On the cognitive understanding of types in modeling languages

Dirk van der Linden[1,2,3] and Henderik A. Proper[1,2,3]

[1] Public Research Centre Henri Tudor, Luxembourg, Luxembourg
[2] Radboud University Nijmegen, the Netherlands
[3] EE-Team, Luxembourg, Luxembourg
dirk.vanderlinden@tudor.lu, e.proper@acm.org

Abstract: We investigate how enterprise modelers see common types (e.g., actor, event, process) used in most modeling languages in terms of their semantic feature structure (e.g., is human, is material). We hypothesize that modelers have specific interpretations for some of these common types that affect their range of conceptually valid instantiations (e.g., actors should not be instantiated as human things). Based on two exploratory psychometric studies performed with enterprise modeling practitioners and computing science students we discuss the way these typical interpretations affect their model(ing) semantics (e.g., results typically having to be modeled as well-described and non-natural entities, restrictions typically as logical necessities), and what consequences these findings have for modeling languages and the use and creation of models themselves, especially in an inherently collaborative effort like enterprise modeling. We conclude by arguing that insights into these conceptualizations are likely useful and should receive more attention and studies.

1 Introduction

While the quality of a conceptual model can be interpreted differently depending on what people expect from it [FHVL12, HFL12], its validity is contingent on its ability to communicate its intended meaning clearly and completely [Rob06]. As models are there to support the building and exchange of knowledge [Sta00], it is important that we can be sure that the semantics in a model reflect the semantics intended by the modeler. This goes further than merely ensuring that the model is capable of expressing those semantics that the original modeler wanted to (with, e.g., test scenarios). We need to additionally ensure that the model does not allow one to express situations that, in the real world, are considered conceptually invalid (or simply illogical) by the original modeler and stakeholders.

We focus here primarily on domain models that are created in order to communicate about a domain with other people. With such models it is important to avoid *mismatched understandings between different modelers and stakeholders* [KSKS12], as they can have a disastrous effect on both the produced models and the modeling process itself. As such, it is important to ensure that a user of the model does not have to guess about the semantics expressed in it. They might not have been involved in the original modeling

process, or come from a different background with their own set of typical understandings, and thus not share the same conceptualization as the original modelers.

What is typically already done to ensure the validity of such domain models is to reach consensus on what type to represent elements from the universe of discourse by, and how to call those representations. However, such discussions often take form of agreeing on the type to be used for such elements (e.g, “Let us model this pizza as a RESOURCE.", “Let us model the delivery of the pizza as a PROCESS."), while not necessarily going into detail on just what that type (‘resource’ or ‘process’) actually *is* in this context. Instead, the deeper semantics often stay implicit. For example, the business process for pizza deliveries might involve restrictions that are logical necessities and cannot be broken (e.g., a pizza can not be delivered before it is made, thus correctly observing temporal dependencies) but also restrictions that can be broken, but perhaps *ought* not to because of moral or financial constraints (e.g., a pizza should be delivered within 30 minutes of its order). It is thus important that whatever type we choose to represent these restrictions accommodate the conceptual distinction between alethic (i.e., logical *necessity*) and deontic (i.e., how the world *ought* to be) modality. But, if we only agree that we model these kinds of restrictions as “rules", we leave this distinction implicit and depend on the (selection of the) modeling language to dictate what exactly the semantic status of both of these rules can be (let alone that users at times deliberately or unconsciously ignore a language’s semantics and invent their own, not always explicit, semantics [HS05]). As some languages accommodate less conceptual distinctions than others, that means relevant information might be lost.

Let us consider actors. If a modeler typically conceptualizes actors as human beings who take actions on their own accord, and wishes to model them so, it is necessary for whatever type is used from the modeling language to support the relevant parts of that conceptualization. In this case, making it explicit that any instantiation of an actor needs to be a single human being, and more specifically, a (in the given context) autonomous human being, for instance when modeling social decision making processes in an enterprise. If the produced model, however, is not explicit about these distinctions, there can be a host of conceptually related, but invalid, instantiations (e.g., an employee who needs permission for every action and can thus not be considered autonomous, a department with multiple persons). Such problems can be prevented by either using a modeling language that accommodates the necessary conceptual distinction, or by explicitly modeling such distinctions manually (e.g., by using unary constraints on the type for each necessary semantic feature). However, being aware of the relevant and necessary conceptual distinctions *before* the model creation process is what is often still lacking.

Thus, in order to ensure that a model can clearly and completely communicate its intended meaning, regardless of who is interpreting it at what time, a certain degree of conceptual alignment is needed. Specifically, an alignment between the conceptualization the original modeler has of his modeling concepts (i.e., the types used by a modeling language to represent the domain concepts) and the ‘official’, used, semantics of those modeling concepts (i.e., the semantics as found in the specification of the used language). This makes it easier to ensure that the modeling language (or dialect thereof) which will be used accommodates the needed conceptual distinctions. What is most lacking for this is either a simple and systematic procedure to uncover such

distinctions (which is difficult as poignant distinctions might simply never be explicitly talked about), or an initial set of commonly occurring distinctions that can be used as guidelines in such a process. When such distinctions have been discovered (either as generalized findings or specifically for a group), they can be of value to the modeling process by e.g., providing model facilitators with input for the facilitation process [RHLR11], or by providing explicits focus points for annotations in the model. Such factors should help enhance the quality of created models and improve consensus amongst stakeholders [SPS07].

Enterprise Modeling (EM) is an excellent environment to perform research into such commonly occurring distinctions. As it is a kind of collaborative modeling [SHP09,RGS+08], which is burdened with the difficulty of requiring "*people within the business to express their views in terms of a modeling language*" [BKV09], who often have a different view on things [RKV08] and then needing to integrate or link all those models written by different people in different languages different languages [Lan04,OB06], it is especially important to be aware of the different conceptualizations they might have of the types used in their models. Furthermore, it is possible that different aspects of the enterprise (e.g., goals, processes, value-exchanges) were modeled by separate groups and need to be integrated without access to (all) the involved people, showcasing again the need to be absolutely sure that the semantics expressed in the models can be relied on. Thus, one can expect a diverse amount of ways that modelers see different concepts.

We will first focus on investigating whether a small group of enterprise modelers have particularly specific conceptualizations of types used by conceptual modeling languages used in EM (e.g., i* GHYA07], BPMN Obj10], e3Value [GYvdR06], RBAC [FCK95], ITML [FHK+09]), and then investigate whether there are common conceptualizations (e.g., actors are considered as singular autonomous human beings, resources as materially existing things) that could be used as pointers for a procedure to uncover conceptual distinctions about these types. With these results we hope to show it is useful to have insights into such conceptualizations, that our approach can be used as a systematic procedure to uncover them, and that larger studies with more participants are warranted.

The rest of this paper is structured as follows. In section II we clarify what the objects of our investigation are, and in section III we set out our research methodology for investigating them. We show the most important results in section IV, discuss them and their consequences in sections V and VI, and finally conclude in section VII.

2 What we investigate

We need to look at someone's concept of a type in order to accurately investigate it. A concept in this context is the understanding that people have of the things they use to model, which is not necessarily the same as the understanding that is prescribed to them (by e.g., language specifications, official standards, or group 'consensus'). This should be done by investigating people, because "*semantic memory for concepts is based on a subject's memories of past experiences with instances of those concepts*" [Gee10] and because people generally do not think in the semantics of a given modeling language, but in the semantics of their own natural language [Sow10]. While we can investigate a

concept, we cannot measure it completely, but can instead characterize it (cf. [MAG+11,Pin07]) by looking which semantic features (i.e., distinctive attributes or properties of something that contribute to its meaning) resonate strongly with it for a person. This gives us information about the range and boundaries of the concept, which directly corresponds to the amount of conceptually valid instantiations of a type.
Different conceptualizations of a type can thus affect both the amount, and the content of the conceptually valid instantiations. When we look at the concept ACTOR with no given context, we could instantiate it with elements as human beings, computer hardware, abstract entities (e.g., agents), and organizational departments. However, when there is some context (i.e., from the characterization of the concept for that type), the amount of possible instantiatiable elements becomes smaller. Assume we know that someone sees actors as human 'things'. Computer hardware and abstract entities are now no longer conceptually valid instantiations. An organizational department might still be a conceptually valid instantiation of that concept as it is not uncommon for people to conceptualize as sets of human beings being essentially human. However, when we add to the context that actors are also considered single (non-composed) things, we do likely have to rule out a department as a conceptually valid actor. The way that an increasing amount of contextual information (in our case, relevant semantic features) affect the range of a concept can be understood in terms of quantum collapse in the semantic space of that concept (cf. [BC05,BW08]), which is backed up by recent psychological work on theory of concepts [GRA08]).

Consequently, we can effectively characterize the conceptual understanding people have of the types common to most modeling languages by investigating their concepts for them in the light of a set of relevant features. To do so, we will base ourselves on earlier work we performed [vdLHLP11] on the specifications of a number of languages and methods covering different aspects used in enterprise modeling (e.g., processes, value exchanges, goals, architecture, performance, security). This resulted in a set of high-level concepts that cover most types, and a set of features on which different languages were found to differ in their typical use of the concept.

The concepts we look at are actors, events, goals, processes, resources, restrictions and results. The different features which can be combined to characterize them are whether things are considered to naturally occur (natural), be human (human), are single things or composed of many (composed), are intentional or unintended (intentional), are logical necessities or not (necessary), are physically existing or not (material), and whether they are vague or well described (vague). A combination of any of these features can then be taken as a characterization of a given concept, for instance an actor being a natural, human, non-composed material thing.

It is important to note that not all features apply to each investigated concept (or are simply not informative), and that we investigate what people consider, regardless of whether it makes sense, such as e.g., people considering processes to be human things instead of abstract chain of events. Nonetheless, such findings of how people actually perceive (e.g., their personal ontology) the world, which might, for instance be choosing to look at a process in terms of the people performing the individual events in it instead of the events occurring can be useful for other purposes (see e.g., [Alm09] for an argument necessitating these endeavors to validate shared ontologies).

3 Method

We use a semantic differential [OST57] in order to investigate the conceptual understandings which participants have of the selected types. It is a widespread method used in different fields and areas of inquiry, and has well-researched specific quality guidelines for the Information Systems [VM07] field to ensure quality and validity of its results. The semantic differential can be used to investigate what connotative meanings apply to an investigated term, e.g., whether 'red' is typically considered good or bad. As input we will use the types and features described in the previous section. Each feature is described by a minimum of 5 adjectives (found through an earlier pilot study) that relates to the specific semantic of the feature to be investigated, ensuring statistical significance [VM07]. In order to ensure we investigate the typical understanding of each type, the differential task for each type is preceded by a semantic priming task in which participants are primed onto their typical understanding or use of the type.

We report on a study investigating practitioners (n=12), which was performed amongst employees of two internationally operating companies that provide support to clients with (re)design of organizations. The participants all had several years of experience as enterprise modelers, creating and using conceptual models and using diverse modeling techniques. Furthermore, we include results from an ongoing longitudinal study (n=10) into the (evolution of) understanding computing and information systems science students have of modeling types. This study started when the students began their university studies and had little to no experience, and will continue throughout their studies.

Results from the semantic differential are processed to give an average score for each type-feature combination stemming from the individual adjectives used for that feature. This results in a vector for each type, containing a score (a numerical value ranging from 2.0 to −2.0) describing for each feature how it relates to that type. Scores ≥ 1.0 were considered positive judgments, scores ≤ -1.0 were considered negative judgments. Other scores were considered neutral. The judgments are then used to calculate a percentage-wise breakdown of the amount of different judgments for each concept.

4 Results

Results for both the practitioners and students are shown in Table 1. The percentages are an aggregate reflecting the amount of negative, neutral and positive results. The amount of neutral responses can be taken as a measure of how open people's typical conceptualization of a concept is, in that it allows for more flexible (and possible amount of) instantiation. On the other hand, the negative and positive responses indicate that an instantiation of the concept would either need, or not need to display a certain feature. For example, when it comes to ACTORS, practitioners have 9% negative responses, 53% neutral and 38% positive. This means that about 9% of the responses indicate a feature that *has to be false* for a typical actor instantiation (e.g., a typical actor should not be a composed thing), 38% of the response indicate a feature that *has to be true* for a typical actor instantiation (e.g., a typical actor should be a human thing), while the remaining 53% allow for features to be either false or true (e.g., an actor can be either material or immaterial thing).

Table 1: Comparison of neutral and polar responses. Practitioners average 43.86% neutral responses, students 38.14%, 39.29% and 36.86%, being on average 5.75% less neutral

	actor	event	goal	process	resource	restriction	result
				Practitioners			
negative	9%	27%	22%	31%	30%	45%	45%
neutral	53%	53%	52%	40%	47%	32%	30%
positive	38%	19%	26%	29%	23%	22%	25%
				Students (phase 1)			
negative	26%	38%	44%	34%	30%	37%	29%
neutral	41%	44%	25%	36%	38%	43%	40%
positive	33%	17%	32%	30%	32%	20%	32%
				Students (phase 2)			
negative	16%	41%	34%	30%	27%	36%	39%
neutral	47%	41%	33%	47%	41%	37%	29%
positive	37%	17%	33%	23%	31%	27%	33%
				Students (phase 3)			
negative	7%	44%	39%	31%	33%	43%	31%
neutral	47%	44%	27%	37%	34%	33%	36%
positive	46%	11%	34%	31%	33%	24%	33%

Table 2 shows a number of strongly polarized type-feature combinations. The percentages here denote how many of the participants judged the type to need (or need not) to display the given (semantic) feature. For example, 91% of practitioners judged goals to need the 'necessary' feature, which means that the majority conceptualize goals as things that need to be achieved (or perhaps gained, if goals are also conceptualized as material objects). It can also be seen that 64% of practitioners judged processes to be non-vague things, meaning that they are (or should) be well-described things. The scores for students are shown as the scores per phase of the longitudinal study, and are only included for those type-feature combinations for which they also display strongly polarized judgments.

5 Discussion

As briefly discussed in the previous section, the results as seen in Tables 1 and 2, directly affect the total amount of conceptually valid instantiations for a given type. The higher the amount of neutral scores, and vice-versa the lower the amount of polarized (and more to the point, strongly polarized) scores, the lesser features need to be exhibited by an instantiation of a type. Most interesting for our purpose are the polarized scores, as they indicate a feature that has to be exhibited by the type, and thus restricts the amount of conceptually valid interpretations for that type.

Table 2: Some strongly (>55%) polarized type-feature combinations in the practitioner sample.

type	*is…*	score (practitioner)	score (student)
goal	necessary	91%	90%,70%,80%
process	necessary	64%	90%,70%,70%
	not vague	64%	
resource	necessary	64%	80%,70%,80%
	not vague	64%	30%,60%,80%
restriction	not natural	73%	
	not human	73%	70%,80%,60%
	necessary	73%	40%,70%,60%
	not vague	82%	90%,60%,90%
result	not natural	64%	
	not human	73%	80%,80%,90%
	intentional	73%	
	not vague	91%	

The results also show that around half of the judgments for the types are of a polar (either positive or negative) nature, indicating that there are many features for all investigated types that are conceived of as respectively strongly typical and strongly atypical. On an individual level this may vary, as specific people might find all features (a)typical for a given type, or find no features particularly typical for a given type (i.e., remain neutral). Such effects are likely correlated with the expertise a modeler has in a specific area, which might exhibit by an increased amount of detailed scrutiny he will give to the types often used by that area. The variation found within the results (see in Table 3) is only significantly different from the average when it comes to actors and events (meaning people are more likely to have similar conceptualizations of those types).

Table 3: Variance of results for each investigated type in the practitioner study

	actor	event	goal	process	resource	restriction	result
variance	0.38	0.57	0.68	0.93	0.73	0.94	0.92

Interestingly enough there is not a significant difference in the relative amount of neutral and polar scores between the practitioners and the students. One could expect that experienced modelers have come across a larger variety of viewpoints (through e.g., the use of different languages and methods and working with a multitude of people), and thus would be more accommodating in their conceptualizations of the investigated types. This is, however, not the case. It might again be correlated with the experience modelers have, in the sense that their often specialized focus leads them to have strong opinions on a specific set of types (e.g., value modelers expressing very specific demands on the features that a resource can and should exhibit). Some specific strongly polarized results are shown for a number of types in Table 2. Only features for which more than 55% of the participants judged them to be necessary (positively or negatively) are taken into

account. This does not mean that a participant is not capable of considering an instantiation of the type that does not conform to these features, but that their typical interpretation (and likely implicit usage) conforms to this set of features. We will discuss some of them in more detail below.

Goals are one of the few types where there is but one absolutely needed feature. Specifically, goals are considered by most (91% of practitioners) to be necessary. What this means is that goals are considered to be logical necessities in sense of their being achieved, i.e., they are the logical consequence of following a certain procedure to achieve them, and should be achieved then (and likely only) then. This is in contrast to the other option of goals being things that 'merely' *should* be achieved. As a consequence, this seems to imply an implicit bias towards hard-goals, as soft-goals are often used to model goals for which the achievement status is (or cannot) be known a priori. The absence of any other needed features is interesting as well. It could have been expected that, in line with the implicit bias towards hard-goals, the majority of practitioners would consider goals as being non-vague (i.e., well-described), yet only 45% does so.

Resources are mostly interesting because of the trend seen in the scores of the investigated students. A slight majority of practitioners deem resources to be necessary and not vague. The necessary feature is consistently shared by students, yet the not vague feature was not shared from the beginning. Over time, students also seemed to deem not being vague a needed feature of resources (going from respectively 30% to 60% to 80% of the investigated students). This seems to imply that resources, are commonly seen as things that need to be well-defined, before they can be reliably (and systematically) used for some other activity. Such typical interpretations might clash with the way resources are handled in some languages (e.g., the information object in ArchiMate [The12], which allows you to represent a functional piece of information needed by some business process, without requiring you to be explicit about what information it contains).

Restrictions already have a larger typical feature set. They are typically considered non-natural, non-human, necessary and non-vague things. The non-vague feature seems obvious in that it demands us to be clear about what a restriction actually does (perhaps making rules more typical restrictions than something like, say, architecture principles). Non-natural clearly implies that they are not naturally occurring restrictions (e.g., natural laws like gravity) but more like restrictions that are 'created' to control or restrict other things. Related to this restrictions are also not considered to be human things, which in this context might mean that the general way of conceptualizing a restriction is seeing it as the abstract entity itself, and not, for instance, the human being enforcing it. More interesting from a modeling language point of view is the, perhaps obvious, view of restrictions being logically necessary things. This means that a typical restriction is something that has to be adhered to, and not something that can be broken. To accommodate that, a modeling language should either have its restriction types be inherently alethic, or allow for a way to explicitly distinguish restrictions with different modality. While there are not many (especially domain-specific) languages that accommodate this, Object Role Modeling (ORM) [Hal05] does have explicit support for denoting elements as being either alethic or deontic, and thus could be a useful choice of

language when a domain involves those non-typical restrictions: ones that do not necessarily have to be adhered to.

Results are typically non-natural, non-human, intentional and non-vague. This seems to fit with the common conceptualization of results, because they are seen as 'new' things, and thus shouldn't be already existing naturally occurring objects. This requires a certain level of ontological scrutiny though, because a tree is certainly a naturally occurring entity, as well as a tree branch that naturally fell off from a storm, but a log that results from cutting the tree into parts with a chainsaw is not. Furthermore, a branch that merely breaks off from a tree would not be considered a result in our sense because its breaking off was not intentional. Some results as modeled by languages might seem not to fit and evoke discussion on their ontological status, as, for instance, the human output concept in ARIS [SN00]. On a first glance, it would seem not to fit the characterization as it is likely a human kind of thing. However, ontologically speaking, the actual thing here is the activity performed by a human, not the human itself. Say, the successful delivery of a pizza. As results are not typically considered material things, this seems thus to be a conceptually valid instantiation of a result.

To show in some more detail how certain combinations of features change the amount of conceptually valid instantiations for a given type some examples are shown in Table 4. In line with the explanation of context (in this case, features) changing the amount of conceptually valid instantiations we show how some of the common types are affected by the results we found.

Table 4: Examples of difference in conceptually valid reflecting some of Table 2's results

type	is a ... feature-wise	example	ounter-example
goal	Necessary thing	Any thing to achieve, whether physical (e.g., producing a re- presentation of a model) or non-physical (e.g., producing the information needed for it) entity to be acquired or some state to be reached, and so on	A non-necessary goal, or thing that is achieved as a side-effect of achieving something else
restriction	Not natural, not human, necessary to adhere to and well-define thing	Legal rules given in a state's laws, well-known and documented natural laws	etiquette, informal dinner rules, or natural laws like gravity
result	Not natural, not human, intentionally achieved well-define thing	A baked pizza, a sawed log, a specification of a modeling language, a representation of the specification of a modeling language	A newly hired person, the outcome of trial-and-error testing

When a type has only few absolutely needed features, the amount of conceptually valid instantiations is significant. Conversely, the amount of conceptually invalid instantiations is quite small. A goal is a good example of this, as the only needed feature

is it being a necessary thing, which means we can instantiate it as an actual physical object to be achieved (i.e., the goal is the pizza that will be baked), the activity of having done so (i.e., the goal is the final state of the pizza baking process) or even more abstract, an abstract object that follows from its creation (i.e., the goal is the information gained when the pizza is baked). The few counter-examples are those where the goal is not necessarily achieved, which could be either something that is not achieved, or something that was achieved by accident (e.g., as a side-effect from another process). The main problem here is that the counter-examples (and examples themselves too, perhaps) are so broad that it is difficult to figure out whether an instantiation was actually a valid instantiation or not.

On the other hand, when a type has many needed features, it is easier to find out whether something is a conceptually valid instantiation or not. When it comes to results, certain objects are clearly valid instantiations, like a freshly baked pizza or a specification of a modeling language. The latter one can actually be a result in two ways, as both the actual specification of the language is a valid result, as well as the representation of the language (i.e., the actual written documents describing it). Counter-examples are then also easy to come by, for instance, the actual person hired as the 'result' of a hiring process is not typically considered a result (by virtue of being human). Furthermore, any 'results' that were the outcome of trial-and-error approaches are also on shaky grounds, as they are not typical results by virtue of not having been necessary.

6 Consequences

The results we have presented entail a number of consequences for the creation and use of models, and modeling language design itself. As it is clear that there are some common type-feature combinations amongst the people we investigated (while the total amount of conceptual diversity is far greater than that), it is important that we revisit the way we model in order to deal with the effects of these different conceptual understandings.

While the data presented and discussed here is reporting on average and shared conceptualizations, the conceptualizations that individual people have can vary wildly. It is important that we take a moment in the modeling process before we start creating models to deal with these differences. A way to do so is by looking at the types that are needed to model the domain under investigation, and then discuss whether certain features (and combinations thereof) apply to it. This can be done by taking known lists of features that have proven to be a source for disagreements and conceptual misunderstandings, such as the features reported on in this work. For example, when we are modeling the process of baking pizzas, we might focus on the resources for a moment. We can then take some time before creating actual models to discuss what we find conceptually valid resources and what we do not (i.e., what we would rather not be possible to express in the model). From such discussions we gain a clearer understanding of how people see such a process and want to model it, which helps us to focus (e.g., by abstracting from such potential resources as time, only modeling material things as resources). By doing so, the semantics of the created model are shared and well understood by all the people involved in the process, not just those who are intimately familiar with the specification of the used modeling language.

Such discussions can go into more detail feature wise as the focus on what to model becomes narrower. For example, when there is already a strong focus on processes, the types most relevant to a process (e.g., processes and restrictions) can be explored more closely by looking at more relevant features in greater detail. For instance, a process model might benefit from having a level of traceability on the restrictions in it [PdKP12], which requires us to discuss whether restrictions are always made by specific people, or departments, and whether that implies that restrictions carry responsibility with them. These aspects require a more fine-grained characterization of the process type than if it is used in a general way. Equally so, when the focus is on modeling value exchanges, then the focus can be on, for instance, characterizing types like resource and actor more clearly by investigating more detailed features (e.g., value exchanges can be analyzed from an ethical point of view, and thus require actors to be moral agents).

As it is clear that there are a number of conceptual distinctions people have for the types we investigated, it is also important that the modeling languages we use allow us to express these distinctions. This can be ensured in a practical way by selecting a modeling language which allows us to explicitly express those distinctions that are important to us. For example, when it is important to make a distinction between alethic and deontic restrictions we could choose for ORM, when it is important to make a distinction between goals that are well-defined or not, we could choose for i* or any of the goal modeling languages that explicitly support hard- and soft goals. In short, before selecting a modeling language and creating models, we need to find out the most important conceptual distinctions the people involved in the modeling process have, and then select a language based on them.

On a more theoretical side, we should also ensure that the modeling languages support these different conceptual distinctions in the specification of their semantics. While this would not necessarily be useful for general purpose languages (e.g., UML, ER), domain-specific languages could benefit from the added detailed semantics. This requires more fine-grained investigations into the conceptualization people have of types as they are used in specific domains. For example, a more detailed characterization of what a resource is could be useful to languages that center around value exchanges (e.g., e3Value), implementations (e.g., ITML) or deployments (e.g., ADeL [Pat11]).

Finally, while the results we have shown are interesting and could be used for a multitude of purposes, care must be taken not to immediately extrapolate the results of this relatively small study and infer general truths from them. Different groups with varying backgrounds and specializations might have different conceptualizations, and thus further studies might uncover additional and conflicting conceptualizations. As such, the contribution of this work has been in showing the demonstrating the possibility to systematically uncover such conceptual distinctions and laying the ground for further investigations.

7 Conclusion & Future Work

We have shown how the conceptualization which someone has of types used by modeling languages can affect the range of conceptually valid instantiations for such types. This can have a negative effect on the validity and usefulness of created models if

we do not take care to discover and communicate these conceptualizations. Furthermore, some languages might be more suited to deal with certain conceptualizations by virtue of explicitly expressing conceptual distinctions that other languages leave implicit. Care should thus be taken to also select a modeling language that fits most with the conceptualizations of the involved modelers and stakeholders. Having shown the potential use of these kind of studies, we aim to extend their thoroughness for generalization purposes by repeating them with a larger amount of practitioners, and also by incorporating groups focused on specific sectors (e.g., modelers in healthcare, government, or telecom) and languages (e.g., the BPM community or the ArchiMate community).

Acknowledgments

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

References

[Alm09] Mauricio Barcellos Almeida. A proposal to evaluate ontology content. *Applied Ontology*, 4:245–265, August 2009.

[BC05] Peter D. Bruza and Richard Coles. *We Will Show Them: Essays in Honour of Dov Gabbay*, chapter Quantum Logic of Semantic Space: An Exploratory Investigation of Context Effects in Practical Reasoning. College Publications, 2005.

[BKV09] Joseph Barjis, GwendolynL. Kolfschoten, and Alexander Verbraeck. Collaborative Enterprise Modeling. In Proper et al., editor, *Advances in Enterprise Engineering II*, volume 28 of *LNBIP*, pages 50–62. Springer Berlin Heidelberg, 2009.

[BW08] Peter D. Bruza and John Woods. Quantum collapse in semantic space : interpreting natural language argumentation. In Peter D. Bruza et al., editor, *Second Quantum Interaction Symposium*, 2008.

[FCK95] David Ferrariolo, Janet Cugini, and Richard Kuhn. Role-based access control (RBAC): Features and motivations. In *Proc. of the 11th Annual Computer Security Applications Conference*, 1995.

[FHK+09] Ulrich Frank, David Heise, Heiko Kattenstroth, Donald Ferguson, Ethan Hadar, and Marvin Waschke. ITML: A Domain-Specific Modeling Language for Supporting Business Driven IT Management. In *Proc. of the 9th OOPSLA workshop on DSM*, 2009.

[FHVL12] Peter Fettke, Constantin Houy, Armella-Lucia Vella, and Peter Loos. Towards the Reconstruction and Evaluation of Conceptual Model Quality Discourses – Methodical Framework and Application in the Context of Model Understandability. In Bider et al., editor, *Enterprise, Business-Process and Information Systems Modeling*, volume 113 of *LNBIP*, pages 406–421. Springer Berlin Heidelberg, 2012.

[Gee10] Dirk Geeraerts. *Theories of Lexical Semantics*. Oxford University Press, 2010.

[GHYA07] Gemma Grau, Jennifer Horkoff, Eric Yu, and Samer Abdulhadi. i* Guide 3.0. Internet, August 2007.

[GRA08] Liane Gabora, Eleanor Rosch, and Diederik Aerts. Toward an Ecological Theory of Concepts. *Ecological Psychology*, 20(1):84–116, 2008.

[GYvdR06] Jaap Gordijn, Eric Yu, and Bas van der Raadt. e-Service Design Using i* and e3value Modeling. *IEEE Software*, 23:26–33, 2006.

[Hal05] Terry Halpin. ORM 2. In *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops*, pages 676–687. Springer, 2005.

[HFL12] Constantin Houy, Peter Fettke, and Peter Loos. Understanding Understandability of Conceptual Models - What Are We Actually Talking about? In Paolo Atzeni, David Cheung, and Sudha Ram, editors, *Conceptual Modeling*, volume 7532 of *Lecture Notes in Computer Science*, pages 64–77. Springer Berlin Heidelberg, 2012.

[HS05] Brian Henderson-Sellers. UML - the Good, the Bad or the Ugly? Perspectives from a panel of experts. *Software and System Modeling*, 4(1):4–13, 2005.

[KSKS12] Julia Kaidalova, Ulf Seigerroth, Tomasz Kaczmarek, and Nikolay Shilov. Practical Challenges of Enterprise Modeling in the Light of Business and IT Alignment. In Sandkuhl et al., editor, *The Practice of Enterprise Modeling*, volume 134 of *LNBIP*, pages 31–45. Springer Berlin Heidelberg, 2012.

[Lan04] Marc M. Lankhorst. Enterprise architecture modelling–the issue of integration. *Advanced Engineering Informatics*, 18(4):205 – 216, 2004.

[MAG+11] Barbara C. Malt, Eef Ameel, Silvia Gennari, Mutsumi Imai, and Asifa Majid. Do words reveal concepts? In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, pages 519–524, 2011.

[OB06] Andreas L. Opdahl and Giuseppe Berio. Interoperable language and model management using the UEML approach. In *Proceedings of the 2006 International Workshop on Global Integrated Model Management*, pages 35–42, New York, NY, USA, 2006.

[Obj10] Object Management Group. Business Process Model and Notation (BPMN) FTF Beta 1 for Version 2.0. Internet, 2010.

[OST57] Charles E. Osgood, George J. Suci, and Percy Tannenbaum. *The Measurement of Meaning*. University of Illinois Press, Urbana, 1957.

[Pat11] Susanne Patig. Modeling Deployment of Enterprise Applications. In Wil Aalst et al., editor, *Information Systems Evolution*, volume 72 of *LNBIP*, pages 253–266. Springer Berlin Heidelberg, 2011.

[PdKP12] G. Plataniotis, S. de Kinderen, and H.A. Proper. EA Anamnesis: Towards an approach for Enterprise Architecture rationalization. In Sheridan Printing, editor, *Proceedings of the The 12th Workshop on Domain-Specific Modeling (DSM12)*. ACM DL, 2012.

[Pin07] S. Pinker. *The stuff of thought: Language as a window into human nature*. Viking Press, 2007.

[RGS+08] M. Rospocher, C. Ghidini, L. Serafini, B. Kump, V. Pammer, S. Lindstaedt, A. Faatz, and T. Ley. Collaborative enterprise integrated modelling. In *5th International Workshop on Semantic Web Applications and Perspectives (SWAP 2008)*, 2008.

[RHLR11] Michael Rosemann, Anders Hjalmarsson, Mikael Lind, and Jan C. Recker. Four facets of a process modeling facilitator. In Dennis Galletta and Ting-Peng Liang, editors, *32nd International Conference on Information Systems*, pages 1–16, Shanghai International Convention Center, Shanghai, December 2011. AIS.

[RKV08] Michiel Renger, Gwendolyn L. Kolfschoten, and Gert-Jan Vreede. Challenges in Collaborative Modeling: A Literature Review. In Jan Dietz et al., editor, *Advances in Enterprise Engineering I*, volume 10 of *Lecture Notes in Business Information Processing*, pages 61–77. Springer Berlin Heidelberg, 2008.

[Rob06] Stewart Robinson. Conceptual modeling for simulation: issues and research requirements. In *Proceedings of the 38th conference on Winter simulation*, WSC '06, pages 792–800. Winter Simulation Conference, 2006.

[SHP09] Denis Ssebuggwawo, Stijn Hoppenbrouwers, and Erik Proper. Interactions, Goals and Rules in a Collaborative Modelling Session. In Anne Persson and Janis Stirna, editors, *The Practice of Enterprise Modeling*, volume 39 of *Lecture Notes in Business Information Processing*, pages 54–68. Springer Berlin Heidelberg, 2009.

[SN00] August-Wilhelm Scheer and Markus Nuettgens. ARIS Architecture and Reference Models for Business Process Management. In *Business Process Management*, volume 1806 of *LNCS*, pages 301–304. Springer Berlin / Heidelberg, 2000.

[Sow10] John Sowa. The Role of Logic and Ontology in Language and Reasoning. In *Theory and Applications of Ontology: Philosophical Perspectives*, pages 231–263. Springer Netherlands, 2010.

[SPS07] Janis Stirna, Anne Persson, and Kurt Sandkuhl. Participative Enterprise Modeling: Experiences and Recommendations. In Krogstie et al., editor, *Advanced Information Systems Engineering*, volume 4495 of *LNCS*, pages 546–560. Springer Berlin Heidelberg, 2007.

[Sta00] Gerry Stahl. A model of collaborative knowledge-building. In *Proc. of 4th Int. Conference of the Learning Sciences (ICLS 2000), Ann Arbor, MI*, pages 70–77, 2000.

[The12] The Open Group. *ArchiMate 2.0 Specification*. Van Haren Publishing, 2012.

[vdLHLP11] D. J. T. van der Linden, S. J. B. A. Hoppenbrouwers, A. Lartseva, and H. A. Proper. Towards an Investigation of the Conceptual Landscape of Enterprise Architecture. In T. Halpin et al., editor, *Enterprise, Business-Process and Information Systems Modeling*, volume 81 of *LNCS*, pages 526–535. Springer Berlin Heidelberg, 2011.

[VM07] Tibert Verhagen and Selmar Meents. A Framework for Developing Semantic Differentials in IS research: Assessing the Meaning of Electronic Marketplace Quality (EMQ). Serie Research Memoranda 0016, VU University Amsterdam, Faculty of Economics, Business Administration and Econometrics, 2007.