Logics for OO information systems: a semantic study of object orientation from a categorial substructural perspective

de Haas, E.

Citation for published version (APA):
Chapter 8

Categories for Profit

Free advice is seldom cheap.

Rule #59 from the Ferengi rules of acquisition ([Behr95])

Let us assume that the best thing that has happened to the field of information systems is the remarkable insight which Edgar F. Codd had in the summer of 1969, when he wrote a research report for his employer IBM, in which he suggested that database technology should have a formal mathematical foundation. The resulting data-model, the relational model, has become very well developed, assumingly because of its clear and rigorous mathematical foundation. It is true, however, that when one asks the users of relational information systems which advantage of the relational information systems they think is the most prominent, hardly anyone mentions its mathematical foundation. Even among developers, we know, the importance of the rigorous foundation is underestimated. We still conjecture that this foundation is the most important factor in developing the relational database to the maturity it has now. Similarly important is a mathematical foundation for object oriented information systems. However, the situation for object oriented information systems differs considerably from the situation for relational information systems. While the concepts of relational systems were based on the mathematical notion of 'relation', the notions of object oriented information systems evolved from practical use. This situation provides a challenge that is not unfamiliar to theoreticians: trying to capture notions from practice in such a manner that it provides a rigorous and precise understanding of the notions.  

So why is it only a few value the mathematics as very important?. The reason probably lies in the supposition that for understanding and working with an

1Consider, for example, research on formal linguistics, capturing notions of natural languages
artificial language it is not necessary to know the full rigorous mathematical semantics, but only some simpler informal easy-to-deal-with intuition. We agree with this statement. But this does not imply that the mathematical semantics is not important, nor that it is the most important. We need some hard ground on which to build the 'easy-to-deal-with' intuition. We will give an analogy\(^2\) to explain this claim.

When one asks a mathematician what complex numbers are, (s)he will explain that they are constructed from ordered pairs of downward closed sets of rational numbers (Dedekind cuts), i.e. the hard thing. When one asks an engineer, (s)he will explain to you the geometric intuition which can be conveyed about the complex plane; the simple thing. This is a robust intuition for which there is an elegant calculus and a nice axiomatization. Dedekind cuts are studied for foundational concerns, i.e. to confirm the correctness of the logic. For working with complex numbers Dedekind cuts are not necessary.

Although one can do many things with an intuition that is not totally rigorous, in the mathematical sense, one still needs it to verify or even build this intuition. Furthermore the (possibly less formal) intuition should in some sense be robust and have features like a calculus for reasoning or even a (not necessarily complete, but at least sound) axiomatization for proving some properties. For designing semantics it is necessary to construct a high level system that forms the basis for the intuition. This system should not be very involved, i.e. by not using many esoteric mathematical constructions, but by giving a direct account of the concepts that are to be understood for using the considered language. For building this system one may use as much heavy mathematical artillery as one wishes.

This chapter we summarize what has been researched in this thesis. We will discuss the analysis and the mathematical foundation we propose for object orientation. Moreover, we will answer the question: what did we gain from all of this? In other words: what is the value of the artifacts from this thesis for the information system analyst, what is the value of these artifacts for the theoretician, and which problems did we solve?

In this thesis we have performed seven tasks:

1. We analyzed the practical context in which notions of object orientation are used.

2. We analyzed object oriented concepts themselves.

3. We constructed a general and formal language for object oriented information systems.

\(^2\)This analogy is borrowed from Dana Scott, explaining the value of his work in semantics.
8.1. The object oriented development practice

4. We defined a formal model (semantics) for this formal language of object orientation.

5. We introduced themes from theoretical computer science and logic for analysis of the formal language and its accompanying model.

6. We analyzed logical properties of our formalization of the concepts from object orientation.

7. We analyzed four philosophical issues using the constructed formal (and therefore precise) language and model for object oriented information.

These seven tasks are summed up in the sections below.

8.1 The object oriented development practice

The analysis of the practical context in chapter 1 provides us with a number of important insights into how object oriented languages are used in software development practice. A software analyst starts with labels that have no meaning, and evolves to a model with objects and types that carry structure and meaning, but the objects may be partially, or even non-wellfoundedly, specified. This insight has a large impact on the way we should interpret the object oriented languages that the analyst uses.

For the practicing software analyst this analysis is nothing more than an interesting view on his daily work. For a theoretician, on the other hand, this analysis provides important requirements on the model for object orientation he wants to construct. We have seen the influence of this insight in the mathematization of the object oriented concepts in this thesis. Notable mathematical concepts that relate to these insights are the notion of a 'link' (or 'aspect' or 'inon'), and the 'extendibility' notion.

8.2 Concepts of object orientation

The analysis of object oriented concepts in chapter 2 gives an overview of the main concepts used in object oriented technology. Because these concepts arose from use in practice, they can be interpreted in several ways, and sometimes are not very precise. We have pointed to potential problems with some of the concepts, and provided an interpretation that is the basis for the formal model of object orientation in this thesis.

For the software analyst this overview provides a thorough view on the notions he uses in practice, and can make him aware of potential problems. For the theoretician this overview can form a basis for his model, as it did for our model.
8.3 A generalized language for object oriented information systems

The major artifact of this thesis is the general language for object oriented information we defined in chapter 3: the language of categorial graphs. This language has both graphical and textual elements, and has a formal syntax. The formal syntax enables one to precisely define semantics for the language constructs and the notions expressed in this language. Moreover we solved the non-trivial problem of 'exploding' and 'imploding' in a graphical language.

A formal language is a necessary artifact for a theoretician to do formal analysis. In practice, the language of categorial graphs gives a designer of an object oriented language the possibility to map his language to a formal one which is suited for formal analysis. Moreover, an information analyst can translate the expression he writes down in his OO modeling language to expressions in the categorial graph language in order to compute mathematical properties of his model. He can, for example, then compute 'satisfiability' of the model he defined with his expressions.

8.4 A semantics for object oriented information systems

In chapter 4 we built a model for object oriented information systems; more specifically, we built a semantics for the language of categorial graphs. This model captures the behavior of entities in object oriented information systems, and thereby 'makes concrete' the notions of object orientation. It is the realization of our analysis. Important new mathematical concepts are the notions of 'object', 'infon' and 'extendibility'. These notions, for example, realize a driving slogan of object orientation: 'every property of an object is an object in its own right'.

A formal semantics is the target artifact of the OO theoretician. It is the construction that rigorously captures his intuition and enables deeper analysis of the notions involved. The model gives the practitioner the possibility to do formal (and thus automated) model checking, using the formal interpretation of the language of categorial graphs. Moreover it enables one to infer properties of a constructed model using the sound syntactic calculus that accompanies the model.

\[3\text{Note that even the standard language UML is evolving. UML version 2.0 is bound to be released!}\]
8.5  Methodology: semantics, logic and applications

Chapter 5 displays an overview of the scientific context in which the research that has resulted in this thesis took place. Here we have introduced for the reader the scientific tools we used for formal analysis; these are 'computer science semantics', 'modal logic' and 'substructural logic'. Moreover we discussed related research and pointed to applications of the theory that has been developed in this thesis.

Such introductions and references are good practice in science.

8.6  Logic of object oriented information

The analysis of the formal language and model of object oriented information systems took place in chapter 6. This analysis has built the hard ground under our intuitions on object oriented concepts. It gives insight into the axiomatics and complexity of notions from object orientation. Moreover we propose an interesting view on how to do logic, using the intuition of object orientation for a general logic.

For the logician these logics provide insight into the logical properties of an interesting domain with practical relevance. Moreover, these logics have purely logical relevance as well. For the software analyst this is the theory that validates his intuition.

8.7  Four philosophical issues

In chapter 7 we analyzed four philosophical issues in the light of the object oriented language for specifying information developed in this thesis. A long history in philosophy has shown that describing parts of the real world is very complex and subject to serious issues. Because the aim of object oriented modeling is also capturing parts of the real world; it is therefore only natural that in object oriented modeling we encounter the same issues. Four of these issues were elaborated.

Most practitioners of object oriented modeling probably do not realize that such issues lie in wait for their models. It is very valuable, however, to realize they do, because these issues really pose problems of consistency on their models. We show that the formal system of categorial graphs gives a clear insight into these problematic issues, which enables one to avoid misinterpretations. Hereby we show its value for analyzing complex matters of describing information in an object oriented manner.