A little theory of abstraction*

Roland Kaschek
Department of Information Systems
Massey University
Palmerston North, New Zealand

January 28, 2004

Abstract

By means of abstraction a notion A is derived from a notion S. The resulting notion A often is called abstraction as well. Abstraction is considered important for information systems development. It often is naively understood as simply prescinding from aspects not considered as important for a task at hand. This approach to abstraction does not take into account two points: Firstly, prior to prescinding from characteristics of a phenomenon this phenomenon must be constructed conceptually. Secondly, the extent of a notion in general only can be determined with respect to a scope that was presupposed. The present paper proposes an approach to abstraction that meets both of these points and restricts abstraction to be carried out only within predefined conceptual frameworks. A few commonly used example frameworks are identified and discussed. The paper aims at helping those who feel that the known abstraction concepts for a task at hand are not satisfactory and want to define better suitable ones.

1 Introduction

It is folk knowledge that abstraction is important in information systems development. Occasionally informatics (or computer science as a whole) is considered as the science of abstraction, see [AU96, p. 1]. However, a theory of abstraction in Informatics, and in Information Systems (IS) in particular, seems not to exist. In the present paper we outline a limited model of abstraction. The model on the one hand is limited in that it ignores the abstractions that are implied by imperfection of human senses and by limitations of human consciousness that leads to the attention being focused on a small number of phenomena and more or less ignore all other phenomena. The model of abstraction on the other hand is limited in the

*The author thanks Jörg Desel from Katholische Universität Eichstätt-Ingolstadt for his helpful comments on an earlier version of this paper.

sense that it rules out abstractions that do not define the extent of the derived notion. Abstractions like "The Good" or "The Evil" are used in every-day reasoning. However, their extent is ill-defined and contested. Therefore these are not within the scope of the model of abstraction that is proposed in the present paper. Our model of abstraction is even narrower as is shown by an example below.

It is not obvious in what respect abstraction is involved in information systems usage. For an explanation of this matter we refer back to the Scandinavian information systems school. This school 30 years ago worked out definitions of the term information systems that are helpful in our respect.

Solving business tasks with the help of information systems is simplified if these systems implement appropriate abstractions. For example, a database might contain a table $EMPLOYEE$ and the tuples of $EMPLOYEE$ represent the employees of the company. Then an individual I is capable of determining the company's number of employees by counting the tuples of $EMPLOYEE$. This surely is simpler than counting the employees directly since these might be many, it might be difficult to assemble them at a given location and no respective counting tool or techniques might be available. Similarly, determining an employee's date of birth (DOB) can be easy if one prescinds from the fact that the respective tuple in $EMPLOYEE$ is not the employee and that the query for the value of the DOB-field is not the same as asking the employee for her or his date of birth. However, asking the employee directly for his or her date of birth might be difficult since s/he might not be available, able or willing to mention her/his date of birth.

According to the Longman Dictionary of Contemporary English to abstract means: "to remove something from somewhere or from a place". Extending the picture [Mat02] defines abstraction as "... a memory process that stores the meaning of a message without storing the exact words and grammatical structures." See also the more encyclopedic sources [Thi92, RM98]. Below we introduce an approach to abstraction that avoids the danger of ill-defined abstraction and covers several abstraction concepts that often are applied in information systems development. The basic idea is that the process of prescinding from aspects of phenomena is restricted to take place within a conceptual framework that must be defined prior to defining the abstraction.

The approach to abstraction taken in the present paper is a constructivistic or activity-oriented one. Rather than assuming that abstractions are around and only need to be identified, we are going to assume that abstractions need to be carried out in a case at hand by an individual. Therefore, we are going to presuppose an individual that performs some cognitive activities and report on what we believe s/he is doing. However, we do not explicitly mention the observer. We furthermore do not presuppose that the observer is omniscient. Rather we report on the findings an observer might have who has discussed with the modeler what s/he is doing.

Outline

The paper is structured as follows: Related work is discussed in the following section. In section 3 we introduce information systems based on the definitions obtained by the Scandinavian IS school and argue that using information systems involves abstraction. After that in section 4 we briefly discuss a meta model for the concept "semantic model". The key idea in the respective definition is to define abstraction concepts as (purposefully chosen) sets of perspectives. In the following section 5 we define and discuss "abstraction" and several predicates from which perspectives can be derived that are frequently used in modelling. The paper is concluded with a resume in section 6 and the references.

2 Related work

To denote that **notion** A is derived by an **abstraction concept** α from notion S we write $\alpha : S \Rightarrow A$. Simplifying the situation significantly, in the sense of Frege, we presuppose notions to be such that they have a sharp and uniquely defined **extent**, i.e., a set of phenomena falling under them. Notions may be composite and more or less complex. Abstraction concepts are a particular kind of notion. For easily accessible overviews on notions refer to [Kel95, Ch. 8] or [Thi92]. The extent of notion X is denoted by $\varepsilon(X)$. To see that abstraction is related to identification and modelling let $\alpha : S \Rightarrow A$ be an abstraction concept. α is related to identification since, given α, by means of S one may refer to A and thus identify A. Abstraction is related to identification at the level of notion extents since the extent $\varepsilon(A)$ of a notion A shall contain only those phenomena that are relevant for a task at hand. Thus these phenomena need to be identified. Abstraction is related to modelling since one may consider A as a model of S. At the level of notion extents one may consider an element of $\varepsilon(A)$ as a model of the elements of $\varepsilon(S)$ and may derive information about the latter from the former one.

Identification was discussed, e.g. in [Tha00]. Note that Thalheim does not define "abstraction". Rather he stays with the traditional way of dealing with it in the conceptual modelling literature, i.e., he lists, using a different terminology, (modelling notions and several) abstraction concepts, i.e., different ways to abstract a notion A from a notion S. The weakness of doing so is that it does not really help in defining new abstraction concepts which might be perceived as essential if the known ones cannot be applied satisfactorily. This however is one of the purposes of the present paper. In the source Thalheim presupposes identification to be understood as distinguishing the particular entity, to which one wishes to refer to, from all others in a given collection of entities. The source though using a different terminology classifies identification as being either associative or conventional. Associative identification means distinguishing uniquely based on instances of characteristics the entity is known to have. Conventional identification by Thalheim was understood as instance of a conventional act of referencing such as pointing and naming. Thalheim points out that associative identification as well as conventional identification is used in conceptual modelling (and in this area are not always consistently dealt with) as well as design and implementation.

The theory of **models** adopted in this paper is the one of Stachowiak, see, e.g. [Sta92, Sta83, Sta73], see furthermore [Lud03]. Further work about modelling that appears to be compatible with Stachowiak's theory is [MCF03, Rot89, Qua85]. Rather than following Stachowiak in all details we use only a part of his theory and simplify it. For the sake of simplicity we consider models as systems of notions. Stachowiak introduces the **model relationship** as -in the simplest form- three place predicate $M(T, S, I)$, specifying that individual I considers the notion system S, i.e., the substitute or model as a **model** of the notion system T, i.e., the thing or original. Stachowiak (using a different terminology) requires as quality aspect of the model relationship that the following assertions are true (see [Sta92]):

1. There exist representations R_T, R_S of T and S respectively that are associated to each other by partial bijections $F : R_T \to R_S$ and $F^* : R_S \to R_T$ such that $F^* \circ F = 1_{dom(F)}$, and $F \circ F^* = 1_{dom(F^*)}$ hold.
2. The individual I regarding his/her aim to solve a problem P_T regarding T first, with help of F, translates P_T into a problem P_S regarding S. I then obtains a solution Σ_S of P_S, and, with help of F^*, translates Σ_S into a solution candidate Σ_T for the problem P_T regarding T.

This solution candidate would then be a solution for P_T, if the modelling process was successful. Stachowiak points out that in general $R_T \setminus dom(F) \neq \emptyset$, and $R_S \setminus dom(F^*) \neq \emptyset$ and that usability of S as a model of T in part is a consequence of these inequalities. Given a model relationship $M(T, S, I)$ one easily constructs an abstraction $\alpha_M : T \Rightarrow S$. However, Stachowiak seems not to have worked out a theory of abstraction.

Work on conceptual modelling such as [SS77b, SS77a, TL82, HK87, PM88, Myl98] regarding abstraction mainly identified and discussed abstraction concepts such as classification, generalization, aggregation and association. The **Zachman framework** (see, [Zac87, SZ92]) contributes to a theory of abstraction in that it points out interrogatives that appear to be important for systems development in companies. A similar approach to abstraction was used in [May94], which focuses on perspectives to discuss association types. The Zachman framework introduces six perspectives or angles on the course of events in an enterprize: "who", "when", "what", "why", "how" and "where". These interrogatives represent abstraction concepts that indicate valuable perspectives on the state of affairs of a given company. These can be applied in a number of cases to various things one wishes to have more complete information about.

The basic idea in the Zachman framework is that, given a well-defined conceptual framework (i.e. the company at hand) abstraction concepts (provided by the mentioned interrogatives) can be introduced within the framework. Outside such framework abstraction concepts are threatened of being extensionally ill-defined.

For example the interrogative "what" (What things are specified by given characteristics?) results in answers that are meaningful to a given individual I if applied to the state of affairs of a given company. Presupposing an unrestricted scope out of which answers to the "what" interrogative can be chosen often would result in an overwhelming amount of -in total- meaningless data.

3 Information Systems

According to [HKL95] Langefors already in 1972 defined the notion information system from a functional point of view as a technically implemented media for recording, storing, disseminating linguistic expressions as well as for deriving linguistic expressions from given ones. Linguistic expressions here are understood as (composite) signs that are valid instances of a particular language. Using information systems thus is understood in this paper as recording, storing and processing sentences of a language in a technically implemented media and retrieving such sentences from this media. Using information systems therefore can be understood as a turn taking of sending and receiving messages, i.e., as a particular kind of communication. In this paper that what is actually communicated, i.e., transferred from sender to receiver is a pattern. Users of information systems by means of these patterns refer to something different from these. Thus, as was already pointed out by [LM78], using information systems involves linguistic perception.

According to the structural point of view definition of the notion information system (see, [HKL95] and [BS98]) such a system has to meet a purpose that is defined by those who arrange an information system to be made effective. Certain business functions shall be supported by using the information system. Those using the information system thus are supposed to have a task at hand and for solving it need some information that they obtain by processing the linguistic expressions they retrieve from the information system.

Though not always recognized, quite a bit of cognitive processing is involved in information system usage. Firstly, based on the business function or purpose of the information system users chose an inquiry fitting the task at hand and launch it against the information system. Secondly, the users try to make efficient use of the system reply to their inquiry. The first one of these steps involves abstraction as it implies the transition from phenomena to predefined generic notions. It further involves a means-ends analysis for determining the system operation best suited to the task at hand. The second of the mentioned steps involves an operation carried out that is inverse to abstraction, i.e., to transition from generic notions to phenomena.

4 Semantic Models

In Information Systems, Informatics and Computer Science, modelling often is considered key. Often models are created by using a so called semantic model. [May94] used the idea of perspective to explain how association in object models can be understood. Object role modelling particularly focused on the idea of association type perspective, i.e. the notion "role". Perspectives introduce a separation of concern into cohesion between objects (occupying the perspectives). This is the basic idea in the meta model represented in figure 1 that defines the notion "semantic model".

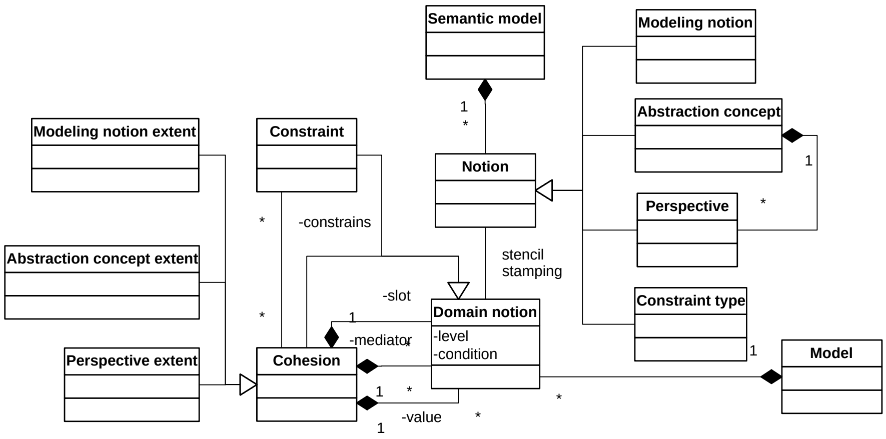

Figure 1: A meta model for semantic models

According to this diagram semantic models consist of notions that either are modelling notions, abstraction concepts, perspectives or constraint types. Abstraction concepts consist of perspectives. E.g. in the Entity Relationship model the abstraction concept "association type" consists of one perspective only: "role". Often it is required that this perspective for a particular association type that is considered as a domain notion is instantiated (at least) two times. For example, regarding the association type "spouse" the the notion "role" is instantiated twice to the role instances "husband" and "wife". The abstraction concepts "generalization" and "aggregation" respectively consist of the two perspectives: "super" vs. "sub", and "whole" vs. "part". To avoid multiple inheritance the "super" perspective is usually limited to being instantiated exactly once while the "sub" perspective may be instantiated several times. Similar conventions are usually observed for "aggregation".

The diagram shows that constraints and cohesions are domain notions. We constrain the diagram by specifying that constraints are stampings of constraint types and that cohesions are stampings of abstraction concepts or of perspectives. Cohesions then are aggregates of domain notions that play roles "slot", "value" and "mediator". Domain notions in the role of value occupy the slot created by a domain notion in the role "slot". Cohesion sometimes is established by a mediator. This, e.g. is the case for entity types that are related to each other by an association type. Similarly, a role instance, i. e., a stamping of the perspective "role" of the abstraction concept "association type" may establish a cohesion between a particular entity type and a particular association type. For example, the role instance "wife" establishes a cohesion between entity type EMPLOYEE and the association type "spouse". There is a corresponding cohesion at the extent level.

The attributes "level" and "condition" may be used to restrict the capability of defining cohesion by a given semantic model. For example, the HERM, see [Tha00], defines the level of entity types as 0. It furthermore defines the level of association types as positive integer. An association type with level l is allowed to establish cohesion between a sequence of database types (i.e., entity types or association types) of level $l' \leq l$ provided one of the sequence items has a level equal to $l - 1$.

Constraints according to the diagram restrict cohesions. Relevant constraint types, e.g. are key dependencies, functional dependencies, extent thresholds and look-up constraints or participation constraints. The former restrict entity type extents or association type extents, i.e., respectively cohesion between an entity type and its instances or between an association type and its instances. Similarly, lookup constraints and participation constraints restrict cohesions between entity type extents or between entity type extents and association type extents.

The meta model in figure 1 was constructed as a framework for data models such as the ER model or the HERM. However, it is likely that it fits semantic models as well that are used for behavior- (State Charts) or causality modelling (Petri Nets) rather than structure modelling.

5 Abstraction

Well known abstraction concepts (or what is here considered to be the same: sets of perspectives) that are used in a number of different situations are classification, generalization and aggregation. See for example [RBP$^+$91] as a text in which these concepts are used in structure modelling and in behavior modelling. Further abstraction concepts are discussed, e.g. in [Myl98] and in [Tha00]. While the elder literature (see, e.g. [Pol88, Che76, SS77b, SS77a, HK87, PM88, HK90]) on semantic modelling deals more with abstraction concepts such as the ones mentioned above the newer literature also deals with somewhat different concepts such as "localization abstraction" (see, e.g. [Tha00, WM02]) or "context" (see, e.g. [KSTZ03, KST03]).

5.1 Abstraction defined

We do not aim at full formalization of our theory. In particular we do not define the concepts "notion", "phenomenon", "predicate" and "role". Rather we assume that these are understood based on the examples used or discussed here. We further do not use a formal language. We only use a partially formalized technical English and presuppose that an a-priori semantics (that we do not define) provides a reasonable meaning for the basic terms we are going to use.

Let m, n be positive integers and $\mathcal{T}$ a set. A **sequence** of length m over $\mathcal{T}$ is a mapping $\mathcal{L} : \{1, \ldots, m\} \to \mathcal{T}$ and thus is a set $\{(i, \mathcal{L}(i)) \mid i \in \{1, \ldots, m\}, \mathcal{L}(i) \in \mathcal{T}\}$. Let $\mathcal{L}$ be a sequence of length m over $\mathcal{T}$. If $n \leq m$ then a **subsequence** $\mathcal{S}$ of length n of $\mathcal{L}$ is a function $\mathcal{S} : \{1, \ldots, n\} \to \mathcal{T}$ that can be extended to $\mathcal{L}$, i.e., for which holds $\mathcal{L} \mid_{\{1,\ldots,n\}} = \mathcal{S}$. For a set S respectively the set of its subsets and the set of words over it is denoted with $\mathcal{P}(S)$ and S^*.

Definition 1 *Let I be a modeler, m be a positive integer, $\mathcal{T}$ a set of notions defined by I and $\mathcal{L}$ a sequence of length m over $\mathcal{T}$. A* ***cohesion predicate*** *$P(\mathcal{L})$ (defined by I) is a predicate that specifies a conceptual cohesion $C_{\mathcal{L}}$, i.e., a mapping $C_{\mathcal{L}} : \mathcal{P}(\{(i, e_i) \mid i \in \{1, \ldots, m\}, e_i \in \varepsilon(\mathcal{L}(i))\}) \to \{true, false\}$ and ascribes a role, i.e., specific function or meaning R_i in $C_{\mathcal{L}}$, to the phenomena in $\varepsilon(T_i)$, for $T_i = \mathcal{L}(i) \in T$, $\forall i \in \{1, \ldots, m\}$. The cohesion $C_{\mathcal{L}}$ is then said to be valid for a set S, iff $C_{\mathcal{L}}(S) = true$ holds.*

Example 1 *Let I be a modeler, $m = 2$, $\mathcal{T} = \{EMPLOYEE\}$, and $\mathcal{L} : \{1, 2\} \to \mathcal{T}$, $i \mapsto EMPLOYEE$. I defines a cohesion predicate $P(\mathcal{L})$ called "spouse" that specifies the cohesion $C_{\mathcal{L}}$ that I considers to be valid regarding regarding an employee set $\{e, f\}$ iff e and f are married to each other. The roles ascribed to EMPLOYEE (modifying the actual legal specification significantly) are $R_1 = wife$, and $R_2 = husband$.*

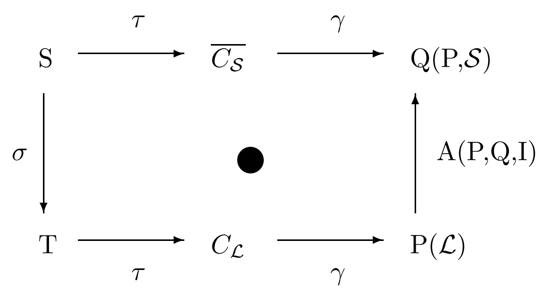

Figure 2: Structure of the situation in definition 2.

Definition 2 *Let I be a modeler. Presuppose that I defines a set $\mathcal{T}$ of notions, a sequence $\mathcal{L}$ of length m over $\mathcal{T}$, and a cohesion predicate $P(\mathcal{L})$. Assume further that the individual I for a sub-sequence $\mathcal{S}$ of $\mathcal{L}$ of length n defines the predicate $Q(P, \mathcal{S}) : \mathcal{P}(\{(i, e_i) \mid i \in \{1, \ldots, n\}, e_i \in \varepsilon(\mathcal{S}(i))\}) \to \{true, false\}$, $S \mapsto true$, iff $\exists T \subseteq \{(i, e_i) \mid i \in \{1, \ldots, m\}, e_i \in \varepsilon(\mathcal{L}(i))\}$, with $S \subseteq T$, and $C_{\mathcal{L}}(T) = true$. Then the predicate $A(P, Q, I)$ means that I proceeds from $P(\mathcal{L})$ to $Q(P, \mathcal{S})$, i.e., that I in $C_{\mathcal{L}}$* ***abstracts*** *from the notions (and their roles) in $\{(i, \mathcal{L}(i)) \mid i \in \{1, \ldots, m\}\} \setminus \{(i, \mathcal{S}(i)) \mid i \in \{1, \ldots, n\}\}$. Then $Q(P, \mathcal{S})$ is called* ***abstraction*** *from $P(\mathcal{L})$.*

The structure of the situation of definition 2 is highlighted in figure 2. In this symbolic commutative diagram σ stands for subset inclusion, τ for a cohesion being valid for a set, and γ for a cohesion being specified by a predicate.

Example 2 *We give an example of a cohesion predicate that presupposing particular universes of discourse either can be abstracted from a given predicate or cannot be abstracted from the given predicate.*

Let I be a modeler and $\mathcal{T} = \{EMPLOYEE\}$ a singleton notion set. Let furthermore be $m = 3$ and $\mathcal{L} : \{1,2,3\} \to \mathcal{T}$, $i \mapsto EMPLOYEE$. Let $P(\mathcal{L})$ be the cohesion predicate "is child of" specifying a cohesion $C_{\mathcal{L}}$ that I considers to be valid for an employee set $\{e, f, g\}$ iff g is a common child of f and g. The (non-optional) roles introduced by $P(\mathcal{L})$ are $R_1 = father$, $R_2 = mother$, and $R_3 = child$. Let furthermore be $n = 2$ and $\mathcal{S} = \{(1, EMPLOYEE), (2, EMPLOYEE)\} \subseteq \mathcal{L}$. Let $P(\mathcal{S})$ be the cohesion predicate "spouse" specifying the cohesion $C_{\mathcal{S}}$ that I considers to be valid for an employee set $\{e, f\}$ iff e and f are married to each other. Let the roles being introduced by $P(\mathcal{S})$ be $R_1^{\mathcal{S}} = husband$ and $R_2^{\mathcal{S}} = wife$. Then $P(\mathcal{S})$ (presupposing an idealized conservative universe of discourse in which employees -sooner or later- have common children iff they -again sooner or later- are married to each other) equals $P(Q, \mathcal{S})$ and thus can be abstracted from $P(\mathcal{L})$ by applying $A(P, Q, I)$. However, regarding a universe of discourse in which employees e, f exist who are married without ever having common children $P(\mathcal{S})$ is not equal to $P(Q, \mathcal{S})$ since in this universe of discourse the cohesion $C_{\mathcal{S}}$ would be considered valid for $\{e, f\}$ but could not be extended to a set $\{e, f, g\}$, for which $C_{\mathcal{L}}$ would be valid, for all employees g. In the modified universe of discourse the "spouse" predicate thus could not be considered as an abstraction of the "is child of" predicate.

Example 3 *We show below that two abstraction concepts that are well-known in conceptual modelling can be understood as abstraction in the sense of definition 2. In the sequel let I be an individual who has identified a universe of discourse UoD. In the sense of an a-priori semantics (as mentioned above) we presuppose that with respect to UoD the meaning of "generalization", "super entity type" and "subentity type" as well as "aggregation", "whole entity type" and "part entity type" are defined (and that these definitions are consistent with the general usage of the terms). Consider for simplicity the case of an extended Entity-Relationship model.*

- Let m be a positive integer and $\mathcal{T}$ be a set of notions from UoD that have been classified as value type. Let further E be an entity type from UoD. Let $\mathcal{L}$ be a sequence of length m over $\mathcal{T} \cup \{E\}$, and $P(\mathcal{L})$ the cohesion predicate that specifies the cohesion $C_{\mathcal{L}}$ being valid for $e \in \varepsilon(E)$ and values $v_i \in \varepsilon(\mathcal{L}(i))$, $\forall i \in \{1, \ldots, m\}$, iff $\{v_1, \ldots, v_m\}$ exactly is the set of attribute values of e. Let $\mathcal{S}$ be a proper subsequence of $\mathcal{L}$ of length n, G an entity type from UoD the attributes of which exactly are the elements $(i, \mathcal{S}(i))$. Then $Q(P, \mathcal{S})$ is the predicate specifying the cohesion $\overline{C_{\mathcal{S}}}$ that for each $g \in \varepsilon(G)$ is valid iff

$\{w_1, \ldots, w_n\}$ is exactly the set of attribute values of g, where $w_i \in \varepsilon(\mathcal{S}(i))$ holds for $i \in \{1, \ldots, n\}$. Then $A(P, Q, I)$ is an abstraction that obtains G as super entity type or generalization of E.

- Let m be a positive integer and $\mathcal{T}$ be a set of notions from UoD that have been classified as entity type. Let furthermore W be an entity type from UoD and $\mathcal{L}$ be a sequence of length m over $\mathcal{T} \cup \{W\}$. Let W be defined as the aggregation of the sequence items different from W. Denote the cohesion predicate with $P(\mathcal{L})$ specifying the cohesion $C_{\mathcal{L}}$ that is valid for a set $\{p_i \mid i \in \{1, \ldots, m\}, p_i \in \varepsilon(\mathcal{L}(i))\}$ of part entities and a whole entity $w \in \varepsilon(W)$ iff the set $\{p_i \mid i \in \{1, \ldots, m\}\}$ exactly is the set of parts of w. Let $(i, L) \in \mathcal{L}$ and $\mathcal{S} = \{(i, L)\}$. Then the whole-part cohesion between $w \in \varepsilon(W)$ and its parts in $\cup_{j \in \{1,\ldots,m\} \setminus \{i\}} \varepsilon(\mathcal{S}(j))$ is $\overline{C_{\mathcal{S}}}$ and the predicate specifying it is $Q(P, \mathcal{S})$. Then the predicate $A(P, Q, I)$ means to obtain a part entity-type $\mathcal{L}(i)$ as an abstraction of the whole entity type W. This kind of abstraction already was investigated in [Kas96].

Example 4 *We show that finite structures over a given signature can be specified by a cohesion predicate. We furthermore show that substructures of finite structures can be understood as abstractions from their super structures. We first introduce the necessary terminology and specify then the cohesion predicate $P(\mathcal{L})$ and show which subsequences $\mathcal{S}$ of $\mathcal{L}$ to chose for deriving substructures by abstraction from their super structures. The construction will make it evident that abstractions can be chosen that do not result in substructures*

Let $Sorts$, Ω and $\Re$ respectively be finite sets of so-called **sorts, function symbols** and **relation symbols**. Then a signature Σ is a four-tuple $(Sorts, \Omega, \Re, arity)$, where $arity$ is a mapping $arity : \Omega \cup \Re \rightarrow Sorts^* \times Sorts$ that assigns to each function symbol or relation symbol ξ its **type** $arity(\xi)$, i.e., the input sorts $s_1, \ldots s_x$ and its output sort s. A **finite structure** $\mathcal{A}$ over Σ is a triple $(\{A_s\}_{s \in Sorts}, \mathcal{O}, \mathcal{R})$ if (1) A_s is a finite set[1], $\forall s \in Sorts$, (2) $\forall o \in \mathcal{O} \exists \omega \in \Omega$, such that $o : A_{s_1} \times \ldots \times A_{s_x} \rightarrow A_s$ and $arity(\omega) = (s_1 \ldots s_x, s)$, and (3) $\forall r \in \mathcal{R} \exists \rho \in \Re$, such that $r \subseteq A_{s_1} \times \ldots \times A_{s_y} \times A_s$ and $arity(\rho) = (s_1 \ldots s_y, s)$. See, e.g. [EMC$^+$99] for more detail on structures. Let $A = \cup_{s \in Sorts} \{s\} \times A_s$ and call it the **support** of $\mathcal{A}$. Let for $x \in \Omega \cup \Re \cup \mathcal{O} \cup \mathcal{R} \cup A$ be x^* the individual notion of x, i.e., the notion with $\varepsilon(x^*) = \{x\}$. A **substructure** $\mathcal{B}$ of a structure $\mathcal{A}$ over a signature Σ is a structure over Σ on a support $B \subseteq A$ such that the restriction of mappings and relations from $\mathcal{A}$ to B respectively is a mapping or relation of $\mathcal{B}$.

Let now be $\Sigma = (Sorts, \Omega, \Re, arity)$ a signature and $\mathcal{A}$ a structure over Σ. Let further be $m = \mid \Omega \mid + \mid \Re \mid + \mid \mathcal{O} \mid + \mid \mathcal{R} \mid + \mid A \mid$ and $\mathcal{T} = \{x^* \mid x \in \Omega \cup \Re \cup \mathcal{O} \cup \mathcal{R} \cup A\}$. Let finally be $\mathcal{L} : \{1, \ldots, m\} \rightarrow \mathcal{T}$ be an injective sequence of length m over $\mathcal{T}$ and $P(\mathcal{L})$ the cohesion predicate specifying the cohesion $C_{\mathcal{L}} : \mathcal{P}(\{(i, e_i) \mid i \in \{1, \ldots, m\}, e_i \in \varepsilon(\mathcal{L}(i))\}) \rightarrow \{true, false\}$ for which the set P_2 of second components of $\cup_{C_{\mathcal{L}}(S)=true} C_{\mathcal{L}}^{-1}(S)$ contains all support elements, all mappings and

[1]Rather than with A_s the set interpreting sort s in $\mathcal{A}$ often is denoted with s^A.

relations from $\mathcal{A}$, and all function- and relation symbols in Σ, and either of the following cases holds for each subset S of $\{(i, e_i) \mid i \in \{1, \ldots, m\}\, e_i \in \varepsilon(\mathcal{L}(i))\}$:

1. S contains exactly one function symbol $\sigma \in \Omega$ with type $arity(\sigma) = (s_1 \ldots s_x, s)$, exactly one mapping $f : A_{s_1} \times \ldots \times A_{s_x} \to A_s$ and nothing else apart from $a_1 \in A_{s_1}, \ldots, a_x \in A_{s_x}, a \in A_s$, with $f(a_1, \ldots, a_x) = a$.

2. S contains exactly one relation symbol $\rho \in \Re$ with type $arity(\rho) = (s_1 \ldots s_y, s)$, exactly one relation $r \subseteq A_{s_1} \times \ldots \times A_{s_y} \times A_s$ and nothing else apart from $a_1 \in A_{s_1}, \ldots, a_y \in A_{s_y}, a_1^s, \ldots, a_z^s \in A_s$ with $r(a_1, \ldots, a_y) = \{a_1^s, \ldots, a_z^s\}$.

The cohesion predicate $P(\mathcal{L})$ specifies the structure $\mathcal{A}$. Choosing subsequences $\mathcal{L}$ of $\mathcal{S}$ that are complete with respect to application of the contained functions and relations results in a cohesion $\overline{C_\mathcal{S}}$ that is specified by the predicate $Q(P, \mathcal{S})$ and determines a substructure of $\mathcal{A}$. Obviously all substructures of $\mathcal{A}$ can thus be abstracted from $\mathcal{A}$ by the procedure characterized by $A(P, Q, I)$.

Remark 1 *Structures are of particular interest for Applied Informatics since they appear to be natural candidates for mathematical models of semantic models. We show below how -at the syntactical level- a simple version of the ER-model (ERM) can be modelled as a signature. The structures over ERM would then be considered as ER-models. It is a straightforward task to invent comparable mathematical models for other semantic models such as Petri Nets or State Charts. Note, however, that in general a variety of candidates can be defined and one needs to make a reasonable choice.*

According to the meta model in figure 1 we define this version of the ER-model to consist of the modelling notions **entity type** and **value type** and the abstraction concepts **association type** and **characteristic type**. We only introduce the one perspective **role** and the constraint type **participation constraint**. These notions respectively are referred to as **E,V,A,C,R** and **P**. We define a signature $ERM = (Sorts, \{\pi_R\}, \{\gamma_E, \gamma_A, \rho\}, arity)$ where $arity$ is defined as $arity : \{\pi_R, \gamma_E, \gamma_A, \rho\} \to Sorts^* \times Sorts$, with $Sorts = \{E, V, A, C, R, P\}$ and $arity(\pi_R) = (R, P)$, $arity(\gamma_E) = (VE, C)$, $arity(\gamma_A) = (VA, C)$, and $arity(\rho) = (EA, R)$.

Let $\mathcal{B}$ be a structure over ERM. Denote its mapping with π_R^B and denote its relations (in the canonical way) with $\gamma_E^B, \gamma_A^B, \rho^B$. Then γ_E^B, γ_A^B respectively specify the characteristics c_e, c_a that associate a given value type v (i.e., an element of V^B) to a given entity type e (i.e., an element of E^B) or a given association type a (i.e. an element of A^B) meaning that c_e or c_a associates v to e or to a. Similarly, the relation ρ^B associates to an entity type e and an association type a the roles $r_1, \ldots, r_s \in R^B$ that e plays in a. Finally the mapping π_R^B associates to each role r the participation constraint p that restricts the participation of the entities $x \in \varepsilon(e)$ in associations $y \in \varepsilon(a)$. Clearly, given an ER-model $\mathcal{B}$ over ERM a database can be specified by interpreting its entity types, association types and value types as the respective notion extent and doing the same thing regarding the function and the relations of $\mathcal{B}$.

Remark 2 *Referring back to our short account of Stachowiak's model theory it is a reasonable idea, to define for a signature Σ and finite structures $\mathcal{A}, \mathcal{B}$ over Σ: $\mathcal{A}$ is a* ***model*** *of $\mathcal{B}$ if there respectively exist substructures $\mathcal{S}_{\mathcal{A}}$ and $\mathcal{S}_{\mathcal{B}}$ of $\mathcal{A}$ and $\mathcal{B}$ that are isomorphic to each other. Then models of structures appear to be abstractions of structures but not vice versa.*

5.2 A few generic conceptual frameworks

Several cohesion predicates in IS are frequently used as conceptual framework regarding which abstraction is carried out. To discuss these let I be an individual introducing them and having a particular universe of discourse UoD defined for each of the conceptual framework in the following bullet points.

- **System**. Let $m = 5$ and $\mathcal{T} = \{S, In, Out, R, C\}$ be a set of notions from UoD such that $\varepsilon(X)$ is a singleton set the element of which is denoted by $\varepsilon(X)$, for each $X \in \mathcal{T}$. Let $\mathcal{L} : \{1, \ldots, 5\} \to \mathcal{T}$, $1 \mapsto S$, $2 \mapsto In$, $3 \mapsto Out$, $4 \mapsto R$, and $5 \mapsto C$. Let $P(\mathcal{L})$ be the cohesion predicate specifying the cohesion $C_{\mathcal{L}}$, which I considers to be valid iff $\varepsilon(In), \varepsilon(Out), \varepsilon(C)$ respectively are sets of so-called **inputs**, **outputs** and **system components** and $\varepsilon(R) \subseteq \varepsilon(In) \times \varepsilon(Out)$ and $\varepsilon(S)$ is the so called **system** that consists of the system components and that realizes the **input-output relation** $\varepsilon(R)$. Let I presuppose this relation being realized in the following way: Firstly, all system components are capable of exchanging (parametric) service requests. Secondly, all system components can meet a (component specific) set of such service requests and reply an appropriate result. Thirdly, for each system input there is a system component that takes this input as a service request. Fourthly, for each system input i in a sequence of service requests and result provisions a system output o is determined such that $(i, o) \in \varepsilon(R)$.

  The individual I may additionally introduce subsequences of $\mathcal{L}$ of length 1 as $\mathcal{S}_4 = \{(4, R)\}$ and $\mathcal{S}_5 = \{(5, C)\}$. These then give rise to the predicates $Q(P, \mathcal{S}_4)$ and $Q(P, \mathcal{S}_5)$ that specify the cohesions $\overline{C_{\mathcal{S}_4}}$ and $\overline{C_{\mathcal{S}_5}}$ and respectively introduce the perspectives "what" and "how" as the abstraction denoted by $A(P, Q_4, I)$ and $A(P, Q_5, I)$.

  The component interaction is often understood as an exchange of matter, energy or information. However, the interaction may also be considered as a purely logical one. See, e.g. [Wym84, VG91, Luh02] as introductory texts into system theory. See finally [Ste90] concerning the origin of the system concept and its relevance during the European reformation and for hermeneutics.

- **Space**. Let $m = 5$ and $\mathcal{T} = \{S, L, O, T, f\}$ be a set of notions from UoD such that $\varepsilon(X)$ is a singleton set the element of which is denoted by $\varepsilon(X)$, for each $X \in \mathcal{T}$. Let $\mathcal{L} : \{1, \ldots, 5\} \to \mathcal{T}$, $1 \mapsto S$, $2 \mapsto L$, $3 \mapsto O$, $4 \mapsto T$, and $5 \mapsto f$. Let $P(\mathcal{L})$ be the cohesion predicate specifying the cohesion $C_{\mathcal{L}}$, which I considers to be valid iff $\varepsilon(S)$ is the so-called **space**, $\varepsilon(L), \varepsilon(O)$ are sets that respectively contain **locations** and **objects**, which may be attached to these locations, $\varepsilon(T) \subseteq \varepsilon(L) \times \varepsilon(L)$ is a location transition relation (specifying which locations of the space are connected) and $\varepsilon(f) : L \to O$ is a partial mapping specifying the objects that are attached to locations within the space.

The individual I may additionally introduce subsequences of $\mathcal{L}$ of length 1 as $\mathcal{S}_2 = \{(2, L)\}$ and $\mathcal{S}_3 = \{(3, O)\}$. These then give rise to the predicates $Q(P, \mathcal{S}_2)$ and $Q(P, \mathcal{S}_3)$ that specify the cohesions $\overline{C_{\mathcal{S}_2}}$ and $\overline{C_{\mathcal{S}_3}}$ and respectively introduce the perspectives "where" and "what" as the abstraction denoted by $A(P, Q_2, I)$ and $A(P, Q_3, I)$.

The locations may be characterized by a scale value for each out of a set of so-called dimension along which the space unfolds. As was the case with the component interaction in a system, the space can be a logical space and the occupation relation be a logical one. For a very brief exposition of Kant's position characterizing our ordinary everyday space as a necessary precondition of all perception see [RM98].

- **Field**. Let $m = 6$ and $\mathcal{T} = \{F, L, S, M, f\}$ be a set of notions from UoD such that $\varepsilon(X)$ is a singleton set the element of which is denoted by $\varepsilon(X)$, for each $X \in \mathcal{T}$. Let $\mathcal{L} : \{1, \ldots, 6\} \to \mathcal{T}$, $1 \mapsto F$, $2 \mapsto L$, $3 \mapsto L$, $4 \mapsto S$, $5 \mapsto M$, and $6 \mapsto f$. Let $P(\mathcal{L})$ be the cohesion predicate specifying the cohesion $C_{\mathcal{L}}$, which I considers to be valid iff $\varepsilon(F)$ is the so called **field** that has a set $\varepsilon(L)$ of locations at which a directed force drives particles of a given mass into a location with a certain strength. The directed force is given by the mapping $\varepsilon(f) : \varepsilon(L) \times \varepsilon(M) \to \varepsilon(L) \times \varepsilon(S)$.

  The individual I may introduce subsequences of $\mathcal{L}$ of length 1 as $\mathcal{S}_2 = \{(2, L)\}$ (source location), $\mathcal{S}_3 = \{(2, L)\}$ (target location), and $\mathcal{S}_4 = \{(4, S)\}$, (strength). These give rise to cohesions $\overline{C_{\mathcal{S}_i}}$ that respectively are specified by predicates $Q(P, \mathcal{S}_i)$, for $i \in \{2, 3, 4\}$. These predicates respectively introduce the abstractions $A(P, Q_i, I)$ that in turn respectively introduce the perspectives "where", "whither" and "how much" on the field.

  The directed force represented by $\varepsilon(f)$ may be considered as constant or subject to change in direction, strength or in how it affects particles. To include this variation into the predicate a further perspective **state** is required.

- **Story**. Let $m = 10$ and $\mathcal{T} = \{B, S, A, O, Z, E, M, p, f, g\}$ be a set of notions from UoD such that $\varepsilon(X)$ is a singleton set the element of which is denoted by $\varepsilon(X)$, for each $X \in \mathcal{T}$. Let $\mathcal{L} : \{1, \ldots, 10\} \to \mathcal{T}$, $1 \mapsto B$, $2 \mapsto S$, $3 \mapsto A$, $4 \mapsto O$, $5 \mapsto Z$, and $6 \mapsto E$, $7 \mapsto M$, $8 \mapsto p$, $9 \mapsto f$, and $10 \mapsto g$. Let $P(\mathcal{L})$ be the cohesion predicate specifying the cohesion $C_{\mathcal{L}}$, which I considers to be valid iff $\varepsilon(B)$ is the so-called **story board** that respectively has sets $\varepsilon(S)$, $\varepsilon(A)$, $\varepsilon(O)$, $\varepsilon(Z)$, $\varepsilon(E)$, $\varepsilon(M)$, of **scenes**, **actors**, **operations**, **tendencies**,

events, and media **actors**, and where $\varepsilon(p) \subseteq \varepsilon(S) \times \varepsilon(S)$ is the so-called plot[2], a relation specifying succession of scenes, and where $\varepsilon(f) : \varepsilon(S) \times \varepsilon(A) \to \varepsilon(Z)$ is a mapping specifying the tendencies of actors in scenes, and where $\varepsilon(g) : \varepsilon(S) \times \varepsilon(A) \times \varepsilon(E) \to \varepsilon(O) \times \varepsilon(M)$ is a mapping specifying the operation and the media used by an actor to do it and that are triggered by an event in a scene.

The individual I may introduce subsequences $\mathcal{S}_i$ of $\mathcal{L}$ of length one as $\mathcal{S}_i = \{(i, \mathcal{L}(i))\}$, for $i \in \{2, \ldots, 7\}$. These give rise to predicates $Q(P, \mathcal{S}_i)$ that specify cohesions $\overline{C_{\mathcal{S}_i}}$ and allows the abstractions $A(P, Q(P, \mathcal{S}_i), I)$ to be carried out. These then define the perspectives "context", "who", "what", "what for", "why" and "whereby" on stories.

These predicates are capable of being used several at a time. For example it is possible to consider a system of spaces and fields or a field superposing a space or similar.

The system predicate has found a number of quite well known applications in IS. To perhaps a lesser degree this holds true for the space predicate: Name-, information-, problem-, and solution spaces are in use. Functional modelling (see [RBP+91]) appears to be an example for the use of the field predicate. However, it is only used in a restricted form where the field strength is constant over all locations of the field and only one particle type, i.e., data is used. Furthermore state charts ([HN96, Har87]) and Petri Nets appear as an example of the field framework. The story predicate was extensively used in story boarding (see, e.g. [KSWM03, SKMW02] and the references given there). In this context the story predicate is used for the development of Web information systems and describes customer interaction with such systems. The Zachman Framework can be seen as significantly overlapping the views of the story framework. See figure 3 for the more important of our conceptual framework interrogatives and the Zachman framework interrogatives being integrated.

An entity that is considered as being used in one of these predicates can be turned into a subject or result of abstraction. One can prescind of everything in the respective predicate that is not related to the role of this entity. In the space framework, e.g. one can prescind from the particular thing that occupies a certain location and only consider the location as important. This is how localization abstraction (see, e.g. [Tha00]) could be explicated.

6 Resume

Above we have argued that abstraction in the sense of prescinding from certain aspects of a phenomenon should take place in a conceptual framework that can

[2]Plots of movies sometimes have more than one end. This is the case, e.g. for "A Perfect Murder" with Michael Douglas starring. We therefore do not restrict the plot to be a linear order. In particular there could quite well be several different beginnings.

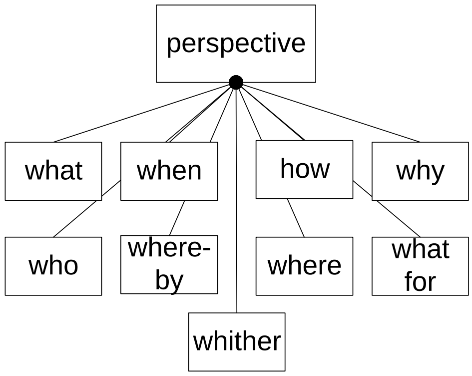

Figure 3: Zachman's basic angles and 3 perspectives added

guarantee that no undesirable effects result from the abstraction carried out. We have discussed the concept of semantic model. A semantic model according to our definitions is a system of modelling notions and abstraction concepts. The latter in turn are defined to consist of perspectives. We have argued that the Zachman framework essentially applies this idea to enterprizes. We have then shown how the idea to carry out abstraction within a predefined conceptual framework can be turned into a definition of (a particular model of) abstraction. We finally have specified a number of generic conceptual frameworks that, as we believe, are frequently used in information systems development as context of abstraction.

References

[AU96] V. Aho, Alfred and D. Ullman, Jeffrey. *Informatik, Datenstrukturen und Konzepte der Abstraktion*. International Thomson Publishing GmbH, Bonn et al., 1996.

[BS98] Peter Bernus and Günther Schmidt. Architecture of information systems. In Peter Bernus, Kai Mertins, and Günther Schmidt, editors, *Handbook on Architectures and Information Systems*, chapter 1, pages 1 – 9. Springer Verlag, Berlin et al., 1998.

[Che76] Peter P. Chen. The Entity-Relationship Model: Toward a Unified View of Data. *ACM Transactions on Database Systems*, 1(1):9–37, 1976.

[EMC+99] Hartmut Ehrig, Bernd Mahr, Felix Cornelius, Martin Große-Rhode, and Philip Zeitz. *Mathematisch-strukturelle Grundlagen der Informatik*. Springer - Verlag, Berlin, Germany et al., 1999.

[Har87] David Harel. Statecharts: a Visual Formalism for Complex Systems. *Science of Computer Programming*, 8:231–274, 1987.

[HK87] Richard Hull and Roger King. Semantic Database Modeling: Survey, Applications, and Research Issues. *ACM Computing Surveys*, 19(3):201–260, 1987.

[HK90] Richard Hull and Roger King. A Tutorial on Semantic Database Modeling. In A. F. Cardenas and D. McLeod, editors, *Research Foundations in Object Oriented and Semantic Database Systems*, pages 1–33. Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[HKL95] Rudy Hirschheim, Heinz K. Klein, and Kalle Lyytinen. *Information Systems Development and Data Modeling, Conceptual and Philosophical Foundations*. Cambridge University Press, Cambridge, 1995.

[HN96] David Harel and Amnon Naamad. The STATEMATE Semantics of Statecharts. *ACM Transactions on Software Engineering and Methodology*, 5(4):293–333, Oktober 1996.

[Kas96] Roland Kaschek. Inheritance as a Conceptual Primitive. In Bernhard Thalheim, editor, *Conceptual Modeling- ER'96*, pages 406–421, Berlin, Heidelberg, 1996. Springer-Verlag.

[Kel95] Rudi Keller. *Zeichentheorie*. Francke Verlag, Tübingen, Basel, 1995.

[KST03] Roland Kaschek, Klaus-Dieter Schewe, and Bernhard Thalheim. Modeling context in web information systems. In *Short Paper of CAiSE'03*, Berlin et al., 2003.

[KSTZ03] Roland Kaschek, Klaus-Dieter Schewe, Bernhard Thalheim, and Lei Zhang. Integrating context in conceptual modeling for web information systems. 2003. WES'03 Proceedings, 2003.

[KSWM03] Roland Kaschek, Klaus-Dieter Schewe, Catherine Wallace, and Claire Matthews. Story boarding for web-based information systems. 2003. Accepted for publication as book chapter.

[LM78] Peter C. Lockemann and Heinrich C. Mayr. *Rechnergestützte Informationssysteme*. Springer-Verlag, Berlin, Heidelberg et al., 1978.

[Lud03] Jochen Ludewig. Models in software engineering - an introduction. *Software and Systems Modeling*, 2(1):5 – 14, March 2003. Reworked version of Ludewig's presentation at "Modellierung 2002".

[Luh02] Niklas Luhmann. *Einführung in die Systemtheorie*. Carl-Auer-Systeme Verlag, Heidelberg, 2002.

[Mat02] Margaret W. Matlin. *Cognition*. John Wiley & Sons, New York et al., 5th. edition, 2002.

[May94] Heinrich C. Mayr. Charakterisierende Objekte: Ein Modell für die Objektorientierte Analyse mit oder ohne Assoziationen. In Udo W. Lipeck and Gottfried Vossen, editors, *Formale Grundlagen für den Entwurf von Informationssystemen*, pages 59 – 62. University of Hannover, 1994. Published in the series "Informatik - Berichte", No. 03/94.

[MCF03] Stephen J. Mellor, Anthony N. Clark, and Takao Futagami. Model-Driven Development. *IEEE Software*, 20(5):14 – 18, September/October 2003.

[Myl98] John Mylopoulos. Characterizing Information Modeling Techniques. In Peter Bernus, Kai Mertins, and Günter Schmidt, editors, *Handbook on Architectures of Information Systems*, chapter 2, pages 17–57. Springer-Verlag, Berlin, Heidelberg, New York, 1998.

[PM88] Joan Peckham and Fred Marjanski. Semantic Data Models. *ACM Computing Surveys*, 20(3):153–189, 1988.

[Pol88] George Polya. *How to Solve It.* Princeton University Press, Princeton, New Jersey, 1988.

[Qua85] Edward S. Quade. Predicting the consequences: Models and modeling. In Hugh Miser and Edward S. Quade, editors, *Handbook of systems analysis: overview of uses, procedures, applications and practice*, pages 191 – 218. Elsevier Science Publishing Co., Inc., New York, 1985.

[RBP+91] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorensen. *Object-Oriented Modeling and Design.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1991.

[RM98] Arnim Regenbogen and Uwe Meyer. *Wörterbuch der philosophischen Begriffe.* Felix Meiner Verlag, Hamburg, 1998. begründet von Friedrich Kirchner und Carl Michaelis, fortgesetzt von Johannes Hoffmeister und von den Autoren vollständig neu herausgegeben.

[Rot89] Jeff Rothenberg. The nature of modeling. In Lawrence E. Widman, Kenneth A. Loparo, and Norman R. Nielson, editors, *Artifical intelligence, simulation, and modeling*, pages 75–92. John Wiley & Sons, Inc., New York et al., 1989.

[SKMW02] Klaus-Dieter Schewe, Roland Kaschek, Claire Matthews, and Catherine Wallace. Modeling web-based banking systems: Story boarding and user profiling. In Heinrich C. Mayr and van den Heuvel, Willem-Jan, editors, *Proceedings of eCoMo 2002*, Springer LNCS, Berlin et al., 2002. Springer - Verlag.

[SS77a] John Miles Smith and Diane C. P. Smith. Database abstractions: Aggregation. *Communications of the ACM*, 20(6):405–413, Juni 1977.

[SS77b] John Miles Smith and Diane C.P. Smith. Database abstractions: Aggregation and generalization. *ACM Transactions on Database Systems*, 2(2):105–133, 1977.

[Sta73] Herbert Stachowiak. *Allgemeine Modelltheorie.* Springer Verlag, Wien, New York, 1973.

[Sta83] Herbert Stachowiak. Erkenntnisstufen zum Systematischen Neopragmatismus und zur Allgemeinen Modelltheorie. In Herbert Stachowiak, editor, *Modelle-Konstruktionen der Wirklichkeit*, pages 87–146. Wilhelm Fink Verlag, München, 1983.

[Sta92] Herbert Stachowiak. Modell. In Helmut Seiffert and Gerard Radnitzky, editors, *Handlexikon Zur Wissenschaftstheorie*, pages 219–222. Deutscher Taschebuch Verlag GmbH & Co. KG, München, 1992.

[Ste90] Karl Steinbacher. System, Systemtheorie. In Hans Jörg Sandkühler, editor, *Europäische Enzyklopädie zu Philosophie und Wissenschaften*, volume 4th., pages 500–506. Felix Meiner Verlag GmbH, Hamburg, 1990.

[SZ92] John F. Sowa and John A. Zachman. Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, 31(3):590 – 616, 1992.

[Tha00] Bernhard Thalheim. *Entity-Relationship Modeling.* Springer-Verlag, Berlin, Heidelberg, 2000.

[Thi92] Christian Thiel. Abstraktion. In Helmut Seiffert and Gerard Radnitzky, editors, *Handlexikon Zur Wissenschaftstheorie*, pages 5 – 7. Deutscher Taschenbuch Verlag GmbH. & Co. KG, München, 1992.

[TL82] Dionysios C. Tsichritzis and Frederick H. Lochovsky. *Data models*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[VG91] John P. Van Gigh. *System Design, Modeling and Metamodeling*. Plenum Press, New York, London, 1991.

[WM02] Catherine Wallace and Claire Matthews. Communication: Key to success on the web. In Heinrich Mayr and van den Heuvel, Willem-Jan, editors, *Proceedings of eCoMo 2002*, Berlin et al., 2002. Springer Verlag.

[Wym84] A. Wymore. Theory of Systems. In C.R. Vick and C.V. Ramamoorthy, editors, *Handbook of Software Engineering*, pages 119–132, New York, 1984. Van Nostrand Reinhold.

[Zac87] John A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3):276 – 292, 1987.