ESTABLISHING SEMANTIC INTEROPERABILITY IN CONTEMPORARY ARCHITECTURAL PARADIGMS

Consolidating loosely coupled semantics

version: v0.4-6

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

Eric Grandry,
??,

Marten van Sinderen,
University of Twente, Enschede, The Netherlands,

Twan Basten,
Eindhoven University of Technology, Eindhoven, The Netherlands,

Abstract

Background: *Access-and-Play semantic interoperability (sIOP) is the next glass ceiling in IT-based business collaboration. Current approaches towards sIOP still rely on conventions on the semantics of the exchanged terms, which can be considered accepted folklore. To break through the ceiling an initial agreement is required on the foundations of semantics and sIOP. Unfortunately, semantics and software are on odds with each other: software can only operate on a token-based machine whereas semantics require an interpretation outside the realm of tokens. This fundamental incompatibility depends for its resolution on the human-in-the-loop. With current artificial intelligence (AI) the fundamental incompatibility cannot be solved. However, the effort of the inevitable human-in-the-loop can be reduced and her position improved. This is a matter of software architecture, and it should address how semantic interoperability can be consolidated by it.*

Objective: *The objective of our study is to identify and formulate the fundamental demands towards access-and-play interoperability, to derive their supporting architectural principles, and its integration in contemporary architectural paradigms.*

Method: *We follow a design-science approach and address the business relevance of the problem, and identify six requirements on sIOP, two of which are concerned with a genuine understanding of semantics that demand for the human-in-the-loop. We assume that the collaborating agents have followed the architectural principles on semantics according to our preliminary study (Brandt et al. 2019), which results in an explicit representation of an atomic semantic monolith for each of the agents: two semantic bridgeheads. We subsequently develop four design principles in order to support interoperability between the semantic bridgeheads, and one design principle to cater for the semantic distinction between a formal semantic correspondence and the necessary*

data transcription during communication. We finally evaluate these principles by designing and formulating an ISO-42010 Architecture Viewpoint and View on sIOP.

Results: *Semantics in software are the result of a reciprocity between data and the software code that operates on them, resulting in a local semantic monolith (Brandt et al. 2019). Data exchange breaks that semantic monolith and hence the aforementioned reciprocity. The main concern of sIOP is to re-establish a valid reciprocity between the internal data processing code from the receiving agent and the external data as received from the producing agent, without extending the semantic monolith from either agents. We show that loosely coupled semantics, semantic alignments and a shared ontological commitment of the applied modelling language can be considered the cornerstone to achieve sIOP. The supporting principles are: (i) assume responsibility for the semantics of one's data, (ii) maintain an explicit ontological commitment, (iii) abstract semantics from the communication syntax, (iv) align the internal and external semantic meaning of the exchanged data, and (v) encapsulate how agents exchange semantic meaning. This results in a loosely coupled semantics that is re-usable for every interoperating peer agent, even those that are not anticipated for during the agent's design. The resulting ISO-42010 Architecture Viewpoint and View on sIOP, including a semantic mediation capability, represents a pattern to consolidate sIOP in contemporary architectural paradigms.*

Conclusions: *The major shortcomings in architectural paradigms to account for an access-and-play sIOP are their negligence of a separation of concerns between the semantic representation and data communication syntax at the one hand and human-authored alignments and the automated mediation process at the other, and establishing the conditions in support of in advance. By their explicit inclusion, we show that loosely coupled semantics can be consolidated in contemporary architectural paradigms, stimulating access-and-play sIOP.*

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 without a prior explicitly established and therefore time-consuming understanding, agility in business collaboration is hampered in all domains. For instance, consider the following (allegedly real) example of an interoperability failure.

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

> 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.[1]

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 Chapter 8 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. "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; in Scheider 2012).

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 that originated outside the monolith impossible, unless again a time consuming (months) semantic adoption process is applied. Moreover, these semantic conventions consider semantic heterogeneity a bug instead of a feature necessary to achieve semantic accuracy. Nevertheless, this

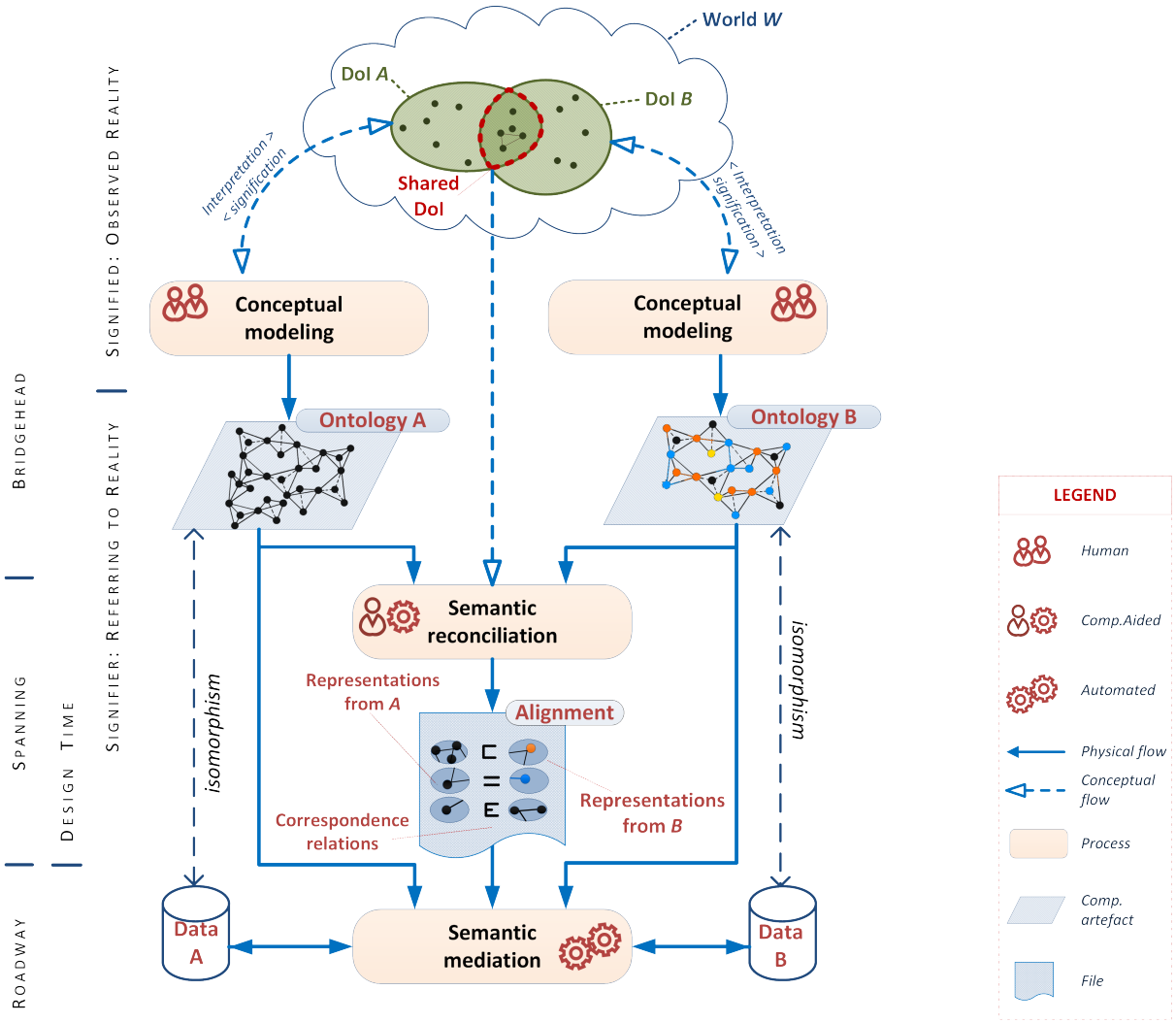
[1] Source: http://catless.ncl.ac.uk/Risks/14.57.html#subj1, accessed May 20, 2018

conventions-based approach towards sIOP is considered accepted folklore, even state of the art 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 collaboration, we consider this approach "too little, too late". Some form of automation is required to resolve these issues, and we place formal semantics at its core.

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 standardised solution patterns that 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 only then we can take down the tight coupling between semantics and the syntax of the shared data scheme. 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 in due time with data not anticipated for during software design, and 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 aligments) resting on them to structurally (semantically) support the interoperability bridge and its traffic, and with a *roadway* (data mediation) enabling the crossing of the (data) 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. This has been depicted in Figure 1.1.

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 disciplines of semiotics, philosophy, modelling and mathematics we elaborate in (Brandt et al. 2019) how to achieve a semantic bridgehead. We formulate the principle of assuming responsibility on the semantics on data, and conclude what preparations about semantics are required for an agent before being able to engage in semantic interoperability (Chapter 2);
- *sIOP concerns (spanning)*: Since computers remain incapable of true understanding, 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 formal semantics to a task that suffices to achieve sIOP, viz. authoring semantic alignments only (Chapter 3);
- *Mediation concerns (roadway)*: We determine the demands for a generic component that allows for communication with the peer agent in one's native vocabulary only, by considering both ontological models and the alignment. Such approach applies the principle *connectivity without dependency* at the semantic level. This consolidates the agent's potential to collaborate to any unforeseen applications without the need to adopt external semantic definitions, and remain scalable in the process (Chapter 4);
- *Evaluation of semantic principles*: In order to consistently address the above concerns, their founding architectural principles have been derived. It is a matter of architectural hygiene to evaluate how these principles .... (**??**);
- *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



(a) (b)

Figure 1.1: Conceptual overview of the relationships in sIOP between the bridgehead (conceptual modelling), its spanning (semantic reconciliation) and roadway (semantic mediation), (a), and a legend explaining the used constructs (b).

considered to consolidate sIOP for contemporary architectural paradigms (Chapter 6).

Based on these contributions we conclude that there is a lot to gain towards access-and-play sIOP, despite the fundamental impediments in ICT to create an automated genuine understanding. By separating syntactical from semantic concerns and turning semiotic fundamentals into architectural principles, loosely coupled formal semantics emerges and can be consolidated in contemporary architectural paradigms. From that position it is only a small step towards embedding standard semantic services into the communication infrastructure. We first describe some background on the semantic foundations in the following section.

Chapter 2

Bridgehead: Semantics

Despite the precise meaning of the term 'semantics' in semantic interoperability, it is clear that sIOP encompasses a communication between at least two actors. This brings a natural responsibility for both actors in the communication, described by (Grice 1975) as the particular purpose of communication, viz. to serve:

1. Quantity: Make your contributions as informative as is required (for the current purpose of the exchange), and not more than is required;
2. Quality: Do not say what you believe to be false, or for which you lack evidence;
3. Relation: Be relevant (to the immediate needs);
4. Manner: Avoid obscurity of expression, ambiguity, and be brief and orderly.

This leads to the definition of a design principle to its effect, applying the normative notation from (Greefhorst and Proper 2011):

Design Principle 2.1 (The responsibility for the semantic meaning of data lays with the source)
When it is reasonable to expect that the software agent will be engaged in collaboration or otherwise will interoperate with (an)other software agent(s), it is the responsibility of the software architect to serve the quantity, relation and manner of the potential interoperability by specifying the semantics of the data in advance.

Type of information: *business, data*
Quality attributes: *semantics, semantic interoperability, usability, efficiency*
Rationale:

1. *Data represent the state of affairs of some part of the world, viewed from a particular perspective of use. Such view is just one particular perspective out of many equally legitimate ones;*
2. *Semantic heterogeneity, a direct consequence of the equally legitimate perspectives on reality, should not be considered a bug to resolve, but a feature to preserve and nurture in order to maximise semantic accuracy and relevancy;*
3. *Accepting semantic heterogeneity implies the probable uniqueness of the agents view on reality;*
4. *Computers are not capable of genuine understanding, hence cannot establish semantics from data and thus require the human-in-the-loop for that;*
5. *The responsibility for formulating the semantics that are expressed by the data can only lay by the software architect that has taken the particular perspective on reality when carving out the entities of interest to the software application;*
6. *On specifying semantics, Grice's maxims on communication, and particularly on serving the quantity, relation and manner of communication, represent the natural constraints to respect;*

7. *Without adherence to this principle, the meaning of the data expressed by the software agent can be considered flawed, inaccurate, incomplete or otherwise insufficient in its support for semantic interoperability.*

Implications:

1. *The specification of the data semantics is only dependent on the agent's own perspective on the application domain, and can therefore be fulfilled before any interoperability with communication peers;*
2. *No matter the number of different communication peers, the software agent needs to specify the semantics of its data elements only once;*
3. *By providing an explicit semantic specification of the data, an agent facilitates other components and agents to connect to it and, consequently, grounds its semantic interoperability with them unequivocally.* ◇

We argued in (Brandt et al. 2019) that since software is incapable of genuine understanding, semantics cannot exist in software. Nevertheless, the software agent acts as transport medium for semantics: for single-user software as the medium that transports the semantics as it was intended by the software engineer to the semantics as it is experienced by the end user at the human-machine interface; for multi-user software as the transport medium for the semantics as intended by one end user at the time of data insertion, to the semantics as experienced by another end user when retrieving the processed data. To act as valid transport medium for semantics, we further stated that the reciprocity between code and data does manifest itself as software semantics. This essential disposition discerns in semantics its *semantic meaning*, i.e., what is said and carried by data, and the *pragmatic meaning*, i.e., to connect with our frame of reference and carried by code. The latter implements comprehension as an inference process that starts from a set of premises and results in a set of conclusions that are warranted by them (Grice 1975). We explained that by observing that data and code are always tightly coupled and since their reciprocity emerges as software behaviour, software malfunction originates (amongst others) from a broken reciprocity, i.e., inconsistencies between data and code. Consequently, when the data and code are representations of the things and laws in the application domain and, hence, represents semantic meaning and pragmatic meaning, their reciprocity represents the degree with which the collective outcome of processing all potential data refers to the intended states of affairs in reality. Any incoherent reciprocity equates to unfaithfulness: semantics that are considered invalid in the application domain.

Despite the quality with which the data and the code are developed individually, we can maximise semantic validity by maximising their reciprocity, viz. demanding maximal coherence between code and data. We have called this the *semantic coherence principle*. The consequence of demanding high coherence between the data and its processing code is in its inevitably emerging monolith, which we denoted as the Atomic Semantic Monolith (ASM): a semantic monolith, for it refers to the monolith's reciprocity between data and its processing code that describe the affairs in the application domain; Atomicity refers to the level of granularity at which the entity that is referred to by the data token is considered a non-dividable whole in the application domain. Where it is the objective of sIOP to address this monolithic nature of the ASM, as we do in the next section, it is the objective of semantics to maximise and maintain the coherence of the ASM, as elaborated in (ibid.).

Regarding the quality of the data and code models we reasoned that the data model should have a backward-looking role (in contrast to forward-looking) (Gonzalez-Perez and Henderson-Sellers 2007), present an ontological mode of modelling as opposed to a linguistic mode (Atkinson and Kühne 2003), and demand a strong type-mode (as opposed to a token-mode) that result in non-transitivity and use the kind of abstraction known as classification (Henderson-Sellers 2012). From those demands, we concluded that for representing semantics ontologies are best suited (Brandt et al. 2019). For example, the trueness of forward-looking models,

i.e., all 42010:2011 models, is established against their meta-models, while the trueness of backward-looking models, i.e., ontologies, is established through the interpretation in the conceptualisation of reality.

Moreover, we showed (ibid.) that the predominant purpose of a model is to describe reality by distinguishing the entities of interest. These distinctions can be modelled, but the (modelling) language itself is also used to convey distinctions. The distinctions that are already articulated by the elementary language constructs define the expressiveness of that language; the more distinctions the language elements can convey, the more differences can be represented by that language. The (fundamental) categories that the language elements can discern apply during modelling as a *commitment*, e.g., the language commits to the intuitive difference between the cup and the coffee that it holds. When the modelling language does not commit to such distinctions, the user of the language is forced to specify these distinctions in the model itself. This so-called *ontological commitment* of the language (Bricker 2016; Guarino et al. 1994,Guarino:1998wq) lays the foundations for the model and its data and, consequently, for the code to process the data. Interoperating peer agents that apply different ontological commitments will therefore show major differences in the construction and internals of their respective ASM's. This observation leads to the following design principle:

Design Principle 2.2 (Maintain an explicit ontological commitment)
The language constructs that are used to formulate a model always represent an ontological commitment, explicitly or implicitly.

Type of information: *business, application, data*
Quality attributes: *semantics, semantic interoperability, reliability*
Rationale:

1. *The particular reciprocity that emerges in the ASM is influenced by the ontological commitment of the modelling language;*
2. *The purpose of sIOP is to re-establish a coherent reciprocity between the external semantic meaning and the internal pragmatic meaning;*
3. *Incompatibility between the ontological commitments of both interoperating agents creates a sIOP concern on the modelling language level;*
4. *sIOP cannot be established without having addressed this language concern;*
5. *This language concern and its related resolution is independent from any particular sIOP case;*
6. *By maintaining an explicit ontological commitment, its incompatibility with other ontological commitments can be addressed in a generic manner.*

Implications:

1. *Since the choice for a specific ontological commitment is only dependent on its applicability to the agent's semantics, its specification therefore is independent from any specific interoperability case;*
2. *No matter the number of different communication peers, the software agent needs to specify its ontological commitment only once;*
3. *By specifying its ontological commitment explicitly, an agent enables the emergence of a standard and related infrastructural components to address this concern.* ◇

Based on the principles and arguments in this section, we defend that software agents that might engage in interoperability should provide for a semantic bridgehead in the form of a domain ontology and a foundational ontology; the latter to explicate the ontological commitment and the former to specify the semantics of the data. Such ontology provides the ability to connect to the semantics of the agent in a computational manner, consolidating the semantic concerns for semantic interoperability.

Chapter 3

Spanning: semantic interoperability

3.1 Objectives of semantic interoperability

Semantic interoperability is about two software agents that share a particular reality in their domains of application, and exchange data that represent a certain state of affairs from that shared reality. Despite the (different) reasons that both agents might have for sharing the data, the only demand that is put on the exchange is to serve what was coined by Grice as the communication's quality: "Do not say what you believe to be false, or for which you lack evidence". Subsequent to the exchange the data will be processed by the receiving agent and it stands to reason that understanding the data precedes their faithful use. In conclusion, semantic interoperability discloses the capability between two software agents to faithfully use exchanged data that accurately represent the state of affairs about a particular shared reality.

We have seen that software semantics is necessarily reduced to the reciprocity that exists between data and code in the so-called atomic semantic monolith. Such ASM guarantees the coherence between the data and their processing code. Unfortunately, when communicating data they are necessarily separated from the ASM they belong to. (Why it is useless to exchange the complete semantic monolith in order to establish sIOP, is left as an exercises to the reader.[1]) The consequence of data exchange on the semantic meaning (data), therefore, is twofold: it loses its coherence with its original pragmatic meaning (data processing code), and, a new reciprocity with the pragmatic meaning belonging to the receiving agent emerges. Unless it can be guaranteed that this new emerging reciprocity is as coherent as it needs to be, semantic interoperability cannot emerge from the data exchange and phantom semantics will emerge in stead. From this we conclude that the main task about establishing sIOP is to re-establish coherence between the external semantic meaning and the internal pragmatic meaning. Note that the resulting semantic monolith of the receiving agent will be different from the original semantic monolith, although they share the same semantic meaning. For example, by exchanging a heartbeat both agents share the a semantic meaning about the number of beats per second, however the pragmatic meaning can vary between an indication of health for an health-care application or an indication of performance potential in a sports application.

The **prime objective** from the perspective of the receiving agent is to assure that the reciprocity between data and code remains truthful to the state of affairs in reality. This relates for both the external data and the data that can be inferred from them. The two possible approaches are, (i) to modify the pragmatic meaning such that it can operate in a valid way on the external semantic meaning, or (ii) to modify the semantic meaning such that it can be operated on in valid way by the existing pragmatic meaning. The first approach clearly breaks one of the fundamental principles of software engineering, *low coupling, high cohesion* (e.g., Hitz and Montazeri 1995), by allowing external definitions to influence internal workings. The second approach only adapts that what was already external to the receiver, and thereby doesn't breach the integrity of its own software. By pursuing the second approach we also maintain a **second objective** for sIOP: ensure

brandtp, 21-10-2019 put otherwise: to adopt an externally defined perspective on reality in order to process its data

[1] Answer: Coupling the semantic monolith would result in both agents to perform the exact same functionality with regards to the data.

that the ASM's from both agents remain independent from each other, viz. establishing a semantical loose coupling between both agents. Such loose coupling does not require one single homogeneous view on reality, which aligns neatly with the **third objective** for sIOP to allow for semantic heterogeneity: distinct agents will probably maintain alternative but equally legitimate points of view on reality, implying that semantic heterogeneity is a feature to preserve necessarily. The **fourth objective** of sIOP is to strive for access-and-play sIOP: ideally, sIOP between agents can be achieved instantaneously, also for unforeseen collaborations. This objective is very hard to achieve because when we accept that software is incapable of genuine understanding, and when we accept that correct use of data is to be preceded by its understanding, a human-in-the-loop to provide that understanding becomes a necessary condition for sIOP. The **fifth objective** of sIOP is to allow for semantic evolution and, consequently, the maintainability of sIOP. Finally, we consider that semantic heterogeneity brings about an issue of scalability, since semantics won't be a centrally coordinated anymore; in stead, semantic definitions will be distributed all over the place. We therefore include as **sixth objective** of sIOP to allow for scalable semantics.

We conclude that from these six objectives, only the first (re-establish reciprocity) and fourth (access-and-play) are concerned with genuine understanding of semantics, leaving the others as engineering challenges. Our focus in the subsequent section is only on achieving the re-establishing reciprocity and access-and-play objectives. We address the latter four as evaluation of the principles that have been introduced to achieve the two core objectives.

3.2 Explicit sIOP by alignment

We maintain the position that computers lack the capability of genuine understanding. We subsequently defend that with the current state of the art in AI the human-in-the-loop remains a necessary element in sIOP in order to cater for the understanding of the semantic differences between the interoperating agents. In its pure sense, an access-and-play capability can therefore not be established. However, current solutions on semantic standards solidify the understanding in the syntax of the data. In this way, semantics are carried by a data schema that is primarily designed to serialise data and to support data transfer by message construction and exchange. We consider this a significant neglect of the principle on separation of concerns, conflating the semantic interoperability concerns with the data communication concerns. The consequence of conflating these concerns is that source code which should concerned primarily with message construction, parsing, storage, and other data communication related tasks, becomes dependent on how semantics influence the syntax. In a message-oriented paradigm, for instance, any difference in structure in order to reflect the local perspective on semantic structure will have a significant impact on how to (de)compose the message. And any new data source to connect to will proliferate into a new software release. We thus observe that the current approach to data understanding results in an architecture which imposes a significant complication on interoperability (and other -ilities as well), impeding access-and-play. And despite the current limitations of AI-software to genuinely understand, a significant gain towards the software agent's access-and-play capabilities can be achieved by untangling the syntax and semantics through separation of the sIOP concerns from the data communication concerns. We propose the following design principle to its effect:

Design Principle 3.1 (Abstract semantics from communication syntax)
When a software agent engages in interoperation with (an)other software agent(s), resolve their semantic differences independently from the syntax of the exchanged data.

Type of information: *data, technology*
Quality attributes: *semantic interoperability, portability, maintainability, efficiency, usability (reuse), reliabil-*

ity, functionality
Rationale:

1. *Data schemata are defined to support the (de)serialisation processes that consolidate the data communications concern;*
2. *Neglecting the principle of separation of concerns solidifies dependency between otherwise disjoint concerns, here the semantic level and the syntactic level of data communication;*
3. *Access-and-play capabilities are supported by assuring minimal impact on software code when introducing semantic modifications;*
4. *Minimising impact on software code that is concerned with data communication is realised by abstracting semantics away from the data schemata.*

Implications:

1. *Separation of concerns has a strong positive effect on software quality, including but not limited to sIOP;*
2. *Removing any dependency between semantics and data syntax enables to support multiple communication paradigms without the need to modify the semantic abstraction;*
3. *Similarly, modifications in the semantic representation, or supporting multiple semantic representations become possible without the need to modify the communication layer;*
4. *Align semantics, not data schemata: Semantic reconciliation is applied at a higher conceptual level and abstracts away from data communication schemata;*
5. *Heterogeneous semantics from multiple data sources are more easily supported;*
6. *Semantic alignments imply the need for a mediation capability between the semantic representations of the communicating agents.* ◇

This principle is in clear contrast with principle A.20 , Data that are exchanged adhere to a canonical data model, as determined in the Principles Catalogue from (Greefhorst and Proper 2011). Indeed principle A.20 reflects the current practises to achieve interoperability. It is not necessarily wrong, since following it results in achieving interoperability, as justified by the many if not all interoperable systems that exist today. However, its application impedes access-and-play interoperability since such capability require more specification and less implementation, more generic infrastructural solutions based on meta-standards as opposed to local solutions that rely on data standards.

brandtp, 18-11-2019 Move this paragraph to section Related Work?

Current sIOP practises already require humans-in-the-loop to reconcile the semantic differences that occur. Often, the subject of reconciliation is the differences in data schemata, and the result of the reconciliation is laid down as a canonical data model. By applying semantic reconciliation on the conceptual level, the dependency on the (data) syntax, and vice-versa, is minimised. Moreover, by representing the result of the reconciliation as an alignment (between ontologies) as opposed to a canonical semantic model (core ontology), the influence of the peer agent's semantics on one's own semantics is minimised as well. An alignment, thus, functions as an interface that enforces loosely coupled semantics by enabling semantic transparency between communicating peers. Reducing the human-in-the-loop to author an alignment only, (i) accelerates the deployment of sIOP by removing all human effort that is concerned with implementation activities, and (ii) decouples the sIOP scope to bilateral alignments only. This process has been depicted in Figure 3.1.

From the perspective of the receiving agent the first objective of sIOP is to guarantee that the reciprocity between the external semantic meaning and (our) internal pragmatic meaning remains faithful to reality. We've determined that we can only convert the external semantic meaning, and we need to do it such that

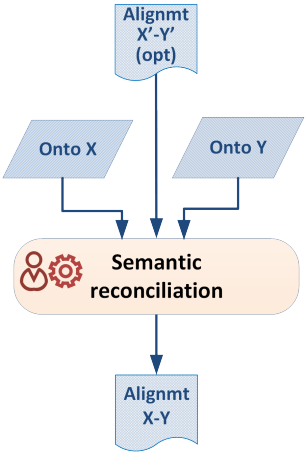


Figure 3.1: Semantic reconciliation results in an alignment between the semantic representations of two ontologies. We defend that semantic reconciliation is a computer-aided but ultimately human-authored task.

the result becomes coherent with the internal semantic meaning. We can assume that the quality of our own agent software is such that the internal semantic meaning is in coherence with the internal pragmatic meaning. Contrarily, we cannot assume that any extension of the internal semantic meaning, no matter how small, will remain in coherence with the internal pragmatic meaning. Founded on this essential disposition we conclude that aligning the external semantic meaning with the internal semantic meaning such that the former does not overlap the latter, is the only approach that can guarantee sIOP. We reflect this with the following principle:

Design Principle 3.2 (Align the internal and external semantic meaning of the exchanged data) *When a software agent engages in interoperation with (an)other software agent(s), establish for the exchanged data a maximal coherence between external semantic meaning and internal pragmatic meaning by formalising the alignment between the external and internal semantic meaning.*

Type of information: *application, data*
Quality attributes: *semantics, semantic interoperability*
Rationale:

1. *On processing external data, semantics manifest as the reciprocity between data and processing code;*
2. *Data are considered isomorphic to the semantic meaning as specified by their source agent;*
3. *Formalising a correspondence relation between the semantic meanings of collaborating peers connects the external semantic meaning with the internal pragmatic meaning;*
4. *Assuring that the internal semantic meaning encompasses the external semantic meaning, or that the consequences of the latter extending the former are insignificant, assures the semantic validity if the correspondence relation.*

Implications:

1. *The conversion from external to internal semantic meaning is specified by a correspondence;*
2. *The collection of all correspondences specify the semantic alignment that holds between a pair of interoperating agents;*
3. *Software agents that are unable to align their semantic meaning with the external semantic meaning cannot engage in sIOP without introducing phantom semantics, with unforeseen consequences in their data processing.* ◇

A *correspondence* specifies as accurately as possible the semantic difference (out of those listed in Appendix A) that exists between a pair of related concepts, i.e., it aligns between the semantic meanings of interoperating agents. By exhaustively addressing all semantic differences that exist between both agents, the set of correspondences collectively specify the *alignment* that holds between two agents. The purpose of the 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', and to capture their potential difference as a relation. To that end we differentiate between two categories of semantic differences:

1. *Conceptual differences*: variations that can be specified as logical relation between (constructions of) concepts from both ontologies, e.g., naming conventions or structural engineering variations;
2. *Value differences*: variations in conventions on how to represent values with or without magnitudes, e.g., differences in value scales, units of measure or nominal properties.

The language used to specify the correspondences must be expressive enough to identify the atomic elements of the ontologies, to combine them into logical combinations as well as to formulate the relationship that holds between them. In (Euzenat et al. 2007; Scharffe et al. 2011), an investigation has been reported towards the requirements for such an alignment language, summarised as follows. A *correspondence* denotes a single particular inter-ontological relation, prescribed, and assumed to represent a semantically valid relation between both concepts, as:

$$\mu = \langle e, e', \theta \rangle$$

with:

- $\theta \in \{=, \sqsubset, \sqsupset, \perp, \between\}$ specifying the *correspondence relation* that holds between entity constructions from the source, e, and the target, e'. The basic correspondence relations denote $=$: semantic equivalence, $\sqsubset$: subsumption of, $\sqsupset$: subsumes, $\perp$: disjointness, and $\between$: overlap. Although more relations can be required to include for a particular use case, such does not invalidate the general principle. Further note the correspondence relation is a directed relationship.
- The source and target *entity constructions*, e, are build on the atomic elements of the ontology language. An entity construction connects concepts by applying:
  - conceptual connectors:
    - logical connectors AND, OR, and NOT;
    - a path connector as a sequence of zero or more Object Relations, R, optionally ending with an Object Property P, summarised as follows: $R^*[P]$;
    - property construction operators (inverse, composition, reflexive, transitive and symmetric closures);
    - constraints (through domain, range, cardinality and value restrictions);
  - value functions:
    - mathematical calculations operating on one or more values having a magnitude for, e.g., unit conversion;
    - transcriptors operating on one or more nominal values without magnitude, e.g., ISO two-letter country code, or blood type.

Without the conceptual connectors it is only possible to address a single concept or individual as defined by the ontology, representing an aggregation level that is relevant for the software agent but might not be relevant in terms of the interoperating agent, and hence, for their mutual sIOP. By application of conceptual connectors the architecture gains the capability to address a specific compound of individuals in either the source or target ontology that relate to the semantic difference at hand. Similarly, with the application of value functions the architecture gains the capability to specify transformations between conventions on value representations and nominal properties.

Chapter 4

Roadway: Mediation

The mediation pattern has already been described in (Gamma et al. 1994), albeit in the context of object-orientation as opposed to sIOP. It is described as "an object that encapsulates how a set of objects interact", and it promotes loose coupling "by keeping objects from referring to each other explicitly" and by enabling you to "vary their interaction independently". In this way, the mediator turns a many-to-many interaction into a many-to-one interaction, each of which is easier to understand, maintain and evolve. The fundamental idea behind the pattern, viz. trading the complexity of the interactions with the complexity in the mediator, can also be applied on a semantic level, and we formulate the following principle to its effect:

Design Principle 4.1 (Encapsulate how agents exchange semantic meaning)
When software agents engage in interoperation, encapsulate how the representation of their semantic meaning should be transcribed without inducing phantom semantics.

Type of information: *business, data*
Quality attributes: *semantic interoperability*
Rationale:

1. *The semantic meaning is codified in (onto)logical representations;*
2. *Keeping agents from referring to each others representation therefore requires transcription between representations;*
3. *A solution where each agent needs to implement one transcription component between its own representation and each of its interoperating peer, increases complexity;*
4. *Encapsulating the transcription into an alignment-based intermediary component results in less communication complexity and relieves the agents from development and maintenance of local wrappers;*

Implications:

1. *A mediator creates representational transparency between communicating agents, keeping agents from using each others representations;*
2. *This enables independent development of the individual agent's semantic meaning;*
3. *The need to enforce a canonical semantic representation, viz. semantic homogeneity, expires, allowing semantic heterogeneity to become the norm;*
4. *Each agent can reuse its semantic bridgehead in any other interoperation;*
5. *Data transcription logic can become a generic service provided by the communication infrastructure;*
6. *Each agent can communicate with any other agent in its own native semantic representation.* ◇

In this reading a mediator encapsulates how a pair of agents represent their semantic meaning and provides for a generic transcription logic to mediate between native semantic representations. However, one paramount issue must be resolved by the transcription logic of the mediator, as follows.

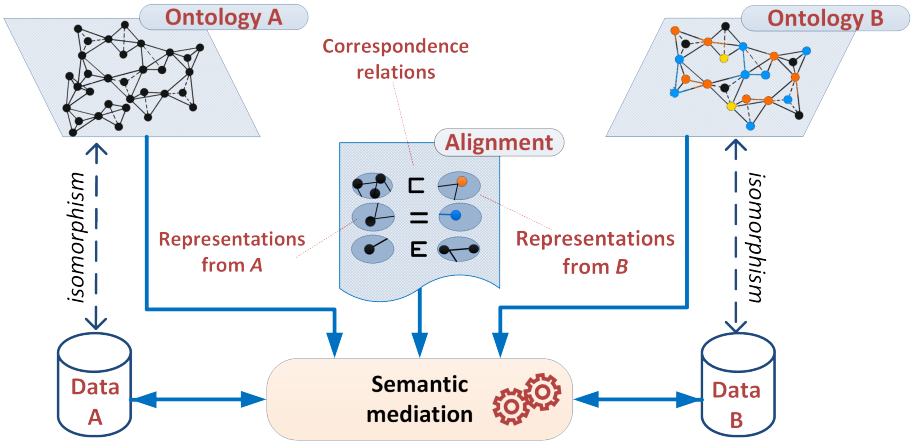


Figure 4.1: Semantic mediation encapsulates how agents exchange semantic meaning. It implements a generic transcription service between the particular representations of data between both agents. It depends only on the representations of the semantic meaning from both agents (ontologies) and the alignment that holds between them.

We have seen in the previous sections that a correspondence assures the semantic validity between the different semantic meanings of both agents. It can do so by token of the broad correspondence relations that can be specified, both for the conceptual and the value differences that may exist between them. Unfortunately, this is in shear contrast with the fact that a transcription can only replace one term for another, which basically implies that an equivalence relation holds between both terms. As can be seen from the correspondence relation, this is not necessarily true. In (Brandt et al. 2018), we make a distinction between a naive transcription, which ignores this inconsistency, and a semantic valid transcription that can establish under what conditions this inconsistency does not produce invalid semantics. We show that these conditions rely on logical constructs only and, consequently, are independent on the actual ontologies that apply. We have formulated these logical constructs as rules that generically apply for any transcription. Unfortunately, a set of conditions remain for which a semantic valid transcription cannot be guaranteed, either due to an incomplete alignment or to (onto)logical incorrect correspondences. It follows that a semantic mediation service must provide for options about the resolution for these remaining issues. We suggest at least the following options: Firstly, as run-time options we consider (i) a means to fall-back to the application of naive transcriptions and continue the data exchange, or (ii) to raise a Transcription Error and refrain from data exchange. Both represent necessary service implementations that might satisfy scenarios that have either less or more stringent semantic demands. Secondly, as design-time option, it serves as a necessary tool to generate an exhaustive list of shortcomings of the transcription as a result of the current ontologies and alignment. Such list will provide the semantics engineer with sufficient information to adapt the alignment, or introduce modifications to one or both of the semantic bridgeheads in order to remove transcription issues that are considered vital to the business. Finally, the third option is to try to resolve the transcription issue by starting a dialogue-based semantic reconciliation process in run-time with the aim to improve upon the shortcomings of the current alignment. Despite our insistence on the need for a human to author the genuine understanding, it might still be worthwhile to have the system meticulously, consistently and methodologically negotiate logical and ontological constructs in order to find additional correspondences (Diggelen 2007).

Chapter 5

Evaluation of sIOP principles

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. We have introduced some new principles to its support in the previous sections, and now evaluate their consequences on objectives two (loose coupling at semantic level), three (allow for semantic heterogeneity), five (support to semantic evolution) and six (scalable semantics).

5.1 Loosely coupled semantics

Consolidating sIOP demands the emergence of *loosely coupled semantics*. As analogy, consider a vehicle with its cargo. Logistics rely on external transfer services that allow the cargo to be transported over different vehicles, from a truck into an aircraft into ship into a truck again, without ruining the cargo. This requires the cargo to be firmly connected to the vehicle, but at the same time to be completely independent from any particular vehicle in order to complete the transport. "Connected but as independent as possible", also known as loosely coupled, is a need for sIOP as well. When software agents interact they exchange meaning. Loosely coupled semantics implies that the semantic meaning (the cargo) remains as independent from the representation of semantics (the vehicle) as possible. Similarly to logistics, sIOP should rely on infrastructural services that can transcribe the semantic representation from its native form into a foreign form without invalidating the semantics that it bears. Loose coupling is known as a strong characteristic which brings many advantages. This applies to its semantic variety as well: agents can now communicate in their own native representations without the need to learn or integrate foreign representations; define semantics once and achieve sIOP many times with many different peers; development of the agents' semantic representations can be locally isolated to fit their particular domain and application; and it enables local re-use which on its turn increases its quality.

Loose coupling in the classical sense is realised through the principles of separation of concerns and transparency. In its original reading separation of concerns turns complex functionalities into simple, atomic and complementary functional capabilities. In a semantic reading separation of concerns is not about maintaining complementary semantics; in fact, the domain of interest of the agents are required to overlap, since interoperation would be completely useless otherwise. Instead, semantic separation of concerns refer to enforce an explicit division between syntax and semantics, as discussed in Section 3.2. Additionally, it refers to keeping each other's representations of the semantic meaning strictly separated. We described how this can be achieved in Chapter 4. The classical results of applying the principle is the emergence of unique functions that are implemented only once and used many times. In its semantic application this results in every software agent to maintain its own semantics. Collectively, all agents make that semantics become distributed all over the place which seems counterintuitive or even plain wrong. We come back to that intuition when we talk about scalability, Section 5.4, but we can already see that this is an indirect

consequence of the demand to enable sIOP with agents that were not anticipated for during software design; this, on its turn, requires that independent development of semantics shall be possible without disabling sIOP.

The classical reading on transparency separates access to the unique functions from the particular design and implementation of the functions. Remaining agnostic to *how* its function are achieved makes it possible to communicate with minimal mutual dependency. Semantic transparency remains agnostic to how semantics are *represented*, which makes it possible to communicate with minimal syntactic dependency and without prior mutual agreements on semantic representation. In its classical reading, transparency requires the introduction of standards in the components' API. Semantic transparency, contradictory, requires the total absence of any standard on representation. In stead, semantic transparency, too, requires to separate semantics from syntax. Furthermore, a need emerges for a semantic oracle that knows how to align distinct representations and to translate between them subsequently. The latter has already been discussed in Chapter 4 while the former is directly related to the human-in-the-loop as authoring authority as discussed in Chapter 3.

From the above discussion we draw the conclusion that all demands necessary to allow for semantic separation of concerns and semantic transparency are met by the sIOP principles **??**. Therefore, loosely coupled semantics should emerge between agents that comply to these principles.

5.2 Semantic heterogeneity

We have already determined that the principle to align semantics as opposed to data schemata, Design Principle 3.1, breaks the conflation of semantics and syntax, enabling to consider semantics on its own terms[1]. Abstaining from a canonical model by introducing a semantic alignment between pairs of semantic meanings, Design Principle 3.2 introduces the capability for each agent to develop its semantic representation in a way that fits its local perspective optimally. By also encapsulating the particular means to provide valid semantic transcriptions only and refrain from naive transcriptions between communicating agents, Design Principle 4.1, the necessary elements to support semantic heterogeneity are present.

5.3 Semantic evolution

Consider an agent who's local semantics are in demand for a change. Assume that the agent has modified its internal pragmatic meaning to accommodate the evolved semantics. This implies a change in what we called the semantic bridgehead, which acts as semantic interface to any sIOP concerns. The semantic change can be considered an altered or additional difference with the semantic bridgehead from interoperating peer agents. In order to not destroy the sIOP that had been established before, it is necessary to reflect the new difference as modifications to the alignments that apply. By having performed the necessary modifications, and by relying on the semantic validity of the own ontology, the mediator is now capable of transcribing the data in accordance to the evolved semantics.

The consequence of allowing semantic heterogeneity is therefore a sufficient condition to also enable local semantic evolution and remaining semantic interoperable with existing counterparts in the process.

[1] Pun not intended

5.4 Scalable sIOP

Many definitions exist to constrain the semantics of *scalability*, academic (e.g., Neuman 1994) and popular[http://www.linfo.org/scalable.html, accessed Nov. 2019][https://en.wikipedia.org/wiki/Scalability, accessed Nov. 2019] alike. Our summary refers to increasing the demand that is placed on a system, and/or adding resources to a system, without experiencing loss of performance or increase in management to an extent that defeats its primary objective. We consider scalable sIOP as the capability to allow for increase in number of communicating agents as well as in the level of semantic heterogeneity without degrading the agent's communication performance or its ability to manage and control the semantic differences with its interoperating peers.

If we consider the agent's communicating performance degradation, we argue that since complexity of the connections have been traded with the complexity in the mediator (Chapter 4), the agent will only experience communication degradation when the mediator experiences transcription latencies that exceed communication parameters. In other words, the performance bottleneck is with the mediator, not with the agent. Transcription latency will result from complexity in the transcription algorithm, or from the number of transcription request that exceeds the capacity of a single mediator. In case of the latter, no particular mechanism in a mediator impedes horizontal scaling to increase the collective processing capacity to match the transcription demands. In case of the former . . . .

brandtp, 20-11-2019 Address the computational demands for a single transcription. Or address it in [@Brandt2018b]ref to it here.

Regarding the ability to manage and control the semantic differences with all interoperating peers, the root cause for potential scalability issues are laid in the need to establish an alignment with each peer agent an agent engages in semantic interoperation with. This might become impractical due to our insistence on the need for a human-in-the-loop to author the alignment. We discern different solutions for different semantic topologies:

i. Star alignments (core domain ontology, aligned to local ontologies) for relative stable and homogeneous domain semantics
   - Good: semantic governance remains controllable independent from the number of actors;
   - Bad: very big semantic monolith, hence, low agility in dynamic environments. Moreover, the more actors involved, the higher the need for semantic compromises, and the lower the overall semantic accuracy.

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: semantic governance may become an issue to the level where it becomes impractical.

iii. Mix-n-Match (coarse-grained star-alignment with intricate bilateral alignments as specialisations to the core domain ontology) 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.

iv. Daisy-chained alignments (when A is aligned to B, and B is aligned to C, A and C are indirectly aligned as well)
   - Good: self-organised alignments emerge, and an instantaneous access-and-play becomes possible;
   - Bad: No guarantees can be given on the completeness of the indirect alignment. Furthermore, more intermediate alignments will increase the chance of impossible end-to-end transcriptions that would not occur with a direct alignment.

In conclusion, scalable sIOP can be guaranteed when considering the communication performance. With respect to the ability of a single agent to manage and control the number of alignments with increasing number of interoperating agents, several options exist to support scalable sIOP although no guarantees can be given.

Chapter 6

ISO42010 viewpoint on sIOP

Chapter 7

Related work

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. Aßmann, 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
10. Götz, S., Beckel, C., Heer, T., & Wehrle, K. (2008). ADAPT: A semantics-oriented protocol architecture. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5343 LNCS, 287–292. https://doi.org/10.1007/978-3-540-92157-8-27
11. Zhou, L., Cheatham, M., Krisnadhi, A., & Hitzler, P. (2018). A Complex Alignment Benchmark: GeoLink Dataset. In D. Vrandecic, K. Bontcheva, M. C. Suárez-Figueroa, V. Presutti, I. Celino, M. Sabou, ... E. Simperl (Eds.), The 17th International Semantic Web Conference, ISWC 2018 (pp. 273–288). Monterey, CA, USA: Springer Nature Switzerland AG. https://doi.org/10.1007/978-3-030-00668-6_17
12. Kühne, T. (2018). Unifying nominal and structural typing. Software and Systems Modeling, 1–15. https://doi.org/10.1007/s10270-018-0660-y
13. Fahad, M., Moalla, N., & Bouras, A. (2012). Detection and resolution of semantic inconsistency and redundancy in an automatic ontology merging system. Journal of Intelligent Information Systems, 39(2), 535–557. https://doi.org/10.1007/s10844-012-0202-y
14. Renner, S. A., Scarano, J. G., & Rosenthal, A. S. (1996). Data interoperability: Standardization or Mediation. 1st IEEE Metadata Conference, 1–8.

15. Pagano, P., Candela, L., Castelli, D., & Paolucci, M. (2013). Data Interoperability. In Data Science Journal (Vol. 12, pp. GRDI19–GRDI25). https://doi.org/10.2481/dsj.GRDI-004
16. Wilkinson, M. D., Dumontier, M., e.a. (2016). Comment: The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data, 3, 1–9. https://doi.org/10.1038/sdata.2016.18

Chapter 8

Discussion & future work

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

Appendix - Documented sIOP failures

Insulin pens versus vials

In a follow-up article in the February 21, 2013 issue, the Institute for Safe Medication Practices (ISMP) warns that simply replacing insulin pens with insulin vials also has pitfalls. For one thing, insulin vials are labelled differently from most other medications. A label of "U-100" on a vial, for example, may be misunderstood to mean that there are 100 units of insulin in that vial, when in fact it means that the concentration of insulin in the vial is 100 units per mL; failure to recognize this fact could lead to a 10-fold overdose.

Source: (**???**)

Twin identification

Tweelingen zijn vaak op dezelfde dag geboren, in dezelfde stad en dragen vaak dezelfde achternaam. In heel veel systemen wordt de combinatie {achternaam, geboortedatum, geboorteplaats} gebruikt ter identificatie van een individu waarvan door de organisatie persoonsgegevens worden opgeslagen. Deze combinatie is dus niet in staat onderscheid te maken tussen de individuen van meerlingen, en vaak zijn de gegevens dan ook vervuild in de zin dat er feiten worden opgeslagen van alle individuen van de meerling. Zelfs bij het overlijden van één van hen zullen diens gegevens nog van toepassing blijven voor de andere individuen van de meerling. Dit leidt niet alleen tot grote ergernis, maar ook tot daadwerkelijk foutieve beslissingen.

Source:

Account deletion

SAP, hèt enterprise contract administration package voor vele organisaties, kent het begrip 'dienstverband'. Binnen TNO afdeling HR wordt SAP gebruikt, ondermeer om de werknemerscontracten vast te leggen. TNO afdeling IS beheert de Microsoft accounts van de werknemers. Accounts zijn afhankelijk van een dienstverband en dus bestaat er een koppeling tussen beide systemen. Echter, in de praktijk bleek dat één persoon meerdere dienstverbanden binnen TNO kon hebben. Op het moment dat een dienstverband afliep, werd het account van de betrokkene weggegooid, terwijl er nog een ander dienstverband liep. Afgezien van het feit dat TNO het SAP-deelpakket dat benodigd is voor het onderkennen van het begrip 'persoon' vanwege kostenreductie niet heeft aangeschaft, ligt de oorzaak van het probleem in oneigelijk gebruik van het concept 'dienstverband' door het identiek te stellen aan een 'persoon' binnen Microsoft accountbeheer.

Source: Interview Lex Beijk

Daylight saving time

At the end of April 1993, when Germany went on summer time, the computer clock of a German 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.

Source: http://catless.ncl.ac.uk/Risks/14.57.html#subj1

- Failure in sIOP
- Failure between : computer clock <-> process controller
- Fault: difference in value scale, since local time does not represent physical time but an arbitrary convention that has lost its characteristic of being continuous.

Sea level reference

Building a railway line from Austria to Germany in the last century, engineers started at both sides, meeting in the mountains. At the meeting point both ends of the railways had a difference in altitude of 0.8 m, because Germany referred to the North Sea level and Austria to the Mediterranean Sea level.

brandtp, 25-3-2015 11:03:29 REFERENTIE toevoegen

Source:

Y2K solution: let's insert month 13

In order to avoid the Y2K problem, the programmers had decided to give 1999 13 months instead of the usual 12. This way, they decided, they could buy time to resolve any unforeseen issue. Coincidently, just before the New Year's, they installed a time server in order to prevent there being a problem when the system time of the computers was compared with the normal time which is broadcast in Germany on a special frequency. They missed one little thing: leading zeros. The operating system delivered dates as 1:12:99 while the time server used 01:12:99 to indicate the first day of the month. Since the time server was not installed until after the 10th of December, this discrepancy wasn't noticed until midnight that the clock struck "month 13", or rather 1:13:99 instead of 01:13:99. This caused another program to cough about a date discrepancy. The people overseeing the systems subsequently tried to reboot the system [this at just past midnight on the first of January! -dww].

Het vervolg is interessant vanuit het perspectief van de consequenties, maar niet meer relevant voor sIOP: But it wouldn't reboot, because two of the computers were not configured properly. If they had ignored the error message, everything would have been okay! Now, while the system was trying to reboot, the folks in the fire trucks got antsy: They used to get acknowledgements for their location reports that they send regularly to indicate their whereabouts. But not this time, thus they just pushed all the buttons, to see if the machine was dead. This flooded the system additionally with repeated status messages. Finally, the network boards come into play. They had been named as the culprits in the first round of finger-pointing. They, too, had just been installed, and were misconfigured. They couldn't handle the traffic and began producing random errors. This confused the part of the system that keeps track of where the equipment currently is located, and then IT died.

Source: http://catless.ncl.ac.uk/Risks/20.93.html#subj6

- Failure in sIOP
- Failure between : local time server <-> local systems
- Fault: data discrepancy as result of representation error

Divide by zero - ready

The 'divide by zero' bug is a bug programmers have to cater for, and most software take this scenario into account. It was with some embarrassment, then, that the Aegis cruiser USS Yorktown suffered a complete failure of its propulsion system and was dead in the water for nearly 3 hours when a crew member typed a 0 (zero) into the on-board database management system which was then used in a division calculation. The software was installed as part of a wider operation to use computers to reduce the man power needed to run some ships. Fortunately, the ship was engaged in maneuvers at the time of the incident, rather than deployed in a combat environment, which could have had more severe consequences.

Source: http://listverse.com/2012/12/24/10-seriously-epic-computer-software-bugs/

- Failure in sIOP
- Failure between : database <-> calculation process
- Fault: Oneigenlijk gebruik van het begrip "0": daadwerkelijk nul versus NotApplicable

Space shuttle inversion

Much to the surprise of Mission Control, the space shuttle Discovery flew upside-down over Maui on 19 June 1985 during an attempted test of a Star-Wars-type laser-beam missile defence experiment. The shuttle's laser reflecting mirror was supposed to be oriented pointing downward at a spot 10,023 feet above sea level on Mona Kea; that number was supplied to the crew in units of feet, and was correctly fed into the onboard guidance system – which unfortunately was expecting units in nautical miles, not feet. Thus the mirror wound up being pointed (upward) to a spot 10,023 nautical miles above sea level. Not surprisingly, the experiment failed.

Source: (Neumann 1985)

Mars Climate Orbiter

For nine months, the Mars Climate Orbiter was speeding through space and speaking to NASA in metrics. But the engineers on the ground were replying in non-metric English. The mathematical mismatch that was not caught until after the $125 million spacecraft, a key part of NASA's Mars exploration program, was sent crashing too low and too fast into the Martian atmosphere. The craft has not been heard from since.

Sources: (Stephenson et al. 1999)

Endeavor mission

A loss was narrowly averted during the maiden flight of Endeavor (STS-49) on May 12, 1992 as the crew attempted to rendezvous with and repair an Intelsat satellite. The software routine used to calculate rendezvous

firings, called the Lambert Targeting Routine, did not converge on a solution due to a mismatch between the precision of the state vector variables, which describe the position and velocity of the Shuttle, and the limits used to bound the calculation. The state vector variables were double precision while the limit variables were single precision. The satellite rescue mission was nearly aborted, but a workaround was found that involved relaying an appropriate state vector value from the ground.

Source: (Leveson et al. 1993)

Ariane 501 disaster

It took the European Space Agency 10 years and $7 billion to produce Ariane 5, a giant rocket capable of hurling a pair of three-ton satellites into orbit with each launch and intended to give Europe overwhelming supremacy in the commercial space business.

All it took to explode that rocket less than a minute into its maiden voyage in June 4, 1996, scattering fiery rubble across the mangrove swamps of French Guiana, was a small computer program trying to stuff a 64-bit number into a 16-bit space.

This shutdown occurred 36.7 seconds after launch, when the guidance system's own computer tried to convert one piece of data – the sideways velocity of the rocket – from a 64-bit format to a 16-bit format. The number was too big, and an overflow error resulted. Steering was controlled by the on-board computer, which mistakenly thought the rocket needed a course change because of numbers coming from the inertial guidance system. That device uses gyroscopes and accelerometers to track motion. The numbers looked like flight data – bizarre and impossible flight data – but were actually a diagnostic error message. The guidance system had in fact shut down.

LT (Launch Time) = 9 o'clock. At 33 min. 59 sec. local time, the "launch window" was "caught" again and finally, the vehicle launched and was running in a normal mode until LT+37 seconds. In the following several seconds there was a dramatic deviation from the given missile trajectory that ended in an explosion. At LT+39 seconds, because of high aerodynamic load due to the "angle of attack" exceeding 20 degrees, the starting accelerators separated from its main stage, which triggered the missile Autodestruct System.

The change of the angle of attack happened because of a malfunction in the nozzle rotation of the solid accelerators, which was caused by a command from an on-board computer *based on the information from the active Navigation System (IRS 2)*. Some of this information was incorrect in principle: what has been *interpreted as flight details was actually diagnostic information* from the IRS 2 firmware.

The protection of all 7 (including BH) variables wasn't provided because the maximum workload for the IRS computer was declared as 80%. The developers had to look for ways to reduce unnecessary evaluation expenses, and they weakened the protection in that fragment where theoretically the accident could not happen. When it occurred, then the exception handling mechanism was activated, which turned out to be completely inadequate. This mechanism supposes three main steps:

- The information about the contingency should be transmitted via the bus to the onboard computer OBC. This indeed happened, but the OBC was not aware that this data was not flight data but status data instead.
- In parallel it was written – together with the whole context – to the reprogramming memory EEPROM (during the investigation it was possible to restore it and read the contents)
- The work of IRS processor should have been aborted.

The last action was a fatal one; it led to the accident despite the fact that the situation was quite normal (even though there was an exception generated due to unsecured overflow).

Sources: (**???**) referenced in (Leveson 2004); https://hownot2code.com/2016/09/02/a-space-error-370-million-for-an-integer-overflow/ http://www.around.com/ariane.html

- Failure in sIOP
- Failure between : Inertial guidance system <-> on-board computer
- Fault: Term overloading, as well as Semantic variety (inconsistent representation)

NORAD missile warning failure

On June 3 and 6, 1980, missile warning system failures occurred when a faulty component in the communications system began writing numbers into blank spaces in the missile warning messages sent out live to various command posts. The blank spaces during an attack indicate the number of attacking missiles and usually contained zeros, but in this case the erroneous numbers generated by the computer indicated a mass attack. These messages were used for communications line testing.

Source: http://archive.gao.gov/f0102/115265.pdf (p.13)

- Failure in sIOP
- Failure between : "faulty" component <-> missile warning system
- Fault: Semantic categorisation; test data versus operational data

Patriot Missile Bug February 25th, 1991

During Operation Desert Shield, the US military deployed the Patriot Missile System as a defence against aircraft and missiles – in this case Iraqi Al Hussein (SCUD) missiles. The tracking software for the Patriot missile uses the velocity of its target and the current time to predict where the target will be from one instant to another. Since various targets may travel at speeds of up to MACH 5, these calculations need to be very accurate. At the time, there was a bug in the targeting software – which meant that over time, the internal clock would 'drift' (much like any clock) further and further from accurate time the longer the system was left running. The bug was actually already known about and was simply fixed by regularly rebooting the system, and thereby resetting the system clock. Unfortunately, those in charge didn't clearly understand how 'regularly' they should reboot the system, and it was left running for 100 hours. When an Iraqi missile was launched, targeting a US airfield in Dhahran, Saudi Arabia, it was detected by the Patriot missile system. However, by this point, the internal clock had drifted out by 0.34 of a second, so when it tried to calculate where the missile would be next, it was looking at an area of the sky over half a kilometer away from missiles true location. It promptly assumed there was no enemy missile after all and cancelled the interception. The missile carried on to its destination where it killed 28 soldiers and injured a further 98.

Source: http://listverse.com/2012/12/24/10-seriously-epic-computer-software-bugs/

- Failure in sIOP
- Failure between : Internal clock <-> targeting software
- Fault: Semantic accuracy; targeting software assumed stable accuracy of internal clock

Undetected hole in ozon layer

Launched in 1978, NASA's Total Ozone Mapping Spectrometer (TOMS) produced profiles of how thick or thin the ozone was at different altitudes and locations. Ironically, it wasn't until October 1985 that a British team of scientists found a significant reduction in ozone over Halley Bay, Antarctica. Using a ground-based Dobson ozone spectrophotometer, the team found that the amount of stratospheric ozone over Halley Bay was about 40 percent less than it had been the previous year. Their finding stunned the science community because they were expecting anthropogenic ozone depletion to occur first at upper levels in the stratosphere (30 to 50 km) and so they anticipated that the initial signal of depletion in a total column of ozone would be weak. NASA researchers hastily reviewed their TOMS data and found that it too had detected a dramatic loss of ozone over all of Antarctica. Why hadn't they discovered the phenomenon earlier? Unfortunately, the TOMS data analysis software had been programmed to flag and set aside data points that deviated greatly from expected measurements and so the initial measurements that should have set off alarms were simply overlooked. In short, the TOMS team failed to detect the ozone depletion years earlier because it was much more severe than scientists expected.

Source: http://earthobservatory.nasa.gov/Features/RemoteSensingAtmosphere/remote_sensing5.php

- Failure in sIOP
- Failure between : measurement system <-> TOMS data analysis
- Fault: Categorisation variety due to wrong assumption in TOMS, although the Semantic veracity of the measurement system was high

Autopilot test

De eerste autopilot werd getest met uitvoerige simulaties. De bemanning werd onaangenaam verrast bij één van de scenarios waarbij de simulator de evenaar passeerde en de autopilot het vliegtuig spontaan ondersteboven liet vliegen (Janssen 1986). Similarly, early American air-traffic control software could not be used in Britain because it was unable to cope with longitudes less than 0.

Source: http://people.engr.ncsu.edu/efg/379/sum03/lectures/wk13/lecture.html

Appendix - sIOP faults

Where sIOP failures show the result of failing semantic interoperability, sIOP faults refer to the underlying causes for these failures to occur. Fundamentally, these causes are situated in **Semantic heterogeneity**, that, according to (Scheider 2012) "occurs when semantic interpretations of user and provider are not compatible with each other". There are many reasons why semantic interpretations can be incompatible with each other, not at the least because their differences indeed are implied. Distinct perspectives on the same reality, each of them equally valid in their own right, will continue to exist. Therefore, we consider semantic heterogeneity a feature as opposed to a bug that should be resolved. We take the philosophical *subjectivistic* stance that semantics *is* heterogeneous, and that one single truth does *not* exist. Its pragmatic consequence is that semantic interoperability cannot be realised by the definition of one single "universal" standard. We allow for the occurrence of semantic heterogeneity, and instead, we want to be able to differentiate between intended and unintended semantic variations.

This section presents an as comprehensive as possible list of faults that are foundational to the emergence of semantic variations. Refer to Table 8.1 for an overview of how these semantic faults introduce the previously described semantic problems.

Table 8.1: Cross-reference between the examples of the semantic failures from the previous appendix (as columns) and the semantic variations that might unintentionally occur (as rows), representing the semantic faults.

	Autopilot test	Account deletion	Twin identification	Space shuttle inversion	Mars climate orbiter	Insulin pens versus vials	Endeavor mission	Daylight saving time	Undetected hole in ozon layer	Patriot Missile bug	Ariane 5 disaster	Sea level reference	Y2K solution: month 13	NORAD missile warning failure	Divide by zero - ready
Different worlds	✓	(1)	(1)				✓								
Different circumstances															
Different conventions	✓			✓	✓	✓			✓	✓	(3)		✓		
Distinct semantic grounding		✓	✓							✓					
Varying use of datafield									✓		(3)				
Term overloading										✓					(6)
Concept overloading															
Temporal scalability								(2)							
Categorisation variety												✓			
Semantic granularity variety															
Low representational homomorphism		✓					✓				(4)	✓			
Representation differences						✓					✓	✓			✓
Lacking semantic provenance								(2)							
Varying semantic intelligibility															
Low semantic precision															
Low semantic veracity											✓				
Varying data characteristics				✓	✓	✓	✓	✓	✓	✓					
Low data quality							✓					(5)[1]	✓		

[1] Is dit een voorbeeld van hoe een particular (hier: de instance data) een andere SoA refereert dan de bijbehorende universal?

	Autopilot test	Account deletion	Twin identification	Space shuttle inversion	Mars climate orbiter	Insulin pens versus vials	Endeavor mission	Daylight saving time	Undetected hole in ozon layer	Patriot Missile bug	Ariane 5 disaster	Sea level reference	Y2K solution: month 13	NORAD missile warning failure	Divide by zero - ready
Varying data identity		✓													
kolomnummer				5						10					15

Notes to the tabel:

1. Difference in 'Weltanschauung'.
2. The sIOP issue did not occur due to temporal scalability, but after the issue occurred the definition of the concept 'anthropogenic ozone depletion' was altered[2].
3. In the description of this case it is not revealed how the connected systems and applications would interpret 13 months in one year; we can however assume that special precautions had to be made to prevent out-of-bound errors.
4. This shows a typical example for construct redundancy, i.e., two ways to represent the first of January, 2000.
5. The data in this case referred to a complete different UoD than intended by the data producer, e.g., low data consistency .
6. The use of zero in this case to represent 'no data' is a typical example of term overloading.

brandtp, 26-4-2015 15:08:16 Toevoegen: medische voorbeelden uit voorgaande hoofdstuk

We will explain the semantic faults in the following subsections.

Different reality

The applications that we consider are about a specific part of reality. Reality refers to everything out there that we can somehow represent in tokens, e.g., a physical, administrative, virtual or even a mythical reality. Applications can only engage in conversation with each other if their realities that each of them is concerned with, overlap. Such overlap, however, can vary in at least two ways:

- Variations in the *Universe of Discourse* (UoD) (see **??**). Any collaboration becomes impossible when addressing different subjects. The situation where a partial overlap exists can be both dangerous as well as profitable. It can show dangerous when the border between the overlapping and disjunct parts is not clear, especially when tokens refer to entities that are divided by the border . Here the examples from **??** apply about the different *germane entities* that are considered in the shared reality.
- When communicating peers do not share an identical cognitive orientation and maintain a different 'Weltanschauung', their realities differ in terms of *what there is* (see **??**). As a result, although they may point to the same thing in reality, e.g., a box on the table, they can mean a very different thing, e.g., the location on the table versus the box itself, or even the cat in the box, dead and alive.

brandtp, 3/21/2017 Example required: shared qualities their values expressed differently (unity), or only partly shared (accuracy and resolution); shared entities that contains parts that are not shared.

het type dingen dat er – ongeacht de UoD – in de wereld bestaat, het zgn. *Wereldbeeld* of 'Weltanschauung'. Wanneer hier tussen de beide gesprekspartners geen overlap bestaat, zullen beide niet allen op verschillende wijze dezelfde UoD conceptualiseren, maar bestaat er tevens geen gezamenlijk kader waarbinnen tot begripsuitwisseling kan worden gekomen.

Omdat het in de praktijk niet erg zinvol is samenwerking te bewerkstelligen omtrent verschillende zaken, en omdat we niet filosofisch radicaal skeptisch zijn, doen we de aanname dat in ons onderzoek deze interoperabiliteitsproblemen niet aan de orde zullen zijn.

Kortom, een SIOp probleem als gevolg van inconsistentie tussen universal en particular

[2] As a result, without semantic provenance in place the interpretation of 'old' data, i.e., recorded before October 1995, would suffer from sIOP due to temporal scalability.

Different circumstances

De situatie waarin de data is ontstaan is van invloed op zowel de wijze van het gebruik als de validiteit van het gebruik. Strikt genomen zijn de omstandigheden niet bepalend voor de betekenisgeving; het concept blijft het concept, onder welke omstandigheid ook. De situatie is strikt genomen alleen relevant voor de pragmatiek, en niet zozeer voor de semantiek. Omdat de eigenschappen van het concept wel anders zijn onder verschillende omstandigheden, bijvoorbeeld de temperatuur van een draaiende of stilstaande motor, èn omdat een toepassing in vele gevallen onvoldoende informatie heeft om iets zinvols te doen als de omstandigheden onbekend zijn, is het verdedigbaar het situationele aspect te betrekken bij de grondoorzaken die van invloed zijn op sIOP. Andersgezegd, sommige pragmatische problemen zijn aan te pakken door een betere, rijkere semantiek te gebruiken, net zoals sommige semantische problemen kunnen worden opgelost door een verrijking in de syntactische laag, zoals een semantische standaard 'hrXML'.

Different conventions

Conventies zijn breedgedragen afspraken en kennen een grote varieteit aan toepassingsgebieden, zoals de notatiewijze voor graden celsius: °C, volgordelijkheid (qwerty), referenties (westerlengte), en veel meer. Ontologische gezien stellen conventies de restcategorie voor: daar waar geen fundamentele of natuurlijke regels voor lijken te bestaan die de eenduidigheid kunnen vaststellen, wordt eenduidigheid gecreëerd door afspraken of praktische referentiekaders. Conventies zijn daarmee ook in potentie ambigu, juist vanwege de arbitraire keuze die gemaakt wordt. Voor gesprekspartners is het dus van belang dat beide dezelfde conventie volgen, èn dezelfde interpretatie ervan hebben.

brandtp, 25-3-2015 10:37:53 Op dit moment lijken de grondoorzaken *Conventies* en *Waardeeigenschappen* erg veel op elkaar. Niet qua definitie, maar wel qua toekenning van problemen. Dat komt omdat binnen waardeeigenschappen de keuze van de waardeschaal conventioneel is. Dat aspect moet niet worden meegenomen binnen deze grondoorzaak.

Semantic conventions [REFERENCE] There exist many semantic conventions that originated from a certain domain but leaked into other domains as well. These conventions are often shortcuts for more correct but also verbose descriptions, or ontologically wrong but broadly accepted definitions of terms. Very similar to the example on direction of river flow is the direction of wind; when we speak about a north-west gale, a wind direction *from* north-west is implied as opposed to an equally valid *to* north-west direction.

Distinct semantic grounding

Semantic grounding (Scheider 2012 ; Steels 2012) refers to how a concept relates to the universe of discourse (UoD) in the world. Absence of semantic grounding leads to absence of crucial information about, e.g., what are the primitives in the model, on what grounds did they become primitives and what knowledge pattern should they be aligned to (Janowicz and Hitzler 2012)?

Wanneer de grondslag van een begrip aanleiding kan geven tot een verschil in interpretatie, dan is het begrip niet eenduidig te herleiden tot dat waar het voor staat. Juridische teksten zijn vaak zeer uitgebreid en maken gebruik van zeer specifieke termen die gegrond zijn in enerzijds de wet (een definitie dus), of rechterlijke uitspraken (jurisprudentie) anderzijds. Hierdoor wordt getracht meervoudige uitleg te voorkómen. De semantic grounding hier volgt dus uit onweerlegbare relaties met wetsteksten. Een veel natuurlijkere versie van de semantic grounding is gebaseerd op perceptie: de term 'bitter' bijvoorbeeld is nauwelijks te omschrijven, doch iedereen heeft toch een eenduidig begrip van de term door het proeven van een grapefruit of andere bittere etenswaar. Eenzelfde voorbeeld is die van kleur, en het sIOP probleem wordt hierbij vooral ervaren bij gesprekken met kleurenblinden. Ter voorkóming van het probleem van semantic grounding, stelt het Semantic Web als eis dat elk concept een IRI heeft die kan worden *resolved*, wat zoveel wil zeggen dat het herleid

kan worden naar een uniek (web)resource dat, net als bij juridische teksten, vastlegt wat met de term wordt bedoeld. Bijvoorbeeld: gewicht van een patient. Daar zijn vele verschillende definities van, zodat er binnen SNOMED-CT dan ook evenveel unieke codes voor zijn. Omdat computationeel alleen kan worden vastgesteld of twee IRI's dezelfde zijn en geen enkele conclusie kan worden afgeleid uit de inhoudelijke definitie, is deze oplossing slechts een eerste stap. Tot slot zijn deze oplossingen sterk afhankelijk van (de rijkwijdte van) de definitie, die meestal beperkt blijft tot een specifiek (sub)domain.

Varying use of datafield

Dit knelpunt treedt typisch op als gevolg van het maken van een shortcut in de software: in plaats van het introduceren van een nieuw syntactisch element en de daarvoor benodigde afhandelingsprocedure, wordt het bestaande syntactische element en diens afhandelingsprocedure een klein beetje opgerekt om andersoortige data te communiceren. Denk hierbij aan status data, bijv. het gebruik van de waarde `0` als codering voor afwezigheid van data, of `31-12-9999` om aan te geven dat er geen einddatum is voorzien. De status wordt op deze wijze gecodeerd in datawaardes die ofwel in de realiteit niet mogelijk zijn, ofwel binnen de toepassing geacht worden nooit voor te zullen komen. Het knelpunt leidt tot problemen wanneer de status data niet als codering van status worden herkend, en derhalve daadwerkelijk als waarde – uitzonderlijke, maar desalniettemin als mogelijke waarde – worden opgevat.

Dit probleem is feitelijk een combinatie van twee andere knelpunten, namelijk een specifieke (lokale) *conventie* die een categorieverdeling aangeeft in de wijze waaraan de *data characteristics* worden toegekend. Dit knelpunt is derhalve breder dan alleen de verwarring tussen gewone data en statusdata; codering kan voor meer dan alleen status worden ingezet, bijvoorbeeld als overgang naar andere data categorieën, of als verschuiving van de schaal naar negatieve waarden.

Term overloading

Term overloading (Janowicz and Hitzler 2012) is a very well known issue, e.g., a bank represents both a financial institute and the side of a river, or, a *large scale* study includes a *large* number of elements under survey, whilst a geographic map with *large scale* covers a *small* area since the scale represents a fraction. There are far more things in the world than we have terms for, therefore it is considered a necessary feature for sIOP to facilitate the means to identify and resolve term overloading.[3]

Concept overloading

Concept overloading (Janowicz and Hitzler 2012) expresses the fact that a conceptualisation is unsound: there is and will be more than one way to conceptualise reality, all of which might be considered correct. For instance, a street is a connection from A to B from the view point of transportation science, while it is a disruptive separation that cuts a habitat into segments from the view point of ecology and conservation.

sensor values can be represented in distinct scale and type, e.g., Fahrenheit versus Kelvin, or multiple distinct but often combined readings can be modelled as one parameter or as a single concept contains multiple values, e.g., blood pressure as "systolic/diastolic" string or as a BP container concept with systolic and

[3] Term overloading is something different than Concept overloading since the former identifies variability due to overloading the semiotic Representamen with multiple Objects, while the latter only overloads the Interpretants with multiple Objects

diastolic attributes. Another example is a strict correlation of multiple parameters into one data item, e.g., speed vs. direction of flow of a river, the latter being identical over 99.9% of its basin except for its estuary. Therefore, direction of the river flow is often implied towards the sea, as opposed to explicitly modelled.

Concept overloading seems very similar to representational homomorphism, especially language excess. However, language excess relates to the semiotic 'representation' axis whilst concept overloading refers to the semiotic 'abstraction' axis.

Temporal scalability

Temporal scalability (Schlieder 2010) characterises the fact that semantics change over time due to semantic and cultural ageing. For instance, the expression "I'll save it in my cloud" was considered nonsense, or at best an empty statement until ICT cloud storage was invented. Other examples include new insights that lead to the introduction of an intermediate category between two existing ones; introduction of new laws or modified business processes, giving rise to modification of definitions. Clearly, this impacts sIOP since conversation partners can ground the term used historically different. Hence, capabilities to read and interpret the concepts have to persist, e.g., when population of classes with instances change over time, or when queries or inferences change their use of instances or classes. Since semantics are subject to evolution, it should at least be considered a necessary feature for the *governance* of sIOP. For instance, stock prices listed at Frankfurt stock exchange dropped almost by 50% during the year turnover from 1998 to 1999. This had nothing to do with the state of the world during those days, but was due to the Europe currency change to Euros instead.

This fault is similar to instance unification, where absence to acknowledge temporal scalability essentially merges time periods as opposed to merging state of affairs as is implied by instance unification.

Categorisation variety

Categorisation variety (Janowicz and Hitzler 2012) originates from implicit categorisation, e.g., lacking a definition for it but using the term as if objectively defined, e.g., *forest*, or assuming locally different purposes for a categorisation, e.g., a pothole in the road (Stuckenschmidt 2012). The categorisation variety almost always originate either from dependency on the consequences that follow from being part of the category, e.g., being a pothole legally induces financial ramifications, or from reuse of implicit semantics that come naturally with universals. The latter is closely related to the aspect of Different Worlds (**??**) where we have rejected the philosophical notion of radical scepticism and assumed an objective conception of reality (however, without universal formalisation).

brandtp, 23-4-2015 14:43:32 adequate evidence missing

Semantic granularity variety

Semantic granularity (Albertoni et al. 2006; Fonseca et al. 2002; Keet 2008) represents the level of detail at which semantics have been defined, and summarises the degree of informativeness that results from both the issues that describe the concept and the structure or schema in which it is embedded.

The term **semantic accuracy**, coined by (Nyberg and Mitamura 1992), "characterises the completeness and correctness with which an entity in reality, once conceptualised and represented by a model, can be interpreted by the recipient". The completeness of a drawing of a tree, for example, will be influenced by

the ex- or inclusion of its roots and of its leafs, while its correctness might suffer from the artist's drawing skills, or creative style, e.g., impressionism versus expressionism. In our opinion, correctness might be better termed *faithfulness*.

We take the stance that where semantic granularity represents an indication of level of detail, semantic accuracy is about the criteria one sets on the grains itself. Difference in semantic granularity between communicating peers shows by details being absent, conceptual incompleteness (e.g., missing issues) or discrepancies in conceptual structures.

Low representational homomorphism

Representational homomorphism[4] addresses the one-to-one correspondence of a model representation to the conceptualisation of the reality that has been modeled, and seeks to its achievement by designing a meta-model, i.e., a language, that is capable of accurately representing the model. **Domain appropriateness** (i.e., truthfulness to reality) and **comprehensibility appropriateness** (i.e., conceptual clarity) (Guizzardi 2007) address the extent to which such representation language is fit for use in a certain domain, and as such represent two quality criteria of a modeling language. These criteria compare the worldview underlying a language (it's meta-model) with a representation of the specific domain conceptualisation that it represents. According to (ibid.) both criteria depend on the level of homomorphism between the meta-model and the conceptualisation. A low homomorphism provides room for misinterpretations and hence low sIOP. Homomorphism is dependent on each one of the following characteristics.

brandtp, 12-4-2015 15:28:19 Bijna alle teksten hier zijn letterlijke kopieën uit [@Guizzardi:2005tn] en moeten herschreven of beter gerefereerd worden.

We discuss for each quality criterion the characteristics of the underlying language and their impact on the model construction.

Soundness and language excess

Construct excess (Wand and Weber 1993) occurs when a language construct does not represent any domain concept. A language `L` is **sound** w.r.t. to a domain `D` iff every modeling primitive in the language has an interpretation in terms of a domain concept in the ontology `O` (Guizzardi et al. 2005). Although construct excess results in the creation of unsound models, soundness at the language level does not prohibit the creation of unsound models.

Since no mapping is defined for the exceeding construct, its meaning becomes uncertain, hence, undermining the clarity of the model. Users of modeling language must be able to make a clear link between a modeling construct and its interpretation in terms of domain concepts. Otherwise, they will be unable to articulate precisely the meaning of the models they generate using the language.

These characteristics are very close to *concept overloading* (see Chapter 8), the distinction being that concept overload is a fact of life that *is* occurring intra- or inter-domain and *will* hamper sIOP, while soundness and construct excesses are tooling issues that are implicit to unsound language design principles that create sIOP issues by representational design-flaws.

[4] This might be better phrased as representational *isomorphism*

Laconicity and language redundancy

Construct redundancy (Wand and Weber 1993) occurs when more than one language construct can be used to represent the same domain concept. A language L is **laconic** w.r.t. to a domain D iff every concept in the ontology O of that domain is represented at most once in the metamodel of that language (Guizzardi et al. 2005). Despite of being related, laconicity and construct redundancy are two different (even opposite) notions. On one hand, construct redundancy does not entail non-laconicity. For example, a language can have two different constructs to represent the same concept, however, in every situation the construct is used in particular models it only represents a single domain element. On the other hand, the lack of construct redundancy in a language does not prevent the creation of non-laconic models in that language. Non-laconicity at the language level can be considered as a special case of construct redundancy that does entail non-laconicity at the model level.

Construct redundancy adds unnecessarily to the complexity of the modeling language, possibly confusing the users. Non-laconicity allows redundant representations that can be interpreted as standing for a different domain element.

Lucidity and language overload

Construct overload, also known as *ontological overload* (Wand and Weber 1993), occurs when a single language construct is used to represent two or more domain concepts. A language L is **lucid** w.r.t. to a domain D iff every modeling primitive in the language represents at most one domain concept in O (Guizzardi et al. 2005). Although construct overload and non-lucid representation seem very similar, these notions albeit related are not identical, as is shown in (Guizzardi et al. 2005 fig. 3). The absence of construct overload does not directly prevent the construction of non-lucid representations in this language. Additionally, construct overload does not entail non-lucidity.

Construct overload is an undesirable property of a modeling language since it causes ambiguity and, hence, undermines clarity. When a construct overload exists, users have to bring additional knowledge not contained in the model to understand which phenomena is being represented. Additionally, a non-lucid representation language allows for ambiguous or obscure interpretations.

Completeness and language expressivity

A limited **language expressivity** (Wand and Weber 1993) results in incomplete models, i.e., there exist phenomena in the considered domain that cannot be represented in the model. A language L is considered **complete** w.r.t. to a domain D iff every concept in the ontology O of that domain is represented in a modeling primitive of that language, e.g., if every concept in a domain conceptualisation is covered by at least one modeling construct of the language (Guizzardi et al. 2005).

In order to compensate for limited language expressivity, users of the language can choose to overload an existing construct in order to represent concepts that originally could not be represented, thus, undermining clarity. Hence, unless some existing construct is overloaded, an incomplete modeling language is bound to produce incomplete models. However, the converse is not true, i.e., a complete modeling language can still be used to produce incomplete models.

Representation differences

Representation differences often occur when semantics have been defined independently, and refers to using different syntax to represent identical concepts. For instance, a representation of a car registration plate represents the registration of that car, independent on its syntactical form, e.g., textual versus graphic encoding where the latter represents different syntax in its own right, e.g., png versus bmp versus jpg. Or another, very familiar example is the use of a different language, something that has been acknowledged already in many applications and for which a syntactic conversion mechanism has been devised, called I18N and L10N.

This is different from the representational homomorphism since a representation difference finds its origin in incompatible choices of syntactical representation, while the representational homomorphism refers to the way the concept can be mapped onto a representation.

Dit is een mooi voorbeeld van stapeling van semiotische driehoeken: De semiotische driehoek die de grafische representatie geeft, heeft daar bovenop een semiotische driehoek die de interpretatie van het beeld van de nummerplaat naar de daadwerkelijke letters van de nummerplaat weergeeft, en daarbovenop weer een driehoek die gaat over de principle of identification ter identificatie van die specifieke auto.

Lacking semantic provenance

Similarly to its definition on the level of `particulars`, we introduce **semantic provenance** on the level of `universals` as the derivation history of a semantic product starting from its original sources. Its purpose is to provide for traceability information about the semioses, e.g., the way that a specific `object` has been perceived, abstracted, and represented into its referring 'representamen'. In this way, semantic provenance is an important information source about the semiotic history. This is close to, or encompasses, purpose of model and/or application. (**???**) states that provenance is information about entities, activities, and people involved in producing a piece of data or thing in the world, which can be used to form assessments about its quality, reliability or trustworthiness.

Varying semantic intelligibility

Semantic intelligibility refers to have, as a recipient, sufficient prior knowledge about the world that the model addresses. Without overlap, at least a partial one, between the reality that is considered by the recipient and the reality that the model is about, the model is deemed incomprehensible for the recipient. For example, if I state that my car is green, I cannot expect the recipient to begin to comprehend that statement if world knowledge about vehicles and colours is absent. Intelligibility is also used by (Neuhaus et al. 2013) to indicate that all intended users need to be able to understand the intended interpretation of the ontology elements (e.g., individuals, classes, relationships) that are relevant to their use case. This view relates to the prior knowledge about the meta-model of the model, i.e., to be able to recognise the foundations that the model is build upon: In the same example, the recipient cannot recognise the car as a bearer for the green colour if knowledge about qualities that inhere in an entity is absent.

Low semantic precision

Semantic precision is defined by (Falbo et al. 2005, @McComb:2003uF) to refer to the specificity with which the data has been meta-annotated to a specific level of discernment. We take the stance that it is not about meta-annotation but about the ability to discern between apparently similar concepts. Semantic precision can thus be viewed as the inverse of categorisation (refer to Chapter 8) in that categorisation is about selecting concepts to fit a predefined set, while precision is about differentiating concepts to belong to different sets.

This cannot be seen independently from semantic granularity, because more detail will provide more opportunity to discern distinct concepts.

Low semantic veracity

Semantic veracity is defined by (Falbo et al. 2005) to refer to "the level of validity of the information, e.g., PI has a higher veracity with 3.1415 than with 3.14[5]". However, in our opinion, semantic veracity is a collective term for all characteristics that contribute to the correspondence between the objects and their real-world referents, and resonates with Grice's quality maxim (Grice 1975). Additionally, (McComb 2003) remarks that "You cannot determine the veracity of the data from only the data; you must also examine the system that collected and verified the data and determine how the data are used (context)".

brandtp, 23-3-2015 17:49:55 Is daarmee SemAcc subsumed by SemVeracity?

In our opinion, semantic veracity is determined by:

- Data characteristics
- Data quality
- Semantic provenance
- background knowledge
- Semantic accuracy

Verification of the semantic veracity includes verification against controlled vocabularies; against internal consistent prior knowledge, e.g., quantity > 0; corroborative checks (provenance); closed loop with intended originator, e.g., provenance; independent audit.

Semantic accuracy (Nyberg and Mitamura 1992) characterises the completeness and correctness with which an entity in reality, once conceptualised and represented by a model, can be interpreted by the recipient. For example, when a drawing is made of a tree, its completeness will be influenced by the ex- or inclusion of its roots, or *all* its leafs, while its correctness might suffer from the artist's drawing skills, or creative style, e.g., impressionism versus expressionism.

Varying data characteristics

Een waarde van een concept wordt uitgedrukt in één of meerdere dimensies. Elke dimensie kent een aantal eigenschappen die niet altijd evident zijn. De betekenis van een dimensie wordt grotendeels bepaald door de eenheid, maar ook door de resolutie waarmee aanliggende waarden worden onderscheiden, alsmede de aard van de dimensie zelf zoals het continu of discreet zijn, en begrensd zijn of niet (en zo ja met welke grenzen dan). Dit is sterk gerelateerd aan *conceptual spaces* van (Gärdenfors 2004), en zoals uitgewerkt door (Probst 2008).

brandtp, 27-3-2015 11:49: Toevoegen: de verschillende elementen die Gärdenfors benoemd in zijn transitie van space naar waaredimensie.

[5] This is not similar to data accuracy: a higher data accuracy can improve semantic veracity, but vice-versa is not true.

Zie: (Probst 2008 fig. 8). Een conceptual space (bijvoorbeeld de daadwerkelijke temperatuur) wordt afgebeeld op een zgn. reference space, een door conventie bepaalde dimensie (bijvoorbeeld Fahrenheit). Hierin wordt voor elke conventie een specifieke invulling gegeven aan afbeeldingsaspecten zoals bijvoorbeeld resolutie. De resolutie van bijvoorbeeld de hartslag wordt weergegeven als een geheel getal, bijv. 115. De semantiek van een getal zoals '115' wordt met standaard conventie geïnterpreteerd als het interval 114.5 en 115.5, en met resolutie '1' in de reference space. Deze resolutie voor hartslag is echter niet relevant en wordt vaak afgerond op vijftallen. Derhalve wordt de conceptual space met grovere resolutie afgebeeld op de reference space, te weten intervallen van 5 eenheden breed die gecentreerd is rond de indicator. De standaard conventie geldt niet meer: de indicator '115' in de reference space refereert hier aan het interval 112.5 en 117.5. Zonder resolutieaanduiding wordt dus meer significantie aangenomen dan waarin de conventionele dimensie daadwerkelijk kan voorzien, en zullen de waardes 113, 114, 116 en 117 zoals die bestaan in de standaard conventie geen deel uitmaken van de resultaten; niet omdat ze niet bestaan, maar omdat er geen indicator voor bestaat.

Het betreft NIET de accuraatheid van een waarde, want dat is een ander (alhoewel samenhangend) concept dat echter direct afhankelijk is van het observatieproces (zie **??**). Daarmee heeft het niet te maken met de waardedimensie zelf maar juist met de metrologische aspecten. Tezamen, kwaliteit van data en eigenschappen van de waardedimensies, geeft dit de semantiek waarover een toepassing moet beschikken om te kunnen bepalen in welke mate de data toepasbaar is voor die specifieke toepassing.

Intuitively overlap exists between Data Characteristics and Semantic Convention due to the fact of how the conceptual spaces are projected onto, and discretized into, a semantic reference space. However, the mere fact of the relationship between conceptual spaces and reference spaces, and its subsequent characteristics, e.g., scale resolution, is what is being relevant here. Only the way *how* it is done, e.g., the naming or the selection of the grounding magnitude, is determined by convention, for instance the distinction between degree Celsius and degree Fahrenheit. So there is definitely a relationship between Data Characteristics and Semantic Convention, but no overlap.

Low data quality

Kwaliteit van data is uiteindelijk een gevolg van het proces dat is toegepast om de data te genereren. Daarom is het dus complementair aan de eigenschappen van de waarde zoals beschreven in **??**. We hebben het hier ook niet over aspecten zoals accessibility, security en dergelijke, want dat is irrelevant voor sIOP in de zin dat we daarvan aannemen dat die al in de syntactische laag zijn opgelost. Kwaliteit van data is feitelijk geen semantisch aspect, want het beïnvloedt niet de duiding van de data maar de toepasbaarheid ervan. In die zin is data kwaliteit dus een *pragmatische* parameter. Echter, de afwezigheid van voldoende kwaliteit van data kan zinvol gebruik ervan verhinderen, kan daarmee een drempel opwerpen voor de toepassing om de data op zinvolle wijze te verwerken. Hiermee raakt het dus aan de grens tussen semantiek en pragmatiek: als we stellen dat semantiek vooraf gaat aan pragmatiek, en voldoende kwaliteit benodigd is alvorens het te kunnen toepassen, dan is datakwaliteit onderdeel van semantiek.

10-4-2015 Niet alleen agv het meetproces: een voorbeeld zoals "plus of min zoveel procent" hoeft geen gevolg te zijn van het generatieproces, toch?

8.0.0.1 Data Correctness {-}

Dit knelpunt , dat door (Nousak and Phelps 2002) gedefinieerd wordt als '**Data Validity**: Data element passes all edits[6] for acceptability', geeft de situatie weer waarbij de waarde qua vorm niet afwijkt van daartoe geaccepteerde criteria (bijv. een persoonsnaam zonder getallen), of qua waarde binnen als normaal geaccepteerde limieten valt (bijv. een negatief huisnummer). Hierbij kan een parallel getrokken worden met

brandtp, 20-4-2015 10:13:01 Hernoemen naar 'Data Integrity

[6] Edits? Vreemd woordgebruik. In literatuur nazoeken

de termen 'correct XML' (correct in accordance with XML syntax) en 'valid XML' (correct XML in accordance to subject XML schema): data correctness betreft de correctheid in relatie tot de betreffende reference space karakteristieken (see Chapter 8), en NIET in relatie tot de werkelijkheid .

brandtp, 20-4-2015 10:58:33 Hiermee is dit geen grondoorzaak meer, maar een evaluatiecriterium.

8.0.0.2 Data Completeness {-}

Data Completeness (Nousak and Phelps 2002): A data element is either always required, or required based on the condition of another data element. Dit knelpunt betreft de noodzaak van de aanwezigheid van de waarde: niet alle data is noodzakelijk, maar de data wordt compleet geacht alléén als alle noodzakelijke data aanwezig is. Hierdoor valt context dus ook binnen de semantiek.

An example on failing this quality criterion is given when a Payroll record misses a value for Person.

8.0.0.3 Data Consistency {-}

Data Consistency (Nousak and Phelps 2002): The authors have defined this as "A data element is free from variation and contradiction based on the condition of another data element". In our opinion, is too small a definition when semantics is involved. We would suggest to not only take a data *internal* perspective, but add an external perspective as well , i.e., that takes into account the coherency with the state of affairs in the world that the data describes.

brandtp, 17-4-2015 20:12:50 Deze keuze impliceert een relatie/mapping op de semiotische as Object->Interpretant

Failure is exemplified as, e.g., a New Hire record has a Hire Date before the birth date of the individual, or, a Leave of Absence is checked, but the employee is at work.

8.0.0.4 Data Uniqueness {-}

Data Uniqueness (Nousak and Phelps 2002): Data element is unique—there are no duplicate values. Dit is identiek aan *Identificatie van Individu*

Failure to this criterion is exemplified by, e.g., two Person records that have the same Social Security Number.

brandtp, 17-4-2015 20:16:59 De term 'coherency' is hier wellicht niet voldoende correct, want bedoel ik hier niet uiteindelijk de correctheid waarmee de data de UoD weergeeft.

8.0.0.5 Time of occurrence {-}

There is a lot to say about the relationship between data and time. **Data Timeliness** is defined by (Nousak and Phelps 2002) as 'The data element represents the most current information resulting from the output of a business event', while expressed by (Batini and Scannapieco) as 'how current data are for the task at hand'. (Batini and Scannapieco) also defines **data volatility**, characterising 'the frequency with which data vary in time'. We take the stance that all there is to know about data and time is about their relationship: is the concept that the data instantiates considered a perdurant (i.e., happening in time) or an endurant (i.e., being in time), and, in case of the former, what is the precise moment that that precise state of affairs occurred. All the rest can be deferred from that,[7] as long as the moment in time is considered a datum in itself with its own specific data quality, e.g., accuracy.

[7] leuk bedacht, Brandt, maar de volatility kun je dus niet vaststellen, want je weet nooit of het het meest current data sample is

Time of occurrence is relevant for semantics of perdurants, at least to establish the order of events, whether that specific SoA is relevant or useless because they are too late or exactly a later or earlier one instead. Endurants do not have such implicit relationship with time, hence time of occurrence is irrelevant for their semantics.

8.0.0.6 Data Accuracy {-}

Data Accuracy is defined by (Nousak and Phelps 2002) as "Data element values are properly assigned". For semantic purposes we would like to refine this and refer to the extent that the concept space has been mapped correctly to the reference space, as indicated by (Probst 2008). A region in the semantic reference space, a *reference region*, serves to approximate an absolute magnitude in some concept dimension. Such reference region is denoted by a term, i.e., the datum. Using this terminology we define the data accuracy as the extent that the datum denotes that specific reference region that approximates the absolute magnitude in question. Note that since the approximation and the denotation is defined by convention, accuracy therefore holds under conventional rules.

8.0.0.7 Data provenance {-}

Data provenance (**???**) describes the derivation history of a data product starting from its original sources. It is a collective term for all aspects related to traceability, responsibility, auditability, accountability and accuracy of data. Provenance gives an important indication about the reliability of the data and is very important for the reuse of (linked) data.

Varying data identity

Identity overload is identical to term overloading, except that it occurs at the instance level: Depending on the context of use, a term can correctly identify one, or incorrectly many individuals. For example, "John" might proof unique and hence sufficient in our family, but insufficiently unique to identify one single person in the city. For this exact reason, the administrative social security number has been invented: a convention that by definition provides the ability to point to one single inhabitant within a nation. Applying a SSN prevents the occurrence of identity overloading.

Instance unification (Hitzler and Harmelen 2010) represents the converse of identity overload in that two different named instances in fact represent identical individuals. Again, one can identify a person by use of his/her social security number, or by use of his/her name. (Nousak and Phelps 2002) defines this as "**Data Uniqueness** : Data element is unique – there are no duplicate values.".

Foundational to both instance unification and identity overloading is the absence of, or difference in selection of, a valid **principle of identity**, defined by (Guizzardi 2005, p98) as "(principle that) supports the judgement whether two particulars are the same, i.e., in which circumstances the identity relation holds.". Incorrect application of this principle results in said ambiguity in the identification of individuals. This implies data incompleteness, or the occurrence of duplicates with the risk of operating on the wrong individual.

Hieraan gerelateerd is het oneigenlijk gebruik van een principle of identity. Hierbij wordt voor de identificatie van individuen 'b' van concept 'B' ten onrechte gebruik gemaakt wordt van de identificatie van individuen 'a' van een gerelateerd concept 'A'. Omdat er vanuit een bepaald gebruiksperspectief een vaste relatie bestaat tussen concept 'A' en het gewenste concept 'B', wordt de indentificatiemethode tbv 'a' gemakshalve ook voor

het gewenste individu 'b' gebruikt. Binnen het gebruiksperspectief wordt dan ten onrechte aangenomen dat die relatie constant blijft, er over tijd geen conflicterende concepten worden toegevoegd, en de opgeslagen gegevens niet buiten de eigen specifieke toepassing wordt gebruikt. Omdat in ieder geval de laatste premisse bij interoperabiliteit niet het geval zal zijn, zal dat onherroepelijk tot een sIOP probleem leiden.

Therefore, it is considered a necessary feature for sIOP to adapt the principle of identity to a mechanism that proves sufficient for the recipient.

References

Note:

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

Albertoni R, Camossi E, Martino M de, Giannini F, Monti M. 2006. Semantic Granularity for the Semantic Web. OTM 2006 work. 1863–1872; doi:10.1007/11915072_93.

Atkinson C, Kühne T. 2003. Model-driven development: a metamodeling foundation. IEEE Softw. 20:36–41; doi:10.1109/MS.2003.1231149.

Batini C, Scannapieco M. *Data Quality*. Springer-Verlag Berlin Heidelberg.

Brandt P, Basten T, Sinderen MJ van. 2018. Semantic mediation: from alignment relations to data transcriptions. Prep.

Brandt P, Grandry E, Basten T. 2019. Consolidating semantics in contemporary software architectural paradigms. Prep.

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.

Diggelen J van. 2007. Achieving semantic interoperability in multi-agent systems: A dialogue-based approach. SIKS 2007-., PhD thesis, Utrecht University, Utrecht.

Euzenat J, Scharffe F, Zimmermann A. 2007. Expressive alignment language and implementation. 70.

Falbo R de A, Ruy FB, Moro RD. 2005. Using Ontologies to Add Semantics to a Software Engineering Environment. SEKE 151–156.

Fonseca F, Egenhofer M, Davis C, Câmara G. 2002. Semantic Granularity in Ontology-Driven Geographic Information Systems. ANN MATH ARTIF INTEL 36:121–151; doi:10.1023/A:1015808104769.

Gamma E, Helm R, Johnson R, Vlissides J. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Gärdenfors P. 2004. *Conceptual spaces: The geometry of thought*. MIT Press, Lund University, Sweden.

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. 1975. Logic and Conversation. In *Syntax semant. 3 speech arts* (P. Cole and J.L. Morganeds.), pp. 41–58, Syntax; semantics 3: Speech arts, Cambridge, MA, USA.

Guarino N, Carrara M, Giaretta P. 1994. Formalizing Ontological Commitments. B. Hayes-Roth and R.E. Korfeds.. Proc. 12th natl. Conf. Artif. Intel. AAAI-94 I: 560–567.

Guizzardi G. 2007. On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. O. Vasilecas, J. Eder, and A. Caplinskaseds. Proc. 2007 conf. Databases inf. Syst. IV sel. Pap. From seventh int. Balt. Conf. DB&IS'2006 155:18–39; doi:10.1017/CBO9781107415324.004.

Guizzardi G. 2005. Ontological foundations for structural conceptual models. PhD thesis, University of Twente, The Netherlands; CTIT, Enschede, The Netherlands.

Guizzardi G, Pires LF, Sinderen MJ van. 2005. An ontology-based approach for evaluating the domain appropriateness and comprehensibility appropriateness of modeling languages. In *Lect. Notes comput. Sci. (Including subser. Lect. Notes artif. Intell. Lect. Notes bioinformatics)* (L. Briand and C. Williamseds.), Vol. 3713 LNCS of, pp. 691–705, Centre for Telematics; Information Technology, University of Twente, Enschede, The Netherlands; Springer Berlin Heidelberg, Centre for Telematics; Information Technology, University of Twente, Enschede, The Netherlands.

Henderson-Sellers B. 2012. *On the mathematics of modelling, metamodelling, ontologies and modelling languages*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hitz M, Montazeri B. 1995. Measuring Coupling and Cohesion In Object-Oriented Systems. Proc. 3rd int. Symp. Appl. Corp. Comput. 50; doi:10.1.1.467.9312.

Hitzler P, Harmelen F van. 2010. A reasonable semantic web. SWJ 1:39–44; doi:10.3233/SW-2010-0010.

Janowicz K, Hitzler P. 2012. The digital earth as knowledge engine. SWJ 3:213–221; doi:DOI 10.3233/SW-2012-0070.

Janssen B. 1986. F-16 Problems.

Keet CM. 2008. A Formal Theory of Granularity - Toward enhancing biological and applied life sciences information system with granularity. DS-2008th-1st ed., PhD thesis, Free University of Bozen-Bolzano, Bozen-Bolzano, Italy.

Kuhn W. 2009. Semantic engineering. In *Res. Trends geogr. Inf. Sci. Lect. Notes geoinf. Cartogr.* (G. Navratiled.), pp. 63–76, Springer Berlin Heidelberg.

Leveson NG. 2004. Role of Software in Spacecraft Accidents. J. Spacecr. Rockets 41:564–575; doi:10.2514/1.11950.

Leveson NG, Charette RN, Claussen B, Droste CS, Fujii RU, Gannon JD, et al. 1993. An Assessment of Space Software Development Shuttle Flight Processes. 206.

McComb D. 2003. *Semantics in Business Systems*. Elsevier Science {&} Technology.

Neuhaus F, Vizedom A, Baclawski K, Bennett M, Dean M, Denny M, et al. 2013. Ontology Summit 2013 Communique.

Neuman BC. 1994. Scale in Distributed Systems. Readings Distrib. Comput. Syst. 463–489.

Neumann PG. 1985. Letter from the Editor; Risks to the Public. ACM SIGSOFT Softw. Eng. Notes 10: 1–25.

Nousak P, Phelps R. 2002. A Scorecard Approach to Improving Data Quality. SUGI27 1–9.

Nyberg EH, Mitamura T. 1992. THE KANT SYSTEM: FAST, ACCURATE, HIGH-QUALITY TRANSLATION IN PRACTICAL DOMAINS. COLING '92 1069–1073; doi:10.3115/993079.993131.

Probst F. 2008. Observations, measurements and semantic reference spaces. AO 3:63–89; doi:10.3233/ao-2008-0046.

Scharffe F, Euzenat J, Zimmermann A. 2011. EDOAL: Expressive and Declarative Ontology Alignment Language.

Scheider S. 2012. Grounding geographic information in perceptual operations. Dissertation, Westfälische Wilhelms-Universität Münster; IOS Press.

Schlieder C. 2010. Digital heritage: Semantic challenges of long-term preservation. SWJ 1:143–147; doi:10.3233/SW-2010-0013.

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.

Stephenson AG, Mulville DR, Bauer FH, Dukeman GA, Norvig P, LaPiana LS, et al. 1999. Mars Climate Orbiter Mishap Investigation Board Phase I Report. 48.

Stuckenschmidt H. 2012. Data Semantics on the Web. J Data Semant 1:1–9; doi:10.1007/s13740-012-0003-z.

Wand Y, Weber R. 1993. On the ontological expressiveness of information systems analysis and design grammars. Inf. Syst. J. 3:217–237; doi:10.1111/j.1365-2575.1993.tb00127.x.