**CONSOLIDATING SEMANTIC INTEROPERABILITY IN SOFTWARE ARCHITECTURES**

**Access-and-play semantic interoperability in contemporary architectural paradigms**

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

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

Johan Lukkien, Eindhoven University of Technology, Eindhoven, The Netherlands

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

**Abstract**

Background: *Access-and-Play SIOp is the next glass ceiling in ICT. We can think of two approaches to break through the ceiling, i.e., using either strong AI or weak AI. Strong AI is not yet available, while weak AI, despite its current applications in Semantic Web or ontologies, has not yet been embedded in contemporary software architectural paradigms. Current approaches towards SIOp can be considered accepted folklore.*

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

Method: *Our approach is based on the discipline of semiotics. After identifying shortcomings in MDA and view-based architectural paradigms and subsequently define them as missing concerns, we develop the necessary guiding architectural principles. We finally investigate their consolidation in view-based and model-driven architectural paradigms. [We evaluate these principles by designing a reference architecture and proof its use in SIOp between two software agents.]*

Results: *The semiotic approach/discipline demonstrates/proofs semantics to be the result of a reciprocity between data and the software code that operates on them. The major shortcomings in architectural paradigms to account for semantic interoperability are their negligence of semiotic fundamentals and, particularly, the absence of an explicit ontological commitment that stands at the root of semantics. Therefore, the concern about a semantic loose coupling should be added to the architectural paradigms. The supporting principles are*

1. *semantic transparency,*
2. *(ii) semantic separation of concerns, and*
3. *(iii) explicit computational semantics. In view-based architectures their consolidation implies a new semantic view, while the MDA paradigm requires an ontological commitment on M3. Both paradigms need to include an alignment-based semantic mediation capability.*

Conclusions: *Access-and-play SIOp can be achieved when considering semiotic fundamentals and adding loosely coupled semantics to contemporary architectural paradigms.*

**Introduction**

Never before, data were so ubiquitous, and was managed access to external data so easy. Current ICT is unable to *use* that same external, non-native data as access-and-play service, which hampers business agility. For instance, brandtp, 2018-04- 15 Include small example

Lack of automated *semantic interoperability* (SIOp) represents the next glass ceiling for ICT. Its most disconcerting consequences are time-to-deliver, flat interoperability failures, or even seemingly correct but quite invalid data analysis leading to devastating system behaviour.

Current SIOp implementations are essentially based on establishing a convention on the semantics of the terms that are exchanged during collaboration. Such convention can be considered a semantic monolith, which makes dealing with external (outside the monolith) data impossible, unless again a time consuming (months) semantic adoption process is initiated. Moreover, semantic conventions consider semantic heterogeneity as a bug instead of a feature to achieve semantic accuracy. Still, this conventions-based approach towards SIOp is accepted folklore in ICT. Due to 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”.

Business agility emerges once you only have to access-and-play, namely (i) to achieve SIOp in due time, (ii) with data that are generated outside the own semantic monolith. In this way, software systems, applications or software components – henceforth collectively denoted as software agents – can realise collaboration in due time, when faced, at some point in their life cycle, with the need to exchange data with other however unanticipated software agents outside the semantic monolith. We compare access-and-play SIOp to a bridge overarching a (semantic) gap: a *bridgehead* on each side of the gap, resting on which a *spanning* is constructed that structurally supports the bridge and its traffic, and a *roadway* for the traffic. Finally, *principles* provide the necessary guidance to the architect for the various design decisions that effectively result in the particular bridge over the particular (semantic) gap. Our contributions to consolidating semantic interoperability in software architectures are fourfold, and represented as architectural principles and concerns, as follows:

* *Principles*: Founding interoperability on loose-coupling is a well accepted best-practice in data communication architectures. We realise loose-coupling at the semantic level by introducing principles on semantic separation of concerns and semantic transparency (Section ##), and show how these principles can be operationalised;
* The *bridgehead* concern: essentially, access-and-play SIOp implies that collaboration of software agents becomes independent from their particular semantic implementation, while assessing their semantics remains possible. However, when we are asked to point at the semantics parts in the code of our software agent, we can’t. Based on the discipline of semiotics, we provide for a clear understanding on software semantics and their contribution to the software agent. We show how such explicit notion on semantics fits with other architectural components (Section ##);
* A *spanning* concern: An alignment between both semantic notions (Section ##);
* The *roadway* concern: We provide for a prototypical implementation of a mediator as the necessary component to automatically translate data when transferred between the semantic monoliths of the collaborating software agents (Section ##);

Based on these contributions we defend that a definitive SIOp should aim at an access-and-play potential as upfront “design to collaborate”

Access-and-play SIOp demands a notion on semantics, often denoted in layman’s terms as the “understanding of the data”. Despite the used terms *smart* or *intelligent*, e.g., smart watches, or intelligent autonomous systems, computers are inherently stupid. The notion of understanding is completely alien to them. In fact, it is the IT engineer who performs the understanding of data upfront, and implements the proper response to such understanding in program code. Nevertheless, we do. For instance, when we are asked to explain how we address the grounding problem in the design of our software agent, we can’t; when we are asked to point at the semantics parts in the code of our software agent, we can’t. The same question however about, e.g., its scalability, will render a lecture with adequate references to the underlying architecture. We thus remain at a loss of how to engineer semantics into software agents. However, without a clear understanding on semantics and its contribution to the software agent, we are lacking the bridgehead within the software agent that is fundamental to the semantic interoperability bridge. Hence, the first architectural concern to consider is the nature of semantics in software, and we will address that in the next section.

* The major impediments to the automation for access-and-play SIOp is laid in the nature of semantics itself: The meaning of a term ultimately relates to what it denotes in reality, the so-called term grounding. This semiotics explanation on semantics confronts us with the inevitable separation between languages (software code, modelling languages and natural languages alike) and reality (entities, e.g., things, events and properties of things), also known as the *grounding problem*. We make optimal use of artificial intelligence (AI), particularly “weak AI”, not to solve this grounding problem but to explicate the semantics of data, such that and automate the translation of data between the semantic monoliths of the collaborating software agents.
* Generic mediator / architectural components
* weak AI stu\_

This relation can be deferred neither from the characteristics of the term itself due to its total arbitrariness, nor from an additional definition since that is represented, again, in terms. In information systems, addressing this fundamental distinction is at best extremely narrow (Steels 2012), or not present at all (Cregan 2007). Despite the progress of artificial intelligence (AI), its capability to build some form of conscience and gain even a beginning of an understanding to relate a term to what it denotes in reality, also known as “strong AI”, does not yet exist and is expected to emerge on the long term only, if ever (Xiuquan Li and Tao Zhang 2017). Its counterpart “weak AI” is based on machinery that relies on language only and can therefore never make the step to reality on its own (Scheider 2012). Still, it’s all we’ve got at this very moment and we’ll have to make do with it for the time being in order to achieve semantics and SIOp. We therefore cannot neglect the existence of the grounding problem and its semiotic origins.