2005

Ontology Based Method Engineering

Andreas Gehlert
Dresden University of Technology, gehlert@wise.wiwi.tu-dresden.de

Uta Buckmann
net-linx Europe GmbH, uta.buckmann@net-linx.com

Werner Esswein
Dresden University of Technology, esswein@wise.wiwi.tu-dresden.de

Ontology-Based Method Engineering

Andreas Gehlert
Dresden University of Technology
Faculty of Business Administration and Management
Chair of System Development
gehlert@wise.wiwi.tu-dresden.de

Uta Buckmann
net-linx Europe GmbH
uta.buckmann@net-linx.com

Werner Esswein
Dresden University of Technology
Faculty of Business Administration and Management
Chair of System Development
esswein@wise.wiwi.tu-dresden.de

ABSTRACT

We need conceptual modelling languages to gain domain knowledge in the requirements engineering and analysis phases of an IS development project. These languages should serve an IS expert as means of communication between him or her and the domain expert. Many different modelling languages have been used for conceptual modelling. Consequently, questions relating to the quality of these languages have arisen. Wand, Weber and others have evaluated these languages using an ontology. Each of the languages was found to contain certain deficits. Because our aim is to construct a language without such deficits, we propose the opposite technique. We develop an ontologically clear modelling language for process modelling with the help of the BWW representational model. In addition to this modelling language, we introduce a process model which guides model creation. Both components form a conceptual modelling method.

Keywords: Method-Engineering, Ontology, BWW-model, Conceptual Modelling

INTRODUCTION

Requirements engineering is the first step in IS development. Within an interaction process with the domain expert, requirements for the information system are usually gained with the help of models as a communication medium. Each model can be thought of as a statement in a language.

A language in the linguistic sense is a system of signs or constructs. Each construct links two components. The first component is an external entity (expression; i. e. the words “entity type”). The second component is a representative mental state or the semantics associated with the first component (i. e. the meaning of the words “entity type”) (Sebeok, Posner and Ray, 1994, p. 936).

The term semantics is used with different meanings. Because of its focus on formal systems in the discipline of computer science, the term semantics is usually defined as a formal system of rules. This *formal semantics* explains the constructs of a language using another formal system (Esswein, Gehlert and Seiffert, 2004, p. 464). This is the prerequisite of an automatic interpretation of the model (Guizzardi, Pires and van Sinderen, 2002). For the domain expert, however, this definition is not very helpful. According to Putnam describing the meaning of a word by its translation into another language does not add any meaning to this word as long as the person reading this translation does not know the other language. For instance, the description of the word “water” by its chemical formula H_2O does not explain what the word “water” means for somebody who is not familiar with Chemistry (Putnam, 1975). In other words, the definition of formal semantics, although very important in the field of computer science, does not help in the communication process with the domain expert, because the domain expert is usually not familiar with formal languages (Ortner and Schienmann, 1996, p. 262).

In contrast, *material semantics* focuses on a real world domain and describes the relationship between language constructs and the conceptualised phenomena of reality. Because every individual has his or her own understanding of reality, material

semantics cannot be defined explicitly. It rather "arises" within a long social and cultural process (Schefe, 1999, p. 127). The semantics of a language depends on each person who speaks this language (Wittgenstein, 1981).

If we attempt to find the morphemes, i. e. the smallest syntactic part which is capable of carrying a meaning of a model, we have to distinguish between at least two different layers. One part of the model semantics is determined by the modelling language. The other part of the model semantics is given by domain language expressions used in these models (Esswein et al., 2004, p. 465).

Weber identifies a number of problems with current modelling languages used for conceptual modelling. All of these problems regard to the semantics of languages. Firstly, the semantics of the modelling constructs (morphemes) is not clear. Secondly, some conceptual modelling languages address both material and formal domains and due to that fact undermine conceptual modelling. Thirdly, the selection of modelling constructs is largely intuitive and rarely based on theoretical understanding. Each of these deficits leads to models which are difficult to understand (Weber, 2003, p. 11).

The problems described above can be solved with conceptual modelling languages based on a sound theory which addresses a material domain. Since there is no such theory yet available in the field of computer science, ontologies developed by different philosophers have been widely adapted to modelling issues (see for instance Wand and Weber, 1990, p. 1282). The Bunge-Wand-Weber Model is highly relevant for this issue, because it was developed for the representation of information systems (Rosemann and Green, 2002, p. 78). In many publications, existing modelling languages have been analysed using the BWW-model (see for example Evermann and Wand, 2001; Green and Rosemann, 2000; Opdahl and Henderson-Sellers, 2002). In all cases, different deficits were found. Consequently, none of the analysed modelling languages could be labelled ontologically clear (each language construct has exactly one corresponding ontological construct) or ontologically complete (each ontological construct is present in the modelling language) (Weber, 1997, p. 92).

A modelling language that is both ontologically clear and complete can be developed in two different ways. One way is to analyse an existing modelling language (as described above) and to gradually remove all its deficits. The other way is to construct a new modelling language on the theoretical basis of an ontology (Rosemann and Green, 2002, p. 77).

The first option includes the risk of misinterpreting the existing language constructs. Consequently, the comparison of the modelling language with the ontology largely depends on the understanding of the researcher (Rosemann, Green and Insulska, 2004, p. 113). The second option avoids that problem, since the meanings of the ontological constructs are clear. Therefore, we have chosen to follow the second approach here.

As a result, we have to explain in which way an ontology can support the definition of a conceptual modelling language (see figure 1). In our terminology, a conceptual modelling language consists of a (small) set of modelling constructs, a definition of their syntax, and a description of their semantics which relates to a material (real world) domain.[1] The abstract syntax of a modelling language is usually described using a meta model (Strahringer, 1996, p. 23).

On the other hand, an upper level ontology defines a (small) set of constructs and the relations between them (Guizzardi, Herre and Wagner, 2002, p. 65; Guizzardi, Pires and van Sindren, 2002). Consequently, the ontology can be the basis for the meta model of the abstract syntax of the conceptual modelling language. In contrast to the constructs of new conceptual modelling languages, the meaning of the constructs of the ontology is already clear.

We use the syntax and semantics already "defined" by the ontology (subsection 0) to construct a conceptual modelling language on the basis of this ontology. Additionally, we add a graphical representation to these constructs to enhance their usability in communication with the domain expert (see Larkin and Simon, 1987 for advantages of graphical representations). Furthermore, we divide the constructs into views to reduce the complexity of the models of this language (subsection 0). Last of all, we construct a process model on the basis of the existential dependencies of the ontological constructs. This process model describes the usage of the conceptual modelling language (see subsection 0). The modelling language and the process model form a *modelling method* (Esswein et al., 2004, p. 465).

[1] Conceptual and "non-conceptual" modelling languages are distinguished by their problem domain. For example, the Entity Relationship Model (ERM) is not a conceptual modelling language because the semantics of this language focuses on a formal domain. Chen describes: "From now on, we shall consider only the entities and relationships (and the information concerning them) which are to enter into the design of a database." (Chen, 1976, p. 11) Chen wants to use his language to design databases (a formal system). Consequently, he defines a formal semantics of his graphical representations as transformation rules to the relational model (see Chen, 1976, pp. 19). If we use the ERM for conceptual modelling, we apply a new (material) semantic. Thus, we do not use the original language but merely its syntax and its graphical notation.

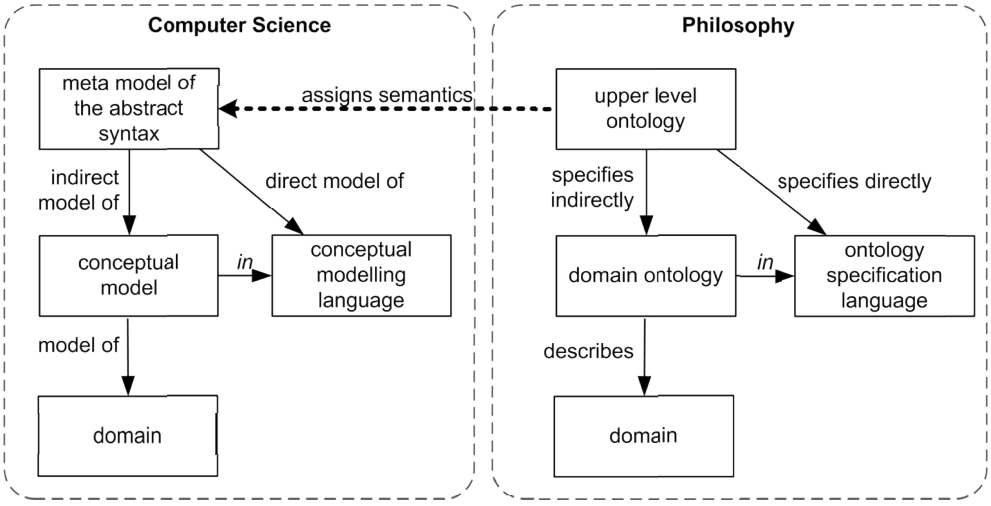

Figure 1. Relations between conceptual and ontological modelling

The presented modelling method is especially interesting for the computer science discipline for the following reasons:

- We show for the first time that it is possible to develop a conceptual modelling method on the basis of the BWW-model.
- Future empirical evaluation of our language allows conclusions to be made about the suitability of the BWW-model as a theoretical foundation for conceptual modelling languages. This is especially interesting since the General Ontological Language (GOL) was established as a competitive ontology (Degen and Herre, 2001; Guarino and van Sideren, 2004; Guizzardi, Herre and Wagner, 2002; Guizzardi, Pires and van Sindren, 2002). Thus the question arises which ontology is more suitable for conceptual modelling (Weber, 2003, p. 16).
- Regarding the formal focus of the BWW-model, its models may be evaluated using formal criteria. As a result, the quality of the models can be enhanced (Soffer and Wand, 2004).

The new modelling method has the following theoretical advantages which are subject to further empirical research:

- As explained above, the new modelling language is ontologically clear.
- The semantics of the modelling constructs is described precisely within the BWW-model, so the scope for different interpretations of the modelling language constructs by its users is minimised. Thus, the possibility of misinterpreting the resulting models is decreased.
- The syntax of the modelling constructs is theoretically founded and not arbitrary. The semantics is well-established because of the wide range of BWW-model users.
- The language only contains constructs necessary for describing real world phenomena. All implementation aspects are disregarded.

In the next section, we develop the new modelling method in detail. We finish the paper with a summary, a reflection of the paper within different epistemological positions and draw consequences for further research activities.

METHOD ENGINEERING WITH ONTOLOGIES

According to Brinkkemper et al., Greiffenberg, Harmsen and Strahringer, method development must be divided into at least three distinct activities (Brinkkemper, Saeki and Harmsen, 1998, p. 384; Greiffenberg, 2004, p. 35; Harmsen, 1997, p. 43; Strahringer, 1996, p. 91): [2]

[2] This paper does not aim to discuss method engineering issues in detail, but rather uses some fundamental insights of this discipline.

1. The first step is the development of the abstract syntax of the modelling language. This syntax defines all language constructs as well as their lawful combinations. It is expressed using a meta model (subsection 0).
2. Each construct of the modelling language is assigned to its graphical representation, because visual languages enhance the communication between the IS and the domain expert (subsection 0).
3. After having defined the language in steps one and two, a process model has to be added. This process model defines the activities that are needed to produce valid models within the modelling language (subsection 0).

This paper, however, does not provide a modelling method which covers all aspects of the BWW-model. Instead, we restrict ourselves to the process view. Subsequently, we follow the approach proposed by Rosemann and Green and focus the BWW-model by concentrating on the elements needed to describe processes (Rosemann and Green, 2000, p. 622). These selected constructs are the basis for the abstract syntax of the modelling language. Therefore, the resulting modelling language seeks to achieve ontological clarity but explicitly not ontological completeness.

Focusing the BWW-model

The BWW-model was developed by Wand and Weber to provide a sound theoretical basis for conceptual modelling. This model was described extensively by its authors (Wand and Weber, 1990; Weber, 1997). Rosemann, Green and Davies (2003) provided meta models of the BWW-model. Accordingly, we only explain the most central concepts which have important consequences for the modelling language.

The central term is the *thing*. Things are manifested by their *properties* which are modelled by *attributes*. The value of these attributes at a certain time is called *state*. Things are coupled if one thing acts on another thing. That means that the states of two things are related to each other (dependent histories). The thing belongs either to the *system* in focus or to its *environment*. A *transformation* refers to a state-change within one thing only. Thus, a transformation always changes the state in one thing. This transformation may, additionally, induce a change in state of a coupled thing.

According to Soffer and Wand, a process can be described by a set of states, a subset of states representing the unstable states, a subset of states representing the stable states and a set of transformation laws describing valid state changes (Soffer and Wand, 2004, p. 524).

Using this definition, a *process* is a set of state changes bringing a system from an unstable state (caused by an external event) to a stable state. This process definition provides the input for the modelling language which we develop in subsection 0:

- A process is based on state changes. These are defined as changes of attribute values of things. Consequently, we have to describe the relevant things and their attributes first (thing identification).
- To distinguish between external and internal states, the identified things must be separated from their environment (system building).
- Additionally, a process must be described by laws (see above). State laws define the lawful state space and transformation laws define the lawful state changes (lawful transformation) or business rules.

Definition of the Abstract Syntax

Based on the information above, we can start to construct the meta model of the modelling language (Strahringer, 1996, p. 91, see also figure 2). We use the Unified Modelling Language (UML) for the representation of the meta model (OMG, 2001)[3]. We reconstructed the BWW-model constructs as UML-classes, properties of the constructs as attributes and associations.

[3] The UML as quasi-standard for object oriented modelling has been used for meta-modelling by the OMG itself. Hence, it suits our modelling needs in this paper. It is used to describe the meta model of the abstract syntax (figure 2) as well as the meta model of the modelling process (figure 5).

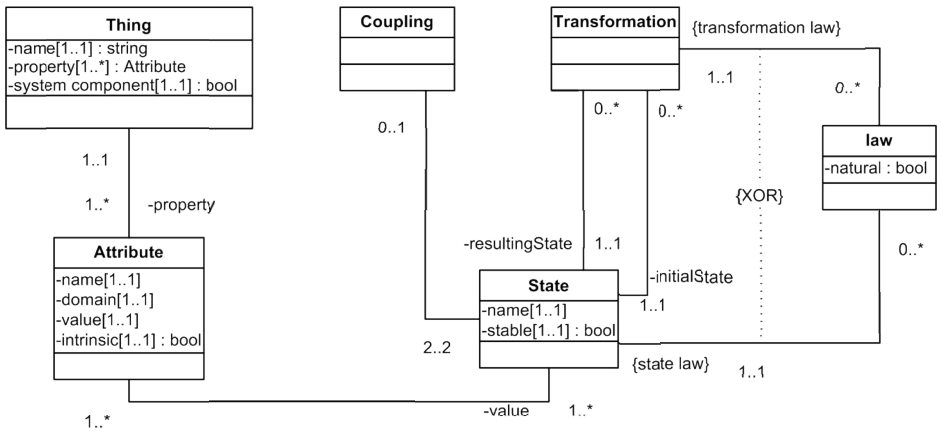

Figure 2. Meta model of the abstract syntax

The actors and resources of a process are represented by a thing (see subsection 0; class `Thing`). Things carry a name (attribute `name`) and can only be identified through their properties (attribute `property`). Furthermore, a thing can either belong to the system in focus or to its environment (attribute `system component`=yes/no).

Properties of things are modelled as attributes (class `attribute`; association between `thing` and `attribute`). Attributes have a name (attribute `name`), a domain (attribute `domain`) and a value (attribute `value`) out of that domain. A thing has at least one intrinsic property (attribute `intrinsic`=yes) which defines it uniquely throughout its lifetime. Additionally, attributes can be attached/detached to/from this thing without affecting its identity (attribute `intrinsic`=no).

If the attribute values of two or more things have a common history, the things are coupled (class `coupling`). Each thing must have at least one coupling (association between `coupling` and `state`). The state is always attached to a thing and represents a named vector (attribute `name`) of the thing's values (association between `state` and `attribute`; Green and Rosemann, 2000, p. 76; Wand and Weber, 1995, p. 210).[4]

The possible state space of a thing is defined as the Cartesian product over its attribute's domains. However, only a subset of all possible states is modelled. These states are constrained by state laws (class `law` in conjunction with the association to `state`). The remaining states form the lawful state space and are either bound to a system thing or to an environmental thing.

The transition from one state to another is called a transformation (class `transformation` and associations to `state`). The set of possible transformations is constrained by transformation laws (association between `transformation` and `law`).

When defining laws, the modeller must state whether a law is created by humans (attribute `natural`=no) or is given by nature (attribute `natural`=yes). Whereas human laws can be broken temporarily, natural laws must always be fulfilled. A broken natural law indicates a major design flaw of the information system (Opdahl and Henderson-Sellers, 2002, p. 52).

Furthermore, we must distinguish between stable and unstable states (attribute `stable`=yes/no). A thing is in a stable state until its state is changed because of a coupling with another thing. Hence, a system is in a stable state if all its things are in stable states. Consequently, two dynamic aspects must be separated within the BWW-model. Firstly, one thing may change from one unstable state to another. Secondly, one thing can be switched from a stable state to another state (stable or not) because of its couplings to another thing. Let us suppose that a system was in a stable state. A process starts if one environmental thing changes its state. Through the coupling(s) to one or more system things a change into an unstable state

[4] To decrease the complexity of a state machine (Harel and Pnueli, 1985, p. 242) Evermann and Wand as well as Fettke and Loos suggested using a functional schema containing only a subset of the thing's attributes (Evermann and Wand, 2001, p. 357; Fettke and Loos, 2003, p. 2947). The possibility to define a state according to a functional schema was not clearly specified by the authors of the BWW-model. To achieve maximum ontological clarity this aspect is disregarded in this paper.

of this (these) system thing(s) is caused. The system things change their states until all of them reach a stable state again and, thereby, terminate the process.

In addition to the abstract syntax defined by the meta model in figure 2, we specify five consistency rules:

R1 A state is always associated with attributes of the same thing.

R2 A transformation is always related to two states of the same thing.

R3 A coupling relates to states of two things.

R4 A state change between things can only occur if they are coupled.

R5 If two transformations have the same unstable starting state they are linked to each other by the XOR operator, since a thing can only be in one state at one time. Although two transformations can have the same starting state, they must have different end states.

Definition of the Graphical Notation

From our process definition in the BWW-model, we can conclude that a process cannot be separated from things. Soffer and Wand call these things actors or resources of that process (Soffer and Wand, 2004, p. 524). This leads to a separation of the language constructs into two views. The resource view contains the specification of things, their attributes, the specification as system or environmental things as well as the couplings between these things. The process view focuses on the states and transformations of these things. All state and transformation laws are specified here. The following simple modelling example introduces the views as well as the graphical notation of the language constructs.

We use the following metaphor as conceptual description for our example: "Readers can only borrow books from the library as long as they do not have any overdue fees. The overdue fee applies to the reader after he or she has not returned the books within 10 days."

In the resource view we model the "book"-thing, which belongs to our system under analysis (underlined) and the "person"-thing belonging to its environment. Both things have intrinsic attributes (underlined) as well as additional attributes. Both things are coupled.

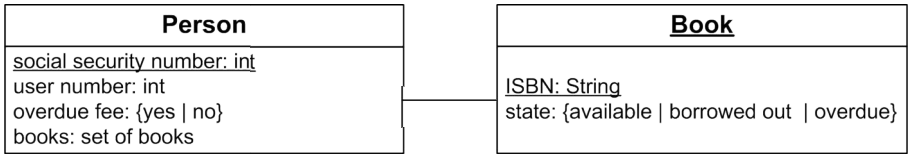

Figure 3: Modelling example – Resource View

In the process view we model relevant states of things (round rectangles) and transformations between these things (arrows). A coupling is expressed by annotating conditions on the transformation that refer to another thing. All natural laws are underlined. A book, for example, cannot be borrowed out if it is not in the library (natural law). On the contrary, the person cannot borrow more books when he or she has an overdue fee (human law).

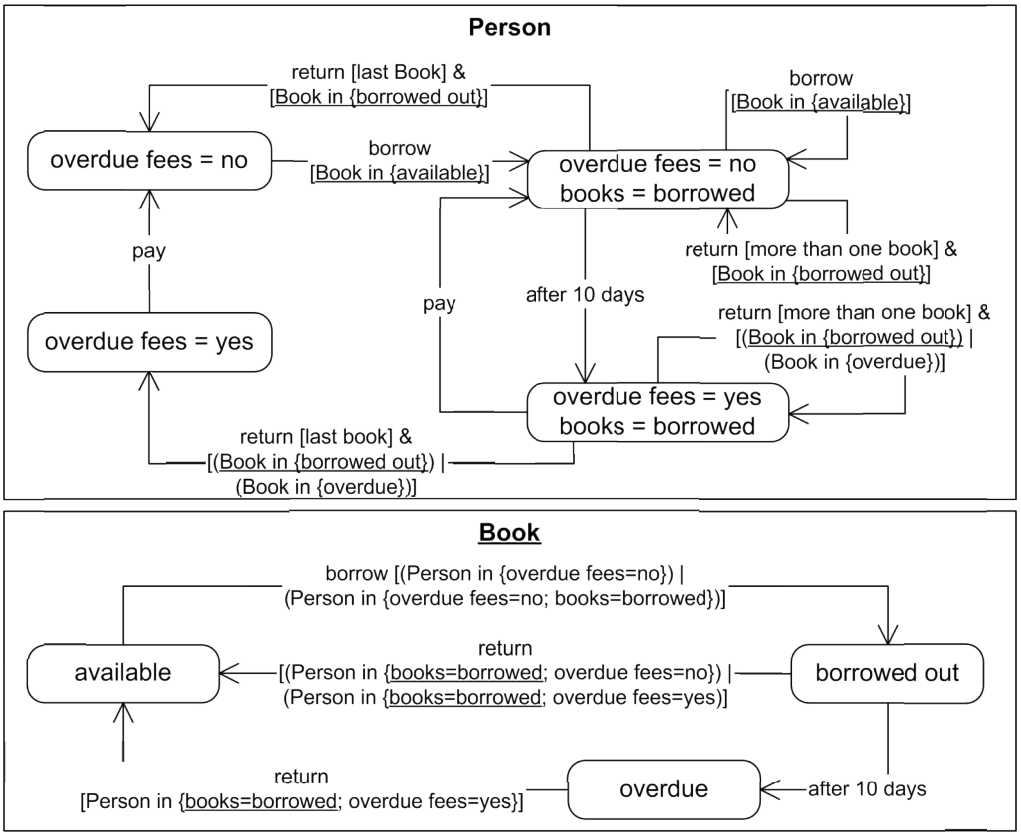

Figure 4: Modelling example – Process View

Process Model

The existential dependencies between the BWW-model constructs analysed in subsection 0 have important consequences for the model construction process. Again we use the UML activity diagram for representing the meta model of the modelling process (see figure 5).

"The world is made up of things. We know about things in the world via their properties." (Weber, 1997, p. 34) This means, that we have to identify the properties of the things first. Second of all, we consider the things that own these properties. Each identified thing can belong to the system in focus or to its environment. The proposed properties are modelled as attributes of the thing (Weber, 1997, p. 34; see also subsection 0). Furthermore, we must decide for each property whether it is intrinsic or not. After the analysis of all properties and things, the models of the resource view are provisionally complete.

Having identified the attributes, we can constrain their values by defining state laws. For each state law, we have to specify it as a human or a natural law. Additionally, we must decide whether the identified lawful states are stable or not.

After the definition of the states, we must decide whether the states of different things depend on each other. If so, we have to model a coupling within the resource view. Within the process view, this coupling is expressed by a condition. A transformation in thing A can only occur if a thing B is in a specific state.

Since the modeller can repeat this process of model extension as often as necessary, the process model grants the flexibility needed for modelling projects. However, it expresses that someone using the modelling language described in subsection 0 cannot logically define states before the analysis of the respective properties, attributes and things is done. So, the process model provides a logical sequence of the modelling activities.

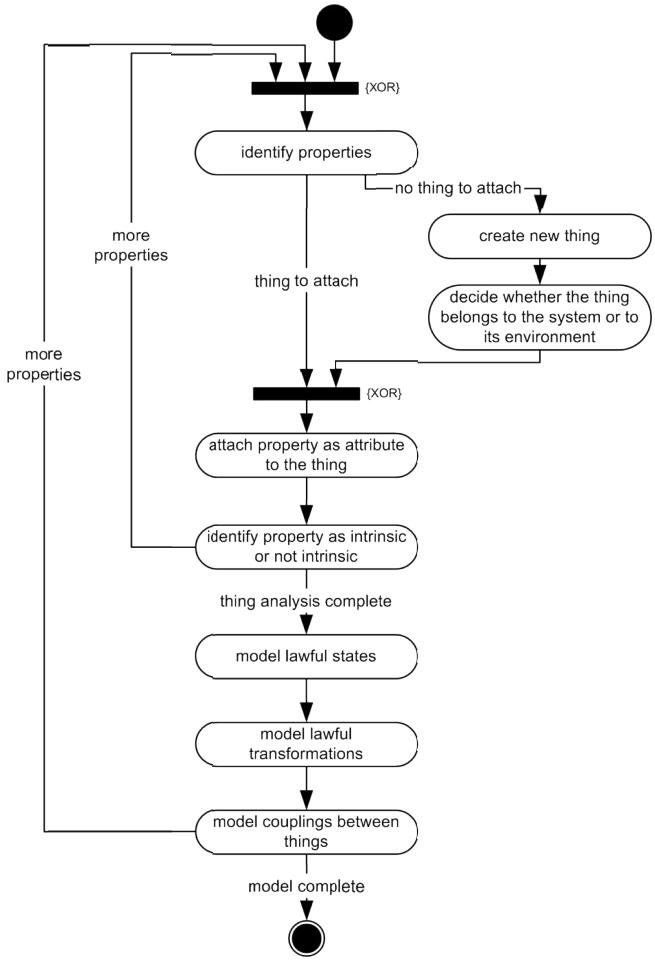

Figure 5: Meta model of the modelling process

CONCLUSIONS

We demonstrated that the BWW-model cannot only be used for assessing exiting modelling languages but also for creating new ones. These new conceptual modelling languages are ontologically clear. Describing the existential dependencies between the language (or ontological) constructs within a process model leads to a complete modelling method as defined by Greiffenberg and Harmsen (Greiffenberg, 2004; Harmsen, 1997).

Furthermore, we pointed out that process modelling within the BWW-model is in fact state modelling. The Harel and Pnueli state machines (Harel and Pnueli, 1985) are, with minor extensions, suitable for conceptual modelling in that sense. The BWW-model, however, focuses on the things and their inherently connected states. Because of this strong binding, process modelling within the BWW model always includes the specification of things (resource view).

These findings might be interpreted differently depending on the epistemological position of the modeller. If he or she follows the constructivist approach, the modeller would deny the overall existence of ontology in the philosophical sense. From this point of view, the BWW-model is "only" one possible conceptual model. The value of the BWW-model, however, lies in providing a language suitable for describing real world phenomena. Existing empirical results have so far supported this thesis (see Bodart, Patel, Sim and Weber, 2001; for more references see Weber, 2003). Furthermore, the formalism provided by this language restricts the modeller in subsequent modelling activities. For instance, if a modeller has decided to model some construct as attribute, it is impossible to assign other attributes to the existing one.

In the light of a realist epistemological position, the decisions of the BWW constructs to be modelled are given by reality. The modeller, in turn, can only choose whether a certain construct should be modelled. For instance, the realist modeller

cannot choose to model a "customer" as a thing, since it represents a mutual property between a "company-thing" and a "person-thing". However, the modeller can choose if he or she wants to model this mutual relationship or not.

In both cases the meanings of the ontological concepts are clear. In the constructivist approach the meaning of the given language constructs is assigned to them by speaking the language (Wittgenstein, 1981). Since there are many publications on the BWW-model, there is a big language community which is familiar with the semantics of these constructs. For the realist, the meaning is the correspondence of real world constructs to BWW constructs. All meanings of the ontological constructs are clear.

However, our approach is only a first step in constructing new and hopefully better conceptual modelling languages. We see further research activities in the following areas:

1. The modelling language, although ontologically clear, lacks ontological completeness. Hence, other aspects of the representation model must be addressed. This work leads to the construction of a multi view language which is also ontologically complete.
2. Current research within the area of the BWW-model mainly concentrates on the representation model, but disregards the state tracking and the good decomposition model. We do not know if our modelling language is consistent with the latter BWW sub-models.
3. Lastly, we must deal with the problems of the domain specific language used in these models.

To fully understand conceptual modelling, we must pay regard to the modelling language, the graphical and the domain language level.

REFERENCES

1. Bodart, F., Patel, A., Sim, M. and Weber, R. (2001) Should optional properties be used in conceptual modelling? A theory and three empirical tests, *Information Systems Research*, 12, 4, 384-405.
2. Brinkkemper, S.; Saeki, M. and Harmsen, F. (1998) Assembly techniques for method engineering, in: Pernici, B. and Thanos, C. (Eds.) *CAiSE'98 Conference Proceedings*, Springer Press, 381-400.
3. Chen, P.P.S. (1976) The entity relationship model - toward a unified view of data, *ACM Transactions on Database Systems*, 1, 1.
4. Degen, W. and Herre, H. (2001) What is an upper level ontology?, in *Workshop on "Ontologies"*, Vienna.
5. Esswein, W., Gehlert, A. and Seiffert, G. (2004) Towards a framework for model migration, in Persson and Stirna, 2004, 463-476.
6. Evermann, J. and Wand, Y. (2001) Towards ontologically based semantics for UML constructs, in Kunii, H. S.; Jajodia, S.; Sølvberg, A. (Eds.) *Conceptual Modeling-ER 2001 Conference Proceedings*, 354-367.
7. Fettke, P. and Loos, P. Ontological evaluation of reference models using the Bunge-Wand-Weber model, in Ninth Americans Conference on Information Systems, 2944-2955
8. Green, P. and Rosemann, M. (2000) Integrated process modeling: An ontological evaluation, *Information Systems*, 25, 2, 73-87.
9. Greiffenberg, S. (2004) Method enigneering in the private and the public sector, Dr. Kovač Press, Hamburg, in German.
10. Guizzardi, G., Herre, H. and Wagner, G. (2002) On the general ontological foundations of conceptual modelling, in: Spaccapietra, S.; March, S. T. and Kambayashi, Y. (Eds.) *Conceptual Modeling-ER 2002 Conference Proceedings*, Springer Press, 65-78.
11. Guizzardi, G., Pires, L. F. and van Sinderen, M. J. (2002) On the role of domain ontologies in the design of domain-specific visual modeling languages, in: Tolcanen, J. P. (Ed.) *OOPSLA 2002 Second Workshop on Domain Specific Visual Languages*, 25-38.
12. Guizzardi, G., Wagner, G., Guarino, N. and van Sideren, M. (2004) An ontologically well-founded profile for UML conceptual models, in Persson and Stirna, 2004, 112-126.

13. Harel, D. and Pnueli, A. (1985) On the development of reactive systems, in Apt, K. R. (Ed.) *Logics and Models of Concurrent Systems*. NATO ASI Series, Vol. F-13, Springer Press, 477-498.
14. Harmsen, A.F. (1997) Situational method engineering, Master's thesis, University of Twente.
15. Kamlah, W. and Lorenzen, P. (1984) Logical propaedeutic: Pre-school of reasonable discourse, Rowman & Littlefield Press.
16. Larkin, J. H. and Simon, H. A (1987) Why a diagram is (sometimes) worth ten thousand words, *Cognitive Science*, 11, 65-99.
17. OMG (2001) OMG Unified Modeling Language specification: Version 1.5, document - formal/03-03-01.
18. Opdahl, A. L. and Henderson-Sellers, B. (2002) Ontological evaluation of the UML using the Bunge-Wand-Weber model, in *Software and Systems Modeling*, Vol. 1, Springer Press, 43-67.
19. Ortner, E. and Schienmann, B. (1996) Normative language approach: A framework for understanding, in: Thalheim, B. (Ed.) *Conceptual Modelling ER '96 Conference Proceedings*, Springer Press, 261-276.
20. Persson, A. and Stirna, J. (Eds.) (2004) *CAiSE'04 Conference Proceedings*, Springer Press.
21. Putnam, H. (1975) The meaning of "meaning", in Gunderson, K. (Ed.): *Language, Mind and Knowledge*, The University of Minnesota Press, Minneapolis, 215-271.
22. Rosemann, M. and Green, P. (2000) Integrating multi-perspective views into ontological analysis, in: Ang, S., Krcmar, H., Orlikowski, W., Weill, P. and Degross, J. (Eds.) *International Conference on Information Systems Proceedings*, 618-627.
23. Rosemann, M. and Green, P. (2002) Developing a meta model for the Bunge-Wand-Weber ontological constructs, *Information Systems*, 27, 2, 75-91.
24. Rosemann, M., Green, P. and Davies, I. (2003) BWW version 4, http://www.uq.edu.au/~uqtchua/ARC/ermetamodel/ index.htm
25. Rosemann, M., Green, P. and Indulska, M. (2004) Towards an enhanced methodology for ontological analyses, in: Grabis, J., Persson, A. and Stirna, J. (Eds.) *CAiSE'04 Forum Proceedings*, 112-121.
26. Schefe, P. (1999) Software engineering and epistemology, *Informatik Spektrum* 22, 2, 122-135, in German.
27. Sebeok, T. A., Posner, R. and Rey, A. (Eds.) (1994) Enceclopedic Dictionary of Semiotics, Volume 2, Mouton de Gruyter Press, Berlin.
28. Soffer, P. and Wand, Y. Goal-driven analysis of process model validity. In: Persson and Stirna, 2004, 521-535.
29. Strahringer, S. (1996) Metamodelling as an instrument of method comparison: An evaluation of object oriented analysis methods, Shaker Press, in German.
30. Wand, Y. and Weber, R. (1990) An ontological model of an information system, *IEEE Transactions on Software Engineering*, 16, 11, 1282-1292.
31. Wand, Y. and Weber, R. (1995) On the deep structure of information systems. *Information Systems Journal*, 5, 203-223.
32. Weber, R. (1997) Ontological Foundations of Information Systems, Coopers & Lybrand Press.
33. Weber, R. (2003) Conceptual modelling and ontology: Possibilities and pitfalls, *Journal of Database Management*, 14, 3, 1-20.
34. Wittgenstein, L. (1981) Tractatus logico-philosophicus, Routledge & Kegan Paul Press, London.