Enterprise Modelling Languages
Just Enough Standardisation?

Marija Bjeković[1,2,3], Hend*erik* A. Proper[1,2,3] and Jean-Sébastien Sottet[1,3]

[1] Public Research Centre Henri Tudor, Luxembourg, Luxembourg
[2] Radboud University Nijmegen, Nijmegen, Netherlands
[3] EE-Team, Luxembourg, Luxembourg*
{marija.bjekovic,erik.proper,jean-sebastien.sottet}@tudor.lu

Abstract. In enterprise modelling, a wide range of models and languages is used to support different purposes. If left uncontrolled, this variety of models and languages can easily result in fragmented perspective on an enterprise, its processes and IT support. A traditional approach to address this problem is to create standard modelling languages that unify and integrate different modelling perspectives, such as e.g. UML, BPMN, and ArchiMate. However, one can observe how, in actual use, the 'standardising' and 'integrating' effect of these languages erodes. This is typically manifested by the emergence of 'variants', 'light weight versions', and extensions of the standard dealing with 'missing aspects'. The empirical data suggests that these 'variants' emerge to compensate the inability of a standard language to aptly fit the needs of specific modelling situations. In this paper, we reconsider the drivers and strategies of modelling language standardisation. Relying on an ongoing research, the paper develops a fundamental understanding of the role of fixed language in the context of conceptual and enterprise modelling. This understanding is then used to analyse the 'variants' in the actual use of a standard process modelling language, and to discuss the potential insights towards its standardisation strategy.

Keywords: model, modelling language, standardisation, modelling pragmatics

1 Introduction

Enterprise models play an important role in the design and operations of enterprises. They typically represent an enterprise from different perspectives, and are used for various purposes, e.g. to study the current state of an enterprise, analyse problems with regard to the current situation, sketch potential future scenarios, design future states of the enterprise, communicate with stakeholders, manage change, etc. If this plethora of models is left uncontrolled, it may result

* The Enterprise Engineering Team (EE-Team) is a collaboration between Public Research Centre Henri Tudor, Radboud University Nijmegen and HAN University of Applied Sciences (www.ee-team.eu).

in a fragmented view on the enterprise, and thus negatively affect the overall coherence of models. The fact that these partial enterprise models are often expressed in different modelling languages makes the coherence even a greater challenge.

The traditional approach of dealing with this situation is to create a unifying modelling language, such as UML for software design, and ArchiMate for enterprise architecture modelling. The assumption behind is that the fragmentation can be prevented by *a priori* integrating relevant perspectives and concepts within a single *standard* modelling language. However, one can observe how, in actual use, the 'standardising' and 'integrating' effect of these languages erodes. This is typically manifested in terms of local 'dialects' [35, 9], 'light weight versions' [35], or several extensions of an existing standard that are intended to deal with 'missing aspects' [15, 19, 40]. The point is further illustrated by the advent of domain-specific [22] and/or purpose-specific modelling languages [6], which allow for the creation of models that are tuned to the needs of specific domains or purposes.

A more realistic strategy to address fragmentation is to create point-to-point bridges (e.g. [41, 55, 17]) between the modelling languages used in enterprise modelling. The bridges between the languages are typically established based on the *standard* language definitions, in order to be able to reuse them across different usage contexts. Nonetheless, the phenomena of 'dialectisation' is not limited to unifying languages, but is also reported for many enterprise modelling languages, such as goal-oriented, value-oriented and process modelling languages [2, 16, 33, 43]. This is also manifested in dialect-like variations of the original modelling language (e.g. [58]), use of workarounds (e.g. using ad hoc notes, narratives, and annotations, e.g. [43]) to compensate for the missing elements in the language/tool. It may even go as far as using home-grown, organisation-specific semi-structured languages instead of the standard ones [2, 35, 31]. This phenomena might question the real potential for reuse of standard language bridges, i.e. the value of bridging languages out of context of their use.

While a non-compliance to modelling standards is typically perceived as undesirable, the reasons underlying the wide spread existence of language 'variants' or 'dialects' are not well understood. The available empirical data suggests that they mainly emerge to compensate for the lack of suitability of the language/tool to aptly fit the needs of specific modelling situations. The arguments underlying the widespread use of e.g. Visio as a modelling tool in practice (e.g. [14, 35, 43]), as well as the growing research interest in modelling language/tool flexibility (e.g. [32, 11]), further strengthen this point.

How similar are these 'variants' of a standard? What are the dimensions of divergence from a standard language? If widespread, should these 'variants'(e.g. light-weight versions used in stakeholder communication [35]) be covered by the original standard? How many of these 'variants' should become part of the standard? Finally, to paraphrase, how much standard language would be enough [58]?

In our view, this calls for reconsidering the drivers of, and approaches applied in, modelling language standardisation. But first and foremost, this requires a deep understanding of the role of modelling language in (conceptual and) enterprise modelling, and of the factors driving its use. We believe that this understanding may provide valuable insights for the scoping and design of modelling languages that are better suited to the practical needs. The present paper aims to contribute to such an understanding from a rather theoretical perspective. Relying on our ongoing research, a fundamental understanding of the role of fixed language in conceptual and enterprise modelling is developed in the paper. This understanding is then used to analyse the 'dialectisation' in the actual use of a standard process modelling language BPMN [39], and to discuss the potential insights towards its standardisation strategy.

The remainder of the paper is organised as follows. Section 2 explores the problem of modelling language standardisation, by confronting the rationales of standardisation to the actual use of standard modelling languages. This section identifies the need to deeply understand the role that modelling language has in conceptual/enterprise modelling, and the factors that determine its added value. Subsequently, the Sections 3 and 4 elaborate our fundamental understanding of this topic. This understanding is then used, in Section 5 to analyse the 'variants' emerging in the actual use of a standard process modelling language, BPMN [39], and to discuss the potential insights towards its standardisation strategy, prior to concluding the paper.

2 What is the Scope of a Modelling Language Standard?

In the field of enterprise modelling, a wide range of *fixed* modelling languages is being defined and used, while some of them also undergo the process of *standardisation*. Despite the ambition of such standards, available empirical studies, e.g. [2, 58, 43, 35, 9, 31] pinpoint at their inability to fit the needs of practical modelling situations. In practice, this is typically overcome by the emergence of different 'variants' of the used standard, which alter the original language definition, by *reducing*, *extending* or *adapting* it to a modelling task at hand.

Is such behaviour due to the very nature of complex and generic standards, or to the failure to include all the relevant aspects into a standard language? Or, is it rather tied to the way language users perceive the value of the modelling language (standard or not), and even to their subjective preferences? Could these derivations from a specific standard be prevented in the first place, e.g. by a different design of modelling languages? If not, how should they be dealt with? In this section, we revisit the common rationales of modelling language standardisation in 2.1, and confront them to the actual use of such standard languages in 2.2. We then identify, in 2.3, one of the possible research directions to gain a deeper understanding of this problem, which is explored in the paper.

2.1 Drivers of Modelling Language Standardisation

The potential benefits of a *fixed* definition of a modelling language are well-known. It provides a foundation for the development of tools and automated model manipulations (e.g. analysis, simulation, model transformation, code generation), thus enabling the increase of productivity and diminishing the error-rate of model manipulations.

The effort of *standardisation* of such languages is often driven by the desire to generalise these potential benefits across one or more application areas (e.g. system engineering, software engineering, or process modelling) etc. More precisely, the review of commonly used modelling standards in enterprise modelling [38, 39, 30] reveals that standardisation is typically driven by the following interrelated rationales:

1. Harmonise and consolidate many similar yet divergent modelling languages for modelling some problem area/system, incorporating the best practices of similar methods/notations [38, 39],
2. Standardise model exchange format between the tools implementing the language specification [38, 39],
3. Provide the standard way to communicate about the problem/system by different stakeholders, and for different uses of models [38, 39],
4. Define a uniform representation for a wide range of uses within some problem area [38, 39, 30],
5. Integrate different perspectives of a system under study within a single standard language [38, 39, 30],
6. Provide standardised bridge of the gap between the graphical language and the appropriate execution format [39], etc.

Indeed, the harmonisation and/or consolidation of similar overlapping modelling languages already existing for modelling some problem area (e.g. business processes) is a common goal of many standards. The drivers here are many, e.g. reducing language and tool related learning and training costs, tool market harmonisation, facilitating tool selection for practitioners, providing standard model exchange format for tool interoperability. Obviously, the more generic the standard, the more reuse potential it is likely to have across different application areas.

Additionally, a standard modelling language has the ambition to standardise the communication about some problem area/system between various stakeholders and for many purposes for which modelling is done. For instance, the BPMN specification states the ambition to provide the *common* language and visual notation for both business and technical users. An *a priori* imposed standard vocabulary is meant to avoid frequent meta-discussions on concepts between model stakeholders, and to facilitate knowledge transfer. This is tightly related with the drive of defining a *uniform representation* for a range of *uses* of models within a problem area.

The 'uniformisation' and 'harmonisation' often also entail the *integration* of different perspectives, i.e. models, of the system under study within a single, standard *unifying* language definition. As many overlapping languages exist to model the system from different perspectives, the unifying language consolidates these languages within the single integrated language, to be used instead of these partial languages. UML and ArchiMate are typical cases of such a strategy. The standardising effect of a unifying language hence also lies in that it *a priori* integrates perspectives for modelling some system. Such a strategy facilitates assuring integration between the models, given that consistency and coherence rules can be embedded in the standard language definition, and tools can *automatically check* these properties.

A standardised and/or integrated language is thus one possible strategy to ensure the return on the modelling effort. We can however observe that drivers underlying standardisation effort are predominantly of technical-economical nature. In our view, the potential benefits of standardisation tend to be overly quickly generalised to the entire range of possibles uses of such a standardised language. Below we discuss different challenges of using standardised languages, based on the available empirical data reporting on an enterprise modelling practice.

2.2 Insights from the Use of Standard Modelling Languages

A key problem in *the use of fixed/standard languages* in enterprise modelling seems to be rooted in the lack of *suitability* of a language for the modelling task at hand.

For instance, the widespread need for simpler and rather informal 'variants' of software and enterprise architecture modelling languages is identified by the practitioners interviewed in [35]. These variants are needed in particular for *stakeholder communication*, which is actually reported as the *primary need by practitioners*, and the need that is the most poorly met by existing architectural languages. Despite the abundance of sophisticated and rather formal architectural languages, practitioners still tend to mostly use UML-based languages. Too much formality, as well as too little support for stakeholder communication are indicated as the main reasons for this. Along the same lines, practitioners raise the need for better tool support for language extensibility, for informal activities such as sketching, for combining models with text, etc.

Similarly, the common use of rather informal 'variants' of general-purpose modelling languages is reported in enterprise modelling practice in e.g. [14, 9, 31]. The need for relaxed versions/dialects of a generic language such as UML is observed, for example, in enterprise modelling situations whose primary goal is *collective knowledge creation* (e.g. developing vision and strategy, scoping the problem, and high-level business design) [9]. As most stakeholders do not have modelling expertise, the language and tools are required to be simple, intuitive, and corresponding to the natural interaction that occurs in such situations [9].

Besides 'variants' of existing standard modelling languages, the use of 'home-grown' or 'ad-hoc' notations is quite common in enterprise modelling practice [2, 35, 9]. For instance, the study of the use of conceptual models in enterprise modelling efforts across IBM [2], reports that business analysts and business architects clearly prefer home-grown and semi-structured models over the usual standard languages. This is typical for *exploration phases* "where things are unclear and ambiguous" [2, p. 1304], and where semi-structured models offer flexibility in terms of delayed commitment to syntax, unconstrained development order, evolvable re-factorable metamodel, as well as a *closer fit to the inherent way of thinking* in these phases [2]. As argued in [2], these notations emerge through the repeated use in similar modelling situations, and gain more structure over time.

In the cases presented so far, we discussed challenges of using standard languages in situations of model-based communication with rather business stakeholders and business concerns. Moreover, in the case of exploration and collective knowledge creation with stakeholders, their involvement and input is crucial for producing 'good' models. For these communication-oriented purposes, standard languages do not seem to be suitable enough: their inherent complexity (in terms of constructs and embedded syntactic-semantic restrictions) seem to rather represent a burden. We have argued in [6, 7] that 'variants' emerging in such situations are in fact purpose-specific variations of the original generic language, which tune the language to the needs of given modelling situations. An extreme case of this tuning are, in our view, these ad-hoc and home-grown notations. Such notations can be seen as the emergent modelling languages, which naturally adapt to the needs of situations in which they are used, and gain structure over the course of recurring use in similar modelling situations.

The need for purpose-specific tuning of the language for a given communication situation is a rather natural principle, and indeed corresponding to the way humans normally use natural language [13]. The standards such as e.g. UML and BPMN have the ambition to define the language and notation which is readily usable for the various purposes in their respective application areas, in particular aiming to provide modelling support for business users and purposes. However, a thorough consideration of how these complex standards can/should accommodate these different purposes is lacking. The practice seems to suggest its response.

The practical use of the BPMN standard [39], widely used in enterprise modelling, is examined in [58, 43]. It reveals that, in practice, a very small subset of the BPMN constructs is widely used (cluster of around six concepts), another six concepts being occasionally used, while a huge number of concepts is rather superfluous and extremely rarely applied. It is shown in [58] how these different language subsets are used for different purposes. The basic subset of core BPMN concepts (i.e. task, flow, start and end event) together with lanes and pools is rather observed for the purposes of process documentation, organisational (re)design and process improvement. A slightly richer set of constructs, including gateways and event conditions, is used for e.g. simulation and work flow engineering. Overall, the practical use of BPMN demonstrates much less complexity than the standard specification.

Indeed, the BPMN standard is assessed by [43] as over-engineered and way too complex compared to the practical needs. Should then BPMN *as a standard* include those superfluous and rarely used concepts? What is the rationale and added-value of including them in the scope of a *standard definition*? Should the standard language rather be geared towards the most common use of such language? For instance, organisation design and documentation, knowledge management and continuous process improvement are reported, in [43], as the primary purposes for which BPMN models are used. However, BPMN does not provide sufficient constructs to express all the relevant organisational aspects [53, 43]. Constructs such as Pools and Lanes are judged as not sufficient for modelling organisational resources in [53]. This finding corroborates the observation by [43] that process models expressed in BPMN are very often extended with the symbols, and even with other models, that capture organisational resources, organisational structure, data, business rules, risks, resources, documents etc. So, if this need is recurrent in the practical use of BPMN, shouldn't these aspects be covered by the standard language?

An additional challenge is present in the use of standard *unifying languages*. Despite the ambition to *a priori* address the integration problem of different perspectives within a standard, the adaptation and extension of such a language with 'missing aspects' can be observed in its actual use. For instance, ArchiMate [30] was initially designed as the enterprise architecture modelling language, which relates relevant 'architectural domains' within a single language umbrella. In the practical use of ArchiMate, it is possible to observe this drive to *extend* ArchiMate models to include specific concerns (e.g. [19, 10]) and/or industry-specific standards (e.g. [4]), to cater for contingencies of a *specific application context*. These extensions and adaptations essentially yield a domain-specific version of the original ArchiMate language.

The challenge here lies in the fact that it is nearly impossible to *a priori* identify all the relevant perspectives that should be part of an integrated/unified language for e.g. enterprise modelling. The relevance of different perspectives is highly context-dependent: different perspectives may be relevant for different industries and enterprises, or even in different transformation projects of the same enterprise. Additionally, over time, new perspectives may become relevant (e.g. cloud, privacy, compliance). At the same time, one can observe how there is a drive to *extend* the ArchiMate *standard* to cover the additional aspects, potentially relevant for enterprise modelling. The move from the ArchiMate 1.0 to the ArchiMate 2.0 standard included two additional aspects, namely motivation and implementation & migration. Further integration between TOGAF and ArchiMate is likely to lead to even more extensions. Moreover, the extensions of a standard with e.g. business policies and rules, are also considered [30]. Should all of these considered extensions become integral part of the ArchiMate language? What are the aspects falling under the competence of the ArchiMate language, and which of them should remain outside the language? Potentially endless extensions of ArchiMate are quite likely to result in a fairly complex and, most probably, over-engineered language, similar to the situation of the BPMN 2.0 standard. In practice, this is likely to result in usability problems, and potentially also in many different simplified or adapted variants of a standard being in place. Whether this is a desirable result of a language standardisation has to be questioned [43].

2.3 Discussion

The discussions so far clearly point, in our opinion, to the challenge of defining the *right* scope of a standard modelling language. How should the practical needs and practical use of a language stand in relation with the standard? What is the added value of including some construct or perspective into a standard language? How should this be decided? Should the scope of a standard language be explicitly limited, and based on which criteria? Going further, more fundamental questions are at stake: What does a standard modelling language seek to standardise: the way of thinking about some class of problems in a problem area, the (visual) representation of models (across their different uses) within a problem area, or model exchange format between the modelling tools? Should the single modelling language 'standardise' all these aspects?

To answer these questions, we argue that it is necessary to clearly understand the role that modelling language plays in modelling, and thus identify the main factors determining its added-value in modelling. This is, in our view, necessary if we want to make scientifically grounded decisions regarding the optimal scope of the fixed/standard modelling language. At the same time, this requires tackling some very fundamental aspects of modelling.

3 What is Modelling?

The next two sections of the paper present the initial version of our *explanatory* theory, which has the ambition to reach a fundamental understanding of the role of fixed language in enterprise modelling. The focus of the theory is on *modelling pragmatics*, i.e. on the use and value of models and modelling languages in the given context, in dependence on the modelling goals [49]. We thus primarily seek to fundamentally understand the phenomena related to the *use of modelling languages* in different situations and for different purposes of modelling. Such a focus requires us to revisit our understanding of the very act of modelling, and the role that purpose has in it. We provide our understanding of these topics in the present section. Section 4 then discusses the role and potential benefits of a fixed modelling language in an enterprise modelling effort.

3.1 Grounding

We understand models as essentially means of communication about some domain of interest, and process of modelling as communication-driven process led by a pragmatic focus [25]. This view is inspired by different related research tackling the fundamental modelling aspects such as [46, 45, 18, 28, 42, 49].

3.2 Model Definition

Though different views on models and modelling exist, as well as many different model definitions, here we elaborate reasons for which we propose the following (general) model definition (based on [46, 45, 18, 48]):

*A **model** is an artefact acknowledged by an observer as representing some domain for a particular purpose.*

By stating that a model is an **artefact**, we exclude *conceptions* [18] or so-called "mental models" from the scope of this definition. The reason for this restriction is practical: in our field, the primary concern is the model-as-artefact resulting from the modelling act.

Conceptions are *abstractions* of the "world" under consideration, adopted from a certain perspective. They share this property with models. However, a conception resides in the mind of a person holding it, and as such is not directly accessible to another human being. To be able to discuss and agree on its content, the conception has to be externalised. While the conceptions reside in mental space, the models are necessarily *represented* in physical/material space (typically using some system of symbols). This *representation* dimension is crucial for any model, as the value of modelling primarily resides in the utility of the model-as-artefact for some purpose.

This said, we do consider conceptions to be fundamental to modelling. This point is thoroughly discussed throughout this section.

The **observer** in our definition refers to the group of people consisting of model creators and model audience. On one extreme, it can refer to the entire society, on the other extreme, to the individual. Though it may not be the general rule, it is very often the case, in an enterprise modelling context, that model creators are at the same time its audience. The observer is the key element in modelling, as it is only by virtue of the observer's appreciation that some artefact comes to be acknowledged as a model of some domain.

Similarly to [18], we define **domain** as any "part" or "aspect" of the "world" *considered relevant by the observer* in the given modelling context. The "world" here may refer not only to the "real" world, but also to hypothetical or imagined worlds. Even more, the domain of a model can be another model as well.

A model always has a **purpose**. This *purposefulness* dimension is present in most of the model definitions, e.g. [46, 45, 48]. Although acknowledged as essential dimension of models, the concept of purpose is rarely defined and its role in the entire modelling process is scantly discussed.

In the following, we discuss our view on the role of purpose in modelling, as well as suggest our definition of this concept.

3.3 Centrality of Purpose

A **modelling situation** is *at least* characterised by the wider context in which the modelling takes place (e.g. a particular organisation, project), the involved observer, and the goals of the situation.

The goals of the modelling situation are not necessarily restricted to the goal of producing the model. Particularly, in enterprise modelling, these goals may also refer to organisational learning, achieving consensus on a topic and reaching some commonly agreed knowledge [8, 34]. The present discussion will focus on these goals that are relative to the desired model-as-artefact.

The reason why an observer creates a model in the first place is to enable some *usage* of that model (e.g. analysis, sketching, execution, contracting etc.) by its intended *audience* (e.g. business analysts, business decision-makers, enterprise architects, process experts, etc.).

We believe these are crucial dimensions underlying the concept of model purpose. As will be discussed later, these dimensions are heavily interdependent. Their combination determines the desired model qualities [8, 34] for the purpose at hand. This is quite important, since the *fitness-for-purpose* determines the *value* of the model for its intended use.

We therefore propose the following definition of the model purpose.

*The **purpose** of the model is a combination of the following dimensions:*

(1) the domain which the model should pertain to, and

(2) the intended use of the model by its intended audience.

In line with [45, 48], that (although usually implicitly present) the purpose should be made explicit within the modelling process. At least the model creator should be aware of the intended usage and audience of the model.

To explain the central role that purpose has in a modelling act, we will first consider the modelling situation where an observer is an individual, illustrated in Figure 1a. When, in this situation, an observer **O** engages in modelling of some "world" under consideration, s/he judges which aspects of that "world" are relevant for the given modelling situation[4]. This process of selecting the relevant and *abstracting away* from the irrelevant aspects of the "world" yields the observer's *conception of the domain*, $\mathbf{c}_d$.

It is here important to underline that this process of abstracting away from irrelevant aspects is always *relative* to the given modelling situation. It is implicitly or explicitly influenced by the purpose $\mathbf{p}$ of the model-to-be $\mathbf{m}_d$ [46, 45, 48]. This is depicted as an influence of the purpose $\mathbf{p}$ on the relation *conception of* in the Figure 1. Secondly, but not less importantly, how an observer creates an abstraction is also very much dependent on his/her pre-conceptions [42], brought by its particular social, cultural, educational and professional background. We come back to this point in Section 4.

To externalise the conception $\mathbf{c}_d$ residing in his/her mind, the observer **O** subsequently tries to shape an artefact (i.e. the model-to-be) in such a way that it adequately *represents*, for the purpose $\mathbf{p}$, his/her conception of the domain $\mathbf{c}_d$.

At this point, it should be noted that the observer's understanding of the purpose $\mathbf{p}$ is essentially a conception as well, i.e. the conception of the purpose of the model-to-be $\mathbf{c}_p$. Even more, the observer **O** also forms the conception of the model-to-be, $\mathbf{c}_m$. The modelling process thus actually consists in the observer's

[4] Obviously, the observer's judgement may be influenced by many different factors, e.g. observer's intentions, experience, previous knowledge, interests, etc. Our discussion excludes the from the consideration the potential conscious *political intentions* underlying the observer's judgement.

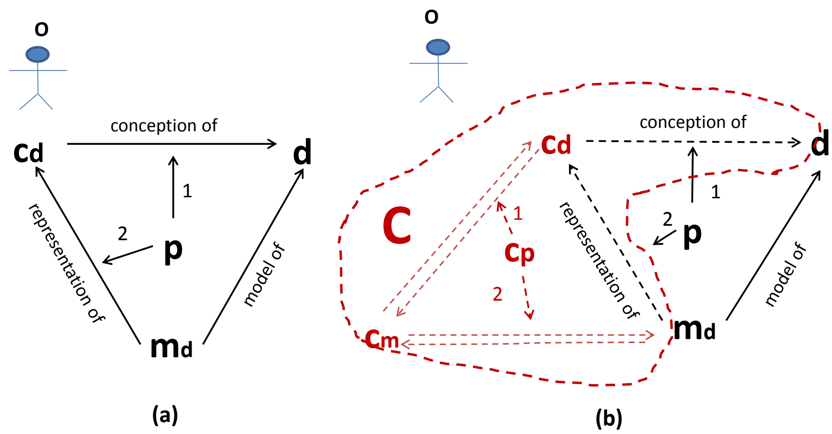

Fig. 1. The act of modelling

gradual alignment of these three conceptions, in parallel with the very shaping of the (model-to-be) artefact. This is illustrated in the Figure 1b.

As this alignment is iterative, the conception of the domain $\mathbf{c}_d$ is not completely stabilised before the artefact (model-to-be) is shaped in a satisfactory manner. It is only at this point that we can speak about the *existence* of the domain **d**, as pointed out in FRISCO [18]. The domain as such does not exist[5] a priori, but emerges in the very act of its modelling, and is being very much shaped by its observer in the given modelling situation.

When the mutual alignment of $\mathbf{c}_d$, $\mathbf{c}_p$ and $\mathbf{c}_m$ is achieved, the artefact comes to be acknowledged as the *representation of* the (conception of the) domain **d** for the purpose **p**. In other words, the observer **O** *acknowledges* that the artefact **m** is a *model of* the domain **d** for the purpose **p**.

Given the modelling situation, the purpose thus determines which features of the domain should be modelled and with which accuracy [45]. This is depicted as the influence of the purpose **p** on the relationship *representation of* in Figure 1. Similar distinction is made in [45], where a model is considered as a special kind of representation, having a *purpose* and a *cost-effectiveness criterion*.

The previous explanation holds for a modelling process where the observer is an individual. In a collaborative modelling situation, a group of **n** human actors is involved in the process of modelling, and is supposed to *jointly* observe some domain **d** and come up with its model $\mathbf{m}_d$, for the purpose **p**. In order to reach a

[5] The term *exist* is used here in the sense of Heidegger's notion of *breaking down*, discussed in [52]. Indeed, "*Heidegger insists that it is meaningless to talk about the existence of objects and their properties in the absence of concernful activity, with its potential for breaking down. What really is is not defined by an objective omniscient observer, nor is it defined by an individual – the writer or computer designer – but rather by a space of potential for human concern and action*" [52, p.37].

shared view on the domain, the purpose and the model-to-be, the co-alignment of potentially $\mathbf{n}\times\mathbf{3}$ conceptions (i.e. $\mathbf{c}_d$, $\mathbf{c}_m$, $\mathbf{c}_p$ per each of $\mathbf{n}$ actors) has to take place. Indeed, this is considered as a critical step in collaborative modelling, where all the discussions, negotiations and agreement about the model need to take place.

As we have seen, the main factor of the alignment of conceptions is the purpose $\mathbf{p}$, i.e., more precisely, the conception of the purpose by an observer, $\mathbf{c}_p$. As the purpose $\mathbf{p}$ is the main discriminant of the value of created model $\mathbf{m}$, we argue that it should be explicitly considered when creating (and using) models. In other words, the purpose should also be modelled. Explicitly considering and expressing the purpose is important not only to facilitate the (usually implicit) aligning of conceptions, but also to enable the understanding of a model by an observer who was not involved in process of its creation.

Furthermore, and in line with [9, 48, 29], we argue that the purpose should be the primary driver of shaping (the choice of) modelling language. The modelling language used for modelling should allow expressing the model in such a way that the model is of value for its intended purpose. Some related work embraces this view regarding model and language quality assessment [34], as well as modelling processes [8]. In our research, this position is explored with regards to the definition and use of modelling languages in modelling.

4 Role of Modelling Language in Modelling

Having introduced our fundamental understanding of the modelling act, this section develops our view on the role of modelling language within modelling. We will use such an understanding to discuss the challenges inherent to definition of a fixed modelling language, and more specifically, to its standardisation.

4.1 Grounding the View on Language

In our view, the language consists of a system of symbols whose primary function, i.e. *raison d'être*, is to act as an instrument of human communication and action. Language is thus *used* to formulate and communicate human conceptions of various aspects of the "world", in various communities and communicative circumstances.

This view on language grounds in the body of knowledge of cognitive linguistics [23], functional perspective on language [13, 12, 52], and semiotics [18, 47]. It is adopted in our study of modelling languages, as the phenomena we are interested in go beyond the isolated study of linguistic code, and puts forward its *use* within different modelling situations. We look at the extent to which a fixed/standardised modelling language allows to effectively formulate and communicate conceptions in a given modelling situation, and how it can be designed to better suit this need.

4.2 Elements of Modelling Language Definition

Traditionally, a modelling language is defined in terms of *abstract syntax*, *concrete syntax* and *semantics*.

The **abstract syntax** of a modelling language defines the modelling constructs and rules for their combination when creating models. The abstract syntax of visual modelling languages is usually represented using metamodels.[6]

The **concrete syntax** or *notation* deals with the representation of a modelling language on medium. The medium itself can be restricted to a specific form, such as graphical, textual, or video, but the notation in general can also be restricted in terms of fonts, icons and layout rules. Concrete syntax defines symbols (according to the medium) and rules for their combination, as well as their correspondence to the abstract syntax of the language. The role of notation in modelling is thorougly discussed in [37].

The **semantics** of a modelling language deals with its meaning. It is conventionally defined in terms of a *semantic domain* and a *semantic mapping* [24]. According to [24], the semantic domain captures the "*decisions about the kinds of things language should express*" [24, p. 68], while the semantic mapping establishes the correspondence from syntactic elements to the semantic domain.

It is often considered that the abstract syntax of the modelling language does not deal with semantics [24]. Nonetheless, the metamodel of the modelling language actually represents a particular **conceptual foundation** of the language, i.e. a specific classification of concepts to be used in discourse about the "world" [18]. The metamodel thus provides a particular *ontological position*, as it filters the view on the "world" one chooses within the modelling language [18]. It is even argued in [18] that all other aspects of the modelling language depend on its conceptual foundation. Thus, the conceptual foundation is an important (if not crucial) aspect of the modelling language semantics. It may be even argued, based on the research in the area of linguistics (e.g. [13, 23]) that syntax and semantics of a language are not that clear-cut.

In the following, we will argue that this traditional approach to modelling language definition, and in particular standardisation, needs to be complemented with another dimension, that of **modelling pragmatics** [49]. Pragmatics is concerned with the use of language signs by the user, in the context/situation in which and the purposes for which they are used, as well as meaning and effect they have in their context of use [49, 1]. Though it is not widely studied [49], we will show that inadequate consideration of pragmatic aspects of modelling, when defining a modelling language, may affect its capacity to effectively perform its function in modelling. We will equally show that an explicit consideration of pragmatic aspects may provide valuable insights for the scoping and a the design of a better-suited modelling languages.

---

[6] The advantages or disadvantages of using metamodelling for representing abstract syntax are discussed in e.g. [24].

4.3 Function of the Modelling Language

Modelling language can be regarded as having a *twofold function* in modelling:

1. *Representational function*, the function of representation system for expressing models, in particular for their mechanical manipulation, and
2. *Linguistic function*, the function of a natural language to be used in conceptualising and communicating about some domain in a particular modelling situation.

This twofold functioning puts different and often conflicting requirements on the modelling language definition.

Representational Function. In its representational function, a modelling language should accommodate the formulation of models, while allowing their mechanical manipulation. For this purpose, the representation system should *at least* have a well defined *abstract* syntax *prior* to developing tools that implement model manipulations. To reuse the implemented manipulations, the representation system is also typically required to remain *fixed* once defined, i.e. not evolving dynamically in different situations of use.

The potential added value of the modelling language, from this perspective, lies primarily in 1) the re-usability of the representation system and of the associated manipulations across different modelling problems, and 2) the extent to which the language specification is machine readable.

The re-usability of a language relates to its *expressiveness* [18], i.e. to how many different (conceptions of) domains a modelling language allows to model. It can largely be influenced *at language design time*, when its designers identify and restrict the *intended set of models* expressible by the representation system. This is done through the choices relative to the (levels of genericity of the) *conceptual foundation* and to the *syntactic-semantic restrictions* incorporated into the language definition.

The requirement of machine readability is driven by the need for the automated manipulation of models. For representations produced in the modelling language to be precisely interpreted (by machines), a *formal*, i.e. precise and unambiguous, definition of both abstract syntax and semantics of the representation system, usually in a mathematical language, is required. In this context, modelling language is seen as a formal language, and its definition is of normative character.

However, the formal perspective on modelling languages focuses on purely referential aspects of meaning [13], and excludes all context-dependent aspects of meaning in language [1, 49], which fall in the area of pragmatics. In particular, this perspective on modelling languages disregards the potential influence of 'labels' from natural language (as part of language's conceptual foundation) on the understanding of the modelling language by an observer, as well as on its use in the process of domain conceptualisation in the given modelling situation. Clearly, from this perspective, a modelling language does not function as a 'human language', but only as an ultimately syntactic carrier of models.

Linguistic Function. In its linguistic function, a modelling language should provide *language* to support the activities of conceptualisation and communication taking place in a modelling situation [27, 42].

From this perspective, the potential added value of a modelling language consists in its capacity to 1) frame the discourse about the domain in a modelling situation, and to 2) facilitate shaping and expressing the conception of a domain formed/agreed by an observer in a model that is fit for its purpose. These dimensions combined provide what is usually referred to as the *suitability* or *utility* [42] of a modelling language.

Whether the given modelling language is suitable for some modelling problems is largely contextual and cannot be fully determined *a priori*. The pragmatic aspects of a modelling situation in which the given modelling language is *actually used* determine the degree of its suitability. This point is developed in the following discussion.

As defined in Section 3, a modelling situation is at least characterised by a wider context of modelling, the goals of the situation (including the purpose of the model), and the observer involved in modelling. Additionally, the fixed/standard modelling language to use, **ML**, may also be (and typically is) selected *prior* to engaging in the modelling effort. As illustrated in Figure 2a, the central question, in this context, is to which extent the selected **ML** is capable to effectively support the creation of conception $\mathbf{c}_d$, as well as its externalisation into **m** suitable for the purpose **p** (in terms of representational system provided by **ML**). Let us try to provide a tentative answer to these questions.

As discussed in 4.2, **ML** provides an embedded filter on the "world", i.e. its conceptual foundation. In the modelling situation, this filter is meant to constrain, or at least influence, the way observer **O** forms the conception of a domain $\mathbf{c}_d$. To which extent an 'externally' imposed language **ML** can interact with the observer's process of conceiving a particular domain **d**?

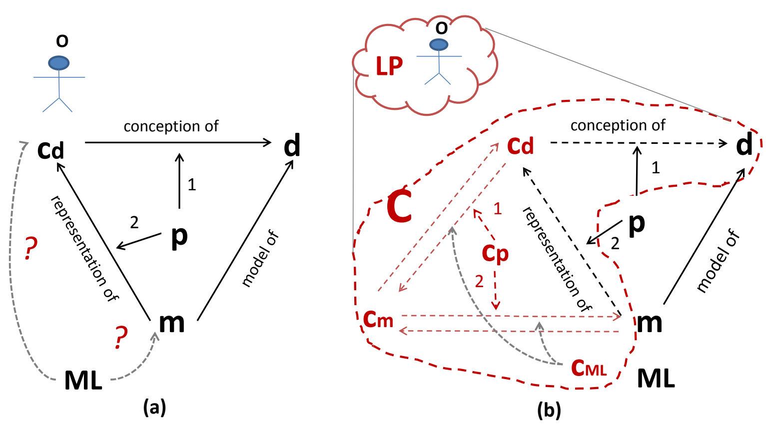

Fig. 2. The role of modelling language in a modelling act

The way in which a particular individual/observer observes and conceptualises the "world" is shaped by the factors of biological, cognitive, cultural, and social (thus also educational and professional) background of the individual [44]. These factors shape the linguistic personality [21] of the individual, which is illustrated as a space **LP** around observer **O** in Figure 2b. This is also addressed as the individual's world view, or the observer's pre-conceptions in [42]. The **LP** of an observer **O** affects his/her *default interpretation* of concepts embedded in **ML** [50], i.e. his/her *default* $\mathbf{c}_{ML}$. The shaping of $\mathbf{c}_{ML}$ is further influenced by the characteristics of the modelling situation: its wider context, e.g. the particular enterprise and/or project, language(s) spoken in (particular groups within) the enterprise [26], model purpose **p**, other participants (in the case of collaborative modelling) etc. In this context, the understanding of conceptual foundation of **ML** follows the principles of human use of natural languages. In the use of natural languages, the words and/or linguistic utterances are given their precise meaning within the entire context of linguistic communication, and the function linguistic utterances have in the communication context [13, 12]. Therefore, the contextualised understanding of **ML**, i.e. *contextualised* $\mathbf{c}_{ML}$, arises by taking into account the entire context in which modelling takes place.

An expert modeller can be expected to have less difficulties in understanding and using a particular **ML**, because of his/her education and consistent experience in modelling. However, it is reasonable to assume that a prototypical stakeholder of enterprise modelling (involved as observer in a modelling situation) requires an increased *mental effort* [5, 3] to understand and use **ML** [56]. In fact, the **LP** of a prototypical enterprise modelling stakeholder differs most likely significantly from the specific ontological position, i.e. level of abstraction, embedded in **ML**. The latter typically provides the level of abstraction higher than the one in which most stakeholders are used to reason. Adopting higher than usual levels of abstraction in reasoning also increases mental effort needed for stakeholders in this task [51]. The training and consistent experience in *using* a **ML** by stakeholders may possibly remedy this, but it is unlikely that each enterprise modelling stakeholder will be trained to the modelling languages used in his/her particular context.

If we add to this picture the typical conceptual complexity of standardised **ML** (see Section 2), it can quickly result in *cognitive overload* [5, 36] observers, when *only* trying to understand the **ML**. Such a situation rapidly hinders the understanding of other elements in the modelling effort [56], i.e. $\mathbf{c}_d$, $\mathbf{c}_m$ and $\mathbf{c}_p$. This is one of the potential sources of problems with the use of **ML** in the practical modelling situations.

Secondly, **ML** is meant to affect observer's creating and aligning of conceptions $\mathbf{c}_d$ and $\mathbf{c}_m$, as illustrated in Figure 2b. As model purpose **p** at the same time drives this process(see Section 3), the given **ML** should provide sufficient constructs to form $\mathbf{c}_d$, and allow to express (by the means of **ML**'s representation system) the $\mathbf{c}_d$ for the purpose **p** into an artefact **m**. Whether a particular **ML** used in a modelling situation allows to do so is *not* only dependent on the selected **ML**, but primarily on the model purpose **p**, and the observer **O**, as discussed so far.

If, in the given modelling situation, **ML** (i.e. $\mathbf{c}_{ML}$) lacks constructs to express all the relevant aspects of $\mathbf{c}_d$ for a purpose **p**, the natural attitude in the modelling practice is to invent a 'dialect' that provides the best *cognitive fit* [37] for the purpose **p** at hand. As the **ML** is typically a *fixed* language, it is not likely that all the possible relevant concepts and constructs for all the modelling situations could be preconceived in such a language anyway. This suggests that the need to adapt the modelling language in the given modelling situation is likely to be present despite the standardisation/unification efforts.

What these discussions suggest is that, in order to maximise its added value in terms of linguistic function, the **ML** should be as closely aligned as possible with the characteristics of the *actual modelling situation* in which it is used, e.g. in terms of the provided ontological position, vocabulary, coverage of specific aspects, level of detail in modelling, specific form and symbols used, etc. This also means that an ideally suitable language could not be *a priori* fixed, but that the adaptability to specific modelling situations has to be *designed in* the language.

Moreover, when defining a fixed modelling language or when selecting it for a particular modelling effort, one should be careful when assuming that the influence of the observer's **LP** on his/her conceptions can be overcome by imposing a filter from **ML** from which to look at the domain. Our discussions rather suggest that the conceptual foundation of **ML** chosen for a modelling situation should be as close as possible to the way stakeholders would discuss the particular modelling problem, if there would be no restrictions on the language to use whatsoever[7].

This is even more critical for language standardisation efforts. The *standardised* **ML** is typically thought as an effective solution to also standardise the (conceptualisation and) communication about some problem area, regardless of the involved stakeholders, and the purposes for which modelling should be done. The discussions so far rather suggest that, because of differences in individuals' world views, the $\mathbf{c}_{ML}$ is likely to be different for each individual, even in the case of standard **ML**. It indeed suggests that meta-discussions are likely unavoidable, if not even necessary to reach a sufficiently similar $\mathbf{c}_{ML}$ between human actors involved in modelling. Furthermore, it also suggests that the degree of complexity of a modelling language has to be manageable by humans, such that the language indeed facilitates rather than hinders the modelling effort.

5 Pragmatics Under the Carpet of Standardisation?

The previous section identifies and discusses the opposing forces that influence the definition of a modelling language. These forces, stemming from its aforementioned functions, evidently need to be carefully balanced so that the modelling

[7] In this context, the notation of **ML** can also play an important role in facilitating understanding and use of **ML**, see [37].

language provides the added value when *used* in modelling situations. Based on the discussions in Sections 2 and 4, we can observe that standardisation efforts mainly aim to maximise the potential benefits of the modelling language in its representation function. In these efforts, its linguistic function is very lightly, if at all, considered.

To support this discussion, we will analyse the BPMN standard and its use (as reported in [57, 58, 43, 53, 20]). The OMG's BPMN is developed with the primary goal "*to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes.[..] In doing so, BPMN will provide a simple means of communicating process information to other business users, process implementers, customers, and suppliers*" [39, p.31].

Business users are identified in the specification document as the primary audience of BPMN models. At the same time, BPMN aims to create standardized bridge between business process design and process implementation, and to enable visualizing process models defined in languages optimized for their execution.

How does the BPMN standard reflect the needs of its primary business users, and how does it support creating models for these 'business-oriented' purposes? According to [58, 43], these purposes in practice are mainly process documentation, continuous process management, knowledge management, and organisation redesign. Furthermore, a very small subset of essential BPMN constructs is used for these purposes [58][8]. However, this does not imply that such a small subset is sufficient for the needs of business users. As reported in [53, 43], practitioners lack constructs for expressing business rules, organisational resources and roles, risks, performance, etc. [43, 53]. This corroborates the pattern of extension of BPMN models observed in [43]: the models are extended with the symbols allowing to capture organisational information, such as data, risks, resources, documents etc. "*This situation points to BPMN being a pure process modeling language. Users, however, often are concerned with enterprise modeling [...] beyond the mere depiction of the control flow of their business operations.*" [43, p.189].

While targeting primarily business users, a closer look at the internal complexity of BPMN standard reveals the language rather geared towards the advanced technical purposes of process modelling, e.g. workflow engineering, process simulation and systems specification etc [43]. However, such an advanced modelling is very rarely applied in practice [43]. Furthermore, a rather significant numbers of constructs of BPMN standard is reported by practitioners as superfluous, e.g. some highly differentiated event constructs, gateways etc [58, 43]. This complexity is reported to negatively affect the ease of use of the language [43, 20]. If reported as overwhelming, should the language complexity then somehow

[8] We have discussed the usage of BPMN constructs for different purposes in more details in Section 2.2 of the paper.

be managed? Should the constructs used only for advanced process modelling purposes be hidden from other users of the language?

Similar observation of the BPMN complexity motivated the work on defining a ‘simplified’ version of the language, namely Simple BPMN (SBPMN) [20], targeted at business users of process modelling. The original standard is simplified by excluding or modifying constructs that require technical knowledge or that can cause confusion when used by business users. It does not surprise that the usability tests performed and reported in [20] asses SBPMN as easier to use for business users than BPMN.

Therefore, we can observe that BPMN targets too broad a range of process modelling purposes, as well as very diverse audience. The approach to its standardisation is oriented towards maximising the benefits of *only* the *representation system* across rather technical uses of process models. However, BPMN does not seem to accommodate well the needs of its business audience: it does not come with relevant constructs for modelling processes from the business perspective. Also, as a fairly complex language it is difficult to use by this audience. In our view, the standardisation is done without adequately considering the pragmatic needs (in terms of audience, model purposes, constructs needed to structure the discourse about the domain) of modelling situations in which BPMN is/should be used. As we have seen, this affects its suitability, i.e. how it performs the linguistic function. The examples of SBPMN and of BPMN extensions covering organisational aspects illustrate the strategies to overcome the lack of BPMN’s suitability. Such ‘variants’ seem to have better chances of cognitive fit for ‘business-oriented’ purposes of process modelling, and they certainly demonstrate better cognitive effectiveness than the full-blown standard.

More importantly, this raises the question whether such ‘simplified variants’, tuned to particular purposes/audiences, should be part of a standard process modelling language, too. Could the BPMN standard fulfil its stated objectives effectively, if it does not include these ‘variants’? Similar findings result also from the study of practical use and support needed by architectural languages [35].

6 Conclusion & Outlook

Instead of sweeping pragmatics under the carpet, how should the pragmatic needs for language support, as well as language use data, stand in relation with the standard? Should the needs of language users be the driving force in creating and scoping the (standard) language?

To shed more light on this question, this paper discussed the role of the modelling language from a fundamental perspective. We identified its twofold role in modelling, and argued that it has to be carefully considered and balanced when defining the (standard) modelling language. In other words, we argued that it makes sense to explicitly consider the pragmatic needs when defining, revising or evolving a fixed modelling language.

What is the general competence area of a modelling language, e.g. process modelling language? Should the process modelling language cover only a single, process, aspect or include aspects such as business rules, resources, and similar aspects related to organisational modelling, which are relevant for purposes such as process documentation, process monitoring and improvement, and re-engineering? The same questions can be asked for any modelling language. Our research suggests that the scope of a fixed modelling language should be in line with the context in which it will be used, i.e. in line with purpose(s) for which models are produced using that language. If enough support is not a priori provided in the language, purpose-specific 'variants' will emerge to compensate for the missing suitability.

If the fixed modelling language aims to cover too many purposes at the same time, this is more challenging, as the language tends to become overly complex if there is a strive to accommodate all the different pragmatic needs. Nonetheless, these needs cannot be swept under the carpet, as they will reappear when the language is used in the actual modelling situations. Should then the standard language for e.g. process modelling be reorganised into multiple (modular) languages within a standard for process modelling? Does this suggest a different approach towards the standardisation of modelling languages?

We believe this is one of the promising directions to explore in the future research. The modular organisation of languages could have language 'chunks' scoped and geared towards purpose(s) for which the 'chunk' is used. The *right* language for the modelling situation at hand could then be woven out of the different 'chunks', based on the pragmatic needs of the actual modelling situation. This also suggests that the pragmatic aspects of the 'chunks' need to be made as explicit as it is possible. Such a modular organisation could also decrease the cognitive load needed for the understanding, selecting and using the language for chosen modelling purposes.

In addition, the needs for *situational adaptability* and *evolution* of the language will always be present, as it is not feasible to pre-define all the possible circumstances in potential modelling situations, nor to predict the evolution of the 'reality' for whose modelling the language support will be needed. The strategy of including all these missing aspects within a *single* language would make it at some point overly complex and virtually unusable. In our view, this is another argument in favour of modular organisation of modelling languages.

Of course, this calls for more research on instruments of modelling language creation, adaptation and combination, as well as more research on the related tool mechanisms supporting this. Indeed, there is a growing research interest in modelling language and tool flexibility [32, 11, 54]. Our research aims to contribute to this stream of research by providing a conceptual framework from which to understand the role and, in particular, the use of the modelling language. This paper argued that the added value of a modelling language cannot be evaluated without considering its use, i.e. pragmatics dimension. Our belief is that only by identifying the factors determining the added value of the language, we can be in the position to make grounded decisions on optimal scope and design of the language.

ACKNOWLEDGEMENTS

This work has been partially sponsored by the *Fonds National de la Recherche Luxembourg* (`www.fnr.lu`), via the PEARL programme.

References

1. Allwood, J.: Some Remarks On the Relationship Between the Semantic and the Pragmatic Web. In: ICPW. (2008) 35–39
2. Anaby-Tavor, A., Amid, D., Fisher, A., Bercovici, A., Ossher, H., Callery, M., Desmond, M., Krasikov, S., Simmonds, I.: Insights into enterprise conceptual modeling. Data Knowl. Eng. **69**(12) (2010) 1302–1318
3. Baddely, A.: Working memory: Theories, models and controversies. Annual Review of Psychology **63** (2012) 1–29
4. Band, I., Bjeković, M., Chaney, C., Lindokken, K., van Dis, E.: Modeling the Insurance Enterprise. Technical Report W12B, The Open Group (2012)
5. Bannert, M.: Managing cognitive load-recent trends in cognitive load theory. Learning and Instruction **12** (2002) 139–146
6. Bjeković, M., Proper, H.A., Sottet, J.S.: Towards a coherent enterprise modelling landscape. In: PoEM (Short Papers), CEUR-WS.org (2012)
7. Bjeković, M., Sottet, J.S., Favre, J.M., Proper, H.: Framework for Natural Enterprise Modelling. In: IEEE CBI Conference. (2013) To appear.
8. Bommel, P.v., Hoppenbrouwers, S., Proper, H., Roelofs, J.: Concepts and strategies for quality of modeling. In: Innovations in Information Systems Modeling. IGI Publishing (2008)
9. Bubenko, J.A.j., Persson, A., Stirna, J.: An Intentional Perspective on Enterprise Modeling. In Salinesi, C., Nurcan, S., Souveyet, C., Ralyté, J., eds.: Intentional Perspectives on Information Systems Engineering. Springer (2010) 215–237
10. Chiprianov, V., Alloush, I., Kermarrec, Y., Rouvrais, S.: Telecommunications Service Creation: Towards Extensions for Enterprise Architecture Modeling Languages. In: ICSOFT (1). (2011) 23–28
11. Cho, H., Sun, Y., Gray, J., White, J.: Key challenges for modeling language creation by demonstration. In: ICSE FlexiTools Workshop. (2011)
12. Clark, H.: Arenas of Language Use. University of Chicago Press (1993)
13. Cruse, A.: Meaning in Language: An Introduction to Semantics and Pragmatics. Oxford Textbooks in Linguistics. Oxford University Press (2011)
14. Davies, I., Green, P.F., Rosemann, M., Indulska, M., Gallo, S.: How do practitioners use conceptual modeling in practice? Data Knowl. Eng. **58**(3) (2006) 358–380
15. de Kinderen, S., Gaaloul, K., Proper, H.A.: Integrating Value Modelling into ArchiMate. In: IESS. (2012) 125–139
16. Elahi, G., Yu, E.S.K., Annosi, M.C.: Modeling Knowledge Transfer in a Software Maintenance Organization - An Experience Report and Critical Analysis. In: PoEM 2008. (2008) 15–29
17. Fabro, M.D.D., Valduriez, P.: Towards the efficient development of model transformations using model weaving and matching transformations. Software and System Modeling **8**(3) (2009) 305–324
18. Falkenberg, E.D., Hesse, W., Lindgreen, P., Nilsson, B.E., Oei, J., Rolland, C., Stamper, R., Van Assche, F., Verrijn-Stuart, A., Voss, K.: FRISCO - A Framework of Information System Concepts - The FRISCO Report. Technical report, IFIP WG 8.1 Task Group FRISCO (1998)

19. Feltus, C., Dubois, E., Proper, H.A., Band, I., Petit, M.: Enhancing the ArchiMate® standard with a responsibility modeling language for access rights management. In: SIN. (2012) 12–19
20. Fernández, H.F., Palacios-González, E., García-Díaz, V., Pelayo, G., Bustelo, B.C., Sanjuán Martínez, O., Cueva Lovelle, J.M.: SBPMN – An easier business process modeling notation for business users. Computer Standards & Interfaces **32**(1) (2010) 18–28
21. Fesenko, O.P.: Structure of Linguistic Personality in the Aspect of the Center-Periphery Theory. Middle-East Journal of Scientific Research **16(3)** (2013) 397–401
22. Frank, U.: Some Guidelines for the Conception of Domain-Specific Modelling Languages. In: EMISA. (2011) 93–106
23. Geeraerts, Dirk and Cuyckens, H., ed.: The Oxford Handbook of Cognitive Linguistics. Oxford University Press (2010)
24. Harel, D., Rumpe, B.: Meaningful Modeling: What's the Semantics of "Semantics"? IEEE Computer **37**(10) (2004) 64–72
25. Hoppenbrouwers, S.J.B.A., Wilmont, I.: Focused Conceptualisation: Framing Questioning and Answering in Model-Oriented Dialogue Games. In: PoEM 2010. (2010) 190–204
26. Hoppenbrouwers, S.: Freezing Language; Conceptualisation processes in ICT supported organisations. PhD thesis, University of Nijmegen, Nijmegen, The Netherlands (2003)
27. Hoppenbrouwers, S., Proper, H., Reijswoud, V.v.: Navigating the Methodology Jungle – The communicative role of modelling techniques in information system development. Computing Letters **1**(3) (2005)
28. Hoppenbrouwers, S., Proper, H., Weide, T.v.d.: A Fundamental View on the Process of Conceptual Modeling. In: Conceptual Modeling - ER 2005. Volume 3716., Springer (2005) 128–143
29. Hoppenbrouwers, S., Proper, H., Weide, T.v.d.: Understanding the Requirements on Modelling Techniques. In: CAiSE. (2005) 262–276
30. Iacob, M.E., Jonkers, H., Lankhorst, M., Proper, H., Quartel, D.: ArchiMate 2.0 Specification. The Open Group (2012)
31. Karlsen, A.: Enterprise Modeling Practice in ICT-Enabled Process Change. In: PoEM 2011. (2011) 208–222
32. Kimelman, D., Hirschman, K.: A Spectrum of Flexibility-Lowering Barriers to Modeling Tool Adoption. In: ICSE FlexiTools Workshop. (2011)
33. Kort, C., Gordjin, J.: Modeling Strategic Partnerships Using the E3value Ontology: A Field Study in the Banking Industry. In: Handbook of Ontologies for Business Interaction. IGI Global (2008) 310–325
34. Krogstie, J., Sindre, G., Jørgensen, H.D.: Process models representing knowledge for action: a revised quality framework. EJIS **15**(1) (2006) 91–102
35. Malavolta, I., Lago, P., Muccini, H., Pelliccione, P., Tang, A.: What Industry Needs from Architectural Languages: A Survey. IEEE Trans. Software Eng. **39**(6) (2013) 869–891
36. Moody, D.L.: Cognitive load effects on end user understanding of conceptual models: An experimental analysis. In: ADBIS. (2004) 129–143
37. Moody, D.: The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Transactions on Software Engineering **35**(6) (2009) 756–779
38. OMG: UML 2.0 Superstructure Specification – Final Adopted Specification. Technical Report ptc/03–08–02 (August 2003)

39. OMG: Business Process Model and Notation (BPMN), Version 2.0 (January 2011)
40. OMG: OMG Systems Modeling Language (OMG SysML), Version 1.3 (June 2012)
41. Opdahl, A., Berio, G., Harzallah, M., Matulevicius, R.: Ontology for Enterprise and Information Systems Modelling. Applied Ontology **7(1)** (2012) 49–92
42. Proper, H.A., Verrijn-Stuart, A.A., Hoppenbrouwers, S.: On Utility-based Selection of Architecture-Modelling Concepts. In: APCCM 2005. (2005) 25–34
43. Recker, J.: Opportunities and constraints: the current struggle with BPMN. Business Proc. Manag. Journal **16**(1) (2010) 181–201
44. Rohrer, T.: Embodiment and Experientialism. In Geeraerts, D. and Cuyckens, H., ed.: The Oxford Handbook of Cognitive Linguistics. Oxford University Press (2010) 25–47
45. Rothenberg, J.: The Nature of Modeling. In: Artificial intelligence, simulation & modeling. John Wiley & Sons, Inc., USA (1989) 75–92
46. Stachowiak, H.: Allgemeine Modelltheorie. Springer, Berlin, Germany (1973)
47. Stamper, R., Liu, K., Hafkamp, M., Ades, Y.: Understanding the roles of signs and norms in organizations - a semiotic approach to information systems design. Behaviour & Information Technology **19**(1) (2000) 15–27
48. Thalheim, B.: The Theory of Conceptual Models, the Theory of Conceptual Modelling and Foundations of Conceptual Modelling. In: Handbook of Conceptual Modeling. Springer (2011) 543–577
49. Thalheim, B.: Syntax, Semantics and Pragmatics of Conceptual Modelling. In: NLDB, Springer (2012) 1–10
50. van der Linden, D.J.T., Hoppenbrouwers, S., Lartseva, A., Molnar, W.: Beyond terminologies: Using psychometrics to validate shared ontologies. Applied Ontology **7(4)** (2012) 471–487
51. Wilmont, I., Hengeveld, S., Barendsen, E., Hoppenbrouwers, S.: Cognitive mechanisms of conceptual modelling - how do people do it? In: ER. (2013) 74–87
52. Winograd, T., Flores, F.: Understanding Computers and Cognition - A New Foundation for Design. Ablex Publishing Corporation (1986)
53. Wohed, P., van der Aalst, W.M.P., Dumas, M., ter Hofstede, A.H.M., Russell, N.: On the Suitability of BPMN for Business Process Modelling. In: Business Process Management. (2006) 161–176
54. Zarwin, Z., Bjeković, M., Favre, J.M., Sottet, J.S., Proper, H.: Natural modelling. Journal Of Object Technology (2014) To Appear.
55. Zivkovic, S., Kühn, H., Karagiannis, D.: Facilitate Modelling Using Method Integration: An Approach Using Mappings and Integration Rules. In: ECIS. (2007) 2038–2049
56. Zugal, S., Pinggera, J., Reijers, H., Reichert, M., Weber, B.: Making the case for measuring mental effort. In: Proceedings of the Second Edition of the International Workshop on Experiences and Empirical Studies in Software Modelling, ACM (2012) 6
57. zur Muehlen, M., Ho, D.T.: Service Process Innovation: A Case Study of BPMN in Practice. In: HICSS. (2008) 372
58. zur Muehlen, M., Recker, J.: How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. In: CAiSE. (2008) 465–479