Applying principles of Ontology-Driven Conceptual Modeling to the interpretation process

Marcelo Jaccoud Amaral[1,*], Vânia Borges[1], João L. R. Moreira[2], and Maria Luiza M. Campos[1]

[1] *Programa de Pós-Graduação em Informática, Universidade Federal do Rio de Janeiro, Rio de janeiro, Brazil*
[2] *University of Twente, De Boelelaan 1105, 1081 HV Amsterdam, NL.*

Abstract

Effective interpretation of symbolic information is a critical challenge, especially as data grows in length and complexity, leading to an unpredictable or inadequately described surrounding context. Software development becomes a challenging task in the lack of a systematic description of the operating context, resulting in imprecise coding and ambiguous documentation. Interoperability standards, designed to align information meaning in heterogeneous environments, face similar issues. This work uses ontology-driven conceptual modeling to formalize the dynamic nature of interpretation as described by the Peircian Semiotics. It presents a generic model of the interpretation process that, by explicitly affirming its dynamic nature, directly contributes to the development of context-aware metadata standards. This formalization can significantly enhance the design of data models and standards, enabling the proper inclusion of contextual metadata, roles, and their dependencies, even within the limitations of current representation languages.

Keywords

Digital Object, Semiotics, Interpretation, Conceptual Model.

1. Introduction

In order to ensure information is reused in another system with the exact same meaning it was intended at the source, interoperability standards strive to register it with detailed documentation in clear language, and, more recently, with machine-readable semantic metadata. Standards for highly reusable data types, such as those directly supported by programming languages, have already coalesced into a few universally adopted ones, like Unicode [1] for text or IEEE 754 [2] for floating-point arithmetic. Most of these standards succeed in their task because they are designed to encode highly abstract types that are rarely used in their pure form, but rather as primitive types to build more complex and actually usable types.

For example, consider the words *cats* and *dogs*, which in most sentences, refer respectively to animals of the species *Felis catus* and *Canis familiaris*. However, when talking about animals in general, they may refer to the whole families *Felidae* or *Canidae*, which include wild cats like cougars and tigers, or coyotes and wolves. Consider now the meaning of the expression "It's raining cats and dogs". Now the same words convey a completely different meaning, that it is raining heavily. Note that this type of expression occurs in every language, and does not necessarily use the same words: in Portuguese, we would say "Está chovendo canivetes" (literally: it is raining pocket knives). Sometimes they are semantically linked: "Chove a cântaros" (lit. it rains in pitchers) correlates to "It's pouring rain" (from a vase). The context may influence the meaning of a word either subtly or radically. Consider also these two sentences:

[1] *Proceedings of the 18th Seminar on Ontology Research in Brazil (ONTOBRAS 2025) and 9th Doctoral and Masters Consortium on Ontologies (WTDO 2025), São José dos Campos (SP), Brazil, September 29 – October 02, 2025.*
* Corresponding author.
jaccoud@ufrj.br (M. J. Amaral); vjborges30@ufrj.br (V. Borges); j.luizrebelomoreira@utwente.nl (J. L. R. Moreira); mluiza@ppgi.ufrj.br (M. L. M. Campos)
0000-0002-1720-2245 (M. J. Amaral); 0000-0002-6717-1168 (V. Borges); 0000-0002-4547-7000 (J. L. R. Moreira); 0000-0002-7930-612X (M. L. M. Campos)

(a) The box could not pass through the window because it was too large.

(b) The box could not pass through the window because it was too small.

Notice that in (a) the pronoun *it* refers to the box, but in (b), the same word, in the same exact syntactic structure, refers to the window. This well-known linguistic problem is also linked to the context: the target of the pronoun is determined not only by the adjective ending the phrase, but also by the readers' knowledge about the domain of discourse.

This context dependence is not a distinct behavior of natural languages. It is present in computer languages (the plus sign may signify addition for numbers, concatenation for strings or character repetition in regular expressions), visual codes (red means danger in a road sign, spelling error in a syntax verifier, closing in the computer dialog window). Music, dance, and all forms of expression exhibit this same contextual behavior. As information grows in complexity, its dependence on context becomes more pronounced, often in ways difficult to formally describe. We believe the writing of documentation, and more broadly, interoperability standards, may benefit from a more detailed understanding of the interpretation process itself. As in most tasks where humans play a role, it is convoluted, complex, and sometimes impossible to automate. Already plagued by low-quality data, most diligent data managers have concerns about interpretations that may be incorrect, lossy, or imprecise.

In this paper, we aim to address these issues through ontology-driven conceptual modeling (ODCM), seeking a deeper understanding of the elements and mechanisms involved in the interpretation process. ODCM is the activity of capturing and formalizing how a community perceives a domain of interest using modeling primitives inherited from a foundational ontology [3]. It plays a fundamental role, helping us to understand, elaborate, negotiate and precisely represent subtle distinctions in our multiple conceptualizations of reality.

This work presents a generic conceptual model for the interpretation process. It aims to uncover the dynamic nature of interpretation, an aspect often disregarded in static relational schemas that assume data will always be used in the manner prescribed by data publishers. We believe that by using proper metadata to better describe the context in which data should be used, we end up with datasets that are more safely directed to software written for that scenario. To design such models, we need to understand the semiotic role of each property and its dependencies, considering the limits imposed by the representation, either human- or machine-related.

This paper is organized as follows: Section 2 presents background information on the terminology required for understanding this work. Section 3 addresses the interpretation process and presents an associated generic model. Section 4 explores this generic model by applying it to the interpretation of digital entities. Section 5 discusses requirements derived from the categorization of signs for metadata involved in the interpretation processes. Section 6 draws some conclusions and proposes directions for future work.

2. Background

The Semiotics of Charles Sanders Peirce (1839–1914) has been the object of many books and dissertations. The reader may refer to our short introduction in the discussion about the digital object concept [4]. A more detailed introduction may be found in [5], which used the Peircian concepts in his dissertation on Trinitology. We shall revise here the main topics regarding the process of interpretation, which Peirce called semiosis.

According to Peirce [6], the atomic piece for interpretation is the *sign*, which happens when something takes the place of something else in some context. Signs emerge continuously in our brains when cognitive signals from our senses match previously stored information regarding things they may represent. Later in his work, Peirce admitted that this process may also happen in the minds of other animals, or even in machines designed to reproduce the behavior of a mind. We interpret dark clouds as a rain forecast, a red light as a command to stop, a picture as the thing it depicts, and an utterance as corresponding words in a language. Signs usually come in chains: the smell of smoke may be interpreted as something being on fire, which in turn may be interpreted as

danger, prompting us to run away or to seek means of extinguishing the fire. Signs do not emerge spontaneously: they require an intentional act by the interpreter agent. Things that happen mechanically, such as the production of electric signals by a microphone or a movement induced by a relay, occur without interpretation; they are simply physical reactions that happen in response to other phenomena. This is an important distinction to which we shall return later when regarding software.

Signs may vary largely in different scenarios. The well-known Saussurian linguistic sign is dyadic, composed of two parts: the signifier and the signified [7]. The Percian sign takes into account that the meaning may vary in different situations, and models the sign not as having 3 parts itself, but as something that emerges from the conjunction of a triad [6]:

- The *representamen* is the relevant aspect of something that depicts or describes something else, things that in common language we use to call symbols and graphic signs. Peirce specifically avoided the common terminology for this role in order to render it generic: any physical or mental manifestation may act as representation of something else.
- The *immediate object* is the relevant aspect of the thing being recognized. It is not an actual object in its plenitude, which is called the *dynamic object*.
- The interpretant embodies the knowledge of the situation, what links the representamen to the immediate object and the possible resulting effects. This is not the product of the sign, but what determines it. It is also not the interpreter, but depends on his competence.

Recalling the previous examples: dark clouds may be considered the representamen of a sign where imminent rain is the immediate object and the interpretant is the knowledge of the observer that the former usually precedes the latter; the redness of a traffic light is the representamen that signifies the danger of keeping on driving (the immediate object) which is determined by the driving knowledge (the interpretant) acquired by the driver and that results in braking the car. These three elements can be identified in every interpretation step.

It is important to note that the sign does not imply truth or correctness, since misinterpretations are also interpretations, and the interpretant is not limited to logical rules or formal knowledge. Some interpretations, and the actions they induce, are based on beliefs and convictions inherent in the agent. One common effect of a sign that has an assertion as its object is the storing of such information for future needs, changing the interpretant itself. Peirce called this recurrent sign-making process *semiosis* and believed this is how we acquire knowledge and pursue truth.

Another persistent element in Peirce's writings is the three universal categories, an abstract classification that he uses repeatedly in many situations. These trichotomies pervade his writings, and he purposely named the categories in a way that avoids confusing them with more concrete, quotidian categories. In a very simplified way, paraphrasing Peirce's words [8], these can be described as follows: *firstness* is what simply exists in itself, without referring to anything; *secondness* is what it is by force of something to which it is second; and *thirdness* is what it is through two other things that it mediates and brings into relation.

When we apply these categories to the sign model, we derive the three basic sign types: icon, index, and symbol. An *icon* is a sign in which the representamen displays firstness and relates to the object by a property of its own similitude, such as an image representing the thing it depicts. An *index* is a sign in which the representamen references or points to the object it represents, displaying secondness, such as a pointing finger, an arrow, an address, a compass needle that indicates North. A *symbol* is a sign in which representamen and object are bound by an arbitrary relation that mediates between them, displaying thirdness, such as words, ideograms, and all sorts of arbitrary rule encoding.

When the categories are reapplied to each type, we can derive other subtypes. We shall use these universal categories as a template to derive important metadata categories that expose these monadic, dyadic, and triadic valences.

3. Generic semiosis

As signs are very general in scope, a formal model of semiosis must address high-level entities in some upper-level ontology of choice. Because most of them define a description as a static relation, we had to resort to very abstract entities. We ended up choosing the Unified Foundational Ontology (UFO) [9], which is a grounded ontology that provides a foundation for domain analysis in conceptual modeling [10]. The UFO categories and relations enable the construction of a robust conceptual model that helps clarify concepts, facilitates meaning negotiation, and precisely defines the ontological semantics of represented notions [11]. UFO provides us with an adequate model for events and also an entity called relator, which is particularly useful in reifying complex relations like the ones involving a sign. UFO classes are expressed here with OntoUML, an ontology-driven modeling language that incorporates the distinctions underlying UFO into UML class diagrams [10]. It introduces various stereotypes (shown between «guillemets») that correspond to the concepts defined in UFO, as well as grammatical formal constraints that reflect UFO axiomatization [9], which allows us to better point out some analogies and patterns using a more concise notation.

To improve clarity, we have highlighted stereotypes related to UFO categories used in this paper for endurants and perdurants. Endurants are entities that exist in time with all their parts. They have essential and accidental properties and can undergo qualitative changes while maintaining their identity. Before proceeding, it is essential to discuss the concepts of sortal and non-sortal, as defined in UFO. Sortals are types that aggregate individuals with the same identity principle. In contrast, non-sortals aggregate individuals with different identity principles. In sortals, the identity is defined by a single «kind» it instantiates. The kind may have specializations that can be either rigid or anti-rigid. The former is stereotyped with «subkind» (e.g., hatchback car as a subkind of car). The latter classifies only contingently their instances and is stereotyped as «role» when grounded on relational properties (e.g., employee as a role of a person within the scope of an employment relationship). For non-sortal endurants, «roleMixin» defines contingent relational properties for individuals of multiple kinds (e.g., 'customer' for the kind person and organization). Intrinsic aspects of individuals, which depend on them or other individuals, are stereotyped as «quality» (e.g., car color) and «mode» (e.g., symptom). Extrinsic aspects are stereotyped with «relator» (e.g., marriage, enrollment). The «event» stereotype refers to perdurants, which are individuals that occur in time, accumulating temporal parts. They are existentially dependent on endurants that participate in them.

The main subject of a Peircian model of interpretation is the sign. We declared it as a perdurant, or, in OntoUML terms, gave it the «event» stereotype. It is important to stress that the sign, which denotes the meaning of something, is not a static object, but a temporal product of an intentional act happening in a particular situation. Even when a situation seems to repeat itself, the meaning may not be the same. Take, for example, the repetition of a mother's plea, "Go tidy your room!". Each utterance, even if using the same volume, results in a different interpretation. In this case, the same sign can never emerge again at a later point in time, because at least one of the components of the triad — the interpretant — is time-dependent. Even when the triad repeats itself, we consider the generated sign as another event, happening in a different time. This distinction between the triad and the sign is an important aspect of the Peircian model: the sign is not the triad, but emerges from it.

To clarify the roles played by the different components of the sign triad, we could just declare all related agents as participants. However, to better understand how each agent contributes to the sign emergence, we decided to reify the triad, a common modeling practice described in [12]. The result, however, is not a simple relator class. It not only brings together three very different participants, but their roles are characterized by complex dispositions embedded in each of them. Figure 1 depicts this general view and helps us unravel each component role separately. If no multiplicity is shown on an association end, it implies exactly 1. In the following text, expressions in **Capitalized Bold Font** refer to the ontology classes in the diagrams.

An important aspect of interpretation is that, being a process carried out by beings with a partial and imperfect view of the world, it works on this limited segmented information that is available to the interpreter. Peirce called these 'cuts' of the real things, and they apply to the three components, even if they are of distinct ontological natures. We have adjusted the terminology here to avoid confusion, since Peirce himself frequently used more than one term for the same concept.

The **Representamen** is "that character of a thing by virtue of which, for the production of a certain mental effect, it may stand in place of another thing" [13]. It summarizes the qualities of a larger thing, which, for practical purposes, we named a **Phenomenon**. A phenomenon is the object of the interpreter's perception and which has the potential for being interpreted as the occurrence or presence of something else, the signified thing. This is not restricted to physical phenomena, such as pictures or sounds, but also encompasses abstract or surreal ones, such as the shadows of monsters in a dark room.

The same cutting pattern happens with the signified component. In this respect, Peirce himself clearly differentiated between the actual full signified object, which he called **Dynamic Object**, and its qualities that are active in the triad, called **Immediate Object**. It is important to preserve this distinction because the actual dynamic object may be anything in the object of discourse, real or imaginary, and it is never known to the interpreter in its plenitude. We deal with and talk about things we only know in part, either by descriptions or direct inspection (their representamens).

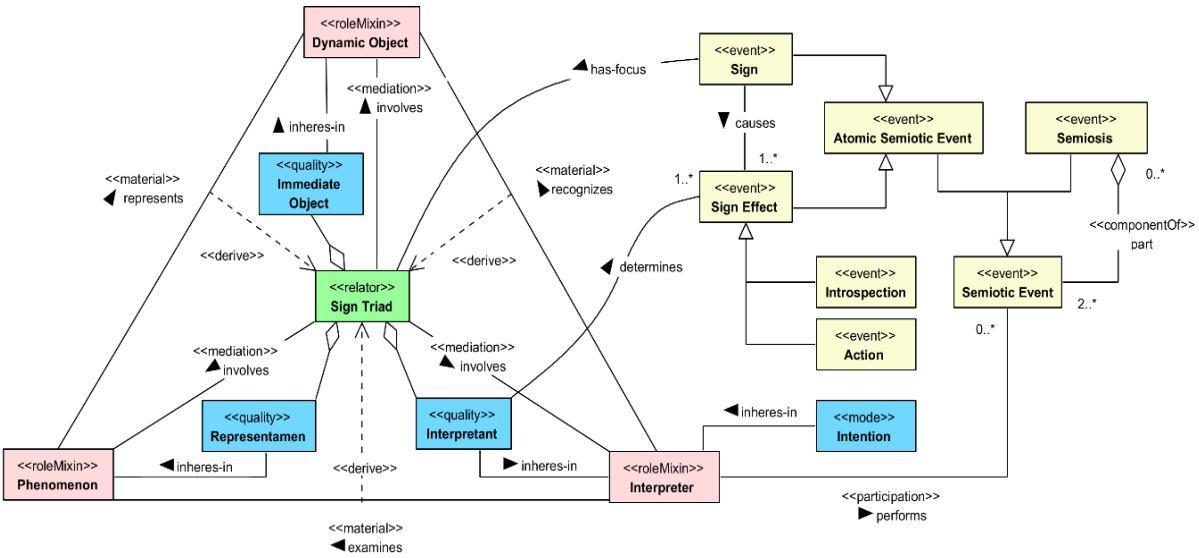

Figure 1: The generic sign model.

The third component is the actual difference from other semiotic models, since Peirce postulated that something can only be said to represent another in a particular context. A static signifier/signified pairing fails to capture innumerable situations where the context changes the meaning. The case of linguistics is exemplary: the same word or expression can mean many different things in different languages, different sentences, or different contexts. And may even signify two things at once. This is not necessarily an error; it may be intentional: a pun only achieves its goal (being funny or outrageous) if two meanings emerge simultaneously. A semiosis must account for all these strange kinds of interpretations. Peirce denoted this contextual part of the triad **Interpretant**, and although he also dissected this complex entity in many subtypes, we shall take a more pragmatic, task-oriented view of keeping the same pattern adopted for the other two components. We consider that, whichever qualities (modes or moments) the interpretant represents, they always inhere in the **Interpreter**, the real-world intentional agent who actually performs the task. These qualities may include:

- Knowledge of the relevant qualities in the representamen, which means acquaintance to important perceivable properties, such as physical modes (e.g. shapes or colors), symbolic codes (e.g. languages, their vocabulary, syntax or schemata), movements (e.g. in dance or sign languages), sounds (e.g. phones, tones, chords and intonation) etc.
- Knowledge of the relevant objects present in the domain, which may lead to the identification of suitable immediate objects.
- Time and spatial context: the meaning of words and gestures may vary radically from one decade to another, or from one place to another, from one user group to another.
- The source of the representamen. For example, the folded hands emoji (🙏) means please, thank you, praying, namaste, añjali mudrā or even a high-five, depending on where you are and with whom you are chatting.
- The principles used to match the representamen and the immediate object and determine the resulting effect of the emergent sign. Peirce frequently called these "habits", to avoid restricting them to logical rules. Interpretation may be based on stochastic functions, good or bad correlations, emotions, beliefs, etc.
- The history of the interpreter's dispositions. The agent's knowledge about a situation may change with time, resulting in different interpretations or actions.

This list may change drastically depending on the situation the interpreter is presented to. An agent who is experienced in translating languages may not have the proficiency to appreciate a complex piece of music or interpret a potentially dangerous situation while driving. The main intrinsic limitation with automating decision procedures using artificial intelligence is related to the colossal amount of information and modeling needed to support even simple scenarios.

Once the **Sign Triad** is formed, by matching the relevant parts in the three components, a **Sign** emerges, which in turn causes or evokes a **Sign Effect**, also determined by the interpretant. There are two basic types of effects: (i) the representamen is replaced based on the resulting immediate object (now in focus), creating a new matching step that will result in a new sign and a sign chain; we named this event **Introspection**, in analogy to the mental process of examining one's own thoughts; and (ii) the sign causes a physical action of some kind. If the resulting object is some assertion of a fact, the effect may be the storing of the information in memory for further use. If the result is a command of some kind, the agent executes it. This type of effect terminates the sign chain. This complex sequence of events constitutes the special process called **Semiosis**.

One should note that artificial symbolic representations, which act as representamens when they are perceived by another agent, are also the result of similar sign chains, in which the objects play different roles. For example, in reading, written characters are interpreted into spoken phrases, and in dictation the heard utterances are interpreted into written symbols. Note that they are not reverse processes; the signs and actions are not the same executed in reverse order, but, if executed correctly, they produce an inverse effect. As any foreign language student knows, it takes time to master both directions and perfect the different skills needed for each language.

This model is purposely abstract in order to be used as a template that must be specialized to reflect elements of the domain of choice. In the next session, we show how this may be applied to the interpretation of data.

4. Digital object semiosis and an application example

Applying Peirce's model to the world of digital computing may arise some doubts and misunderstandings. If the sign results from an intentional disposition from the interpreter, how would a machine with no volition perform a similar task? Souza [14], in her work on human-computer interaction, provides a clue to answering this question. She argues that our common perception of conversing with a user interface is, in fact, an illusion. Instead, the interface's

responses are a direct consequence of how its human designers programmed it to logically react, mimicking anticipated human behavior. So, we are not actually talking to the machine, but talking to the designers via the machine, in the way they intended the conversation to unfold. We can say the machine acts as a delegate to the programmers, automatically responding to user inputs in the way the programmers would react. Also, a computer program does not spontaneously initiate; it is intentionally started by a human, either directly or indirectly. This also holds for very complex interfaces such as AI-driven assistants — there is a large infrastructure behind the interface, carefully constructed to behave in a human-like fashion, but the responses are all driven by the intentional acts of the many humans involved in its construction. So, if a piece of software is built to behave according to the will of one or many humans, we can say it inherits such intentional dispositions from the programmers. The software will always behave in the way it was programmed to behave and will do so because it is intentionally started and used. We can thus say that any interpretation that eventually happens by means of such code is intentional.

An important aspect of the digital environment is that it is very much uniform, since everything is represented by bit-encoded sequences. Because there is no natural source of data, every bit sequence present in the universe of a digital system comes either from some other system or from an analog-to-digital converter (ADC), a device that produces digital data from some analog sensor. For example, a microphone transforms sound waves into an analog signal, which is fed to an ADC circuit that regularly generates data encoded in some digital format like pulse code modulation (PCM). Also, everything that exists in the digital world needs a digital-analog converter (DAC) or similar circuitry that reacts to data stored in some memory and produces a physical reaction. For example, a sound DAC captures octets regularly from a serial interface and generates an analog electrical signal that may be later transformed into sound by a loudspeaker. So, our interpretation model deals exclusively with things that are derived from bit sequences. This is not a conceptual restriction, since in the generic model all components of the sign triad were also informational in nature, but a format restriction: in a digital system, everything *must* be represented by discrete bit sequences.

Porting our generic model into the digital space requires that each concrete class be replaced with some sort of subclass from a superclass we call **Bit Sequence.** To facilitate visualization, Figure 2 illustrates a model with the entities associated with this superclass, highlighting those involved in the process. Note that bit sequences and their subclasses are all of an informational nature, and these bits are always realized in some physical media, in this case, the computer memory. The derived model is illustrated in Figure 3.

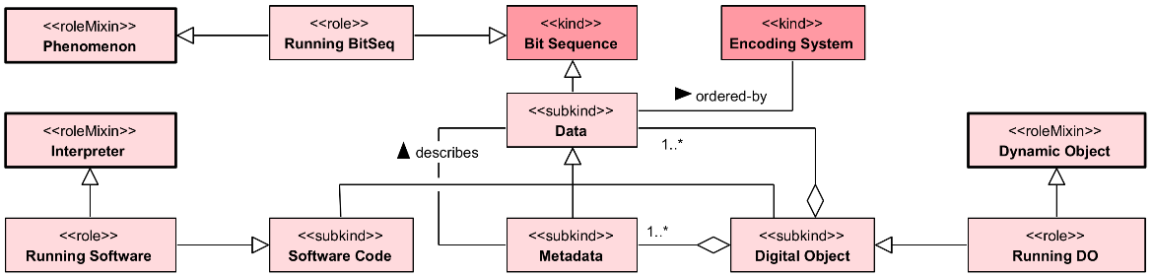

Figure 2: Bit sequence taxonomy and relations.

The definition of **Data** in this ontology is derived from [15], which borrows from DOLCE. It is a sequence of bits ordered according to some **Encoding System**. Although this may be used with all kinds of data, this example illustrates how it would work for a **Digital Object** (DO), which we define as in [4] as *"a finite bit sequence that is identified and has a set of assigned descriptors (also bit sequences) that support or aid in its interpretation"*. These descriptors were represented here by the common concept of **Metadata**, data that describes another piece of data. To simplify the example, we considered only DOs that are data, excluding those that may describe untyped or random bit sequences, which may appear in some scenarios.

Because the computer engine can only deal with bits, everything is a bit sequence that encodes some other entity. This includes descriptions of entities and all sorts of metainformation. Interpretation, thus, is the process of converting between representation spaces with different encodings. For example, in interpreting a program source code within a specific context (text encoding, programming language, syntax), the bit sequence for the string "0x34" is translated into a different bit sequence that represents the integer number *52*. Note that the computer does not actually know what *52* means, because it has no idea of what an integer number is. However, its circuitry knows how to do integer arithmetic on a limited range of integers by manipulating the bits with digital logic dates, and so it is able to do integer arithmetic without knowing what an integer is.

In fact, we have built software that can deal with thousands of other entity bit representations. But the computer can only correctly operate on data that is properly encoded, and representations must be constantly adapted in memory. Consider the example of software itself. It starts as ideas in a human mind, which are then encoded in text, the source code. Once the source code is input to a machine, it becomes a bit sequence, text encoded in some character encoding standard, such as UTF-8. The computer cannot run such a program encoded in text; it needs to interpret the text into an executable form, and it does so using another software called a compiler. The resulting bit sequence is now an object code, but still not ready to be executed. Another piece of software, part of the operating system, needs to copy such code into a specific memory address, adjust boundaries, and start a process to execute the code. The software is now a dynamic sequence of bits that continuously change (the quintessential Turing strip) until the program terminates. We use the same term — software — for all these bit sequences in different encodings, but they are distinct instances with different representations (source and object code) and even different ontological natures (a static plan, the program, and a mutable sequence manipulated by the running process).

Figure 3 shows the relevant parts of the interpretation process that, in different granularities, operate in such encoding transformations. The representamen is the relevant **BitSeq Aspect**s of the **Running BitSeq** that is being interpreted. When dealing with digital objects, the immediate object is the **DO Description**, which is the metadata associated with a particular **Running DO**. The interpretant is the context-aware **Algorithm** that the **Running Software** realizes**.** This triad is here a form of pattern match that results in the sign, the emergence of a **DO Occurrence**. The same algorithm determines which **DO Command** will result as an effect. This sequence of occurrences represents the **Software Process.**

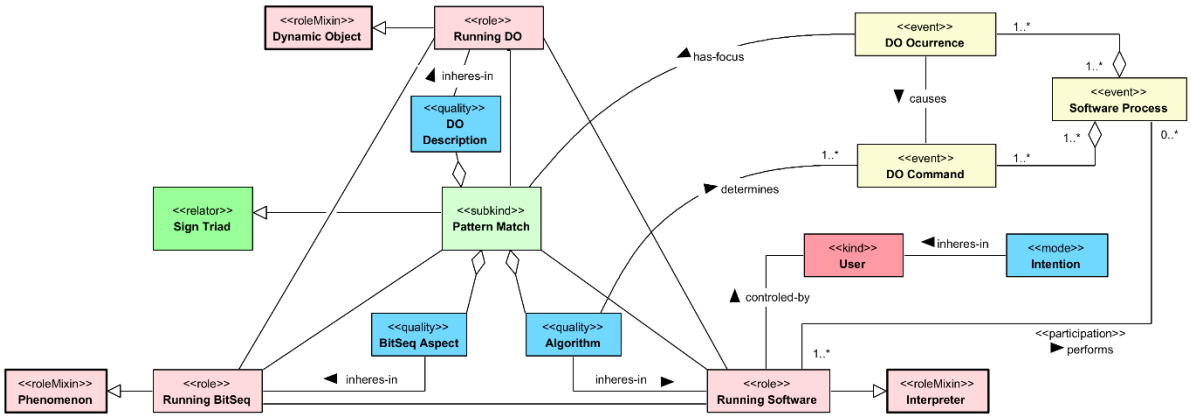

Figure 3: Digital sign model.

To help visualize this model applied to a real system, consider an XML parser, whose task is to interpret an XML-conforming text according to the semantics of an XML Schema. There are two possible approaches: the Document Object Model (DOM) parser [16] and the Simple API for XML (SAX) parser [17], as seen in Figure 4.

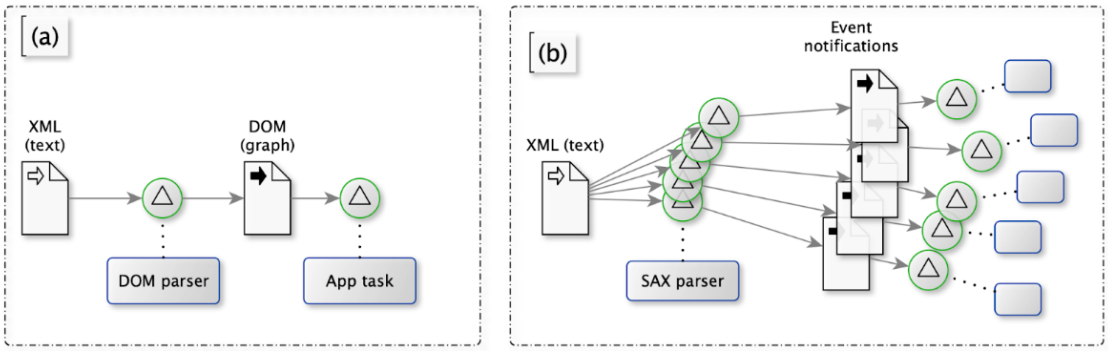

Figure 4: Example of DOM and SAX parsers.

The DOM parser (a) uses only its knowledge of syntax rules to determine and produce a new DOM graph, which is a new representation for the same information. Another routine, which knows nothing about XML, but needs to know the DOM encoding and the semantics inherent in the schema. The DOM sign focuses on the whole XML file at once, producing a single DOM graph, so the application has the whole context of the XML file to work with.

The SAX parser (b), which is used for very big files, translates every structural XML component (starting tag, attribute, ending tag, comment etc.) in discrete events, passing their representations as notifications to particular subroutines that will only be able to work with a smaller context of the XML tree. These are simpler signs, which result in less memory consumption and smaller routines, but are potentially restricted in interpreting the data because of the shrunken context.

In alignment to Peirce's view, parsing has already been considered as a knowledge acquisition method [18]. In a SAX parser, the schema and other previous information collected while traversing the XML tree may be cached to help resolve cross-references or improve the semantic resolution of terms. Every relevant information added to the interpretant should result in a better interpretation.

5. Metadata requirements

For each element of a data structure that requires interpretation, the interpretant must comprise knowledge of its ontological nature and the context for the operation. This information is provided statically by the software code or dynamically by metadata. Knowing how much metadata to provide is important to guarantee a proper interpretation. How much metadata is needed for each type of sign? Not coincidentally, there is a parallel with Peirce's sign typology and universal categories.

The universal categories may be applied to partition domains in segments that are characterized by the self (showing firstness), by something else (showing secondness), or by relating two other things (displaying thirdness). Dozens of such trichotomies appear in Peirce's writings, and many more have been developed by his disciples in many different domains. ***Table 1*** lists a few of a long list collected by [19] that are of interest to our analysis. The last three were used by Peirce to develop a more granular categorization of signs, which we have used to derive requirements for the metadata involved in the interpretation process.

Besides the already seen icon/index/symbol distinction, based on the way representamen and object are matched, Pierce considered how the sign is used: (i) a *qualisign* is based on intrinsic qualities that convey character, possibility, chance, disposition, not facts; (ii) a *sinsign* represents occurrence, reality, factuality, individuality, instantiation, and conveys haecceity and identity; and (iii) a *legisign* displays continuity, generality, relationship and changes in space-time. All symbols are legisigns.

Table1

Trichotomies based on the universal categories

Domain/Topic	Firstness	Secondness	Thirdness
Categories	Monads	Particulars	Universals
Modeling	Attributes	Individuals	Types
Predicates	Internal	External	Conceptual
Metaphysical concepts	Spontaneity	Dependence	Mediation
Relations	Attribute	External relation	Representation
Characterization	Uniqueness	Otherness	Composition
Inference	Emotional/Chaotic	Energetic/Stochastic	Logical/Formal
Signs	Representamen	Immediate Object	Interpretant
Sing-object	Icon	Index	Symbol
Sign use	Qualisign	Sinsign	Legisign
Interpretant mode	Rheme	Dicisign	Argument

The last trichotomy considers what drives the interpretant in the process. A *rheme* represents purpose and focuses on marks or the character of the representamen. A *dicisign* stands for fact, assertion, and, as such, focuses on the object. Only symbols and indexes behave this way, since icons are based on their own attributes. An *argument* expresses reason, rules, habits, and the influence of context. Examples are premises, conclusions, and a sequence of statements. Only symbols may act as arguments.

These sign trichotomies help us identify the type of knowledge required for the interpretant to perform its job, as well as the corresponding metadata. Table 2 outlines the metadata requirements in the case of digital objects.

Table 2

Metadata requirements by sign types

Sign subtype	Knowledge	Metadata
Icon	How bit sequences may be retrieved and compared. The "appearance" of a bit sequence is restricted by its length and the bit order.	None. The hardware is configured to handle bit sequences in a specific mechanical manner, and no interpretation is involved.
Index	How a bit sequence may refer to another one.	Standards for addressing bit sequences. Handled mainly by the operating system and network protocols. However, special standards may be developed for specific scenarios.
Symbol	How bit sequences encode other types.	Binary encoding standards. Formats and packing standards. Local defined types.
Qualisign	How are different possibilities encoded?	Knowledge of the possible arbitrary values an attribute may assume, which are provided by the Mode definitions.
Sinsign	How instances are qualified.	Standards for identifying resources, such as URIs, UUIDs, CRCs etc.

Sign subtype	Knowledge	Metadata
Legisign	How relations are encoded.	Standards for defining relations between objects, including the relations that classify them.
Rheme	How purpose is encoded.	Schemas and ontologies that characterize icons, indexes and symbols.
Dicisign	How facts are stated.	Languages used to describe entities, their types and associations. These affect only indexes and symbols.
Argument	How to reason.	Languages used to describe rules, laws, algorithms, behaviour, beliefs etc. in ways a CPU can replicate.

It is worth noting that the context dependence varies according to the complexity of the sign. On one side, qualisigns, which are based only on intrinsic properties, can only be iconic (nothing to index or mediate) and rhematic (fixed purpose), and thus require very little contextual information. Low-level routines in the software handle most of this low-level encoding, since the possible scenarios for bit encoding and memory manipulation are fixed and cannot be changed. On the other extreme, an argument needs many different standards to be efficiently interpreted, including the contextual information on which the algorithms apply. Some metadata annotation properties which are considered simple, like the Resource Description Framework (RDF) Schema comment[2] or the Simple Knowledge Organization System (SKOS) definition[3] are actually arguments that map from an artificial language (RDF or SKOS) into some natural language used to convey the intention of the author. Any system that needs to interpret such information will need a sophisticated interpretant armed with computational linguistic capabilities. This sort of metadata is clearly aimed at the human reader. Still, there is a lot of research using such information to help modelers in building or aligning ontologies [20, 21] or building natural language interfaces [22].

6. Conclusion and future work

The analysis proposed in this paper has revealed that various factors influence the interpretation of symbolic information encoded in digital systems. Although we automate this procedure with software and employ many methods to assure the quality of its intended execution, every exchange of data to or from another system implies a change of context that needs to be managed. It is not enough to guarantee that encodings and formats are compatible, but also that other contextual information is aligned, such as ontology versions, inference machines and rules. In the case of human interaction, localization parameters such as language and formatting are the same. This is especially important when data is exchanged, not in a neutral format, but one designed for human consumption.

The distinct aspects of knowledge required during interpretation depend on the various components of the sign triad, and this knowledge must be integrated into the running software for the system to provide the intended interpretation. This knowledge may already be embedded in the form encoding and format conformance, but it may also be loaded dynamically, in response to changes in the environment of accumulated knowledge. Proper validation of these new data is crucial to ensure the consistency of the running software. This is especially important with knowledge graphs, which cannot be fed conflicting information, and language models, which cannot receive their own output as feedback.

[2] https://www.w3.org/TR/rdf-schema/#ch_comment
[3] https://www.w3.org/TR/skos-reference/#notes

We hope this ontology helps the derivation of interpretation scenarios for other domains, but especially the case of proper metadata specification for digital objects, which lacks better validation procedures and for completeness and suitability. Incoherent metadata is an omen of incorrect data processing.

As future work, we intend to demonstrate how these principles can be applied to an RDF-based knowledge base and its associated services and show how we can derive Shapes Constraint Language[4] (SHACL) validations to pinpoint lags or inconsistencies within datasets, or problems resulting from the simultaneous use of multiple datasets with incompatible metadata.

Acknowledgements

This work has been partially supported with research grants from RNP, CAPES (Process number 88887.613048/2021-00), and FINEP/DCT/FAPEB (nº 2904/20-01.20.0272.00).

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] The Unicode Consortium. The Unicode Standard, Version 16.0.0, (South San Francisco: The Unicode Consortium, 2024. ISBN 978-1-936213-34-4) https://www.unicode.org/versions/Unicode16.0.0/.

[2] IEEE Computer Society (2019-07-22). IEEE Standard for Floating-Point Arithmetic. IEEE STD 754-2019. IEEE. pp. 1–84. doi:10.1109/IEEESTD.2019.8766229. ISBN 978-1-5044-5924-2. IEEE Std 754-2019.

[3] T. P. Sales. Ontology Validation for Managers. (2014). 312 f. – Universidade Federal do Espírito Santo, Vitória, ES, 2014.

[4] M. J. Amaral, V. Borges, M. L. M. Campos. A Terminological and Semiotic Review of the Digital Object Concept, ER (2023). doi:10.1007/978-3-031-47262-6_5.

[5] A. Robinson, A. God and the world of signs: Trinity, evolution, and the metaphysical semiotics of CS Peirce (Vol. 2). Brill (2010). ISBN 978 90 04 18799 3.

[6] J. Hoopes (ed.). Peirce on Signs – Writings on Semiotic by Charles Sanders Peirce. The University of North Carolina Press (1991). https://muse.jhu.edu/book/41103.

[7] F. de Saussure. Curso de Linguística Geral (Cours de linguistique générale). Portuguese translation by A. Chelini, J.P. Paes and I. Blikstein. 28th. ed. Editora Cultrix (2012). ISBN 978-85-316-0102-6.

[8] C. S. Peirce, The Collected Papers of Charles Sanders Peirce, 8 Volumes, Cambridge, MA. Harvard University Press (1931) — p. 2.356.

[9] G. Guizzardi, C. M. Fonseca, J. P. A. Almeida, T. P. Sales, A.B. Benevides, D. Porello, Types and taxonomic structures in conceptual modeling: a novel ontological theory and engineering support. Data & Knowledge Engineering, (2021). doi: 10.1016/j.datak.2021.101891.

[10] G. Guizzardi, A. B. Benevides, C. M. Fonseca, D. Porello, J. P. A. Almeida, T. P. Sales. UFO: Unified foundational ontology. Applied ontology (2022). doi:10.3233/AO-210256.

[11] G. Guizzardi, H. A. Proper. On understanding the value of domain modeling. In: 15th International Workshop on Value Modelling and Business Ontologies, VMBO (2021).

[12] N. Guarino, T. P. Sales, G. Guizzardi. Reification and Truthmaking Patterns, ER (2018). doi:10.1007/978-3-030-00847-5_13.

[13] G. A. Benedict, What Are Representamens? Transactions of the Charles S. Peirce Society (1985). http://www.jstor.org/stable/40320088.

[14] C. S. de Souza, The semiotic engineering of human-computer interaction. MIT press (2005).

[4] https://www.w3.org/TR/shacl/

[15] D. Oberle. Semantic Management of Middleware. Springer (2006). ISBN 978-0-387-27630-4.
[16] T. Leithead. DOM Parsing and Serialization [Internet]. W3C (2016). URL: https://www.w3.org/TR/DOM-Parsing/.
[17] S. M. Foo, W. M. Lee. (2002). Simple API for XML (SAX). doi: 10.1007/978-1-4302-0829-7_7.
[18] A. Wallis, G. Nelson. Syntactic Parsing as a Knowledge Acquisition Problem. Proceedings of the 10th European Workshop on Knowledge Acquisition, Modeling and Management, EKAW'97, pgs. 285-300. Springer (1997).
[19] M. K. Bergman. A Knowledge Representation Practionary — Guidelines Based on Charles Sanders Peirce, p.143-7. Springer (2018). doi:10.1007/978-3-319-98092-8.
[20] B. Biébow, S. Szulman. TERMINAE: A Linguistics-Based Tool for the Building of a Domain Ontology Proceedings of the 11th European Workshop on Knowledge Acquisition, Modeling and Management, EKAW'99, pgs. 49-66. Springer (1999).
[21] S. J. Ali, V. Naganathan, D. Bork. Establishing Traceability Between Natural Language Requirements and Software Artifacts by Combining RAG and LLMs. Proceedings of the 43rd International Conference on Conceptual Modeling, ER 2024, pg.295-314. Springer (2025).
[22] A. Oliveira, E. Nascimento, J. Pinheiro et al. Small, Medium, and Large Language Models for Text-to-SQL. Proceedings of the 43rd International Conference on Conceptual Modeling, ER 2024, pg. 276-294. Springer (2025).