Towards Formal Knowledge Intelligibility at the Semiotic Level

Statement of interest for the ECAI Workshop on applied semiotics (ASC-2000)

Jérôme Euzenat[1]

Abstract. Exchange of formalised knowledge through computers is developing fast. It is assumed that using knowledge will increase the efficiency of the systems by providing a better understanding of exchanged information. However, intelligibility is by no way ensured by the use of a semantically defined language. This statement of interest explains why and calls for the involvement of the semioticians for tackling this problem.

1 INTELLIGIBILITY

With the expansion of computer networks, the exchange of elaborate knowledge, information and data is developing. To achieve a more precise expression, the knowledge can be represented in a formally defined language (which can range from XML — structured, general but without semantics — to knowledge representation languages — structured, semantically characterised, but with a narrow scope of expression). Between the emitter of a piece of knowledge and its receiver, the computer can reorganise, manipulate, filter, combine with other knowledge chunks and ultimately present it to the reader.

A transformation is a computational way of generating representations from other representations (not necessarily in the same language). Transformations are used for putting representations together or generating specific representations for a particular need. For instance, it can be useful to define a transformation which delivers a documentation to a customer but which hides some information from the initial source (e.g., if a design document includes cost study, it is not advised to communicate this information to sub-contractors).

Conversely, when the elaboration of a representation is a collaborative and continuous process (in the concurrent engineering framework), it is necessary to apply treatments which do not challenge the current stage of development. It is thus relevant to preserve the content.

However, the intelligibility (being the proper understanding of the knowledge by people) of the resulting message is not ensured through transformations. Ensuring the suitability and intelligibility of knowledge for the users, requires to develop an abstract understanding of the preservation of intelligibility through transformations.

There are several levels required for understanding a message in a formal language (here described by the coincidence of meaning for users but which can be established through explicit translation of the meanings):

lexical (or terminological) in which the users must be sure that they use the words (or the terms) with the same meaning. This level is explicitly taken into account, and strengthened, by work on shared ontologies [3];

semantic in which the users must be sure that they assign the same meaning to the constructs of the language used. This topic is well understood in computer science (see §2);

semiotic (or pragmatic) in which the users must be sure that they use the same rules of interpretations of the sentences.

The last level is not necessary for dealing with computers because they can be told to use particular rules usually specified with the semantics. But, when the computers have to interact with human beings, the problem of telling them exactly what is necessary for them to understand what is meant cannot be left to semantics (see §4).

The remainder presents what is used in computer science for providing intelligibility at a semantic level and the insufficiencies of this approach when human beings must interpret the representations taken as signs (§2). Then three examples of these problems are given (§3) in order to explain what computer science would expect from semiotics as a ground for a computational semiotics (§4).

The presentation is very informal and may not be accurate for semioticians, but its goal is to explain the actual problems from the computer science point of view in order to gather comments from the semiotics point of view.

2 THE COMPUTER SCIENCE ANSWER: SEMANTICS

Some people think that by using identifiers with meaningful (for them) names instead of id#356, id#873, they "put more semantics in the system". For the system, in fact, the result is the same. Moreover this do not do anything with semantics: all this is syntax. In fact, the semantics of a language has to be given in relation with the meaning and this cannot be done by syntax.

Formal semantics has been developed for logic languages [7] and adapted to computer programming languages [5] and knowledge representation systems [6]. Logicians have developed powerful ways to provide the semantics of a language by model theory. Roughly, they provide a structural way to interpret a language in a domain. From a domain D, an interpretation function I is defined over the language to the domain. Ultimately, the sentences can be interpreted over the booleans (i.e. they are true or false), but this is not an obligation. Instead of assigning one interpretation to a particular term, the scheme describes *the way* to ascribe meaning. Hence, for a particular

[1] INRIA Rhône-Alpes, 655 avenue de l'Europe, 38330 Montbonnot Saint-Martin (France), http://www.inrialpes.fr/exmo
Jerome.Euzenat@inrialpes.fr

© 2000 Jérôme Euzenat

language and a particular domain there can be many interpretations.

A model, of a set of expressions in the language, is an interpretation is consistent with all the expressions (sometimes more precisely, it can be an interpretation that interprets each expression as true). The meaning of a term, is then the intersection of its interpretations in all the models and a sentence is said to be a consequence of this set of expressions if it is true in every model.

The interesting feature is that, if someone has in mind a privileged interpretation of a set of expressions, it must be a model. Then every processing on the language, which is based on all the models, is coherent with the user's interpretation.

This scheme is suited to the use of a system by one user because it is expected that (s)he interprets in a coherent way the terms (identifiers) (s)he used. We investigate the embedding in XML of the semantics of the language used. For that purpose, a language has been defined for associating set-based model-theoretic semantics of the element defined in the document type description. Thus, the form can be preserved as much as possible (form is relevant to human understanding) and interoperability is ensured through the availability of the semantics (correct computer treatment mainly depends on semantics). The description of the semantics allows tool-makers to understand the semantics of the mark-up or to check the correctness of their tools.

However, when several users communicate, the understanding cannot be ensured by the semantic embedding only. Work developed in the field of consensual ontology construction could help to solve the problem of term interpretation, but the problem of construct interpretation remains.

The problem can also be found in purely artificial systems. Agents are autonomous programs able to communicate together. Of course, for being interesting, the agent approach must apply to agents which have not been programmed to work together. The agent community has thus started to build Agent Communication Languages based on the pragmatic speech-acts theory. It soon appeared that the constructs of the language were not clear enough for computer treatment. So people started to define a semantics for these speech-acts. But the cleanest the semantics, the less generic the languages. They lack the versatility of natural languages which allow one to adapt to new situations. The current trend in ACL is the design of languages with a broad meaning and the use of protocols (i.e. rules of interpretation) for correctly understanding the meaning of the sentences in the context of a conversation [4].

3 EXAMPLES

Below are three examples in which the lack of a particular treatment of the interpretation of a representation by individuals prevents intelligibility.

Classes

In a context related to knowledge formalisation, a user can express knowledge under the form of class hierarchies and first order clauses and then communicate it by using an interoperability language. But if this last language expresses all the knowledge with clauses (though preserving the semantics of the assertions), the initial user will hardly recognise (and hardly understand) the semantically equivalent result (see figure 1). Hence, when a transformation translates between formal languages, good understanding cannot be ensured by meaning preservation.

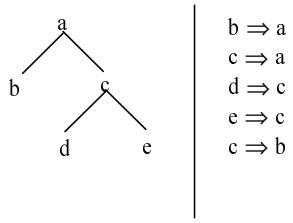

Figure 1: do these representations mean the same?

Marks

In the context of an electronic reviewing system for a conference, an author is provided with the feedback "Acceptable" or 6/10 about a paper (s)he submitted. These comments are a poor way of rendering the choice of a reviewer among "9. Wonderful, 8. Excellent, 7. Good, 6. Acceptable, 5. Fair, 4. Average, 3. Weak, 2. Bad, 1. Very bad, 0. Dreadful" (see figure 1). In order to be understood, the potential signs comprised in the messages should be transmitted with the rules of interpretation (or their enunciation context).

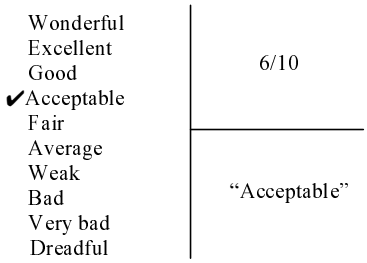

Figure 2: do these representations mean the same?

It is noteworthy that the computer which performs the transformation does not loose any information: it can come back to the initial situation.

Order

In this example, inspired from [2], a set of elements of an unordered domain (or a partially ordered structure) must be displayed. If this set is presented as a list on a HTML page (or a paper), it is not obvious to the target user that the list is *not* ordered. This is to the point that it is usually resorted to tricks such as ordering elements with a meaningless order (e.g. alphabetical) or to mention "in no particular order".

The opposite problem arises in computer science when people add redundant links in inheritance hierarchies because they forget that they are non transitive representations of a transitive structure.

 © 2000 Jérôme Euzenat

4 IS COMPUTATIONAL SEMIOTICS THE ANSWER?

In all the examples above, there is, beside a strict preservation of the semantics, a deficiency in understanding due to a semiotic mismatch, i.e. a failure to apply the adequate rules for sign interpretation.

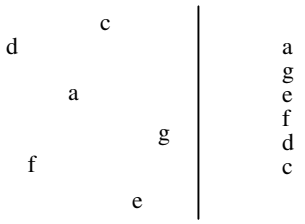

Figure 3: do these representations mean the same?

Solving these problems might require some semiotic treatment for ensuring the meaning reconstruction. It would require an apparatus for expressing and understanding the rules of sign interpretation. This treatment would come as a complement to the sheer semantic treatment used in the knowledge representation area.

The semiotic mismatch can be stated in several ways. There can first be mismatch in the understanding of the computer when it acquires information from people who assign (possibly incorrect) rules of interpretations to the computer. People usually project their own rules to the computer. There can also be mismatch when people interpret information coming from the computer. In both cases, an application of computational semiotics would be to use knowledge about possible interpretation rules for reducing (if not disallowing) the chance of mismatch.

The mismatch can be further analysed in weakening or strengthening what is expressed by the computer. Usually, computers are programmed to reduce the approximation and thus it is not possible to weaken their output when interpreting it. However, this leads to the expectation, from people, that computers always show everything they "know". This is another rule of interpretation that must be taken into account when designing systems which try to adapt their output to what users want!

The term "computational semiotics" is used by a variety of people. The basic idea is that the computer (in fact any computing device) can be used to manipulate the meaning of expressions. For some authors, it is restricted to manipulate data and to display it in a meaningful way. For others, it can be equated with artificial intelligence as a whole. For yet others, it aims at reproducing on computers the emergence of signs in a society (semiosis).

The approach considered here is that computational semiotics is defined by "the existence of an algorithm, a given formal semiotic structure and a mapping function between both" [1]. From this it can be possible to reduce the semiotic mismatch between computer and people. Having the rules of interpretations associated with the languages, just like the semantics is associated to the languages, would allow to ensure meaning reconstructibility.

The idea of algebraic semiotics [2], has been put forth by Joseph Goguen in order to consider representations (or specifications) in relation with their use. To that extent the representations are called sign systems. They include the syntax and semantics of the considered language and — as far as possible — information for interpreting the representation.

A semiotic morphism transforms such a sign system (the source) into another (the target) supposed more suited to a particular use. Algebraic semiotics studies sign systems and morphisms in the framework of category theory.

However, there is a lack of general ways of expressing rules of interpretation and of corpora of such rules.

CONCLUSION

Obviously, in the communication between people and computers, intelligibility cannot be ensured by semantics alone. The users will apply pragmatic rules of interpretation that must be taken into account when interpreting the signs produced by the computer.

Of course, rather than a definitive answer to the question, it would be profitable to know the state of the art on applying semiotics to communication in these matters and what is considered as consensus or as the edge.

It is not clear that there is even a common semiotic corpus that can be applied by computer scientists. Rather, collaboration on particularly important case studies might be profitable to both communities.

REFERENCES

[1] Gerd Döben-Henisch, A possible scientific framework for computational semiotic systems, Institüt für neue Medien, Frankfurt am Main (DE), 1998 http://www.inm.de/kip/SEMIOTIC/cs-framework-19.11.98.html

[2] Joseph Goguen, 'An Introduction to Algebraic Semiotics, with Applications to User Interface Design', *Lecture notes in computer science* 1562:242-291, (1999).

[3] *Formal ontologies in information systems*, ed., Nicola Guarino, IOS Press, Amsterdam (NL), 1998.

[4] Yannis Labrou, Tim Finin, Yun Peng, The current landscape of agent communication languages, *IEEE Intelligent systems* **14**(2):45-52, (1999).

[5] Peter Mosses, 'Denotational semantics', in *Hanbook of theoretical computer science*, ed., Jean van Leeuwen, chapter 11, 575-631, Elsevier, (1990).

[6] Bernhard Nebel, *Reasoning and revision in hybrid representation systems,* Springer Verlag, Heidelberg (DE), 1990.

[7] Alfred Tarski, *Logic, semantics, metamathematics,* Clarendon press, Oxford (UK), 1956.

 © 2000 Jérôme Euzenat