

# CONSOLIDATING SEMANTIC INTEROPERABILITY IN CONTEMPORARY ARCHITECTURAL PARADIGMS

version: v0.3-31

Paul Brandt,

Eindhoven University of Technology; Netherlands Organization of Applied Scientific Research TNO, Den Haag, The Netherlands,

Eric Grandry,

Luxembourg Institute of Science and Technology, Esch-sur-Alzette, Luxembourg,

Twan Basten,

Eindhoven University of Technology, Eindhoven, The Netherlands,

## Abstract

*Context: Access-and-Play semantic interoperability (sIOP) is the next glass ceiling in IT-based business collaboration. Current approaches towards sIOP rely on conventions on the semantics of the exchanged terms, which can be considered accepted folklore. Approaches to break through the ceiling require some level of automation, and artificial intelligence (AI) can make a difference.*

*Objective: The objective of this study is to identify and define AI-based fundamental guidance towards access-and-play sIOP in contemporary architectural paradigms.*

*Method: Our approach is based on the discipline of semiotics. We identify semiotic shortcomings in architectures, establish a semiotic explanation on software semantics and, subsequently, on sIOP. Based on these considerations, we develop guiding architectural principles in support of software semantics and sIOP. We evaluate these principles by designing and formulating an ISO-42010 Architecture Viewpoint and View on sIOP.*

*Results: The semiotic approach demonstrates semantics in software to be the result of a reciprocity between data and the software code that operates on them. Data exchange breaks that reciprocity and the main concern of sIOP is to re-establish a valid reciprocity. Loosely coupled semantics, semantic alignments and an ontological commitment of the modelling language can be considered the cornerstone to achieve sIOP. The supporting principles are (i) semantic transparency, (ii) semantic separation of concerns, and (iii) explicit computational semantics. The resulting ISO-42010 Architecture Viewpoint and View on sIOP, including a semantic mediation capability, can be considered a pattern to consolidate sIOP in contemporary architectural paradigms.*

*Conclusions: The major shortcomings in architectural paradigms to account for sIOP are their negligence of semiotic fundamentals and the absence of an explicit ontological commitment that stands at the root of semantics. By their explicit inclusion, access-and-play sIOP can be consolidated in contemporary architectural*

*Paul Brandt – /*

*paradigms.*

# Chapter 1

## Introduction

Never before, data were so ubiquitous, and managed access to external data was so easy. Because current ICT is unable to *use* all that same external, non-native data as access-and-play service, agility in business collaboration is hampered in all domains. For instance, consider the following (allegedly real) example of an interoperability failure.

A German steel producer upgraded its industrial process robot. Since the majority of the steel production process is dependent on time, from a security point of view the decision was made to not rely on their own internal clocks but to use the German *Braunschweig Funkuhr* time radio signal as source for the exact time instead. At the end of April 1993, when Germany went on summer time, the computer clock of the steel producer went from 1:59 AM to 3:00 AM in one minute. This resulted in a production line allowing molten ingots to cool for one hour less than normal. When the process controller thought the cooling time had expired, his actions splattered still-molten steel, damaging part of the facility.<sup>1</sup>

In this simple example a tiny difference in the meaning of *time* between the steel producer and the national time provider hampered interoperability to the extend of damaging the steel facility. This tiny difference rooted in the assumption by the steel producer that *time* expressed a continuous scale whilst for the Braunschweig Funkuhr, *time* denoted instant clock time for that time zone and therefore represented a non-continuous scale. In order to achieve that both collaborators, here the Braunschweig Funkuhr and the steel producer, can actually *use* their peers data, the need exists to design and implement wrappers that remove any inconsistency between the variations that may occur in terms, structures, dimensions and what have you. Many such variations exist, leading to a range of failures in so-called *semantic interoperability* (sIOP) and Section/Appendix ## provides for a short overview of sIOP-faults. Unfortunately, it is fundamentally impossible to automate the production of wrappers, because we need a genuine *understanding* upfront, which computers still cannot do.

The most disconcerting consequences of a lack of (automated) sIOP are time-to-deliver, flat interoperability failures, and even seemingly correct but quite invalid data analysis probably leading to faulty system behaviour. Current sIOP implementations are essentially based on the (time-consuming) process of establishing a (local) convention on the semantics of the terms that are exchanged during collaboration, requiring custom solutions and collaboration-dependent software adaptations. Such conventions can be considered a semantic monolith, which makes dealing with data outside the monolith impossible, unless again a time consuming (months) semantic adoption process is applied. Moreover, these semantic conventions consider semantic heterogeneity as a bug instead of a feature necessary to achieve semantic accuracy. But still, this conventions-based approach towards sIOP is accepted folklore in ICT. In view of the large uptake of the Internet, the Internet of Things (IoT), cloud computing and big data, and in view of economical pressure to intensify enterprise

brandtp,  
9/5/2018 We  
can apply  
another ex-  
ample, I'm  
open to that.  
Indeed a  
TOOP exam-  
ple could be  
appropriate.  
However, I  
cannot think  
of one but  
maybe Eric  
can?

brandtp,  
9/5/2018 Add  
sIOP-faults  
as appendix.

<sup>1</sup> Source: <http://catless.ncl.ac.uk/Risks/14.57.html#subjl>, accessed May 20, 2018

collaboration, we consider this approach “too little, too late”. Some form of automation is required to resolve these issues, and we place artificial intelligence (AI) at its core. With the separation (coined by Searle 1980) between *strong* AI (a system that can *think* and has a *mind*, in the philosophical definition of the term) or *weak* AI (a system that can only *act* like it thinks and has a mind), we take the position that –unfortunately– strong AI is not yet available, if ever (Xiuquan Li and Tao Zhang 2017). We therefore make do with weak AI and show that despite its limitations it still has got a fundamental role to fulfil as carrier for semantics and sIOP, both facilitating the required automation. Weak AI, despite its current applications in Semantic Web or ontologies, has not yet been embedded in contemporary software architectural paradigms.

In comparison, scalability was a big architectural concern in the past, requiring custom solutions as well. In response to this concern, scalability was standardised in the form of architectural patterns, and finally totally embedded and hidden into the infrastructure. Similarly, sIOP can be considered the architectural concern of this decade. We first need to provide a standardised solution pattern to address semantic concerns before we can embed it in a technological infrastructure. Only then we can claim that sIOP becomes transparent to the developer and that the semantic monolith is taken down. Where scalability resulted in a huge increase in performance-demanding applications against a fraction of the original costs and effort, business agility will emerge once the semantic monolith is removed and semantic services exist at the infrastructural level. Then sIOP becomes an access-and-play operation that can be achieved with data not anticipated for during software design in due time, at any point in their life cycle. Metaphorically speaking, we consider sIOP as a *bridge* overarching a (semantic) gap: with *bridgeheads* (semantic concerns) on each side of the gap, with a *spanning* (semantic alignments) resting on them to structurally support the bridge and its traffic, and with a *roadway* (data mediation) enabling the crossing of the traffic. Finally, architectural *principles* provide the necessary guidance to the architect for the various design decisions that effectively result in a particular bridge over a particular (semantic) gap. Our contributions to consolidating semantic interoperability in software architectures are fivefold, and represented as architectural principles and concerns, as follows:

- *Semantic concerns (bridgehead)*: Abstracting semantics from a tacit software implication into a tangible, computational and distinct artifact provides us with the potential to connect to it and to make comparisons with the semantic artifact of the peer software agent. Based on the discipline of semiotics, we explain why semantics are irrelevant to software. Instead, we should focus on the reciprocity between data and the data processing code of software. This explains, too, the shortcomings of the current approach towards software semantics that rely on prescriptive information models. We argue that the application of ontologies and ontological commitment are fundamental to remedy current semantic shortcomings (Chapter 3);
- *Weak AI concerns (spanning)*: Since “strong AI” does not yet exist, sIOP remains in demand of human intervention in order to reconcile the semantic differences between collaborating software agents. However, human intervention is time consuming. We reduce the necessary human intervention to complement weak AI to a task that suffices to achieve sIOP, viz. authoring semantic alignments only (Chapter 4);
- *Mediation concerns (roadway)*: We provide for a prototypical implementation of a mediator as the necessary component to automatically translate data when transferred between the collaborating software agents (Chapter 5);
- *Principles*: We base sIOP on establishing loose-coupling at the semantic level by introducing principles on semantic separation of concerns and semantic transparency (Chapter 6), and show how these principles can be operationalised;
- *ISO42010 Architecture Viewpoint*: We verify the applicability of the above concerns and principles by formulating their architectural consequences as a specific ISO42010\* \*sIOP Viewpoint, and we show their proper position in the total architecture as corresponding sIOP view. As ISO42010 is considered a set of best practises for architecture description, and therefore is used with architecture frameworks such as MoDAF, TOGAF, DoDAF, RM-ODP and so on, we conclude that our sIOP Viewpoint and View can be considered to consolidate sIOP for contemporary architectural paradigms (Chapter 7).

Based on these contributions we defend that access-and-play sIOP can be embedded and hidden in infrastructural services when considering semiotic fundamentals and adding loosely coupled formal semantics to contemporary architectural paradigms. To that end, we first describe the semiotic fundamentals in Chapter 2.

# Chapter 2

## The semiotic and philosophical foundations of semantics

**Synopsis:** Purpose of this section: To establish an informal but concrete notion on (i) the semiotic triangle and the relations between its nodes, and (ii) ontological commitment and its relation to modelling languages.

### 2.1 Semiotics

**Synopsis:** Present concrete notion on (i) the semiotic triangle and (ii) the relations between its nodes. Show abstraction/generalisation as stacking of triangles vertically, and representational meta-levels as stacking them horizontally.

The discipline of semiotics is the study of signs, reality and meaning. The meaning of a *token* (text, graphics, sound) ultimately relates to what it denotes in reality (the *entity*), whilst this relation cannot be deferred from the shape, structure or other characteristics of the token itself due to its total arbitrariness. De Saussure used a dyadic model, the *semiotic sign*, that stressed that the token and the entity in reality were as inseparable as the two sides of a piece of paper (Saussure 1959). This 'self-containment of the sign' remains one of the major principles of semiotics. Constructing the semiotic sign from its distinct parts is called *semeiosis*. The token, in combination with their ability for semeiosis, provides humans with the tool to converse with each other. Semantics, then, emerges as a result of the semeiosis that connects the distinct parts of the inseparable semiotic sign.

Sanders Peirce (in: Sowa 2000) developed a triadic model of the semiotic sign, the semiotic triangle: a representamen (the token), an object (the entity), and the interpretant which expresses the mental and, hence, individual sense making (Figure 2.1(a)). The semiotic triangle was used and modified by Ullmann (Ullmann 1962), Ogden and Richards (Ogden and Richards 1989) and many others, also in recent years (Kuhn 2009). We introduce our modifications, as depicted in Figure 2.1(b), which mainly focus on naming conventions in IT architectures, as follows.

We prefer the use of *entity* due to the ambiguous nature of *object* in IT. We consider an entity to stand for a thing or event, but also a category of entities, a relation between entities and a property of an entity. We refer to the *interpretant* component as the *conceptualisation*, to underline the individual conceptualisation that is being formed during requirements analysis and conceptual modeling. And we prefer *token* over *representamen*, and consider it both an atomic element and a particular composition of atomic elements. We include denotations for the edges that are connected to the conceptualisation vertex, and names that underline the individual and mental nature of the sense making. These names are directional, and must be read as the transformation that takes place in that direction. Finally, we add the causal characteristics that the edges represent, introduced by (Ogden and Richards 1989), as *adequacy*, *correctness* and *trueness*.

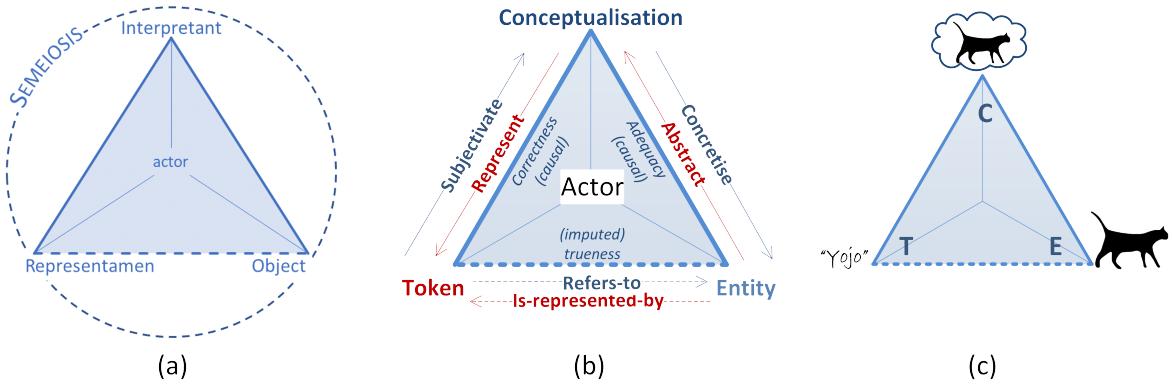


Figure 2.1. The triadic model of the semiotic sign, according to Peirce (a), and modified by us (b). Example (c) shows the concept of a cat named “Yojo”

The connection between the token and the entity is a dashed line because its existence is indirectly only through the conceptualisation and does not exist in any direct means. With “sign” we mean the semiotic self-contained sign. A well-known example of a sign is depicted in Figure 2.1(c): when we talk about “Yojo”, our cognition interprets it as our cat.

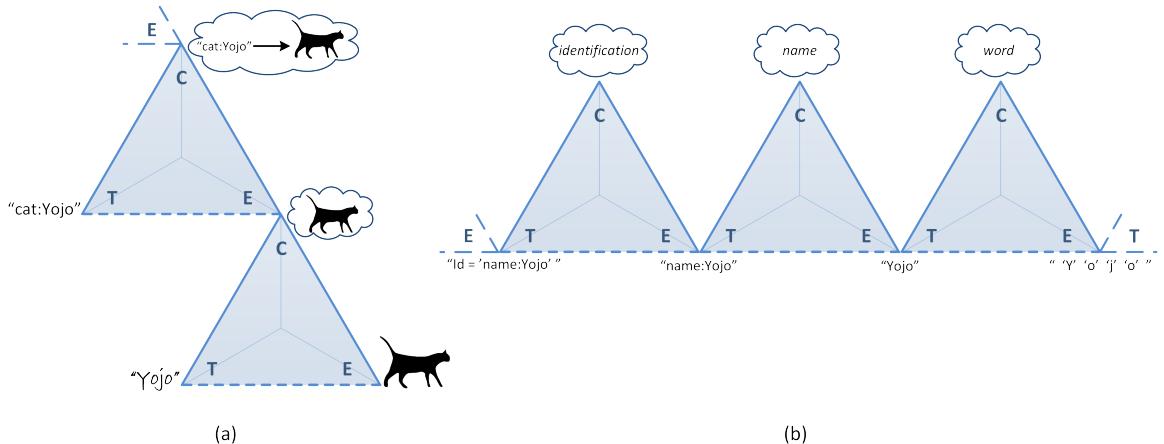


Figure 2.2. Linking triadic models together.

Multiple triangles can be linked together in various ways (Peirce in: Sowa 2000). By stacking them together (Figure 2.2(a)) a conceptualisation is made of “representing an entity”: the original concept of a **cat** named “Yojo” (Figure 2.1(c)) is being conceptualised as the concept of a **cat named “Yojo”** and represented by **cat:Yojo**. Eco (Eco 1976) uses the term *unlimited semeiosis* for the succession of stacking signs that emerge from that, ad infinitum. We consider unlimited semeiosis as addressing a dimension of comprehension about abstraction and generalisation, with an eventual finish in the ultimate **Thing** concept.

Linking the triangles horizontally results in different representational meta-levels, depicted in Figure 2.2(b): From right to left, the characters “Y” “o” “j” and “o” are conceptualised as a single **word** and represented as “Yojo”, which is conceptualised as a **name** and represented as “name:Yojo”, which is conceptualised as an **identifier** that might be represented as “Id=name:Yojo”.

## 2.2 Ontological commitment

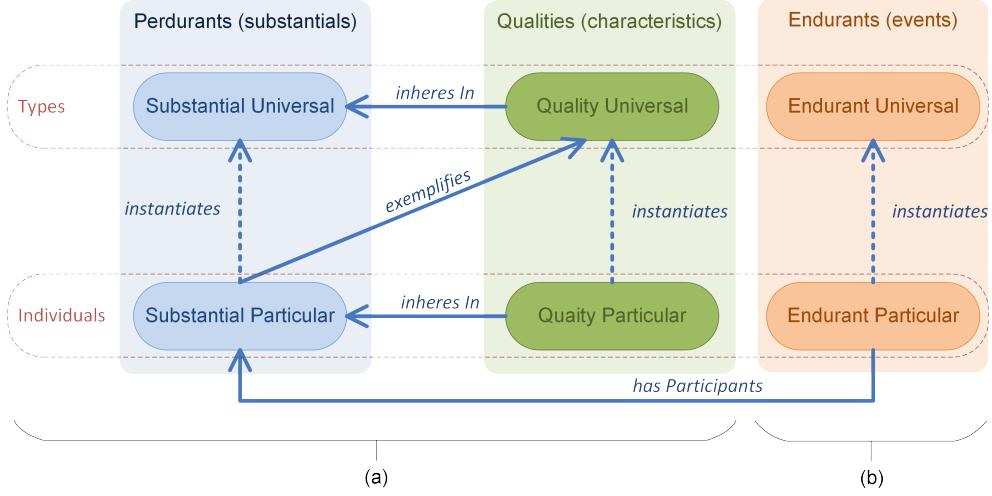
**Synopsis:** Purpose of this section is to show the relevance of a (modelling) language: a language is used to convey distinctions. The distinctions that are articulated by a language are denoted as its ontological commitment. Modelling languages need to show distinctions that are of relevance to the purpose of modelling. The predominant purpose of modelling is to describe (a particular domain in) reality by distinguishing the entities of interest. The ontological commitment for a modelling language therefore should be of ontological nature, "(...) not in order to know *\*what there is\**, but in order to know what a given remark or doctrine, ours or someone else's, *\*says\** there is" [Quine:1953er]. The ontological square or its extension into the ontological sextet [describe it] can be considered the most basic one, which have been specialized into several flavours, all called upper-level, or top-level, or foundational, ontologies, e.g., UFO, BFO, DOLCE and more.

Show how MDE/MDA applies an ontological commitment as defined in M3, which facilitates an equivalent distinction as the ontological square, but less than the ontological sextet.

Optionally, provide for the distinctions/theories that are foundational to the two most appropriate ontological commitments / foundational ontologies: BFO and UFO.

A (modelling) language is used to convey distinctions. The predominant purpose of modelling is to describe (a particular domain in) reality by distinguishing the entities of interest. The distinctions that are thus articulated by a language are denoted as its *ontological commitment*. The ontological nature of the commitment is characterised by: "(...) not in order to know *what there is*, but in order to know what a given remark or doctrine, ours or someone else's, *says there is*" (Quine 1961). For domain analysts, the resulting model (of the conceptualisation) represents the commitment that the domain users have about the entities that they consider relevant to discern: the *domain commitment*. In philosophy (Bricker 2016), similar questions are asked that relate to "life, the universe and everything" as opposed to a single domain of application: What kind of entities exist? Are the universal aspects that seem to be shared between individual entities, e.g. weight, taken to be *sui generis*? These are questions of ontology. From a more pragmatic view these questions can be asked in a more constrained, methodological nature: What kinds of entity exist *according to a given theory or discourse*? This is *meta-ontology* and the answers provide a generic framework of distinction that represents a useful meta-model to the domain analyst, denoted the *ontological commitment*. The ontological square (???) can be considered the most basic ontological commitment. It is obtained by considering two formal but independent distinctions: that between *types (Universals)* and *individuals (Particulars)*; and that between *characteristics (Properties)* and their *bearers (Substrates)*. The four resulting categories and their causal relations are depicted in ??(a). Its extension into the ontological sextet (Smith 2005) is just to acknowledge that at some point the influence of time should be incorporated as well (??(b)). ja, jullie gebruiken ook zoiet (is niet OC, maar C) maar dat is te beperkt en niet gebaseerd op realiteit (filosofie) maar op engineering sec .they can not differentiate for example, between the cup and the coffee that is holds. of zoClearly, similar formal distinctions can be observed in contemporary modelling paradigms at their uppermost meta-level: OMG's MOF M3 metamodel distinguishes between the *Association* and the *Class*, while the Resource Description Framework's language distinguishes primarily between a *subject*, an *object*, and the directed *property* relation between them. All these ontological commitments are very generic, intended to be applicable for all circumstances. Unfortunately, being very generic, the ontological commitments remain very sparse regarding their capability as a language to distinguish; for all examples above, it is impossible to differentiate, for example, between the cup and the coffee that

it holds, while intuitively, these are very different from each other. The more extended an ontological commitment becomes, the more it can distinguish as a language between things that we are interested



in.

The ontological

sextet has been specialised into several flavours, all called { upper-level | top-level | foundational }<sup>1</sup> ontologies, e.g., UFO (Guizzardi et al. 2015), SUMO, YAMAYATO, BFO, DOLCE and more, and with their similarities and differences between them (Jansen 2008). Not all of them are intended to provide for a complete foundation (UFO, BFO); several are only facilitating (DOLCE), or the result of (YAMAYATO), research into particular philosophical theories. No matter the reason for their existence, they all assume a certain philosophical viewpoint with the purpose to accurately describe categories that exist in reality according to their viewpoint. We consider this the philosophical cornerstone for semantics: we can assess the semantic validity of any proposition if and only if the underlying ontological commitment can be referred to. Furthermore, any assessment towards semantic interoperability of two semantic theories cannot be made without an assessment of the similarity between their underlying ontological commitments. Note, however, that “We look to (...) Ontology not in order to know *what there is*, but in order to know what a given remark or doctrine, ours or someone else’s, *says there is*” (Quine 1961). It then remains the responsibility of the (domain) analyst or ontology engineer to select the viewpoint that is most appropriate for its purpose, in order to select the modelling language that describes the types of things that exist in the domain of application the most accurate.

<sup>1</sup> Please select the term you like best.

# Chapter 3

## Bridgehead: Semantics

**Synopsis:** Purpose of this section:

1. From a semiotic perspective, explain what we mean with semantics in software agents, i.e., the reciprocity between data and data processing code (both represented as Term from the semiotic Sign).
2. Establish that for representing semantics, descriptive models (i.e., ontologies) trump prescriptive models (all 42010 models).
3. Conclude that ontologies need their place as single point of reference (truthiness) in architectures, and identify their relationship with the rest of the architectures, i.e., all other prescriptive models. Note the issue on Open World Assumption (ontologies) and CWA (prescriptive models).

Text

### 3.1 What is software semantics

**Synopsis:** The purpose of this section is to explain software semantics as the "reciprocity between data and software code", and show that to some extent, set theory can replace the conceptualisation node. Conclude that the reciprocity between data and data processing code represents the smallest (atomic) semantic monolith.

Additionally, optionally, show:

1. the relationship with Grice's distinction in semantics as "what is said" and "pragmatic meaning": DONE
2. OO as initial implementation to consolidate this reciprocity, and the class as implementation of the atomic semantic monolith;

We take the position that weak AI is essentially a token-based machine without the ability to close the gap between token and reality. Also called the Grounding Problem (Harnad 1990), addressing this fundamental issue in software engineering about semantics is at best extremely narrow (Steels 2012), or not present at all (Cregan 2007). This implies that the semiotic triangle is denied its conceptualisation vertex, and the sign remains incomplete. This is confirmed by the software engineering discipline herself implicitly, since it consistently speaks of 'models that represent reality' in a certain purposeful context *without* factoring the conceptualisation into the equation (Aßmann et al. 2006). Consequently, the edges that connect the

conceptualisation remain vague or necessarily conflate on the relationship between the model and reality, depicted in Figure 3.1. In terms of (Quine 1961) above, we have beheaded the sign and cut-off our “knowledge about our given remark or doctrine”, we deleted that what “we say there is”. We have removed the “ontological level” (Guarino 1994), and with that our “terminological competence [that] can be gained by formally expressing the ontological commitment of a knowledge base” (ibid.). However, since we make do with weak AI and therefore with this beheaded sign necessarily, we must conclude that genuine semantics can not ever exist in current software agents.

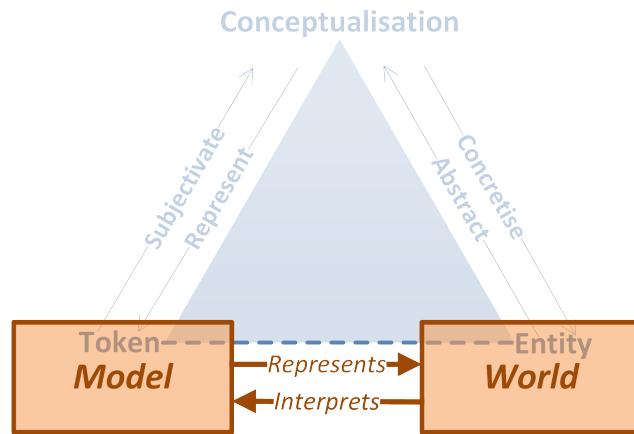


Figure 3.1. Software engineering applies a beheaded semiotic triangle in which its edges remain vague or conflate in the single relation between model and reality.

During the use of a software agent the semeiosis is taken care of by the human-in-the-loop, viz. the end user at the human-machine interface (HMI) whom interprets the tokens that are displayed (subjectivation). During development of a software agent the semeiosis is taken care of by another human-in-the-loop, viz. the software engineer whom implicitly performs the conceptualisation and explicitly represents this conceptualisation into tokens, i.e., *models*. Consequently, all models are representations of the engineers' conceptualisations. From the many models that software engineering typically generates we focus on a pair of models that constitute the engineers' semantics: the information or data models that refer to the *information entities* in reality, paired with the process or business models that represent the *event entities* that operate on the information entities. Data processing is in its bare form nothing more than tokens that follow a specific language grammar. This bare form is a representation of its quintessence, viz. a run-time notion on the proper way to operate on the data. Together, these models comprise the smallest atomic union that can represent meaning, indicated by (Grice 1989) as a twofold: the *semantic* meaning, or “what is said”, and the *pragmatic* meaning, what we like to understand as “how it relates to our intentions”.

### Definition 3.1 (Atomic semantic monolith)

An *Atomic Semantic Monolith (ASM)* denotes the smallest, highest grained pair of models (a data model and a data processing model) that remains faithful to the entity in reality that it refers to. ■

At the modelling level, semantics still exist by virtue of the designer. However, when the software agent is subsequently compiled, its binary code originate from the process model of the model pair (operations, algorithms), and the memory allocation for the data originates from the information model of the model pair (size, format, encoding). At this binary level the software engineer has left the building, and with her the conceptualisation vertex and the subsequent capability for semeiosis and, thus, semantics. In other words, at binary level we have lost the capability to verify the semantic coherence between the code and the data while the reciprocity between data and software code determines the semantic validity of the data processing. For instance, consider a data element  $t$  to represent temperature, and a data algorithm to establish fever,

brandtp,  
8/30/2018  
Make use  
of 42010  
terminology

brandtp,  
8/30/2018 ...  
ditto ...

e.g.,  $t > 37.0 \rightarrow \text{fever}$ . The one and only means to keep the software from failing is that both the data and the algorithm (i) are expressed in the same unit of dimension ( $^{\circ}\text{C}$  in this example), apply the same (ii) resolution and (iii) accuracy, to name a few obvious constraints. We, therefore, take the stance that semantics can only exist in software by virtue of the semeiosis by the human-in-the-loop, while in the software agent itself semantics are necessarily reduced to the reciprocity between data and software code. Still, the software agent acts as transport medium for the semantics as intended by the software engineer to the semantics as experienced by the end user at the HMI. We therefore consider the coherence between data models and data processing models essential for enforcing the software agent to maintain a semantic valid reciprocity between binary code and the data it operates on.

This leads to the definition of a (normative (Greeffhorst and Proper 2011)) design principle to its effect:

**Design Principle 3.1 (Semantic coherence principle)**

*Establish explicit coherence between the models that are contained in a semantic monolith.*

**Type of information:** business

**Quality attributes:** (semantic) accuracy, reusability, manageability, understandability

**Rationale:**

1. *Semantics in software agents are necessarily reduced to, and emerge from, the reciprocity between the data and the binary code that operates on them;*
2. *Without explicitly addressing – at modelling level – all facets that influence the coherence between the data on the one hand, and the operations that apply on them on the other, the software agent cannot guarantee to maintain the reciprocity between them at the binary level;*
3. *Without maintaining the reciprocity between binary code and the data it operates on, the semeiosis performed by the end user on the result of the data processing and their subsequent semantics cannot be guaranteed to be similar as intended by the software engineer.*

**Implications:**

1. *The coherence principle is a necessary condition for supporting semantic interoperability;*
2. *The scope of semantic validity & accuracy is addressed explicitly and can be referred to;*
3. *Reuse of data often implies reuse of the data processing code, and vice versa. Having established explicit coherence improves the quality of data and code reuse, and facilitates the verification that the scope of the semantic validity & accuracy applies in the new context as well;*
4. *manageability ...?*
5. *understandability ...?*

◊

Coherence between models can be established with use of a single unique reference against which the truth of the expressions of both models can be verified. In semiotics, this single unique reference is considered reality, as indicated in Figure 2.1(b) by the *trueness* characteristic. Except as toy example in (Steels 2012), this is clearly not possible. The *correctness* characteristic is the only alternative left, taking the conceptualisation node as its principle point of reference, as depicted in Figure 3.2(b). This is exactly what the mathematical branch of *formal semantics* achieves (Gamut 1991; Genesereth and Nilsson 1987) with its three main characteristics, depicted in Figure 3.2(a), viz. connecting (i) an abstract syntax of a language to (ii) a domain of interpretation (usually a set theoretic framework) by defining (iii) an interpretation function from the abstract syntax onto the set theoretic framework. In terms of the semiotic triangle, Figure 2.1(b), this implies the following:

- (i) the *representation* node represents models that can be formulated by use of an abstract syntax (and grammar) as its modelling language. In this reading, a model is a particular constellation of tokens that represent a particular state of affairs;
- (ii) a particular *conceptualisation* can be mathematically formulated as a specific constellation of (unnamed) individuals, sets of individuals, and sets of sets;
- (iii) the *subjectivation* edge can be formulated as the interpretation function that assigns a mapping from modelling language tokens onto the set elements, enabling the evaluation of a specific model against the intended conceptualisation from (i).

In this way, the cognitive quality of the conceptualisation can be substituted with set theory. Formulating the conceptualisation as a set theoretic model essentially remains a representation, albeit a mathematical one. One can argue that such substitution does not resolve the grounding problem, and appropriately so. Still, mathematics provide for a very exact way to express oneself, reducing the ambiguity that comes implicitly with any other language, and such mathematical constructs come as close to the conceptualisation as we possibly can get with a token-based machine. Furthermore, logical constructs used at the syntactical level can be interpreted into set theoretic operations, facilitating the evaluation of (complicated) expressions. Formal semantics thus provides a domain of discourse about a particular conceptualisation as principle point of reference for both the data model and the data processing model. The particular conceptualisation is then replaced with a particular Domain of Interest (DoI), and both subjectivation relations are replaced with their interpretation function on the abstract syntax of the models.

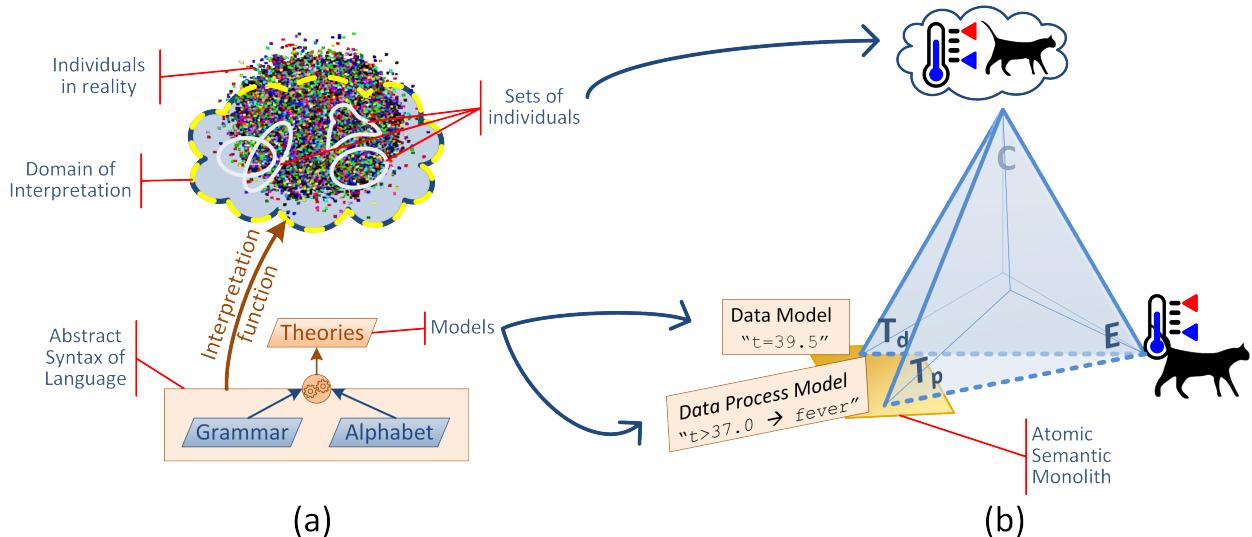


Figure 3.2. Maintaining the reciprocity between data and data processing models through a single semantic reference, viz. the conceptualisation (b), represented as DoI, viz. a selection of individuals with domain specific characteristics defined as sets (a).

In conclusion, we explain software semantics as the reciprocity between data and software code, realised by maintaining the coherence between pairs of data and data processing models, by applying formal semantics to formulate a particular conceptualisation as a DoI that can act as semantic reference, and interpretation functions which perform the subjectivation from the data and operation models to that reference.

<2nd Principle: Make the ASM as small as possible, but not smaller than required to express a semantic element. Too vague, yet. Not necessary to convey the primary message, imo.>

<Elaborate on OO to consolidate the reciprocity; take the class as example of a semantic monolith, the

minimal, atomic one.>

### 3.2 Explicit semantics

**Synopsis:** The purpose of this section is to establish that for representing semantics, descriptive models (i.e., ontologies) trump prescriptive models (all 42010 models)

Firstly, describe that formal semantics can to some extend take the role as conceptualisation, and hence using set theory as conceptualisation is the best option we have to address semantics. Consequently, show that semantic ambiguity then "only" follows from 4 construction issues (already presented in text below).

Secondly, make the distinction between descriptive and prescriptive models [Henderson & Sellers 2012], and in [Aßmann 2006]: "Specification models focus on the specification, control, and generation of systems; ontologies focus on description and conceptualization (conceptual modelling) of things. Both kinds of models have in common the qualities of abstraction and causal connection."

Second argument shows that (i) the trueness of a model is laid in reality, which is impossible to achieve, and that (ii) the next best we can achieve is establishing the correctness of a model against its conceptualisation node. Finally, observe that (iii) 4 issues will influence the correctness of the model. Then conclude that the best tool to control these issues are the logics from descriptive models (viz. ontologies) at the one hand, and its use as ontological commitment for the prescriptive models (viz. the 42010 models)

Conclusions:

1. ontologies are more appropriate artefacts for conceptual models than system models
2. as in [Aßmann 2006]: "Specification models focus on the specification, control, and generation of systems; ontologies focus on description and conceptualization (conceptual modelling) of things. Both kinds of models have in common the qualities of abstraction and causal connection."

Once we have established a conceptualisation, it needs a representation because, despite being a mathematical representation, the sets and individuals in the DoI remain nameless. It is necessary to represent them in a (pair of) model(s) that is tangible, computational and a distinct artifact in order for the software agent to use it and, eventually, make comparisons with the semantic artifact of the peer software agent. We denote the model that represents a conceptualisation a *conceptual model*. The conceptual model that describes semantics best is a model that is as faithful to reality as possible. This begs the questions about the nature of a conceptual model and the nature of faithfulness. To start with the latter and as indicated in Figure 2.1(b), we equate faithfulness with trueness between the representation and the entity in reality. Since this aspect is a characteristic of an indirect relationship, it can only emerge as a result of the adequacy and correctness characteristics. The former one concerns the conceptualisation aspect; in this section we address the correctness and show how differences can emerge between the conceptualisation and the model when constructing the latter. In literature, four construct issues are considered between a conceptualisations and its representation in a model, depicted in Figure 3.5. An exhaustive treatment can be found in (Guizzardi 2005), see also (Carvalho e Silva et al. 2012; or Azevedo et al. 2015).

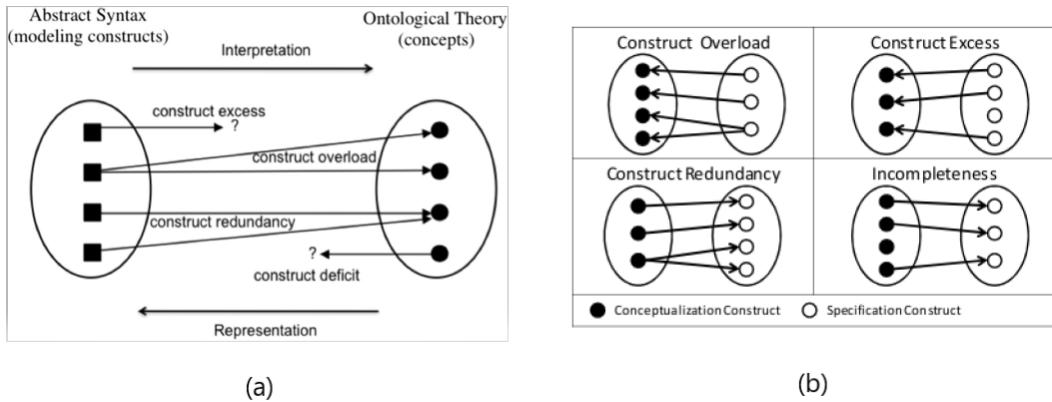


Figure 3.3. Four different types of construction issues that come with formal semantics. Which one is more clear: (a) (Carvalho e Silva et al. 2012) or (b) (Azevedo et al. 2015)?

The more precise the representation of the conceptualisation, the higher its *comprehensibility appropriateness*.\* Semantic accuracy improves when minimising the four construct issues:

- *Construct overload*, or *non-lucidity*, emerges when the interpretation function maps an element from the abstract syntax onto more than one element (individual, subset) from the conceptualisation: One token can mean more than one thing, e.g., **bank**;
- *Construct excess*, or *unsoundness*, represents an abstract syntax element that does not map onto an element from the conceptualisation: One token does not have a meaning at all;
- *Construct redundancy*, or *non-laconicity*, occurs when more than one abstract syntax element can be used to represent an element from the conceptualisation: More tokens mean the same, e.g., ...;
- *Construct deficit*, or *incompleteness*, specifies the situation where a conceptual element does not map onto a token: It is impossible to express a concept because the language has no token for it, e.g., .... (example intentionally left blank).

The best representation of a conceptualisation, then, is a model that is lucid, sound, laconic and complete, i.e., the interpretation/representation mapping is isomorphic. This is difficult to achieve, and due to the complete, accurate and complementary categories of how its carves-out the DoI, using the ontological commitment of the modelling language proves a useful tool in that respect.

In contemporary architectural paradigms, models are being used as first class citizens to the architectures, MDA, IEEE-1471 and ISP RM/ODP alike. It is therefore imperative to decide upon the major characteristics of a model. The distinction between descriptive and prescriptive models is broadly accepted, where a prescriptive model is used to *specify* its subject, and a descriptive model is used to *describe* its subject. As explained by (Gonzalez-Perez and Henderson-Sellers 2007), this distinction only addresses how the model is being used and does not express a complementary, disjunct property. In stead, a genuine dichotomy emerges when considering the *role* that the model plays (*ibid.*). Then, a specification model generally takes a forward-looking role, specifying how the subject is supposed to be. A descriptive model, on the other hand, generally takes a backward-looking role and addresses the subject as it currently is, or once has been. In (Alßmann et al. 2006), the relationship that a forward-looking model has with its subject is characterised as an *instance-of* relation, and that of a backward-looking model as an *is-represented-by* relation. Although we acknowledge the differences that exist, we consider both terms confusing when semiotics are involved since we cannot substantiate their suggested equivalence with the instantiation relation from the ontological square/sextet or the representation relation from the semiotic triangle. Instead, we observe that the characteristics of the *refers-to* relation will specialise into a *declarative* relation for a forward-looking model: The entity is

\*brandtp,  
9/19/2018 de-  
scribe/explain  
the term

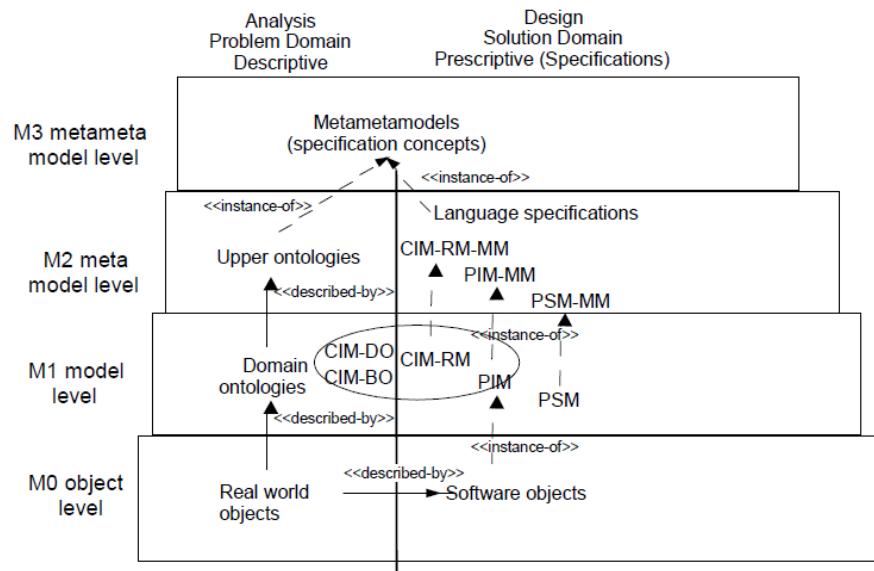
brandtp,  
9/19/2018  
find proper  
example

deemed to follow the rules that are specified by the model. Similarly, for a backward-looking model that provides for a representation of “what a given remark or doctrine, ours or someone else’s, says there is”, the refers-to relation specialises into a *descriptive* relation. We thus observe that the refers-to relation is a causal relation, the direction of which is dependent on the model’s characteristic: declarative towards the entity for a forward-looking model, while descriptive towards the token for a backward-looking model. As a consequence, we conclude that for forward-looking models their truth lies in their meta-models, as depicted in Figure 2.2(b). *Mutatis mutandis*, the truth for backward-looking models lies in the entities they describe.

1. Explain: difference between ontologies and models

- i. Models lack an elaborate ontological commitment, domain ontologies naturally evolve on foundational ontologies (that express an ontological commitment)
- ii. For prescriptive models, truth lies in meta-models (good for deterministic behaviour); ontologies are descriptive models for which the truth lies in reality (good for semantics)
- iii. ontologies have open world assumption (semantic under-specification), models have closed world assumption (data remain consistent, good for performance)
- iv. Models specify systems, ontologies conceptualise reality (entities)
  - Try to also connect the onto/model distinction with above principles
  - Induced problem: from OWA (domain ontologies) to CWA (information/data models), viz. how to get closure?
  - Principle: use domain and business ontologies for “computational independent”-ish models (Aßmann et al. 2006)
  - Bridge descriptive → prescriptive models by grounding all prescriptive model elements with concepts from descriptive model (see e.g. Figure 3.6).

In the remainder of this text we will refer to the formulation of the reference conceptualisation as a *conceptual model*.



**Fig. 9.8.** A proposal for the role of ontologies in meta-pyramid of MDE and the MDA

*Figure 3.4. The use of ontologies in MDA, from (Aßmann et al. 2006)*

### 3.2.1 The nature of faithfulness

To start with the latter and as indicated in Figure 2.1(b), we equate faithfulness with trueness between the representation and the entity in reality. Since this aspect is a characteristic of an indirect relationship, it can only emerge as a result of the adequacy and correctness characteristics in that same Figure. Adequacy concerns the quality of the abstraction, which we do not address here. Correctness concerns the quality of the representation, and in this section we show how differences can emerge between the conceptualisation and the model when constructing the latter from the former.

In literature, four construct issues are considered between a conceptualisations and its representation in a model, depicted in Figure 3.5. An exhaustive treatment can be found in (Guizzardi 2005), see also (Carvalho e Silva et al. 2012; or Azevedo et al. 2015).

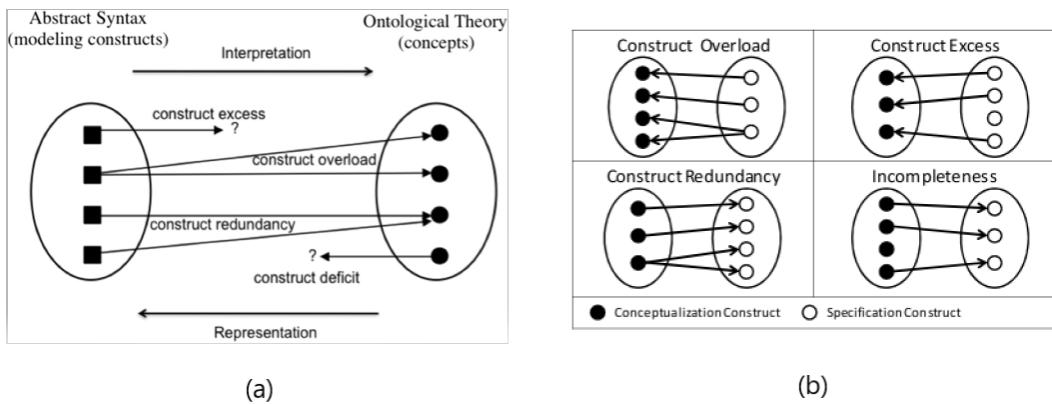


Figure 3.5. Four different types of construction issues that come with formal semantics. (a) (Carvalho e Silva et al. 2012), (b) (Azevedo et al. 2015)

The correctness with which tokens represent the conceptualisation improves when the four construct issues are minimised:

- *Construct overload*, or *non-lucidity*, emerges when the interpretation function maps an element from the abstract syntax onto more than one element (individual, subset) from the conceptualisation: One token can mean more than one thing, e.g.,  $\mathcal{I}(\text{"bank"}) = \{ \text{S}, \text{F}, \text{E} \}$ , where **S** stands for sofa, **F** for a financial institute, and **E** for a rising edge;
- *Construct excess*, or *unsoundness*, represents an abstract syntax element that does not map onto an element from the conceptualisation: One token does not have a meaning at all,  $\mathcal{I}(\text{"pfh"}) = \perp$ , where the *falsum* concept ( $\perp$ ) denotes the empty set (the valuation of which is always False);
- *Construct redundancy*, or *non-laconicity*, occurs when more than one abstract syntax element can be used to represent an element from the conceptualisation: More tokens mean the same, e.g.,  $\mathcal{I}(\text{"xx?"}) = \text{zz?}$  and  $\mathcal{I}(\text{"yy?"}) = \text{zz?}$ ;
- *Construct deficit*, or *incompleteness*, specifies the situation where a conceptual element does not map onto a token: It is impossible to express a concept because the language has no token for it, e.g.,  $\mathcal{I}^{-1}(\text{c}) = \text{ }$ , with **c** some conceptual element and token  intentionally left blank.

The best representation of a conceptualisation, then, is a model that is lucid, sound, laconic and complete, i.e., the subjectivation/representation mapping is isomorphic. This strongly depends on the modelling language, and particularly its appropriateness on how it carves-out the reality of the DoI, viz. its ontological commitment. We turn this into the following design principle.

### **Design Principle 3.2 (Domain appropriateness principle)**

*The faithfulness of the semantic model depends on the ontological commitment of the modelling language: the more appropriate it carves out the domain of interest, the more faithful the domain model can represent the conceptualisation.*

**Type of information:** business

**Quality attributes:** (semantic) accuracy, re-usability, manageability, understandability

**Rationale:**

1. *A software agent cannot act effectively without a model of its DoI that represents the DoI. The trueness with which that model refers to the DoI directly reflects the quality of (the results of) the agent's actions. Trueness emerges as the concatenation of the \*adequacy\* of the abstraction from reality and the \*correctness\* with which the conceptualisation is represented;*
2. *The correctness of the representation of the conceptualisation improves by minimising the construct issues that occur in the domain model;*
3. *Construct issues are a direct result of the necessity to model (represent) a characteristic aspect of a concept with a language element that is not suited or even present to express that characteristic. Unable to correctly express a characteristic is a direct consequence of the inadequacy of the underlying ontological commitment;*

**Implications:**

1. *The more specific the ontological commitment gets, the higher the semantic accuracy with which the domain model can refer to the DoI;*
2. *The more specific the ontological commitment gets, the higher its complexity which might even turn into a NP-complexity;*
3. *The ability of a software agent for semantic interoperability increases with the domain appropriateness of the domain model;*
4. *The achieved semantic interoperability for a pair of agents depends highly on the correspondence between the ontological commitments of their domain models;*
5. *The re-usability of the agent improves with*
6. *manageability ...?*
7. *understandability ...?*

◊

It is the responsibility of the software engineer to select a modelling language that provides a balance between semantic faithfulness at the one hand and computational (and comprehensibility) complexity for representing the domain model.

#### 3.2.2 The nature of models

In contemporary architectural paradigms, models are being used as first class citizens to the architectures, MDA, IEEE-1471 and ISP RM/ODP alike. It is therefore imperative to decide upon the major characteristics of a model. A in-depth treatise of models, meta-models, modelling languages and ontologies can be found in (Henderson-Sellers 2012), where the authors identify 6 modes of a model, derived from three dichotomies: the type vs. token modes, the forward- vs. backward-looking modes, and the ontological vs. linguistic modes, as follows.

- Type models represent different categories that exist in the modelled world, whilst token models represent the individuals from the modelling world that are of interest. This distinction essentially represent the *universals – particulars* dichotomy as identified in the ontological square (refer to ??). As a result, the relationship that is instantiated with its modelled world by a type model is an *instance-of* relationship, while for a token model this is an *equivalence* relationship;
- The distinction between descriptive and prescriptive models is broadly accepted, where a prescriptive model is used to *specify* its subject, and a descriptive model is used to *describe* its subject. As explained by (Gonzalez-Perez and Henderson-Sellers 2007), a specification model generally takes a *forward-looking* role, specifying how the subject is supposed to be. A descriptive model, on the other hand, generally takes a *backward-looking* role and addresses the subject as it currently is, or once has been. We observe that the characteristics of the *refers-to* relation become declarative for a forward-looking model whereas they become descriptive for a backward-looking model. Therefore, the *refers-to* relation is a causal relation, the direction of which is dependent on the model's characteristic. This is consequential for the location of the truth of a model: For forward-looking models their truth lies in their meta-models. *Mutatis mutandis*, the truth for backward-looking models lies in the entities they describe.
- The distinction between ontological and linguistic mode of a modelling was first coined by (Atkinson and Kuhne 2003). The distinction often refers to meta-models and ontologies. Essentially, it is about the confusion that emerges between relationships that model *instance-of* relations (a particular tree that gives rise to a universal Tree), *categorisation* relations (one or more criteria determine the group an individual belongs to, e.g., evergreens versus deciduous trees) and *generalisation* relations (a taxonomic hierarchy, e.g., in the Linnaean system of classification an oak belongs to *family:Fagaceae*, *order:Fagales*, *subclass:Hamamelididae*, etc., up to *kingdom:Plantae*). Linguistic meta-modelling assumes OMG's four layer MOF hierarchy; for each of the adjacent pair of layers the meta layer is considered to model the types that the lower layer can apply in its own model. Adjacent layers are therefore connected through a type-instance relationship, denoted as *linguistic instance-of* but better relabelled *conforms-to* [Bézivin (2004); Favre2004a] in order to prevent ambiguities by overloading the *instance-of* label. Contrarily, ontological meta-modelling acknowledges only one single genuine *instance-of* relationship between the universals and their individuals. At the DoI, the conceptualisation also recognises subsets of supersets, which translates to the generalisation relationship (Kühne 2006). Although the author clearly identifies in (Atkinson and Kuhne 2003; and Atkinson et al. 2011) the necessity for both modes of modelling, the method of their integration is not without issues (Henderson-Sellers 2012). We follow the latter in their method of integrating both modelling modes.

An ontology is considered a backward-looking model, which describes almost always a domain of interest as opposed to systems or components thereof. We summarise the distinction between ontologies and (forward-looking) models in ??.

Table 3.1. The main differences between an ontology and a (forward-looking) model. {tab:onto-versus-model}

Subject	Ontology	Model
Type vs. tokens mode	The main design is a type mode, carrying concepts and their relations in terms of Universals. Inclusion of individuals is very well possible, providing for a token mode as well.	Type mode and token mode can be applied independently from each other, or merged into one model as well.
Prescriptive vs. descriptive mode	Mainly backward looking	Mainly forward looking. Backward looking models often apply a MOF-oriented ontological commitment or similar.

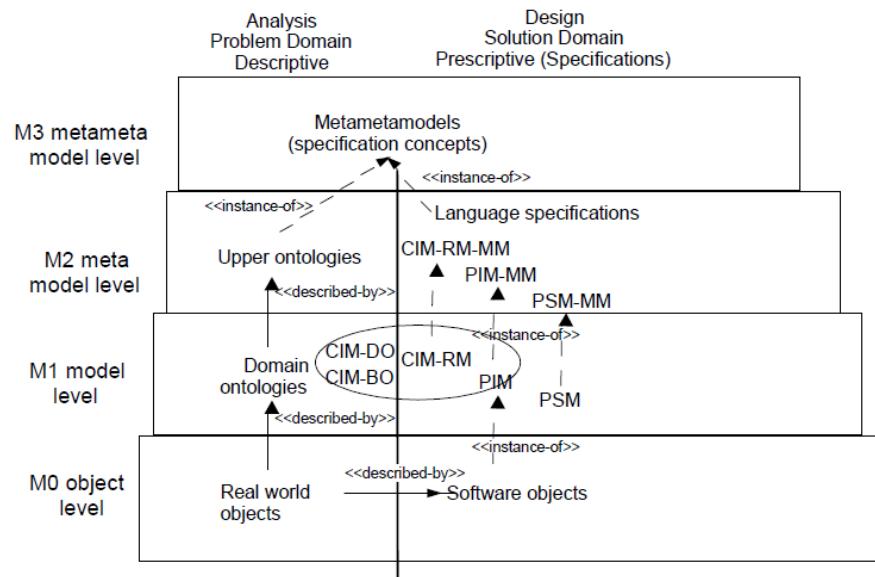
Subject	Ontology	Model
Ontological vs. linguistic mode	Ontological modelling.	Mainly linguistic modelling.
Logical assumption	Open world assumption (OWA) maintains a semantic under-specification	Closed world assumption (CWA) maintains data consistency and query performance
Source of truth	Reality	Meta-model
Focus	Explanatory & understanding	Constructive & realisation
Type of Typing	Subtyping, a.k.a. “descriptive typing” <b>Kuhne2018</b> , i.e., membership of an instance to type T is only dependent on whether or not it has the required properties. Formally: $T.\epsilon = \{ x   T.\iota(x) \}$	“Nominal typing”, explicit typing relationships between individuals and their types. Formally: $\forall x : (x.type = T) \rightarrow T.\iota(x)$
Constrain purpose	Axioms constrain the possible interpretations, and scope the automated reasoning.	Constraints enforce integrity of the data, and are used to optimise processing performance
Objects of interest	The categories of being which are assumed to exist in a certain domain independently of particular information demands	It is the goal of the information level to define one among the many possible structures of propositions (information) about a domain and to characterise a certain information demand.

We have already indicated in Section 2.2 the importance of the ontological commitment for the modelling language. Contemporary modelling frameworks yield meta-models that have a clear engineering focus which is very suitable for forward-looking, prescriptive (system) models. At the same time, these ontological commitments fall short for backward-looking, descriptive models. We consider domain ontologies, which naturally evolve on foundational ontologies (that express an ontological commitment), a necessary component to faithfully describe the domain of interpretation and formally represent the domain of interest. It is in the interest of the

1. Explain: difference between ontologies and models
  - i. Models lack an elaborate ontological commitment, domain ontologies naturally evolve on foundational ontologies (that express an ontological commitment)
  - ii. For prescriptive models, truth lies in meta-models (good for deterministic behaviour); ontologies are descriptive models for which the truth lies in reality (good for semantics)
  - iii. ontologies have open world assumption (semantic under-specification), models have closed world assumption (data remain consistent, good for performance)
  - iv. Models specify systems, ontologies conceptualise reality (entities)
    - Try to also connect the onto/model distinction with above principles
    - Induced problem: from OWA (domain ontologies) to CWA (information/data models), viz. how to get closure?
    - Principle: use domain and business ontologies for “computational independent”-ish models (Aßmann et al. 2006)
    - Bridge descriptive  $\rightarrow$  prescriptive models by grounding all prescriptive model elements with concepts from descriptive model (see e.g. Figure 3.6).
  - v. Different aggregation levels of ASM:

- vi. Differentiate between an information model with “semantic meaning” from the process model with “pragmatic meaning”, and maintain their semantic coherence by having one ontology available as their single unique reference (either formally the domain of interpretation for  $f_d$  and  $f_p$ , or pragmatically, using one single ontology as conceptual model).

In the remainder of this text we will refer to the formulation of the reference conceptualisation as a *conceptual model*.



**Fig. 9.8.** A proposal for the role of ontologies in meta-pyramid of MDE and the MDA

*Figure 3.6. The use of ontologies in MDA, from (Aßmann et al. 2006)*

# Chapter 4

## Spanning: Alignments

### 4.1 What is semantic interoperability

#### Synopsis

1. Explain sIOP:
  - a) that consequence of data exchange = breaking the atomic semantic monolith = breaking the reciprocity by partitioning the data from its original code, and explain that standards are large semantic monoliths that roofs that point but are as manouverable as an oil tanker, and
  - b) that sIOP demands that despite this partitioning the reciprocity between the code of the receiving agent and the external data shall be re-installed.
  - c) Optionally, give a definition on phantom semantics
2. We therefore need to extend the semantic coherence Principe into a sIOP coherence principle with a sIOP rational that the result of the semeiosis on receiving agent A' does not conflict with the outcome of the semeiosis on agent A: Without maintaining the reciprocity between binary code and the data it operates on, the semeiosis performed by software engineer A' on the result of the data processing and their subsequent semantics cannot be guaranteed to be similar as intended by the software engineer.
3. Explain the difference with semantic standards which are basically large semantic monoliths
4. Introduce Principle: ontological commitment as minimal standard for sIOP, “(...) not in order to know \*what there is\*, but in order to know what a given remark or doctrine, ours or someone else's, \*says\* there is” [@Quine:1953er]

\end{synopsis}

Rephrase: despite the notoriously difficult philosophical questions involved, semantic interoperability can be seen as an engineering problem, namely that of effectively constraining interpretations towards the ones that are considered allowable (Kuhn 2009).

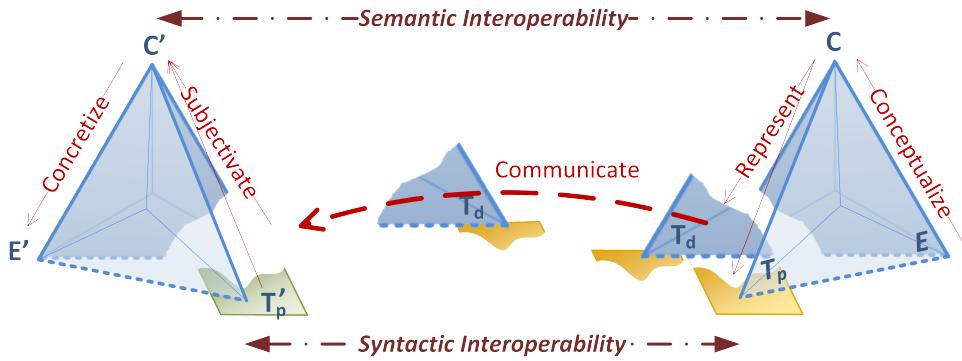


Figure 4.1. The various forms of interoperability

## 4.2 Explicit sIOP by alignments

**Synopsis:** Thus:

1. Follow the coherence principle and conclude that the models from which \*external\* data and \*receiving\* data processing code are derived, need to be brought into coherence with each other.
2. The coherence principle already enforced a single unique reference for each agent. Re-installing coherence demands a semantic alignment between those single unique references.
3. The purpose of that alignment is to establish how the truth of expressions that are formulated in terms of agent A, can be established by using formulations in terms of agent A' against the single unique reference from A'.
4. The language that is used for expressing the alignment should be fit for its purpose. Refer to EDOAL [Scharffe2011] as the currently most complete one.

Text

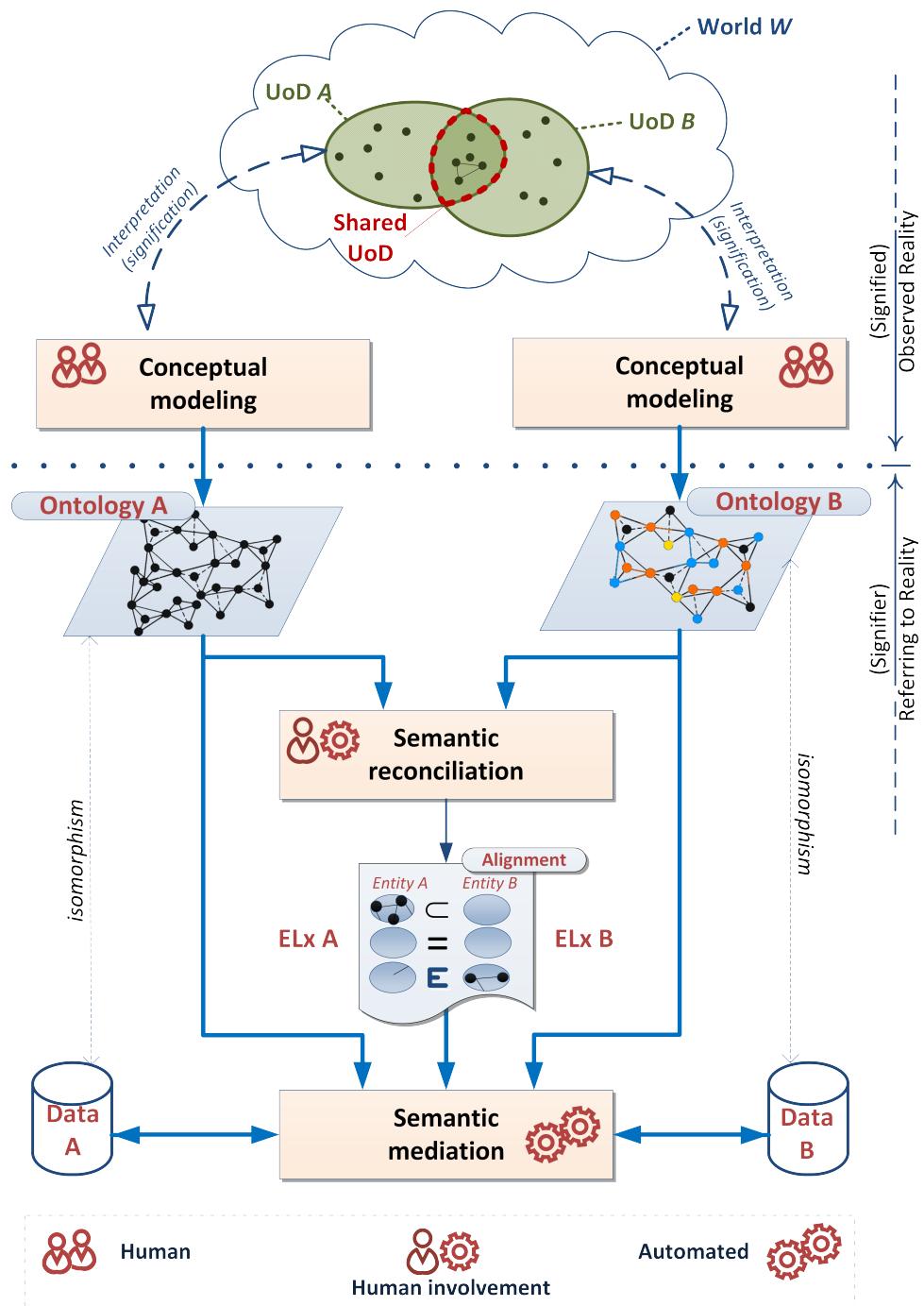


Figure 4.2. The three semantic concerns are related: conceptual modelling, semantic reconciliation, and semantic mediation

# Chapter 5

## Roadway: Mediation

**Synopsis:** Purpose of this section: Establish requirements for a \*generic\* mediator.

With mediation we denote the process of transcribing a data term that originates from Agent A into a data term that matches a term familiar to Agent A', based on both agents' ontologies and the alignment between them. The main issue here is that although many different types of relation can be defined between the concepts of ontology A and A', e.g., superset of, a transcription of a token from A into a token from A' is a complete replacement and, hence, implements an equivalence relation. In [@Brandt2018b], we show a semantic valid transcription process. The requirements of a mediator are:

1. Being a generic service
2. Fully defined by two ontologies and their alignments
3. Allows for semantic valid transcriptions only, where 'validity' refers to absence of inducing phantom semantics.
4. Appropriate behaviour for non-translatable content, which should apply only as result of an incomplete alignment, a logical incorrect alignment, or attempts to communicate content that is considered irrelevant for the receiving agent.

# Chapter 6

## sIOP Principles

**Synopsis:** Purpose of this section:

- Show the semantic architecture as an additional layer that is orthogonal to current layers, expressing a separation of concerns between syntax and semantics (see [@Brandt:2013jh])
- Define loosely coupled semantics as a result of applying the 2 principles 'semantic separation of concerns' and 'semantic transparency' (see [@Brandt:2013jh])
- Repeat the Coherence Principle, the sIOP Coherence Principle, and the Ontological Commitment Principle, and show how they fit in the semantic architecture
- Better structure informal text below, towards more Principle definitions.

The main (business) requirement is to achieve sIOP as quickly as possible, with as minimal effort as possible, for collaborations that had not been foreseen and consequently could not be anticipated for during design time of the (two or more) software agents.

Consequently, the software agents have been developed totally and completely independent from each other. This raises the following semantic concerns:

1. Loosely coupled semantics:

- i. Define semantics once during software design phase, and achieve sIOP many times with many different peers
- ii. EW Dijkstra: Connected but as independent as possible. In its original reading this implies only defining the *what* but leaving the *how* transparent. For semantics the implication is a more abstract one: the semantics of what is being communicated shall remain transparent to *how* it is represented. More specifically, agents shall rely on an external oracle that can change the semantic vehicle from its original source native representation to the destined target representation, without changing the semantic cargo. Agents, then, can communicate in their own native representations without the need to learn or integrate their peers' representations.

2. Scalable sIOP:

- i. Variable in number of peers
- ii. Variable in level of semantic heterogeneity

3. Semantic concerns are foundational to sIOP (see Figure 4.2 for three related ones):

- i. Explicit and computational semantics by *conceptual modelling*: Bridgehead
- ii. Managed and controlled sIOP by *semantic reconciliation*: Spanning

- iii. Automated sIOP by *semantic mediation*: Roadway. Address semantic issue about the non-equivalence between an alignment and a transcription (refer to **Brandt2018b**)

*Ad. Dijkstra's "Connected but as independent as possible".* Complement weak AI with human brain:

- use AI where possible (computational semantics for software agent; supporting semantic reconciliation)
- use human brain where necessary (but not more): ontology engineering @ design time; alignment authoring @ pre-runtime

### ***Achieve loosely coupled semantics***

Loose coupling is founded on principles about (i) separation of concerns, and (ii) transparency:

- Principle *Separation of concerns*:
  - Classical:
    - i. Decompose system in parts
    - ii. with minimal functional overlap
  - Semantical:
    - i. Separate your own semantics (i.e., conceptualisations, viz. let each software agent manage its own abstraction from reality)
    - ii. from establishing sIOP
- Principle *Transparency*
  - Classical:
    - i. Agnostic to *how* its functions are being achieved
    - ii. Communicate with minimal mutual dependency
  - Semantical:
    - i. Agnostic to *how* semantics are being achieved
    - ii. Communicate with minimal syntactic dependency, i.e., without agreements on semantic representation

Formulate the principles in the format according to (Greefhorst and Proper 2011)

*Ad. semantic separation of concern.* Where in its classical application the result of applying the principle is that atomic functions are defined, designed and implemented only once and remain unique, in its semantic application the result of applying this principle is that every software agent maintains its own semantics. Semantics are, therefore, distributed all over the place. This seems counterintuitive or even plain wrong, however, it is necessary for complying with the concern about semantic scalability (in support of heterogeneous semantics). Besides that, it is a direct consequence of the demand to allow for independent software development

- Principle: specify ontological commitment as basic
- Refer to (and partly reuse?) semantic architecture from (Brandt et al. 2013), depicted in Figure 6.1

### ***Achieve Scalable sIOP***

Ensure that different semantic topologies remain possible:

- i. Star alignments (central domain ontology, aligned to local ontologies) for relative stable and homogeneous domain semantics

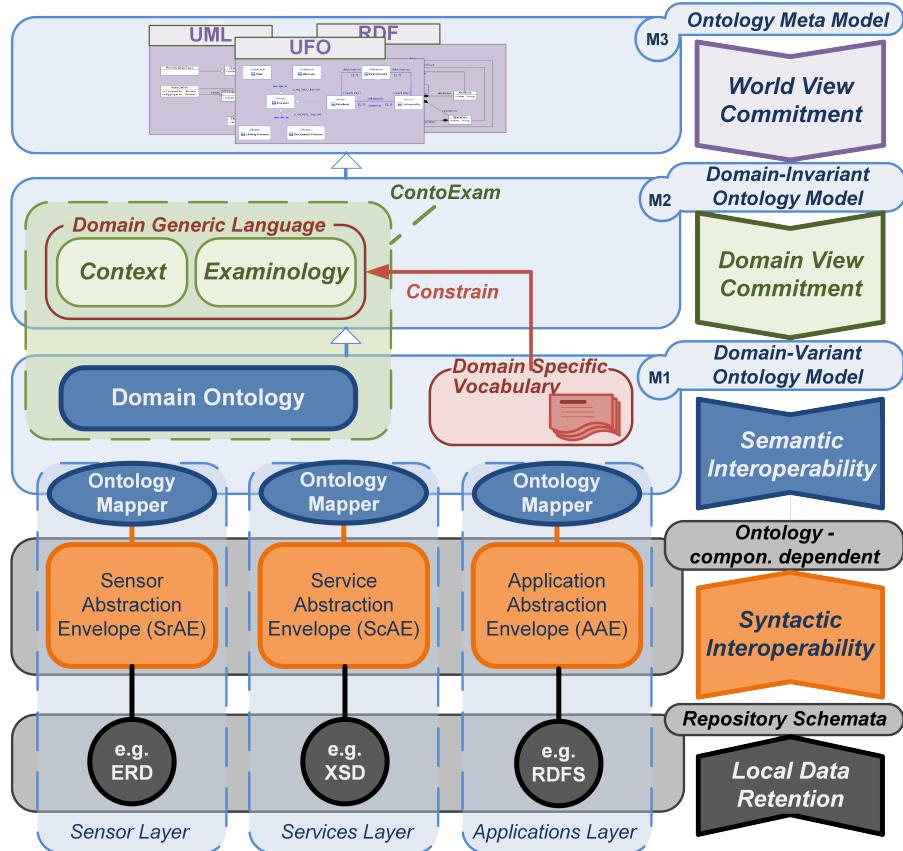


Figure 6.1. An architecture for loosely coupled semantics, founded on semantic SoC and semantic transparency (Brandt et al. 2013)

- Good: easy semantic governance
  - Bad: very big semantic monolith, hence, low agility in dynamic environments
- ii. Mesh alignments (bilateral alignments) for very dynamic and heterogeneous (domain) semantics, or low number of peers
- Good: quickly established bilateral sIOP; granularity-on-demand, viz. intricate where necessary, coarse-grained where possible
  - Bad: complicated semantic governance
- iii. Mix-n-Match (coarse-grained star-alignment with specialised bilateral alignments) for the 70% bulk \*  
 Good: controllable semantic governance; after central alignment, quickly established bilateral sIOP \* Bad: slightly more complicated mediation due to double alignment support

# Chapter 7

## ISO42010 viewpoint on sIOP

**Synopsis:** Consolidate the ideas on the bridgehead, spanning, roadway and principles into an additional ISO42010 Architectural Viewpoint (sIOP) that summarises all previous Sections as concerns on semantics and sIOP. \*\*\*Preferrably written by Eric.\*\*\*

# Chapter 8

## Related work

**Synopsis:** Group the papers into 3 (?) categories, and discuss their strong and weak points in relation to SIOP and architecture in general, and our paper specifically.

Discuss the following papers:

1. M. B. Almeida, C. P. Pessanha, and R. Barcelos, "Information Architecture for Organizations: An Ontological Approach," in *Ontology in Information Science*, C. Thomas, Ed. IntechOpen, 2018, pp. 1-27.
2. S. Yang, J. Guo, and R. Wei, "Semantic interoperability with heterogeneous information systems on the internet through automatic tabular document exchange," *Inf. Syst.*, vol. 69, pp. 195-217, Sep. 2017.
3. U. Alßmann, S. Zschaler, and G. Wagner, "Ontologies, Meta-models, and the Model-Driven Paradigm," in *Ontologies for Software Engineering and Software Technology*, C. Calero, F. Ruiz, and M. Piattini, Eds. Springer-Verlag Berlin Heidelberg, 2006, pp. 249-273.
4. C. Atkinson and T. Kühne, "The Essence of Multilevel Metamodeling," *LNCS*, vol. 2185, pp. 19-33, 2001.
5. H. Carvalho e Silva, R. de Cassia Cordeiro de Castro, M. J. Negreiros Gomes, and A. Salles Garcia, *Well-Founded IT Architecture Ontology: An Approach from a Service Continuity Perspective*, vol. 294. Springer-Verlag Berlin Heidelberg, 2012.
6. R. Carraretto, "Separating Ontological and Informational Concerns : A Model-driven Approach for Conceptual Modeling," Federal University of Espírito Santo, 2012.
7. C. L. B. Azevedo, M. E. Iacob, J. P. A. Almeida, M. J. van Sinderen, L. F. Pires, and G. Guizzardi, "Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate," *Inf. Syst.*, vol. 54, pp. 235-262, 2015.
8. M. B. Almeida, C. P. Pessanha, and R. Barcelos, "Information Architecture for Organizations: An Ontological Approach," in *Ontology in Information Science*, C. Thomas, Ed. IntechOpen, 2018, pp. 1-27.
9. D. Gasevic, D. Djuric, and V. Devedzic, Eds., *Model Driven Architecture and Ontology Development*. Springer Berlin Heidelberg New York, 2006. Particularly Part II: The Model Driven Architecture and Ontologies

# Chapter 9

## Discussion & future work

**Synopsis:** Address shortcomings that we discover throughout writing the sections.

Conclude that by identifying a specific 42010 viewpoint on sIOP, a necessary condition towards the preparation of a sIOP capability in a software agent has been identified which can be applied to all MDE and view-based software architectures.

## References

Note: \_\_\_\_\_

- Atkinson C, Kennel B, Goß B. 2011. The Level-Agnostic Modeling Language. *Softw. Lang. Eng.* 2010, 3rd int. Conf. 266–275; doi:10.1007/978-3-642-19440-5\_16.
- Atkinson C, Kuhne T. 2003. Model-driven development: a metamodeling foundation. *IEEE Softw.* 20:36–41; doi:10.1109/MS.2003.1231149.
- Azevedo CL, Iacob ME, Almeida JPA, Sinderen MJ van, Pires LF, Guizzardi G. 2015. Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate. *Inf. Syst.* 54:235–262; doi:10.1016/j.is.2015.04.008.
- Aßmann U, Zschaler S, Wagner G. 2006. Ontologies, Meta-models, and the Model-Driven Paradigm. In *Ontol. Softw. Eng. Softw. Technol.* (C. Calero, F. Ruiz, and M. Piattinieds. ), pp. 249–273, Springer-Verlag Berlin Heidelberg.
- Bézivin J. 2004. In search of a basic principle for Model Driven Engineering. *Spec. Novatica Issue - UML Model Eng.* 5: 21–24.
- Brandt P, Basten T, Stuijk S. 2013. Semantic interoperability in sensor applications making sense of sensor data. ... *healthc. E- ...* 5: 34–41.
- Bricker P. 2016. Ontological Commitment. In *Stanford encycl. Philos.* (E.N. Zaltaed. ), \url{https://plato.stanford.edu/archives/win- commitment/}; Metaphysics Research Lab, Stanford University.
- Carvalho e Silva H, Cassia Cordeiro de Castro R de, Negreiros Gomes MJ, Salles Garcia A. 2012. *Well-Founded IT Architecture Ontology: An Approach from a Service Continuity Perspective*. R. Benlamried.. Springer-Verlag Berlin Heidelberg.
- Cregan AM. 2007. Symbol grounding for the semantic web. W. Franconi, E and Kifer, M and Mayed.. *Semant. WEB res. Appl. Proc.* 4519: 429–442.
- Eco U. 1976. *A theory of semiotics*. Indiana University Press / London: Macmillan, Bloomington, IN.
- Gamut L. 1991. *Logic, Language and Meaning, volume 1: Introduction to Logic*. The University of Chicago Press.
- Genesereth MR, Nilsson NJ. 1987. *Logical foundations of artificial intelligence*. Morgan Kaufmann Publishers Inc.
- Gonzalez-Perez C, Henderson-Sellers B. 2007. Modelling software development methodologies: A conceptual foundation. *J. Syst. Softw.* 80:1778–1796; doi:10.1016/j.jss.2007.02.048.
- Greefhorst D, Proper E. 2011. *Architecture Principles, The Cornerstones of Enterprise Architecture*. Springer Berlin Heidelberg.
- Grice HP. 1989. Logic and Conversation. In *Stud. W. Words*, pp. 22–40, Harvard University Press, Cambridge,

brandtp,  
9/5/2018 Also  
show the  
ref-id per  
reference  
\*duh\*

- MA, USA.
- Guarino N. 1994. The Ontological Level. R. Casati, B. Smith, and G. Whiteheads. *Philos. Cogn. Sci. Proc.* 16th int. Wittgenstein symp. 443–456; doi:10.1007/978-3-642-02463-4.
- Guizzardi G. 2005. Ontological foundations for structural conceptual models. PhD thesis, University of Twente, The Netherlands; CTIT, Enschede, The Netherlands.
- Guizzardi G, Wagner G, Almeida JPA, Guizzardi RSS. 2015. Towards ontological foundations for conceptual modeling: The unified foundational ontology (UFO) story. *Appl. Ontol.* 10:259–271; doi:10.3233/AO-150157.
- Harnad S. 1990. The Symbol Grounding Problem. *Physica D* 42. 335–346.
- Henderson-Sellers B. 2012. *On the mathematics of modelling, metamodelling, ontologies and modelling languages*. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Jansen L. 2008. Categories : The Top-Level Ontology. In *Appl. Ontol. An introd.* (K. Munn and B. Smitheds. ), pp. 173–196, De Gruyter.
- Kuhn W. 2009. Semantic engineering. In *Res. Trends geogr. Inf. Sci. Lect. Notes geoinf. Cartogr.* (G. Navratiled. ), pp. 63–76, Springer Berlin Heidelberg.
- Kühne T. 2006. Matters of (meta-) modeling. *Softw. Syst. Model.* 5:369–385; doi:10.1007/s10270-006-0017-9.
- Ogden CK, Richards IA. 1989. *The Meaning of Meaning: A Study of the Influence of Language upon Thought and of the Science of Symbolism*. with a pre. Harcourt Brace Jovanovich, New York, USA.
- Quine WVO. 1961. From a logical point of view. *Br. Dent. J.* 195: 229.
- Searle JR. 1980. Minds, brains, and programs. *Behav. Brain Sci.* 3: 417–424.
- Smith B. 2005. Against fantology. In *Exp. Anal.* (M. Reicher and J. Marekeds. ), pp. 153–170, HPT&ÖPV, Vienna.
- Sowa JF. 2000. Ontology, metadata, and semiotics. *Lect. Notes Comput. Sci.* (including Subser. *Lect. Notes Artif. Intell. Lect. Notes Bioinformatics*) 1867:55–81; doi:10.1007/10722280\_5.
- Steels L. 2012. The symbol grounding problem has been solved, so what's next. In *Symb. Embodiment debates mean. Cogn.* (M. de Vega, A. Glenberg, and A. Graessereds. ), pp. 223–244, Oxford University Press, Oxford, UK.
- Xiuquan Li, Tao Zhang. 2017. An exploration on artificial intelligence application: From security, privacy and ethic perspective. J. Zhu, E.-B. Lin, and T. Lieds. 2017 ieee 2nd int. Conf. Cloud comput. Big data anal. 416–420; doi:10.1109/ICCCBDA.2017.7951949.