CONSOLIDATING SEMANTIC INTEROPERABILITY IN CONTEMPORARY ARCHITECTURAL PARADIGMS

Establishing loosely coupled semantics

version: v0.4-3

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. We provide guidance to the design of an interoperability bridge over the semantic gap between two software agents by maintaining the connection with the semantic bridgeheads of them both.*

Method: *We assume the collaborating agents to have followed the architectural principles on semantics according to our preliminary study (Brandt et al. 2019). This results in an explicit representation of an atomic semantic monolith for each of the agents: two semantic bridgeheads. Our approach is based on a (computer assisted) formulation of a formal alignment between them: the spanning between both bridgeheads. Finally, the bridge's roadway consists of a generic mediation component that automatically transcribes the exchanged data between the representations from both bridgeheads. Based on this approach, we develop the guiding* *architectural principles with the purpose of consolidating sIOP in contemporary architectures. We 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) semantic transparency, (iii) semantic separation of concerns, and (iv) a shared ontological commitment. 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 human-authored alignments and the automated mediation process at the one hand, and establishing the conditions in support of loosely coupled semantics at the other. By their explicit inclusion, we show that access-and-play sIOP can be consolidated in contemporary architectural paradigms.*

Chapter 1

Introduction

Never before, data were so ubiquitous, and managed access to external data was so easy. Because current ICT is unable to *use* all that same external, non-native data as access-and-play service 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 Section/Appendix ## provides for a short overview of sIOP-faults. Unfortunately, it is fundamentally impossible to automate the production of wrappers, because we need a genuine *understanding* upfront, which computers still cannot do. "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).

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

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 fourfold, 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);
- *ISO42010 Architecture Viewpoint*: We verify the applicability of the above concerns and principles by formulating their architectural consequences as a specific ISO42010 sIOP Viewpoint, and we show their proper position in the total architecture as corresponding sIOP view. As ISO42010 is considered a set of best practises for architecture description, and therefore is used with architecture frameworks such as MoDAF, TOGAF, DoDAF, RM-ODP and so on, we conclude that our sIOP Viewpoint and View can be considered to consolidate sIOP for contemporary architectural paradigms (Chapter 6).

Based on these contributions we defend that access-and-play sIOP can be embedded and hidden in

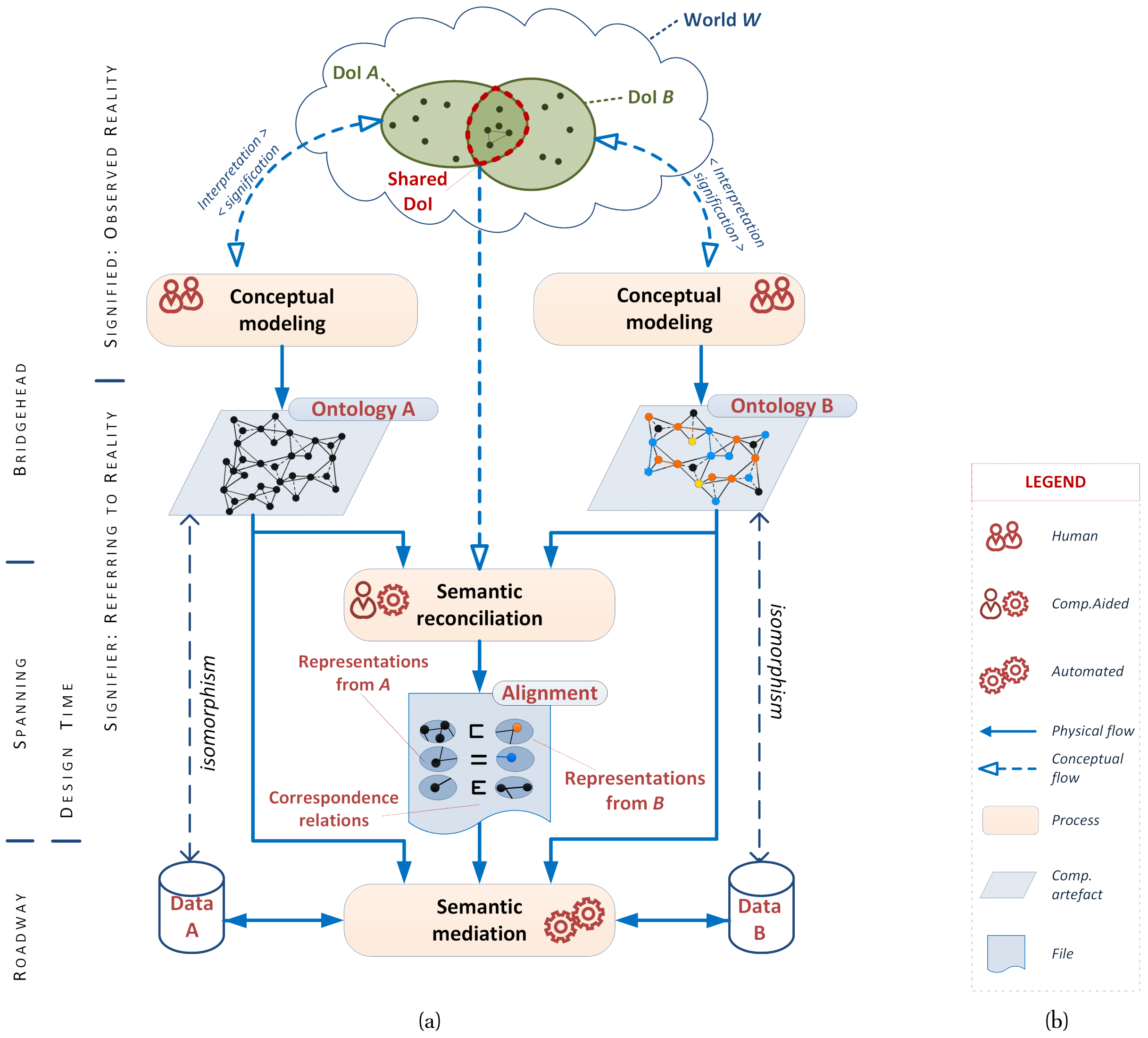

Figure 1.1: Conceptual overview of the relationships in sIOP between the bridgehead (conceptual modelling), its spanning (semantic reconciliation) and roadway (semantic mediation), (a), and a legend explaining the used constructs (b).

infrastructural services when considering semiotic fundamentals and adding loosely coupled formal semantics to contemporary architectural paradigms. To that end, we first describe some background on the semantic foundations in the following section.

Chapter 2

Bridgehead: Semantics

Synopsis: Purpose of this section:

What preparations about semantics are required for an agent before being able to engage in semantic interoperability?

Approach: Summarise [@Brandt2018a], and notably address:

1. explain what we mean with semantics in software agents, i.e., the reciprocity between data and data processing code. Reciprocity denotes the degree with which the collective outcome of processing all potential data tokens by the process token refers to the intended states of affairs in reality.
2. valid semantics demand a coherent reciprocity;
3. representing semantics require descriptive models (i.e., ontologies);
4. Conclude that coherence between the data and its processing code makes for an inevitable semantic monolith. 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;
5. Introduce Principle: ontological commitment as minimal standard for sIOP, "(...) not in order to know *what there is*, but in order to know what a given remark or doctrine, ours or someone else's, *says* there is" [@Quine:1953er];
6. Conclude that ontologies need their place as single point of reference (trueness) in architectures;

For sIOP, we assume the existence of a semantic bridgehead per agent, i.e., ontology, following the principle of taking responsibility for one's own data.

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, further its semantic interoperability with them.* ◇

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

Synopsis: Purpose of this section:

1. Explain sIOP:
   a) the consequence of data exchange = breaking the atomic semantic monolith = breaking the coherence between data and code, i.e., data leaving this monolith is the root cause of current sIOP problems;
   b) receiving external data implies a new reciprocity to emerge with the code of the receiving agent, and
   c) sIOP demands that the external data shall be brought into coherence with the code of the receiving agent; without coherence, phantom semantics will emerge.
2. sIOP objectives:
   a) prime objective is to assure that, from the perspective of the receiving agent and after receipt of data, the reciprocity between data and code remain truthful to the state of affairs in reality. This relates to both the external data and the data that can be inferred from them;
   b) allow for heterogeneity: since distinct agents will maintain alternative but equally legitimate points of view on reality, semantic heterogeneity is a feature to preserve necessarily;
   c) maintain loose coupling between both agents: the ASM's from both agents shall remain independent from each other;
   d) (allow for maintainability and semantic evolution)
   e) strive for access-and-play sIOP: ideally, sIOP between agents can be achieved instantaneously, also for unforeseen collaborations
3. approach to sIOP:
   a) Alignments:

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 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.

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

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.

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

3.2 Explicit sIOP by alignment

Synopsis: Purpose for this section:

Explain how to re-install coherence between "external" data and "internal" data processing code. The most obvious approach is to match the semantic meaning of the external data with that of the internal data, and rely on a generic mediator that translates the data from the vocabulary of the sender to the vocabulary of the receiver.

1. provide for logic-based correspondences between the exchanged concepts from the ontologies of the sending and receiving agent;
2. assure that the correspondence language is sufficiently expressive to align all the types of differences in representation that can occur;
3. Follow the coherence principle and conclude that the models from which *external* data and *receiving* data processing code are derived, need to be brought into coherence with each other.
4. The coherence principle already enforced a single unique semantic reference for each agent. Re-installing coherence demands a semantic alignment between those single unique references.
5. The purpose of that alignment is to establish how the truth of expressions that are formulated in terms of agent A, can be estabished by using formulations in terms of agent A′.
6. List the languages that are used for expressing the alignment should be fit for its purpose, incl. owl contrstructs. Refer to EDOAL [@Scharffe2011] as the currently most complete one.
7. Explain the difference with semantic standards:
   a) ASM's involves data *and* code, standards describe data in terms of their semantic meaning only;
   b) standards are currently applied to produce one single data schema that specify message syntax and structure;
   c) standards are large and as manouverable as an oil tanker

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), reliability, 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.

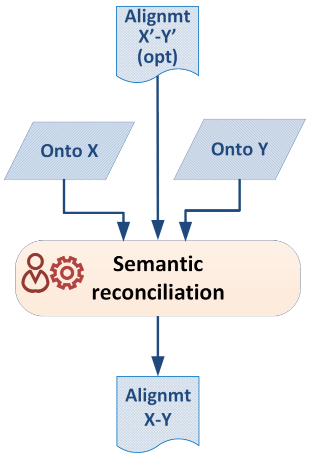

Figure 3.1: Semantic reconciliation results in an alignment between the semantic representations of two ontologies. We defend that semantic reconciliation is a computer-aided but ultimately human-authored task.

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 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.* $\diamond$

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

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

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

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

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 can only be 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 unnecessary complications;*
4. *An intermediary that maintains a separation between representations creates representational transparency between communicating agents;*

Implications:

1. *A mediator keeps agents from using each others representations during interaction;*
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. *The agents do not need to implement semantic wrappers or similar data transcription logic since this becomes a generic service from the communication infrastructure;*
6. *Each agent can communicate with any other agent in its own native semantic representation;*
7. *Loose coupling at the semantic level emerges.* ◇

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.

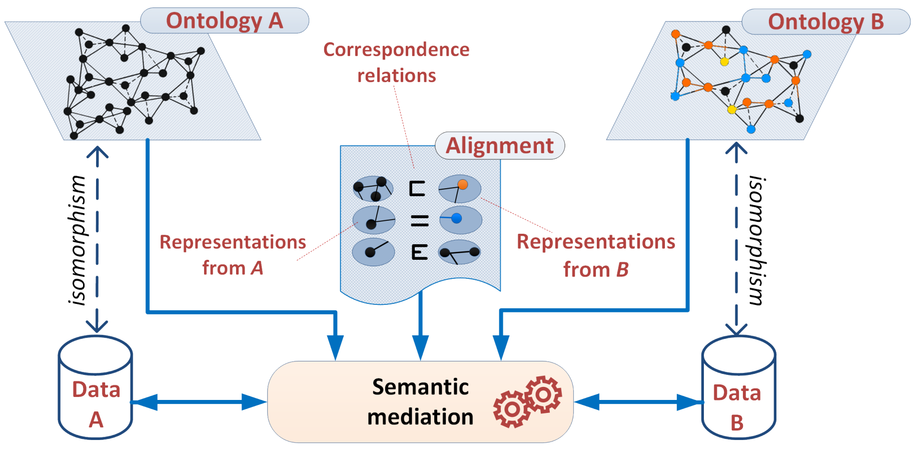

Figure 4.1: Semantic mediation encapsulates how agents exchange semantic meaning. It implements a generic transcription service between the particular representations of data between both agents. It depends only on the representations of the semantic meaning from both agents (ontologies) and the alignment 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 fall-back to the application of naive transcriptions, 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 that will emerge with 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

Principle evaluation

Synopsis: Purpose of this section:

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

The main (business) requirement is to achieve sIOP as quickly as possible, with as minimal effort as possible, for collaborations that had not been foreseen and consequently could not be anticipated for during design time of the (two or more) software agents. 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*: Loose coupling in the classical sense is realised through the principles of separation of concerns, and transparency. In its original reading this implies separating, without duplication, *what* functionality the components are capable of, while leaving *how* such capability is realised completely transparent. Contrarily, semantic separation of concerns is not about maintaining complementary semantics; in fact, the agents are required to have an overlap in the domain of interest, since interoperation would be completely useless otherwise. Instead, separation of concerns refer to enforce an explicit separation between syntax and semantics. Additionally, it refers to keeping each other's representations of the semantic meaning strictly separated. Semantic transparency, then, refers to remaining transparent about how the semantic meaning is represented. As analogy, consider a vehicle (the representation of semantics) with its cargo (the semantic meaning that it bears). Logistics rely on an external transfer service that can change the vehicle from a truck into a ship or aircraft, without ruining the cargo. Similarly, software agents should rely on an infrastructural mediation service that can transcribe the semantic representation from its native form into a foreign form without introducing invalid semantics. Then, agents can communicate in their own native representations without the need to learn or integrate foreign representations.

Below sections are under construction.

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

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

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

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

Achieve loosely coupled semantics

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

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

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

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

5.2 Semantic heterogeneity

We have already determined that the principle to align semantics as opposed to data schemata 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 (ontologies) 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, 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.

5.4 Scalable sIOP

Synopsis: Purpose of this section:

scalability by topology

- Clarify the use of the alignment and mediation approach as an atomic, peer-to-peer sIOP pattern.
- Show how multiple agents can collaborate by cascading, or by introducing a core semantic specification, or by augmenting a centralised topology with a peer-2-peer topology

Scalable sIOP

[1] Pun not intended

Ensure that different semantic topologies remain possible:

i. Star alignments (central domain ontology, aligned to local ontologies) for relative stable and homogeneous domain semantics
   - Good: easy semantic governance
   - Bad: very big semantic monolith, hence, low agility in dynamic environments

ii. Mesh alignments (bilateral alignments) for very dynamic and heterogeneous (domain) semantics, or low number of peers
   - Good: quickly established bilateral sIOP; granularity-on-demand, viz. intricate where necessary, coarse-grained where possible
   - Bad: complicated semantic governance

iii. Mix-n-Match (coarse-grained star-alignment with specialised bilateral alignments) for the 70% bulk
   - Good: controllable semantic governance; after central alignment, quickly established bilateral sIOP
   - Bad: slightly more complicated mediation due to double alignment support

Chapter 6

ISO42010 viewpoint on sIOP

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

Chapter 7

Related work

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

Discuss the following papers:

1. M. B. Almeida, C. P. Pessanha, and R. Barcelos, "Information Architecture for Organizations: An Ontological Approach," in Ontology in Information Science, C. Thomas, Ed. IntechOpen, 2018, pp. 1–27.
2. S. Yang, J. Guo, and R. Wei, "Semantic interoperability with heterogeneous information systems on the internet through automatic tabular document exchange," Inf. Syst., vol. 69, pp. 195–217, Sep. 2017.
3. U. 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

Chapter 8

Discussion & future work

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

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

References

Note:

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

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

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.

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

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.

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.

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

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.