Information Entities and Artifact Ontology

Hans Weigand[1], Paul Johannesson[2]

[1] *Tilburg University, Tilburg, The Netherlands*
[2] *Stockholm University, Stockholm, Sweden*

Abstract

Information entities include technical specifications, data records, and software code, but also e.g., musical scores and novels. The ontological status of information entities has been addressed in several fields, including philosophy and linguistics. In our earlier work, we have developed a general artifact ontology and proposed a textual artifact ontology as a special case. In this paper, the textual artifact ontology is compared with some other (recent) applied ontology approaches, specifically IAO and the ontology of representation proposed by Mizoguchi and Borgo. The goal is to explore the added value of the artifact ontology in the information entity debate and to identify any limitations.

Keywords

Information Entities, Artifact Ontology, Representation

1. Introduction

In a recent paper [10], we have introduced a general DSR artifact ontology, rooted in the foundational ontology UFO [3]. Under artifacts, we include symbolic artifacts like software. In this way, we were able to introduce an Information System (IS) artifact ontology not as a project of its own but linked to philosophy of technology in general. Of course, symbolic, or textual artifacts have some peculiarities. Their material form seems less important than in the case of other technical artifacts. They have something – information content – that characterizes them and for which the material form is not essential. The question of the relationship between the paper copy of a novel and the novel as something created by the artist, or, in the IS field, between data and information, is not that simple and has been the topic of research in fields like applied ontology, philosophy, linguistics, and the history of the art. For instance, Thomasson [9] has discussed questions like whether novels cease to exist when their last copy is destroyed, and whether they are abstract entities that exist in a Platonian eternal world or come to existence at some point in time.

Sanfilippo [8] provides a comprehensive state-of-the-art overview of ontologies for information entities. We summarize here the main general applied ontologies and leave out the library science ones. **YAMATO** [5] distinguishes four main classes: Representation Form, Content, Representation and Representing thing. A *Representation* (e.g., quicksort program) CONSISTS OF a *Content* (e.g., the quicksort algorithm) IN A *Form* (e.g., programming language Java) and is to be distinguished from a *Representing thing*, which is an individual material object (e.g., paper, or a computer file) on which the Representation is written. The Representing thing is classified as an artifact, it has a *medium* (medium is a role) and it is a *realization* of Representation. In YAMATO, a Representation is not dependent on the realization. It also allows to talk about Content independent from the Form, which classifies the approach as semi-Platonian. The distinction between Content and Form allows to say that an Italian novel and its Dutch translation have the same content but differ in form.

The **Information Artifact Ontology** based on BFO [1] focuses on the notion of *Information Content Entity* (ICE). A material *Information Bearer* (a copy of the novel) IS BEARER OF *Information Carrier*

(the text of that copy) that CONCRETIZES the Information Content Entity (the novel). Information Carrier is modeled as a Quality that *inheres in* the Information Bearer. IAO also distinguishes an ABOUT relationship from ICE to Entity to capture that an ICE has a reference. There is some unclarity about IAOs claim that ICEs are always veridical, as logical notions of truth and falsehood are not worked out. IAO acknowledges that different agents may have different interpretations of what the ICE is about, so the question is whether the ABOUT relationship is in fact a derived notion. In contrast to YAMATO, IAO does not distinguish between form and content of the ICE. IAO can be called a textualist approach that holds that a novel is equal to its linguistic structure.

In **DOLCE**, *Information Object* ISA social object (ISA non-physical object). Dolce distinguishes *Description* from Information object. Information object is ORDERED BY Description, when Description means a language that is used to represent the Information Object. It also EXPRESSES a Description, when Description means a narrative, or content of the Information Object. Furthermore, the Information object is REALIZED by *Physical Representation* and ABOUT an Entity.

A recent work of Mizoguchi & Borgo [6] aims to deepen the notion of **Representation**. Like in YAMATO, Representation consists of *Form* and *Content*. What is new is that their proposal distinguishes a form realization and a content realization. In our interpretation, the motivation is as follows. In the cases of a procedure (a software program, a recipe, or a music score), it sounds strange to say that it is *realized* by the representing thing (the text of the procedure). The procedure is realized by the actual performance. The former is a partial realization at most; there is something in the representation that still awaits realization. To account for that ontologically, the representing thing is not the realization of the representation anymore, as in YAMATO, but only the *form* realization, while the *content* realization of the procedure is its execution. In the same vein, the content realization of the music score is the music. For the novel, there is no performance (reading does not count as performance, it is claimed), so it has an empty content realization. This distinction in two kinds of realization looks like a neat solution, but it comes at a price – a shift in the meaning of Form. The Form is no longer the format how the content is represented/encoded (e.g., in Java), but it becomes the content (encoded) in a form (e.g., the Java quicksort program). If Form would be the format only, then the Form realization of all Java programs would be the same – Java. So, Form must be redefined. At first sight, it is not immediately clear where the "old" Form has gone, but later the authors refer to *methods*: a representation assumes that there is a method of encoding the content into the form. The Java language can be viewed as a method. It seems to us that the method concept contradicts the earlier statement that an ontological representation object is "necessarily composed of two ontological parts" (form and content). Another problem is that on the one hand, the representing thing is called the Form realization, and on the other hand it is defined mathematically as the sum of medium and representation (where representation is form *and* content, as we remember).[2] It can be highlighted as a positive point of the proposal that the functional or pragmatic nature of representations comes into the picture: representation not only denotes, but also specifies. There is a performance aspect that is not found in the other information entity ontologies.

The objective of this paper is to work out the Artifact Ontology to information entities and to compare it with the above-mentioned existing information entity applied ontologies. In particular, we ask whether Artifact Ontology is able to capture essential features of information entities and what the added value is of an artifact approach with respect to these approaches that are not using an artifact concept in the sense that ontological relationships of the artifact concept are inherited by the information bearer/representing thing. The structure of the paper is as follows. Section 2 recaptures the artifact ontology and how it represents some typical kinds of information entities. In section 3, we make a comparison with IAO, YAMATO and Mizoguchi/Borgo. Section 4 is the preliminary conclusion.

2. Artifact Ontology

In this section, we summarize our general Design Science ontology of artifacts (DS-AO) and its specialization for symbolic artifacts [10]. The ontology is grounded in the foundational ontology UFO

---

[2] The use of mathematical sums in the formalization is not lucky, we think. It obscures the ontological relationships more than that it reveals them.

[24] and modeled in Onto-UML and was developed to formalize the artifact concept in Design Science Research in the first place. In 2.3., we present how DS-AO would deal with information entities such as music scores and novels.

We start with some basic definitions. A **Technical object** is a *non-agentive physical object* (individual) that is intentionally created, usually for practical use. A technical object, as any physical object, has *capacities* and these are *realized* in its use. Some of these capacities are the result of the creation and not present in its source material. A **Technical Artifact universal** like "the Diesel engine" is a type that has a **design** specification consisting of a (at least one) *make plan*, a (at least one) *use plan* and a *capacity specification*. A particular physical object is an artifact instance of a technical artifact universal if its structure conforms to the make plan, it manifests the specified capacities, and its purpose corresponds to the use plan. So, note that not every technical object is an artifact instance. We acknowledge the existence of a technical artifact universal only if it has instances. During the design phase, the technical artifact universal does not exist yet, there is only an intended artifact universal. An artifact universal usually has many **variants**.

2.1. DS-AO design ontology

Fig. 1 gives a global overview of the DS-AO ontology. Engineering distinguishes between inner design (internal structure) and outer design (external behavior). In DS-AO, we consider both as essential. The capacity attribution is the interface between the inner design context and the outer design context. On the one hand, the capacity of the artifact instance (stereotyped as mode, more specifically, a disposition) is the outcome of the make event and on the other hand, it is realized in the use event, resulting in a use effect. The use plan is not decomposed here but typically includes the *use condition*, a reference to the *use practice* and the way of working (*method*). The key components of the make plan are the *component* list, the *structure* (how the components are combined) and the *method* (in which steps).
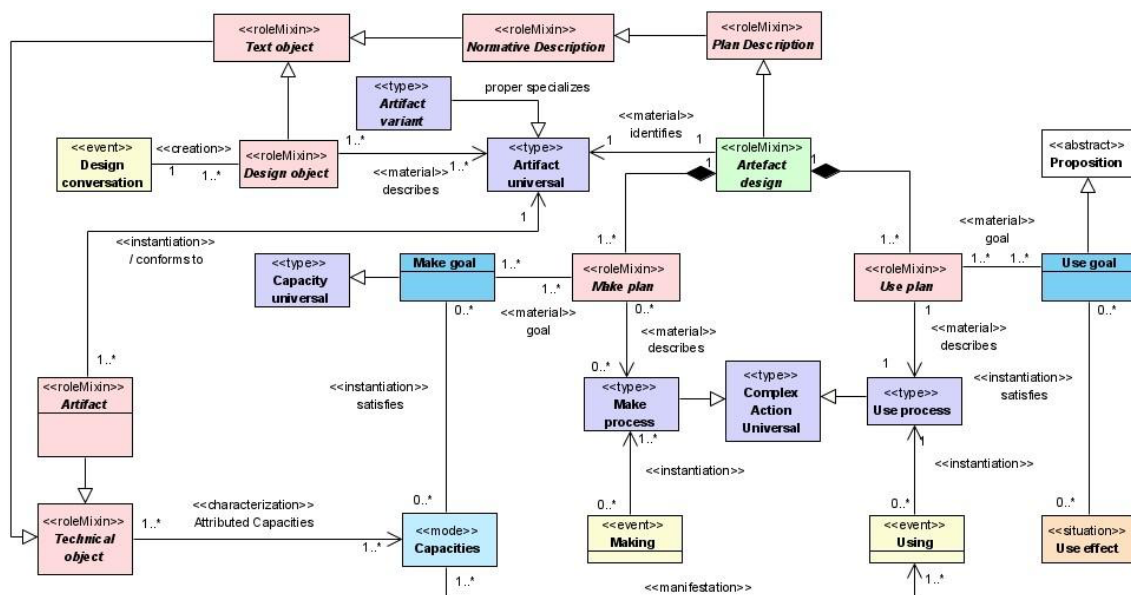

Figure 1: DS-AO - design ontology excerpt

Use plan and make plan are normative plan descriptions (UFO-C) that make up the artifact design identifying an artifact universal, e.g., the Diesel engine or the neural network, here stereotyped as an OntoUML *type*. The artifact universal evolves over time when the design is adapted and usually exists in many *variants*. The artifact universal and its design are modeled by means of *design objects* created and adapted in the *design conversation* between designers and users. Typical examples are drawings,

conceptual models and specifications. Design objects can be informal or based on a formal (model) language. They are usually not complete specifications but partial models[3].

2.2. Symbolic artifacts

A **text object** such as the blue book on my desk is also a *technical object*. Text is to be understood in a wide sense, i.e., a variety of information encoding modes (such as alphabetic text, program code, music score). Any text object has a physical basis (e.g., paper, disk block), a structure (of sign components), and an intended use. In the Saussurean tradition of human language, a sign is relating signifier to signified. The text object is a signifier, and the signified corresponds to the use effect that is, the evoked concept in the mind of the Hearer as the manifestation of the signifier's capacity. The capacity of the text is precisely this: the ability to be read and, ultimately, to evoke certain concepts. These concepts create a mental model in the Hearer (Figure 2). In other words, the capacity is what is traditionally called the text content.
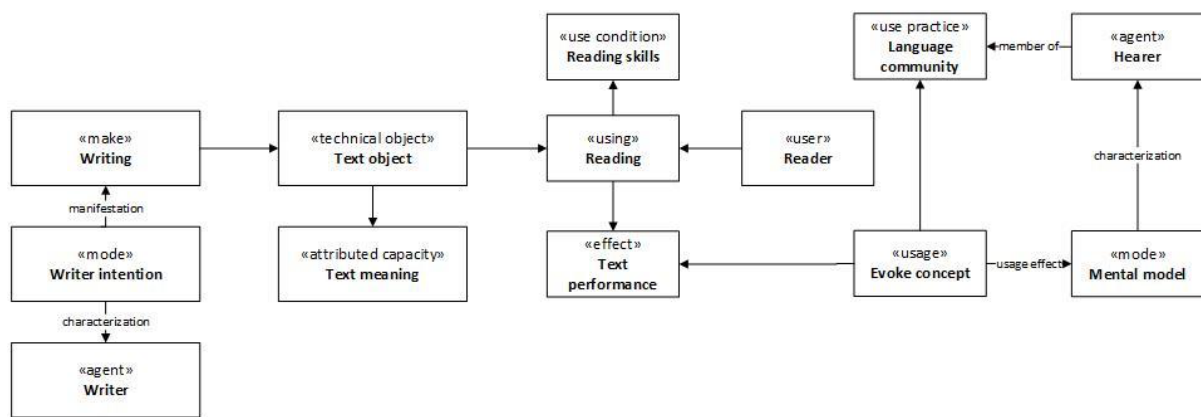

Figure 2: Text as a technical object in a language community

Natural language is not a technical object[4], but a text object is, based on writing technology. As with any technical object there is a designer/author intention (the function) that must be grasped by the user/reader if he wants to use it. The use can be, for instance, to run away, in the case of a danger sign. This is a clear case of Wittgenstein's dictum that meaning is use. In the case of alphabetic signs, the text is used (that is, read) by someone which subsequently changes the mental state of the Hearers (possibly, the reader himself). The effectiveness of the text object in achieving that goal depends on the shared language and shared education in reading/writing. This is a use condition without which the text capacity cannot manifest itself.

If the text object is conforming to an artifact universal, then the latter is a symbolic artifact. Text objects can be copied easily, but the existence of multiple physical copies is not a *sufficient* criterion for positing a symbolic artifact universal. There must be an artifact design that is defined in terms of a class of situations, not a specific one like in the case of John's email yesterday.

2.3. Information entities in DS-AO

We now consider how DS-AO deals with the information entities that were discussed in Section 1.

---

[3] The Artifact design is a description that is assumed to *identify* the Artifact universal. In contrast, the design objects (e.g., a bunch of UML diagrams) typically *circumscribe* the Artifact universal and in doing so, *approximate* the design. Note that in many cases, we have only design objects, and the Artifact design remains implicit. In Design Science, the aim is to have an explicit Artifact design.

[4] DS-AO makes a fundamental distinction between written text and speech. Clearly, speech is not a technical object. To account for speech, we would need a linguistic ontology, complimentary to the artifact ontology. Tentatively, we posit that it should be centered around language acts (locutionary, illocutionary, …) and a language situation in which these acts inhere - a social relationship.

- like in all information entity ontologies, the physical information bearer is related to but distinct from the information content. In DS-AO, this is the relationship between a text object and the capacity that inheres in it (instance level), and the relationship between the artifact design and its capacity specification. Normally, the former complies to the latter, but the capacity of an instance may deviate slightly (for instance, an ink spot or an author signature in a book). The distinction between individual and design is usually not made in information entity ontologies but cannot be ignored.

- in the information entity examples of Section 1 (musical score, novel, procedure, also software program), we are dealing with artifact universals. These are designed for being instantiated and used in multiple contexts. Artifact universals have a make and use plan and are designed by means of design objects – this also holds for these kinds of information entities.

- in the case of a *novel* such as Dante's *Comedia*, the novel/poem is an *artifact universal* under the category "novel" that is identified by its *design*. The designing happens in the writing of the novel, resulting in a manuscript (*design object*). The publisher is responsible for the production of the copies (*artifact instance*), that is, *technical objects* (book, e-book) with a content (*capacity*). The content should conform sufficiently to the design, otherwise it is a misprint. The novel is intended to be read by a language community with some shared background (*use condition*). Its use is to induce a reader experience that corresponds to some extent with the novelist experience. The many translations of the *Comedia* are artifact universals themselves that stand in a *variant* relationship to the original. Recall that the design activity is rooted in a design conversation between designers and users, and so we agree that novels and their translations are socio-cultural entities in the sense of [4,9]. However, we keep a distinction between the novel as artifact and private or public interpretations *of* the novel (like Goodman & Elgin [2])[5].

- in the case of *music* such as Beethoven's 5th symphony (5hB) – or opera, dance, drama, for that matter – we claim that the music is not an artifact (neither the sound nor the performance), but the music composition as designed by Beethoven clearly is. The music composition can be analyzed along the lines just described for the novel, so as an artifact universal with many instances (individual music score pages). The distinction that Figure 2 makes between Reader and Hearer corresponds to the difference between performer and listeners. It is hard to say whether the score shows the performer the music to be realized or the actions she must take to realize it. The oldest examples of music score showed the strings to be touched. Nowadays, the modern music score, the music instruments and the music education are so much intertwined that the score is functioning as a procedure, although it can be argued that there is an iconic relationship between the musical notes and the music sounds. We prefer to conceptualize music as more than sound patterns, namely as the interaction of performers and listeners mediated by sounds. A jazz performance is a musical interaction that does without a composition, so music is not existence-dependent on compositions. The musical performances of 5thB together may be said to form a type [7] but note that the relationship between the music score and the performance is a normative one – we can call it realization (see below). The performer may deviate from the score, intentionally or by mistake. If the deviation is too big, the performance cannot be considered use plan compliant.

- a *written procedure* is analyzed in the same ways as the music composition. However, many procedures are about making a technical object. In that case, the *use plan* of the procedure coincides with the *make plan* of the technical object. We do not consider the actions performed when following the procedure to be artifacts. In DS-AO, they are *use events* that should conform to the content of the procedure text object.

An *alphabetic letter* is a mini artifact universal, e.g., "the" letter A. Each letter occurrence stands for the use of the letter as part of the text structure. The letter instance is a small *technical (text) object*, with a material base and a *capacity*, namely, the possibility to pronounce it. The letter universal has many *variants* (fonts). The capacity must satisfy the *design*, a specification of the format (and sometimes of the drawing sequence in time – the make plan). The design is approximated by means of a *design object*, in this case some standard letter drawing of a font designer. The occurrence may not satisfy (or not anymore) all design requirements, when a human reader or a character recognition

---

[5] Thomasson [9] disagrees with the textualist approach of Goodman & Elgin and identifies a literary work with its interpretations. However, she distinguishes the literary work from its text, and from its composition (the manuscript), so there is at least basic agreement on the ontological distinctions.

program tries to recognize it as a letter. The ontology distinguishes between a concrete letter on paper (text object), its potential for interpretation (in DS-AO, its capacity), and the letter as design specification. Characteristic for an alphabetic letter is its pronunciation that is captured by its *use plan*. In practice, pronunciation is somewhat more complicated and often applies to combinations of letters.

  - A *programming language* like "the" Java language is an artifact universal basically consisting of a set of constructs (such as "if", "=") that function as minimal artifacts and "make plan" rules to compose "a" Java program specification. *A Java program specification* instantiates *"the" Java language* to design/specify an *algorithm or program* (artifact universal). A Java program specification has the role of design object for that program. The individual *software executables* (made by automated compilation) instantiate the program. The executable contains machine instructions and is used as an instrument by the *machine or virtual machine* (another artifact instance) to execute the actions that change the internal memory of the machine (according to the use plan of the designed program). There is more to be said about programs and programming languages but as they are not worked out in the other information entity approaches of Section 1, we leave it as such.

3. Comparison
3.1. IAO

It is relatively easy to map the IAO concepts on DS-AO. The text object (artifact instance) corresponds to Information Bearer, and its textual capacity to Information Carrier – both latter two are a kind of Qualities/Modes. Information Content Entity corresponds to symbolic artifact universal. However, IAO makes no distinction between the artifact universal, its design specification and its design object, as DS-AO does, and does not (explicitly) recognize artifact variants. It also does not distinguish, it seems, between a text object that instantiates an artifact universal and ones that do not. In some papers, IAO distinguishes an Information Structure Entity for the *format* of a representation, e.g., JPG. In DS-AO, a textual object is typically composed of atomic textual objects, like in the case of a text written alphabetically. These atomic elements correspond to the ISE.

3.2. YAMATO

The Representing Thing, which is classified as an Artifact in Yamato, corresponds to the text object (artifact instance) in DS-AO. It is the realization of a Representation, a semi-abstract entity consisting of Form and Content. This Representation corresponds roughly to the artifact universal and its design, where the Content/Form distinction corresponds to the difference between the artifact universal and its make plan (or one of its make plans). However, when it is said that Representation forms are for instance natural or programming languages, then confusion arises between the make plan language – in DS-AO, the atomic components it uses – and the composition of these components that form a specific make plan. Compare the difference between the Java language and a Java specification of Quicksort. We recall that in DOLCE there is a similar ambiguity in the notion of Description, but there, the two meanings are differentiated by the relationships to Information Object (ordered by vs. expresses). YAMATO assumes that Content can exist without a single carrier, while in DS-AO we require an artifact universal to have at least one instance in order to exist. On this point, DS-AO is less quasi-Platonic and more realist. The way YAMATO expresses that the Representing Thing has a medium role goes against UFO, but there is agreement with DS-AO on the difference between the text object as the physical object produced and its artifact role.

3.3. Mizoguchi/Borgo

As we remarked in the above, the extension of Mizoguchi and Borgo [5] on Form realization vs. Content realization is interesting by drawing attention to the performance aspect of representations. DS-AO promotes an artifact perspective of information entities, and so can only welcome attention for their function or *telos* in use. Still, it is not easy to map the proposal to DS-AO. At first sight, it seems that the Form realization relationship corresponds to the relationship between artifact design and instance.

As we have seen, "Content" is a normative description in the case of music or procedure that is realized by sounds or actions. In DS-AO, the printed procedure has a capacity to support a user in producing something or "realizing" some performance. The performance takes place in the use context. For that reason, it can be argued that "Content" corresponds roughly to the use plan in DS-AO. In DS-AO, the use events that *comply with* the use plan are the events that *manifest* the capacity. Note that this holds not only for procedure artifacts but also for hammers and ships. However, there is a difference between the former symbolic artifacts and the latter. A hammer object does not explicitly say how it should be used, whereas the procedure object is doing just that. It is an interesting distinctive property of information entities that we also recognize in software artifacts and cyber-physical applications that they can steer their use – in DS-AO terms, that they (can) incorporate their use plan. The term "realization" that is used by Mizoguchi and Borgo is quite appropriate when talking about use events. At the same time, one can say without stretching that a text object (or more precisely, the make events that result in the text object) realizes the make plan. For that reason, we conclude that Form realization corresponds with the make plan to make event relationship. It occurs to us that DS-AO can account for the notion of Content and Form realization, in a way that is fully consistent with the artifact perspective, because this perspective already makes a distinction between make plan and use plan. One difference is that we consider the reading of a novel also as a use event, so we do not agree that novels have no content realization. In the case of design objects, DS-AO distinguishes between the model language and the artifact that is described by the design object. Another difference is that in our interpretation, both Content and Form are normative, so it is not correct to call one a description and the other a specification. In summary, we claim that the artifact perspective not only captures but also gives a more general and more precise account of what the Mizoguchi/Borgo proposal intends to say.

4. Conclusion

In this work-in-progress paper, we have analyzed how DS-AO compares with current Information Entity ontologies and how it deals with some ontological challenges of "representation". On the basis of our analysis so far, we conclude that the artefact ontology DS-AO is able to account for the core information entity concepts in a more generic way then the current applied ontologies on this topic. It is surprising that the latter recognize the artifact status of information bearers but use it only as a classification. At least, it would be good to indicate how elements of the artifact ontology, such as capacity and use plan, relate to elements of the information entity ontology.

Many ontological questions have not been addressed yet. Some approaches may disagree with our choice that speech, music, dance etc. are not artifacts, if only because they are not physical/material entities. However, there are widely diverging opinions on the ontological status of a musical work. Another question is the ontological status of fictional characters.

Information systems are a kind of information entity and so the present inquiry is also relevant for an IS ontology and an ontology of information products and services. Traditional business and accounting ontologies often take the manufacturing company as their reference, or service companies. The question is whether they deal adequately with the specifics of information products. What does it mean to control an information resource? Does Facebook own my personal data?

A final observation that we make is that when we consider for instance a music performance, many different artifacts play a role. The music score is one of them, but so is the music instrument and perhaps the power supply and the speakers in the case of electronic instruments. In the case of a software program, the execution/realization requires a computer device and an interface device. The relationships that exist between information entity artifacts and other artifacts needs to be taken into account in the further development of information entity ontology.

5. References

[1] Ceusters, W, Smith, B. (2015) Aboutness: Towards Foundations for the Information Artifact Ontology. Proc. of the Sixth Int. Conf on Biomedical Ontology (ICBO) CEUR 1515.
[2] Goodman, N., Elgin, C. (1986). Interpretation and identity: Can the work survive the world? *Critical Inquiry 12*, 564-575.

[3] Guizzardi, G., de Almeida Falbo, R., Guizzardi, R.S.: Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology. In: Proc. CIbSE, pp. 127-140 (2008)

[4] Masolo, C., Sanfilippo, E. M., Ferrario, R., & Pierazzo, E. (2021). Texts, Compositions, and Works: A Socio-Cultural Perspective on Information Entities. FOUST 2021: 5th Workshop on Foundational Ontology held at JOWO 2021.

[5] Mizoguchi R. (2010). YAMATO: yet another more advanced top-level ontology. Proceedings of the Sixth Australasian Ontology Workshop; 2010: 1-16.

[6] Mizoguchi, R., & Borgo, S. (2021). Towards an ontology of representation. In *Formal Ontology in Information Systems* (pp. 48-63). IOS Press.

[7] Nussbaum, C. (2003). Kinds, Types, and Musical Ontology. *The Journal of Aesthetics and Art Criticism*, *61*(3), 273–291. http://www.jstor.org/stable/1559178

[8] Sanfilippo EM. (2021). Ontologies for information entities: State of the art and open challenges. *Applied ontology*. 2021; 16 (2):111-135.

[9] Thomasson, AL. (2015). The ontology of literary works. In The Routledge Companion to Philosophy of Literature, Routledge; 349–358.

[10] Weigand, H., Johannesson, P., & Andersson, B. (2021). An artifact ontology for design science research. *Data & Knowledge Engineering*, 133, 101878.