

Atom model

Preliminary considerations

There must exist an underlying model for representation of knowledge regardless its representations in concrete language syntaxes or its serialization (example: RDF) . But it also must allow to inter operable model representations to co-exist given this underlying common representation.

Atom(s) should be regarded as 'units of knowledge'. Given Peirce's semiotic semantics there are three relations and two operators for basically state knowledge. Then they derive into three more primitive operators.

One must be able to say, for example:
`x.typeOf(y)`, `x.nameOf(y, z)`, `x.valueOf(y, z, w)`

Then, due the arrangement of Atoms, given previous relations or using them to define structure, we can populate them considering, for example, a parent-child relationship between atoms. This could mean that in the atoms object model the parent of an atom could be regarded as its type, the child of the atom its values, or, being an atom, it names its children regarding its parent.

Then, another useful consideration would be that of regarding this hierarchical structure as a list, with head and tail semantics (like functional programming lists) while this arrangement being useful to infer more relationships.

Like the diagram below, we could consider Atom(s) like being one of six kinds (or roles) in a knowledge structure. First, we have semiotics related name (sign), value (object), and type (concept), and they allow us to express classes (set of values sharing the same name), instances (values grouped by same type) and contexts in which resolution of ambiguous meaning could be achieved.

Something to note is that, being Atom(s) instance of one of these six kinds, they can play the 'role' of any other kind of Atom entity in an statement, for example, of a Context, where a Name and a Type are expected.

Term definition, from, lets say, given a set of primitive terms (those terms who form part of a binary pair that is inseparable in terms of defining one on terms of the negation of the other) can be achieved given this primitive pairs and composition through semantic functions. Populating the knowledge base a priori with this knowledge is fundamental to make it work with more involved lexicon.

This model, through aggregation, should compose, for example, instances of a Value for a given Name, as many Types it could be evaluated as. Or, given a Class, the instance would exist for the same class 'Name' as many times there are values for that class.

Diagram 1.

