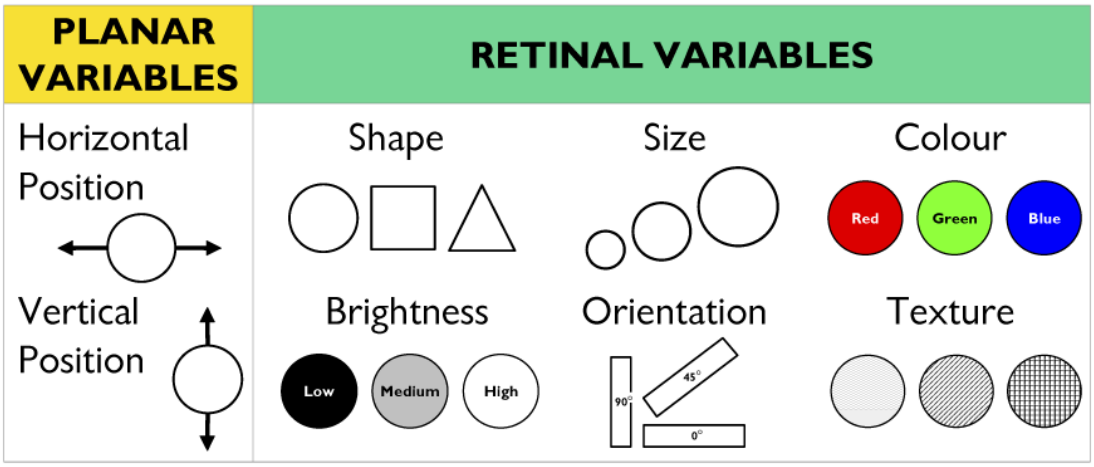


Fig. 2: Bertins visual alphabet - taken from [10, p. 761]

B. Related Work

[14] provide a longitudinal study on the evolution of different process modeling languages based on the Bunge-Wand-Weber (BWW) representation model [15]. Thus, the goal of their research was to comprehensively evaluate the evolutionary character of languages as well as strengths and weaknesses of the process modeling languages as a whole. By referring to the Bunge-Wand-Weber representation model, the focus is on the ontological aspects (such as construct deficit, construct overload, construct redundancy, and construct excess) of a modeling language. Thus, using BWW, one is able to evaluate the syntactic aspects of a modeling language - not the notational ones. Similar research, has been performed by [16] where the authors apply the BWW good decomposition model to evaluate decompositions of complex business process models.

In a similar vein, research exists that targets the analysis of semantic and syntactic consistency requirements between modeling languages [17], [18], [19] and the proposal of consistency management approaches [20]. Here is where notational aspects are of minor importance, as the authors focus on syntactic and semantic issues. Work that solely analyzes the syntactic expressiveness of modeling languages is also broadly given - see e.g., the work of [21] that analyzes the capabilities of expressing unary relationships with the UML.

In [22], the authors evaluate the UML 2.0 family of languages. While being limited to the UML, the research applies a set of principles, therefore realizing a multi-dimensional evaluation based not only on visual expressiveness but also on *semiotic clarity*, *perceptual discriminability*, *perceptual immediacy*, and *graphic parsimony*. Most of these principles are subjective by nature, the research in this paper instead focuses on the objective principle of visual expressiveness.

Genon et al. [5] provide a comprehensive evaluation of the cognitive effectiveness of the BPMN 2.0 notation. [6] investigate domain-specific extensions of the BPMN standard. One of their criteria for evaluating the standard conformity of the BPMN extensions is the concrete syntax (i.e., the notation). They conclude that 30,4% of the analyzed BPMN extensions also comprise notational aspects.

Scientific publications which analyze the expressiveness and inconsistencies of selected modeling languages are also available. For example, in [23] inconsistencies of the UML were analyzed. In [24], selected metamodels used for *situational method engineering* were analyzed. However, representation techniques and styles were not analyzed. Moreover, an increasing amount of research can be found on the empirical evaluation of the usability of modeling languages, e.g. [25].

This research instead focuses on three aspects: the techniques used to specify notational aspects, an evaluation of the visual expressiveness of currently used visual modeling languages in computer and information science, and the definition of recommendations to further improve the latter two (i.e., the specification and the visual expressiveness of notational aspects). The intent is not to blame the responsible ones but rather to point to possible improvements by critically reflecting on the status quo. Moreover, researchers and maintaining institutions shall be enabled to reflect on their practices.

III. Research Questions and Research Methodology

For conducting the analysis, we followed a three-phase research method [26], [27] comprising *planning phase*, *conduction phase*, and *result phase* (see Fig. 3). In the planning phase the research questions as well as the inclusion/exclusion criteria are defined. The conduction phase then describes the execution of the analysis. Finally, the results are presented in the result phase. The planning and the conduction phase are described in the following two subsections. The rest of the paper then presents and discusses the result phase.

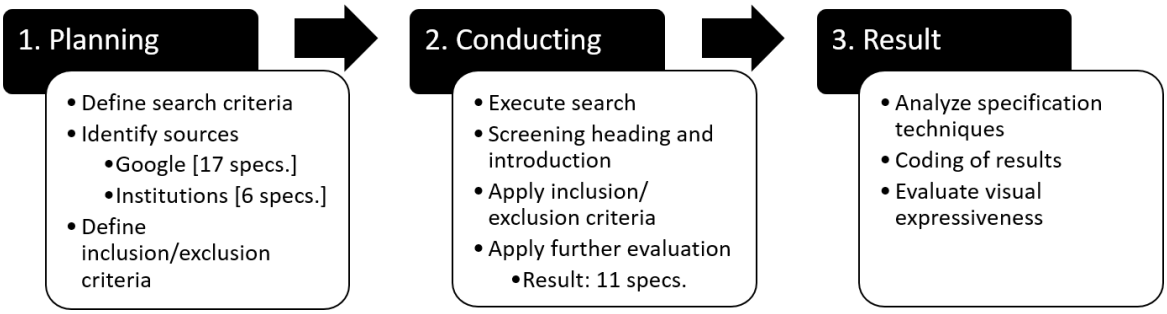


Fig. 3: Search and analysis process followed

A. Planning the Analysis

The term visual modeling language (cf. [9]) is applied for languages which allow the creation of diagrammatic models [28] - called visual models in the following. The goal of this analysis is to answer the following research questions:

- *Which techniques are used for the specification of notational aspects of standard visual modeling languages?*
- *How high is the visual expressiveness of standard visual modeling languages?*

Standard, in this context, refers to visual modeling languages which are heavily used in research and industry - and therefore specified by international institutions such as the OMG [29] or the Open Group [30]. Answering these research questions requires a systematic analysis of modeling language specifications. We considered specification documents which fulfill the following inclusion criteria (IC):

- IC 1: Document is declared as *specification*, *definition*, or *standard*
- IC 2: Document describes a visual modeling language
- IC 3: Document describes notational aspects
- IC 4: Document is freely accessible

Our approach for finding such specifications was twofold: First, we systematically analyzed the specifications published on the websites of OMG and OpenGoup as these two establish the majority of standards for computer and information science visual modeling languages. Second, we conducted a systematic keyword search on www.google.com with the query *(Modeling Language)* AND *(Specification OR Definition OR Standard OR Description OR Documentation)*. The search was conducted in June 2017. For each combination of terms, we analyzed the first ten search result pages. Fig. 3 visualizes the conducted search and analysis process.

We did not focus on scientific databases such as DBLP or Google Scholar in our analysis, as modeling language specifications are usually published by the maintaining institutions. We explicitly excluded modeling languages that have been developed in pure scientific context, e.g., ConML [31], or publications that only propose incomplete specifications or extensions (e.g., UML profiles) of modeling languages. Lastly, we excluded specifications that were not freely available, e.g., ISO/IES 24744, not written in English, and/or not revised since 01.01.2012. Documents which were published before 2012 were ignored because the absence of updates is an indicator that the modeling language is not commonly used anymore and/or not further maintained. The following set of exclusion criteria were applied to ensure that only complete and current modeling languages are considered:

- EC 1: Document is an extension of another language
- EC 2: Document does not specify a notation
- EC 3: Document was published before 01.01.2012
- EC 4: Document is not in English
- EC 5: Document is not freely accessible

B. Conducting the Analysis

The results of the Google keyword search referred several times to pages such as Rosetta Standards[1] or IDEF[2] which enforce a registration before one can access the specifications. If the registration was free of charge we created an account. The Climate Science Modelling Language (CSML)[3] was not accessible and therefore it was not analyzed. The Object Process Methodology (OPM) specification is not freely available. However, a working draft version is accessible which we used for our analysis.

All specifications that met all inclusion criteria were classified as being relevant. Heading and introduction sections were analyzed by the authors in order to ensure that the inclusion criteria are fulfilled. Two specification documents were found for the BPMN: one from the OMG and one from the ISO. Similarly, there exists an UML specification from ISO and one from OMG. In both cases, the specification documents from the OMG were used. The Decision Modeling Notation (DMN) introduces two languages: the Decision Requirements Diagram (DRD) which has a graph-based structure and decision tables which employ a pure tabular notation. Thus, only the DRD was considered relevant. In total, 17 relevant specifications were found. On the website of OMG and OpenGroup, we identified *further* 6 specifications which we considered relevant.

In a second step, the 23 found specifications have been evaluated along the exclusion criteria by reading the introduction section and by cross-reading the following sections.

[1]https://resources.gs1us.org/rosettanet, last accessed: 04.02.2018
[2]http://www.idef.com/, last accessed: 04.02.2018
[3]http://csml.badc.rl.ac.uk/, last accessed: 05.02.2018

TABLE I: Analyzed modeling language specifications

	Version	Maintaining Institution	Domain	Pages	Ref.
ArchiMate	3	Open Group	Enterprise Architecture	181	[32]
BPMN - Business Process Model and Notation	2.02	OMG	Business Processes	532	[33]
CMMN - Case Management Model and Notation	1.1	OMG	Case Management	144	[34]
DMN - Decision Model and Notation	1.1	OMG	Business Decisions	182	[35]
IFML - Interaction Flow Modeling Language	1.0	OMG	User Interactions	144	[36]
LML - Lifecycle Modeling Language	1.1	LML	Systems Engineering	70	[37]
OPM[1]- Object Process Methodology	522	ISO	Automation Systems and Integration	183	[38]
S2ML - System Structure Modeling Language	1.0	OpenAltaRica	Prototypes	53	[39]
UML - Unified Modeling Language	2.5	OMG	Software Systems	794	[40]
URN - User Requirements Notation	Z.151	ITU-T	Requirements Engineering	216	[41]
VDML - Value Delivery Modeling Language	1.5	OMG	Value Creation	150	[42]

[1] Working Draft Specification (*Public Available Specification* - ISO stage 20: Preparatory)

Furthermore, the table of contents has been analyzed to identify the scope of the specifications. This evaluation step classified the specifications using three categories: *complying to the exclusion criteria*, *not complying to the exclusion criteria* and *further evaluation needed*. For example, the System Structure Modeling Language (S2ML) specification required further evaluation as it has a strong focus on the introduced textual language - not on the visual modeling language. Also the business motivation model (BMM) specification required further evaluation because of the limited notation[4]. The specifications belonging to this category were again evaluated independently by the authors and afterwards discussed to come to a decision. Finally, 11 specifications which comply with the search criteria have been identified. Table I provides an overview of them.

The dominance of the OMG is obvious: six out of the 11 specifications are maintained by the OMG. The S2ML is declared as specification, maintained by the OpenAltaRica institution. The other specifications are maintained by the Open Group, Lifecycle Modeling Language (LML) Steering Committee, International Telecommunication Union Telecommunication Standardization Sector (ITU-T) and ISO. The average number of pages for the investigate specifications is approximately 240 with the System Structure Modeling Language (S2ML) having 53 pages as the lower bound and the Unified Modeling Language (UML) having 794 pages as the upper bound.

IV. Notation Specification Techniques and Visual Expressiveness in Standard Visual Conceptual Modeling Languages

The notation of a modeling language, i.e., its graphical visualization, plays an important role for the purpose of communication and understanding [43]. This section first introduces a generic framework for the analysis of notation specification techniques in Section IV-A. Afterwards, a systematic analysis is performed by applying this framework to the visual modeling language specifications. A summary of the findings and the applied notation specification techniques is then given in Section IV-B. Finally, Section IV-C provides an evaluation of the modeling languages with respect to their visual expressiveness. Albeit the general assumption that it is obvious that a visual modeling language specification needs to include notational aspects, the following analysis and evaluation will show the variability in both, notation specification techniques, and visual expressiveness.

A. Notation Analysis Framework

Before analyzing the modeling language specifications it is necessary to establish a set of generic analysis criteria. These criteria are partly derived from the investigated modeling language specifications. Furthermore, a distinction is being made between dynamic and static notation as introduced by [11]. "If the notation of a languages element is fixed at all times, we refer to a static notation, if the notation can change depending on the current state (i.e. attribute value) of the element, we refer to a dynamic notation." [11]. Applying this set of criteria to all identified modeling language specifications enables comparison of the results and eliminates a possible subjective bias. The set of analysis criteria is as follows:

Notation Samples Usage of sample models to introduce the notation of modeling constructs.

Natural Language Notation Guidelines Usage of natural language to introduce the notation of modeling constructs, i.e. by stating that "construct x is represented by a blue rectangle".

Coordinate System Usage of a coordinate system to precisely specify height and width of modeling constructs, and optionally the position of labels.

Dynamic Notation Usage of a notation that reflects current attribute values. For example, a BPMN intermediate event is visualized differently depending on the event type (e.g., message, error, throwing/catching).

Alternative Notation Usage of optional and alternative notations, thereby enabling the user-specific or context-specific customization of graphical visualizations.

Conformance Level Usage of specific conformance levels which enforce the adherence to the non-optional notation guidelines.

[4] http://www.omg.org/spec/BMM/1.3/PDF/, last accessed: 05.02.2018

B. Analysis Results

Only the URN specification [41] introduces the notation using a coordinate system (see Fig. 4). The IFML and the LML specifications [36], [37] use only samples for the notation specification. All other specifications use sample visualizations supplemented with natural language notation guidelines. The BPMN specification also uses natural language notation guidelines, see e.g., the notation specification of the metaclass *Task*: *A task is a rounded corner rectangle that must be drawn with a single thin line* [33, p. 154].

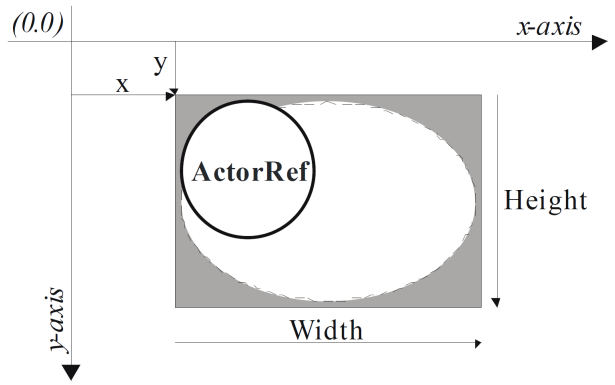


Fig. 4: Notation specification in URN - adapted [41, p. 49]

Some modeling languages support a dynamic notation. An example of a dynamic notation taken from the URN specification [41] is depicted in Fig. 5. The left two boxes represent instances of the metaclass *Intentional* which have two different values for the attribute *type*. Similar, on the right side, two instances of the metaclass *actor* are depicted. Here the value of the attribute importance is only visualized, if it is greater than 0.

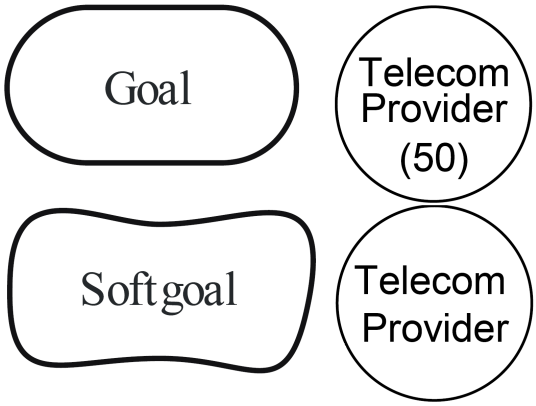


Fig. 5: Dynamic notation of intentions and actors in URN - adapted [41, p. 21-22]

Alternative notations are introduced in almost all specifications we analyzed. In contrast to dynamic notations, alternative notations do not depend on attribute values. Here, the modeler can choose one of the alternatives based on his/her preference. An example is given in Fig. 6 for the ArchiMate constructs *Business process* and *Business function*.

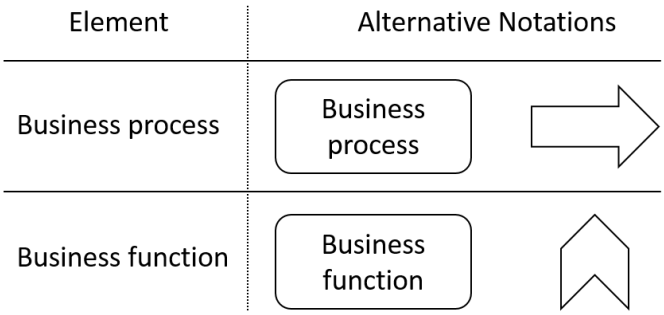


Fig. 6: Alternative notation in ArchiMate - adapted [32, p. 69]

The analysis also revealed, that additionally to the core specification of the modeling language notation, specification documents consider to some extent also different conformance levels. Conformance then relates to different levels of conformity e.g., for tool vendors interested in the development of modeling tools. Some analyzed languages distinguish multiple conformance levels whereas some do not consider conformance at all, or require fill conformance.

Table II summarizes the key findings. Dynamic notation is used in six out of eleven specifications. Nearly all specifications introduce at least one alternative notation, thereby disrespecting the principle of semiotic clarity, more precisely the *symbol redundancy* anomaly [22, p. 23]. All specifications provide samples of the notation; 9 out of 11 supplement natural language explanations. Coordinate systems are only used for the URN [41]. 8 out of the 11 investigate specifications comprise conformance-related aspects.

The ArchiMate specification [32, p. 10] explicitly defines that *the use of color is left to the modeler. However, they can be used freely to stress certain aspects in models*. This is interesting, as in the specification itself, colors are used to distinguish between the three layers of the ArchiMate framework: yellow is used for objects belonging to the business layer, blue for the application layer, and green for the technology layer. All elements belonging to the business layer are colored yellow in the examples presented in the specification. While the layers represent one dimension of the ArchiMate framework, the structure represents the second dimension. The shape of the objects represents to which structure the corresponding construct belongs. *Behaviour* objects are represented by round boxes, *active and passive* structure elements are represented by boxes with square corners, and motivation objects are *usually denoted using boxes with diagonal corners* [32, p. 18]. In the ArchiMate specification, *passive* structure objects are represented with a box with squared corners and an additional horizontal line. ArchiMate follows the philosophy that a different notation should be used for different model users. Therefore the specification introduces a viewpoint mechanism which allows to create user-specific notations. The standard notation introduced in the specification can be used - but it is not binding - see [32, p. 10]. ArchiMate introduces several alternative notations as depicted in Fig. 6.

TABLE II: Analysis of the notation and notation specification techniques ((y)es=used, (n)o=not used)

	Notation Samples	Natural Language Notation Guidelines	Coordinate System	Dynamic Notation	Alternative Notation	Conformance Level
ArchiMate	y	y	n	n	y	y
BPMN	y	y	n	y	y	y
CMMN	y	y	n	y	n	y
DMN (DRD)	y	y	n	n	y	y
IFML	y	n	n	n	y[1]	y
LML	y	n	n	n	y	n
OPM	y	y	n	y	n	y
S2ML	y	y	n	n	y	n
UML	y	y	n	y	y	y
URN	y	y	y	y	y	n
VDML	y	y	n	y	y	y

[1] one alternative notation only

BPMN uses different shapes for different types of metaclasses, e.g., events are represented by a circle and activities are represented by a rectangle. The brightness is e.g. used to distinguish between catching events, not filled, and throwing events, black filled. Most of the BPMN notation guidelines are recommendations and therefore optional. A certain conformance level is introduced whereby the referenced notation guidelines are optional. For example, in section 2.2.3 of the specification [33, p. 8], it is specified that BPMN process diagrams *shall* use the introduced graphical elements and that there is *flexibility in size, color, line style, and text positions of graphical elements*. In section 2.5.3 a similar statement is given for BPMN choreography diagrams. Additionally, section 7.5 of the specification [33, p. 39-45] contains rudimentary notation guidelines - which are heavily referenced within the sections of the specification. It reveals that most of the design guidelines introduced in the specification are optional. For example, color is not defined for the modeling language but may be used for BPMN models. Also, the line style introduced in the specification is optional as well as labels of modeling elements which can be placed at any location *depending on the preference of the modeler or modeling tool vendor* [33, p. 39]. Strict notation guidelines are rare - e.g. it is described that *markers for throwing Events must have a dark fill* [33, p. 39] or that the name of a pool *must be separated from the contents of the pool by a single line* [33, p. 111]. The specification defines alternative notations - there are for example alternative notations for exclusive and event-based gateways (see Fig. 7). For the notation of e.g. events dynamic notation is used which depends on the value of *EventDefinition*.

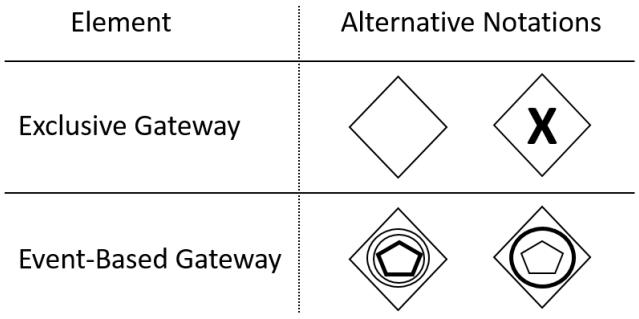
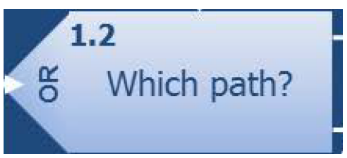


Fig. 7: Alternative notation in BPMN - adapted [33, p. 31]

The CMMN specification [34] introduces a specific notation conformance level whereby - similar to the BPMN - flexibility is retained in e.g., size, color and line style. Only the binding notation guidelines have to be fulfilled in order to create specification-compliant tools. The CMMN makes also use of dynamic notations, e.g., the notation of the instances of the metaclass *Task* depends on attribute value *blocking*.

The DMN specification [35] also specifies flexibility in the notation guidelines for size, color and other notational aspects. The notation itself is specified via samples and natural language, e.g., the name of instances of the metaclass *Decision* should be visualized inside the shape of the element, or that instances of the *KnowledgeSource* metaclass have inter-alia three straight sides. The specification makes also use of alternative notations e.g. for the metaclass *InputData*. DMN does not employ any dynamic notation.

The IFML specification [36, p. 1] introduces a notation conformance level for tool vendors which allows standard-defined notation to be *created, read, updated, and deleted*. However, the notation is not described in a precise way: Unlike other specifications, no natural language notation guidelines are introduced. The notation is only introduced via a table comprising sample notations. Alternative notations are used e.g. for the *Event* metaclass.



(a) Alternative 1 of a OR in LML - adapted [37, p. 52]

(b) Alternative 2 of a OR in LML - adapted [37, p. 51]

Fig. 8: Different notation alternatives of *OR* in LML

The strict distinction between notation and language constructs makes the LML specification unique. LML specifies [37] a very simple notation which aims to keep the complexity low. While a *common visualization* is introduced *other visualizations are allowed and encouraged as they aid in expressing the information, which is the real goal of any language visualizations* [37, p. 5]. The other specifications use at least one symbol for each non-abstract construct as

TABLE III: Visual expressiveness of the notation defined in modeling language specifications

	Visual Expressiveness	Used Variables [13]
ArchiMate	4	Color, Shape, Positions (horizontal/vertical)
BPMN	4	Brightness, Shape, Positions (horizontal/vertical)
CMMN	3	Shape, Positions (horizontal/vertical))
DMN (DRD)	3	Shape, Positions (horizontal/vertical)
IFML	4	Brightness, Shape, Positions (horizontal/vertical)
LML[1]	2	Positions (horizontal/vertical)
OPM	4	Color, Shape, Positions (horizontal/vertical)
S2ML	2	Positions (horizontal/vertical)
UML	4	Brightness, Shape, Positions (horizontal/vertical)
URN	5	Size, Brightness, Shape, Positions (horizontal/vertical)
VDML	3	Shape, Positions (horizontal/vertical)

[1] no standard notation introduced - only examples

well as for each connector. The LML is not that precise. Different samples of LML diagrams are introduced which contain different notations (see Fig. 8a and Fig. 8b for selected examples with different notations for the OR construct). It seems like the specification does not aim at defining unique notations. The notation of the risk matrix, a metaclass of the LML, is explicitly declared as optional in [37, p. 59].

The notation guidelines introduced in the OPM [38] are binding. The OPM specification introduces a specific conformance criteria for tools. The notation is specified using samples as well as natural language notation guidelines. The OPM specification proposes a dynamic notation. For example, the boarder line changes based on an attribute value [38, p. 16-17].

The S2ML uses samples for the specification of the notation. The modeling elements are represented via simple boxes. Additionally, natural language is used to introduce the notation. S2ML also proposes alternative notations for some constructs.

The notation of the UML [40] is binding and introduces a specific notation conformance level. Alternative notations are used, e.g. two different notations are introduced for the construct *Actor* used in Use-Case diagrams. Furthermore, the UML specification makes use of dynamic notations. For example, the construct *PackageImport* is used as connector and has an attribute *visibility*. Depending on the value of this attribute the notation of the connector changes: if the attribute has the value *public* the connector is annotated with the label *import*, otherwise it is annotated with the label *access*. The UML specification [40] is the only specification in this analysis which introduces a precise description of the visualization of the labels. Thereby, the UML reverts to the EBNF formalism. No other specification document describes in such precise way how to represent labels. An example of the label of the construct *extension point* of Use-Case diagrams is shown below [40, p. 640]. ExtensionPoint is denoted by a text string *within the UseCase oval symbol as follows*:

- $< extensionpoint > \ ::= \ < name > [:< explanation >]$

The URN specification [41] introduces binding notation guidelines. It is never mentioned, that the notation is optional or can be modified by tool vendors. If a binding notation is not given for an element, tool vendors have to define ways how to create, access and modify instances of these constructs - e.g. by using a property window. The notation in the URN specification is described using a high level of precision: Notation samples are used as well as natural language notation guidelines. Furthermore, precise specifications of rendering objects by introducing a coordinate system as shown in Fig. 4 are given. Here, it is precisely defined how the x and y coordinates as well as width and height are rendered. Like most of the other specifications, the URN introduces alternative notations e.g. for the constructs *actorRef*.

The notation guidelines of the VDML specification [42] are binding. The VDML specification introduces alternative notations where two instances of a construct are represented differently in two different model types/diagrams.

C. Visual Expressiveness Evaluation

In addition to the findings reported previously, an evaluation of the modeling languages according to the visual expressiveness is performed in the following. The results of this evaluation are summarized in Table III.

The evaluation reveals, that all investigated visual modeling language specifications consider the horizontal and vertical positions variable. Consequently, the minimal visual expressiveness is 2 which is the case for S2ML and LML. Besides, all other nine specifications employ the *shape* variable. With BPMN, IFML, UML, and URN, four specifications additionally describe the *brightness* variable. Only ArchiMate and OPM furthermore use the variable *color*. The notation of the URN specification [41] has the highest visual expressiveness - value 5 - in the analysis.

It seems intuitive to criticize the low values for visual expressiveness. However, some observations need to be considered while interpreting these results. On the one hand, Bertin's alphabet, albeit being extensively used to evaluate modeling languages, was not designed for that particular purpose. In contrast, the work of Bertin was targeted towards the domain of cartography in the 1980s. Consequently, when considering the domain of conceptual modeling in 2018, some variables can be neglected, some need to be analyzed more thoroughly, and others might need to be added to update the alphabet to fit the new context. For example, the use of color might only be feasible when the number of concepts of a modeling language, and thereby the number of different colors used, is not too high. ArchiMate seems to tackle this scaling problem

by using one color for all concepts of an ArchiMate layer instead of using one color for only one concept. Similarly, considerations should be conducted for all other variables.

What concerns color, it must be mentioned, that a lot of people are color-blind. Thus, introducing color as the only distinguishing factor for different concepts might be controversy to the goal of achieving wide and unconstrained adoption. Thus, as e.g. done in the newest ArchiMate standard, color and text can be both optionally used to indicate the layer a specific modeling concept belongs to.

With the movement of modeling methods from being applied with pen and paper towards full-fledged modeling and model-driven development environments, some variables that are disrespected until now, e.g., texture should be revisited. Aspects such as the ability to draw models by hand need to be questioned nowadays. Modeling is nowadays performed in web browsers and models are also used on mobile devices. Such changes in the context where modeling is applied and where models are used, together with the divers set of involved stakeholders with differing purposes, need to be part of future research on visual expressiveness evaluations.

V. Insights and Recommendations

The preceding sections covered the results of a systematic analysis of current visual modeling language notation specifications. In addition, a survey was conducted to evaluate their visual expressiveness. It can be concluded, that the notation of visual metamodels is currently not very mature. All analyzed language specifications revert to simple box-and-line diagrams. One exception is the ArchiMate specification which uses different background colors for different constructs.

There is research to do in order to further improve the visual expressiveness of modeling languages. The findings of the systematic analysis and evaluation underpin the hypothesis as e.g., reported in [44]. The authors state that there exists a "gap of a missing colour systematization for conceptual modeling" [44]. As a results, the authors of [44] propose two different scenarios and the different roles color plays in them: pop-out for parallel processing of multiple colored artifacts and high visual distance for sequential processing. Catalogs of color combinations for both scenarios aid in creating visual models for efficient comprehension by human beings. It is interesting to note that color is currently still only of minor importance for visual modeling languages. However, as a large amount people are color-blind, alternative visualizations, not relying on color as the single differentiating factor, need to be employed.

When it comes to the development of tool support for visual modeling languages, one needs to translate the somewhat imprecise and incomplete modeling language specifications into a machine-processable format. In this translation process, tool developers are confronted with the challenge of balancing the conflicting goals of being standard conforming and providing high usability. When thinking e.g., about if and how to visualize also important attribute values directly within the shapes, therefore omitting the need to double click on model elements. This would directly improve usability. However, such aspects are hardly considered in current modeling language specifications. Besides the intuitiveness of providing sample visualizations, tool vendors need precise and unambiguous specifications to create conforming tools - and ultimately conforming and interchangeable models created with them.

It should be acknowledged, that some of the modeling language maintaining institutions -e.g., the Object Management Group (OMG) - already came to the awareness some of the issues raised in this paper. Using the Diagram Definition (DD) [45], one is able to formally specify the graphical notation of diagram elements by providing a mapping between the abstract syntax (constructs of the Meta Object Facility metamodel [46]) of the language and the provided diagram graphics of the DD. Moreover, diagram interchange formats can be specified in the Diagram Definition standard. Interestingly, besides the necessity of such a standard, it has not been updated from version 1.1 which was published in June 2015.

All specifications except the LML specification [37] use combined elements. Thereby, elements are positioned on the top of each / within each other. For example, in BPMN, events can be placed on the boarder of activities; or in ArchiMate, where groups can be used which encompass other ArchiMate elements. The BPMN notation encompasses grouping elements and combined elements. The variables *texture* and *orientation* (see Fig. 2) are currently not used by any of the investigated visual modeling language specifications, maybe due to their background of being comfortably applicable with pen and paper. It will be interesting to see if those aspects will be considered in the future, when modeling tool platforms improve their abilities, e.g. for multi-dimensional visualizations and navigation through model corpora.

Based on the experience of analyzing the 11 visual modeling languages, some recommendations for improvements are derived. As modeling languages are under continuous revision and extension, we believe this research contributes to the realization of improved and complete specifications in the future.

- **Completeness.** The analysis revealed not only the heterogeneous set of techniques used in the specifications but also a lack of completeness. This considers e.g., that representation of attributes or the representation of connectors. This is a serious deficit as only complete specifications enable proper understanding, utilization, and tooling. Moreover, existing theory about e.g., principles of semiotic clarity [22, p. 23] are disrespected by 9 out of 11 investigated specifications.
  We recommend to check whether for each metamodel construct the notation is specified completely according to Bertin's visual alphabet [13], [10].
- **Consistency.** The analysis revealed several weaknesses e.g., with respect to the inconsistent usage of alternative notations, leading to an increased effort to comprehend the specification.

We recommend to consistently use one/alternative notations throughout the whole specification document.

- **Technique Mix.** All analyzed specifications use two or more techniques throughout the document (samples, natural language, coordinate systems, etc.). When applied in a meaningful way, this combination effectuates the positive aspects (e.g., with respect to visual expressiveness and intuitive comprehension) of each of them while comprising a complete modeling language notation specification.
  We recommend to use a mix of specification techniques in order to utilize the respective strengths of the techniques. Samples are powerful for intuitive comprehension, however they need to be accompanied by more precise techniques like the coordinate-system.
- **Alternative notations.** Modeling nowadays is aims to undergo a transition from being an elicit discipline performed by experts towards being applied by the masses [47]. Consequently, heterogeneous stakeholders with divers knowledge backgrounds and purposes are involve in modeling. Using alternative notations is one way to approach this heterogeneity in conceptual modeling. Thus, one notation might be beneficial for experts that want to generate code form a model, whereas an iconic and colorful notation is beneficial for business users that want to understand e.g., the process behavior.
  We recommend to use alternative notations in cases where heterogeneous stakeholders with different purposes are involved in modeling. Such alternative notations should be motivated and their purpose needs to be clearly stated in the specification documents.
- **Preamble.** What is important is that the reader of the specification is aware of the structure and techniques used within the specification. This contributes to a better and faster comprehension of the actual modeling language.
  We recommend to provide an extended preamble of the specifications where not only the used terminology but also the applied specification techniques are briefly introduced. Moreover, conformance levels shall be detailed in the very beginning - as being done for most OMG standards.

VI. Conclusion

This paper reported on a systematic analysis of currently used standard visual modeling language specifications with a focus on the specification of notational aspects. Based on the analysis, a generic framework of specification techniques has been derived. This framework has then been applied to provide an objective analysis of the languages. Finally, the visual expressiveness of the modeling languages was evaluated by applying the Physics of Notation.

Despite the wide adoption in academia and industry, the visual expressiveness of currently used visual modeling languages can be further improved. This paper showed strengths and weaknesses of current specifications. Based on these findings, concrete recommendations for improving modeling language notation specifications are given.

In our future research we will apply the framework to modeling language specifications from an academic background, e.g., by investigating the domain-specific modeling methods realized within the Open Models Laboratory [2], [48]. It is expected, that e.g., color is more widely used in such modeling languages. Moreover, an empirical study is planned that investigates whether the different variables of Bertin's alphabet have a different importance for the intuitive understanding of modeling languages by human beings (cf. [49], [50]). Lastly, we plan to use the findings of this research to improve modeling language specifications in the future by establishing an awareness for both, researchers and maintaining institutions.

Acknowledgment

Part of this research has been funded through the South Africa / Austria Joint Scientific and Technological Cooperation program with the project number ZA 11/2017.

References

[1] M. Rosemann, "Potential pitfalls of process modeling: part A," *Business Process Management Journal*, vol. 12, no. 2, pp. 249–254, 2006.
[2] D. Karagiannis, H. C. Mayr, and J. Mylopoulos, Eds., *Domain-Specific Conceptual Modeling, Concepts, Methods and Tools*. Springer, 2016.
[3] D. Harel and B. Rumpe, "Modeling languages: Syntax, semantics and all that stuff - Part I: The Basic Stuff," Technical report, Tech. Rep., 2000.
[4] J. Gulden, D. van der Linden, and B. Aysolmaz, "A Research Agenda on Visualizations in Information Systems Engineering," in *Proceedings of the 11th International Conference on Evaluation of Novel Software Approaches to Software Engineering*. SciTePress, 2016.
[5] N. Genon, P. Heymans, and D. Amyot, "Analysing the Cognitive Effectiveness of the BPMN 2.0 Visual Notation," in *International Conference on Software Language Engineering*. Springer, 2010, pp. 377–396.
[6] R. Braun and W. Esswein, "Classification of domain-specific bpmn extensions," in *IFIP Working Conference on The Practice of Enterprise Modeling*. Springer, 2014, pp. 42–57.
[7] B. Pittl and D. Bork, "Modeling Digital Enterprise Ecosystems with ArchiMate: A Mobility Provision Case Study," in *International Conference on Serviceology*. Springer, 2017, pp. 178–189.
[8] M. Brambilla, J. Cabot, J. L. Cánovas Izquierdo, and A. Mauri, "Better call the crowd: using crowdsourcing to shape the notation of domain-specific languages," in *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*. ACM, 2017, pp. 129–138.
[9] D. Karagiannis and H. Kühn, "Metamodelling platforms," in *E-Commerce and Web Technologies, Third International Conference, EC-Web 2002, Aix-en-Provence, France, September 2-6, 2002, Proceedings*, 2002, p. 182.
[10] D. Moody, "The 'physics' of notations: toward a scientific basis for constructing visual notations in software engineering," *IEEE Transactions on Software Engineering*, vol. 35, no. 6, pp. 756–779, 2009.
[11] D. Bork and H.-G. Fill, "Formal Aspects of Enterprise Modeling Methods: A Comparison Framework," in *2014 47th Hawaii International Conference on System Sciences*. IEEE, 2014, pp. 3400–3409.
[12] A. Kleppe, *Software language engineering: creating domain-specific languages using metamodels*. Pearson Education, 2008.
[13] J. Bertin, *Semiology of Graphics - Diagrams, Networks, Maps*. ESRI, 2010.
[14] M. Rosemann, J. Recker, M. Indulska, and P. Green, "A study of the evolution of the representational capabilities of process modeling grammars," in *International Conference on Advanced Information Systems Engineering*. Springer, 2006, pp. 447–461.
[15] Y. Wand and R. Weber, "On the deep structure of information systems," *Information Systems Journal*, vol. 5, no. 3, pp. 203–223, 1995.
[16] F. Johannsen and S. Leist, "Wand and Webers decomposition model in the context of business process modeling," *Business & Information Systems Engineering*, vol. 4, no. 5, pp. 271–286, 2012.

[17] A. Awadid and S. Nurcan, "Towards enhancing business process modeling formalisms of EKD with consistency consideration," in *Research Challenges in Information Science (RCIS), 2016 IEEE Tenth International Conference on*. IEEE, 2016, pp. 1–12.

[18] D. Bork, R. Buchmann, and D. Karagiannis, "Preserving multi-view consistency in diagrammatic knowledge representation," in *International Conference on Knowledge Science, Engineering and Management*. Springer, 2015, pp. 177–182.

[19] D. Karagiannis, R. A. Buchmann, and D. Bork, "Managing Consistency in Multi-View Enterprise Models: An Approach Based on Semantic Queries," in *Proceedings of the Twenty-Fourth European Conference on Information Systems (ECIS), Istanbul, Turkey*, 2016, Research Papers. 53.

[20] A. Awadid, "Supporting the consistency in multi-perspective Business Process Modeling: A mapping approach," in *Research Challenges in Information Science (RCIS), 2017 11th International Conference on*. IEEE, 2017, pp. 414–419.

[21] C. Gonzalez-Perez and P. Martín-Rodilla, "A metamodel and code generation approach for symmetric unary associations," in *Research Challenges in Information Science (RCIS), 2017 11th International Conference on*. IEEE, 2017, pp. 84–94.

[22] D. Moody and J. van Hillegersberg, "Evaluating the visual syntax of UML: An analysis of the cognitive effectiveness of the UML family of diagrams," in *International Conference on Software Language Engineering*. Springer, 2008, pp. 16–34.

[23] D. Allaki, M. Dahchour, and A. En-Nouaary, "Managing Inconsistencies in UML Models: A Systematic Literature Review," *JSW*, vol. 12, no. 6, pp. 454–471, 2017.

[24] B. Henderson-Sellers and J. Ralyté, "Situational Method Engineering: State-of-the-Art Review," *J. UCS*, vol. 16, no. 3, pp. 424–478, 2010.

[25] F. Kathrin, "USER EVALUATION OF SYMBOLS FOR CORE BUSINESS PROCESS MODELING CONCEPTS," in *Proceedings of the 25th European Conference on Information Systems (ECIS), Guimares, Portugal, June 5-10*, 2017, pp. 581–594.

[26] S. Keele *et al.*, "Guidelines for performing systematic literature reviews in software engineering," in *Technical report, Ver. 2.3 EBSE Technical Report. EBSE*. sn, 2007.

[27] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Information & Software Technology*, vol. 55, no. 12, pp. 2049–2075, 2013.

[28] R. A. Buchmann and D. Karagiannis, "Enriching Linked Data with Semantics from Domain-Specific Diagrammatic Models," *Business & Information Systems Engineering*, vol. 58, no. 5, pp. 341–353, 2016.

[29] Object Management Group, "OMG Specifications Catalog," 2018, last checked, 2018-02-05. [Online]. Available: http://www.omg.org/spec/

[30] The Open Group Group, "The Open Group Standards," 2018, last checked, 2018-02-05. [Online]. Available: http://www.opengroup.org/standards

[31] C. Gonzalez-Perez, "A conceptual modelling language for the humanities and social sciences," in *2012 Sixth International Conference on Research Challenges in Information Science (RCIS)*, 2012, pp. 1–6.

[32] ArchiMate, "Archimate," *The Open Group*, 2016, accessed on 2018-02-05. [Online]. Available: https://www2.opengroup.org/ogsys/catalog/C162

[33] BPMN, "Business Process Model and Notation (BPMN)," *OMG*, 2013, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/BPMN/2.0.2/PDF/

[34] CMMN, "Case Management Model and Notation (CMMN)," *OMG*, 2016, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/CMMN/1.1/PDF/

[35] DMN, "Decision Modeling Notation (DMN)," *OMG*, 2016, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/DMN/1.1/PDF/

[36] IFML, "Interaction Flow Modeling Language (IFML)," *OMG*, 2015, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/IFML/1.0/PDF/

[37] LML, "Lifecycle Modeling Language (LML)," *LML*, 2015, accessed: 2018-02-05. [Online]. Available: http://www.lifecyclemodeling.org/specification/

[38] OPM, "Object Process Methodology (OPM)," *ISO*, 2014, accessed: 2018-02-05. [Online]. Available: https://www.iso.org/standard/62274.html

[39] S2ML, "System Structure Modeling Language (S2ML)," -, 2015, accessed: 2018-02-05. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01234903/document

[40] UML, "Unified Modeling Langague Specification (UML)," *OMG*, 2015, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/UML/2.5/PDF/

[41] URN, "User requirements notation (urn)," *ITU-T*, 2012, accessed: 2018-02-05. [Online]. Available: https://www.itu.int/rec/T-REC-Z.151-201210-I/en

[42] VDML, "Value Delivery Modeling Language (VDML)," *OMG*, 2015, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/VDML/1.0/PDF/

[43] J. Mylopoulos, "Conceptual modelling and telos," *Conceptual Modelling, Databases, and CASE: an Integrated View of Information System Development, New York: John Wiley & Sons*, pp. 49–68, 1992.

[44] J. Stark, R. Braun, and W. Esswein, "Systemizing Colour for Conceptual Modeling," in *Proceedings der 13. Internationalen Tagung Wirtschaftsinformatik (WI 2017), St. Gallen*, 2017, pp. 256–270.

[45] DD, "Diagram Definition (DD)," *OMG*, 2015, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/DD/1.1/

[46] MOF, "Meta Object Facility (MOF)," *OMG*, 2016, accessed: 2018-02-05. [Online]. Available: http://www.omg.org/spec/MOF

[47] K. Sandkuhl, H.-G. Fill, S. Hoppenbrouwers, J. Krogstie, F. Matthes, A. Opdahl, G. Schwabe, Ö. Uludag, and R. Winter, "From Expert Discipline to Common Practice: A Vision and Research Agenda for Extending the Reach of Enterprise Modeling," *Business & Information Systems Engineering*, vol. 60, no. 1, pp. 69–80, 2018.

[48] D. Bork and E.-T. Miron, "OMiLAB - An Open Innovation Community for Modeling Method Engineering," in *8th International Conference of Management and Industrial Engineering (ICMIE'2017)*, A. Niculescu, O. D. Negoita, and B. Tiganoaia, Eds., 2017, pp. 64–77.

[49] G. Jošt, J. Huber, M. Heričko, and G. Polančič, "An empirical investigation of intuitive understandability of process diagrams," *Computer Standards & Interfaces*, vol. 48, pp. 90–111, 2016.

[50] D. Van Der Linden and I. Hadar, "Cognitive effectiveness of conceptual modeling languages: Examining professional modelers," in *Empirical Requirements Engineering (EmpiRE), 2015 IEEE Fifth International Workshop on*. IEEE, 2015, pp. 9–12.