Structured Modeling with uncertainty

P. van Bommel, H.A. (Erik) Proper and Th.P. van der Weide

Institute for Computing and Information Sciences, Radboud University Nijmegen
The Netherlands, {P.vanBommel, E.Proper, Th.P.vanderWeide}@cs.ru.nl

Abstract. This paper starts with the description of the modeling process as a dialog, and describes the associated formal functions, including the feedback supporting the growing mutual understanding. The dialog has a procedural and an informational aspect. For this latter a controlled grammar is used, that has a user friendly and a system friendly side. These sides are related via an elementary syntactical transformation. Assuming some elementary requirements on the dialog participants, we prove the main theorem for information modeling effectiveness. We also propose a system of metrics to support the modeling process. In terms of these metrics, modeling heuristics can be described and evaluated. We demonstrate our ideas by a simple sample session.

1 Introduction

We start from a fundamental view on the structure of the modeling process (see figure 1). The kernel of this process is a dialog in which the participants exchange information, trying to develop a common understanding. We assume a modeling dialog to consist of dialog actions, performed by the participants. Each dialog action is a contribution of one of the participants of the dialog. For convenience

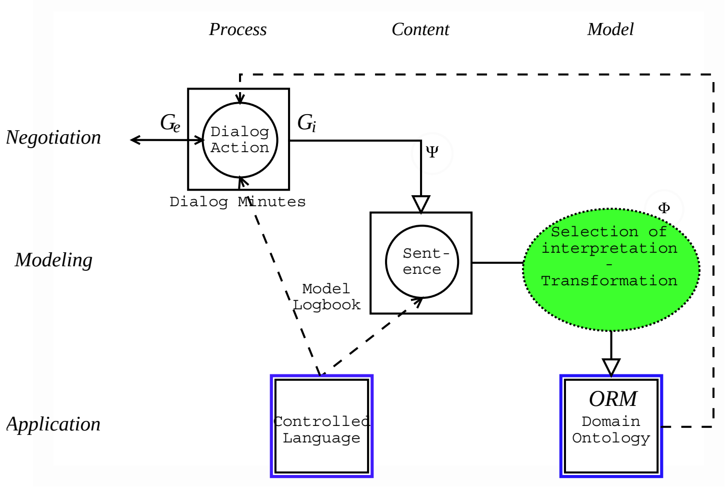

Fig. 1. The modeling dialog

we will restrict ourselves in this paper to dialogs with two special participants, the domain expert and the system analyst. In this paper we will discuss a simple dialog system with a limited repertoire of dialog actions. We will also sketch the extension to the chatbox model, a model for more complex dialogs.

In our simple dialog system, a dialog consists of a sequence of dialog actions. The dialog minutes contain a description of the actions performed, and are assumed to be an agreed and complete representation of the exchange of information between domain expert and system analyst sofar. The ordering of the dialog actions is by dialog time. The structure of this dialog is described by:

*dialog :: (question, answer)**
question :: message.
answer :: message.
message :: time: sentence.

Later we will discuss the structure of sentences. Note that due to the alternation structure of the dialogue, there is no need to qualify messages with their speaker.

The actual information that is conveyed via the dialog has some underlying format, based on a particular controlled language format. In practice, the dialog participants 'speak' this controlled language (possibly supported by the dialog system). We will assume this language to be available in two formats. The grammar G_i describes the format that is best suited for automatic processing, while G_e provides a beautified version of such sentences. These two formats are related by elementary linguistic transformations. We will discuss an example in a later section.

During the dialog, new constructs will be added to the this controlled language as a result of a growing mutual understanding. This mechanism is displayed in figure 1 as a feedback cycle from the domain ontology into the dialog action.

1.1 The dialog minutes

The dialog minutes are transformed by a function called Ψ into the model logbook. This logbook contains sentences that provide a consistent description of the Universe of Discourse as far as this is available in the dialog minutes. We will be interested in cases where this description is a partial description in progress, and examples are being used to provide an extensional domain description from which an intentional one (the domain ontology) is to be derived.

Due to the strict nature of the controlled language, we will restrict ourselves to cases where the model logbook can be processed via formal transformations. We use the function Φ to denote such a transformation.

Inside Φ, there may be several interpretation strategies that help the analyst to use different levels of abstraction. For example, initially, both domain expert and system analyst will prefer a global interpretation as their awareness will probably be at a visceral level (see [1]), and possibly even at a conscious level. In this terminology, the first step should be directed towards reaching the formalized level, after which requirements engineering will bring them to the compromised level.

1.2 The participants

Typically the system analyst will have a number of viewpoints, each of which provides some particular focus on the information provided sofar. Note that this information itself will also cover the various viewpoints from the domain expert. As we assume the system analyst as controlling the dialog, the system analyst will select the most promising view, for further questioning. If the domain expert is at a visceral level, then the system analyst will choose to communicate in terms of examples, which the domain expert is supposed to be capable to handle at this level of awareness.

During the modeling dialog, the informal specification evolves from an incomplete and 'vague' domain description into a formal and precise specification of the domain knowledge (see also [2]). At a general level, terms like 'uncertainty' and 'vagueness' indicate that some information is missing, leading to improper or incomplete understanding. Looking in more detail, various forms of missing or invalid information can be distinguished. A typical example is Smithson's taxonomy of ignorance [3] (figure 2). Although this taxonomy can be argued to be an 'arbitrary' one, it clearly indicates the existence of different types of ignorance, each having its own properties.

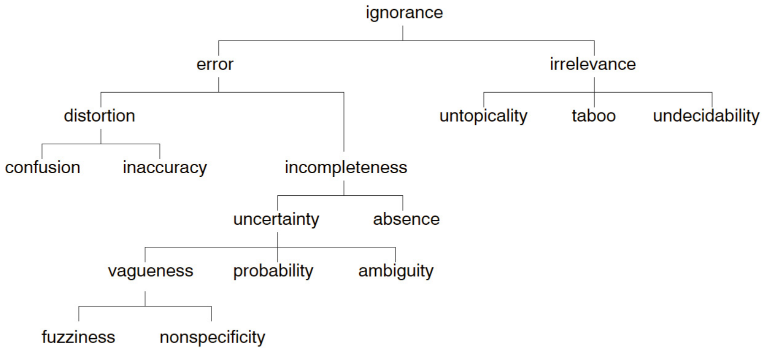

Fig. 2. Smithson's taxonomy

We feel that a modeling technique should handle the uncertainties that the modeling process is affected with. By choosing a proper controlled language, the system analyst may reduce uncertainty, at the cost of an extra effort for the domain expert. In our example session we will not consider uncertainty related with *irrelevance*, *distortion*, *probability*, *ambiguity* or *fuzziness*. A number of combinations of dialog structure and controlled language can be distinguished:

1. Base fact-oriented
   There is a simple elicitation/validation dialog, typically as being used in

the NIAM approach. The controlled language used is FQNF (fully qualified sentence normalform)
2. Base action-oriented
   Also in this case there is a simple elicitation/validation dialog, see [4]. The controlled language used is FQNF (fully qualified sentence normalform)
3. Extended fact-oriented
   A more advanced elicitation/validation dialog is being used. See for example [5] and [6]. The extension is the introduction of elementary dialog actions for negotiation. The controlled language used is UNF (unqualified sentence normalform). We show how the mechanism of uncertainty can be used to describe this situation.

In this paper, we will focus on *Base fact-oriented*. First we will discuss a formal approach to information modeling, discuss the *Main Theorem for Information Modeling Effectiveness*, and introduce some metrics for typical modeling constructs. After this, we will show in a sample session how this could work.

2 Formal requirements

In the fact-based approach we assume the model logbook, the result of the modeling dialog, to be a set of sample sentences. A sample sentence is an example of a description of a conception. This set of sample sentences is the base for the ORM-style modeling process. Basically, the way of working of ORM-modeling is that sample sentences are transformed into populated schema fragments, which (if consistent) are integrated to an overall model, the so-called domain ontology. The quality of the resulting model is directly related to the completeness of this set of sample statements.

In [4] a simple dialog model has been introduced that supports the way of working associated with ORM-modeling. In this model, two participants are assumed, a *domain expert* and a *system analyst*. Their communication channel follows the paradigm of the phone heuristic, only allowing formalized textual information to be exchanged. The system analyst is dominant in this dialog, and either asks another sample sentence, or offers a model description for validation. Underspecification is handled by the system analyst by offering sample populations to the domain expert for validation.

2.1 The main goal of modeling

The goal of the system analyst may be described as follows:

> *Find a minimal generative device (information grammar) capable to generate/accept the sentences of the informal specification, that is maximally expressive.*

Being minimal is to be motivated from the informal specification itself, in the sense that each formal concept is grounded in the informal specification. A forteriori, the introduction of each formal concept then can be related to specific items in the dialogue document without which the introduction of this concept would not be imperative. Being maximally expressive invites the system analyst to introduce abstractions whenever sample sentences seem to have a similar deep structure.

Information modeling deals with all these aspects of modeling, as depicted in figure 1. A theory for information modeling should have as a main theorem:

Theorem 1 (Main Theorem for Information Modeling Effectiveness). *The probability of a model being inadequate, as a function of the dialogue length, tends to zero for the combination of a qualified domain expert and a qualified system analyst.*

In [4] the validity of this theorem is demonstrated for the simple dialog model under the assumption that the behavior of domain expert and system expert is governed by a number of explicitly stated cognitive requirements.

2.2 Generative power

In the remainder of this section, for sample statements we assume some controlled language format. We will describe a fully qualified language, referred to as Natural Language Normalform. The dialog minutes consist of a sequence dialog actions. Let D be such a sequence. From D the set $\Psi(D)$ of sample sentences is derived. Note that this function Ψ will not be monotonic: if sequence D_2 is an extension of D_1, then we can not derive $\Psi(D_1) \subseteq \Psi(D_2)$.

Next we focus on all grammars that may generate this set $S = \Psi(D)$ of sentences:

$$\Upsilon(S) = \{G \mid S \subseteq \mathcal{L}(G)\}$$

Adding extra sample statements poses extra requirements on generating grammars:

$$\Upsilon(S + \sigma) \subseteq \Upsilon(S)$$

where we use $S + \sigma$ to denote the set that results after adding σ to set S.

A formal grammar, denoted as $\langle S, N, T, R\rangle$, is specified by its set of nonterminal symbols (N) including the start symbol S, its terminal symbols (T), and a set R of rules. We call grammar $G_1 = \langle S_1, N_1, T_1, R_1\rangle$ a (structural) subgrammar of grammar $G_2 = \langle S_2, N_2, T_2, R_2\rangle$ if there exists an injection ϕ from terminal and nonterminal symbols from G_1 into those of G_2 mapping S_1 onto S_2 such that the rules from G_1 are injectively mapped on derivations within G_2:

$$\langle lhs, rhs\rangle \in R_1 \Rightarrow \phi(lhs) \rightarrow^*_{G_2} \phi(rhs)$$

Being a subgrammar obviously is a partial order on the set $\Upsilon(S)$ of grammars. The minimal element is the following grammar $G_0(S)$ that start symbol denoted as B, and the following set of rules:

$$\min(S) = \{B \rightarrow s \mid s \in S\}$$

Proof. Obviously $S \subseteq \mathcal{L}(G_0)$. Let G be any grammar from $\Upsilon(S)$, then we map the start symbol B of $G_0(S)$ onto the start symbol of grammar G, and the result follows directly from the definition of the set $\Upsilon(S)$.

The modeling approach will select one of the grammars from $\Upsilon(S)$ as the result of the modeling activity. We will refer to this particular grammar as $\mathsf{G}(S)$.

2.3 Expressiveness

Next let σ be some sentence outside S. The structural distance $\Delta(\sigma, S)$ between σ and S is defined as the minimal distance between σ and any of the sentences from S:

$$\Delta(\sigma, S) = \min \left\{ d(\sigma, s) \;\middle|\; s \in S \right\}$$

Note that this corresponds to the individual approach to compare an instance with a set of instances (see [7]). Structural difference between two sentences is defined on the basis of their parsings in terms of the grammar from which they have been generated. In this particular situation we have sentences from the internal controlled language.

Comparing parse trees is not easy, a number of approaches are related to the comparison of XML parse trees in the context of Information Retrieval (see for example [8]). A special measure is the so-called twig-measure, introduced in the context of index expressions ([9]). Let T_1 and T_2 be two parse trees, then their distance is determined as the Jaccard distance between the expressions representing their nodes and edges. Let N_i be the set of expressions representing the nodes of T_i, and E_i the expressions representations of the edges ($i = 1, 2$), then the twig distance between T_1 and T_2 amounts to:

$$t(T_1, T_2) = \frac{|N_1 \cap N_2| + |E_1 \cap E_2|}{|N_1 \cup N_2| + |E_1 \cup E_2|}$$

By defoliation we remove the leaves from a tree. For index expressions, this means that instances are removed from the parse tree. It will be useful to measure the distance between two parse trees by comparing their defoliated versions. This way we base our comparison on the deep structure of the sentences. Let $\delta(T)$ denote the defoliated parse tree, then we introduce:

$$d(T_1, T_2) = t(\delta(T_1), \delta(T_2))$$

2.4 Proving the main theorem

In order to compare the situation after processing the sentence σ, we have to compare the grammars $\mathsf{G}(S)$ and $\mathsf{G}(S + \sigma)$. However, it is very reasonable to base this comparison on the sample sentences provided. Therefore we propose the following definition:

$$\Delta(\mathsf{G}(S), \mathsf{G}(S + \sigma)) = \frac{1}{|S|} \sum_{s \in S} d(P_{\mathsf{G}(S)}(s), P_{\mathsf{G}(S+\sigma)}(s))$$

where $P_G(s)$ is a parse tree of sentence s in terms of grammar G. First we note that if σ is generated by $G(S)$, then $G(S) = G(S+\sigma)$, and thus $\Delta(G(S), G(S+\sigma)) = 0$.

If a domain expert is sufficiently well known with the domain, then it is to be expected that the more this domain expert has revealed about the domain, the less likely it is that a next sentence will be very different from the information provided earlier. So we may assume:

$$P(\Delta(\sigma, S) < \epsilon) \rightarrow 1 \text{ if } |S| \rightarrow \infty$$

This rule is referred to as the *weak law of elicitation*. This law can be seen as the cognitive requirement a domain expert is supposed to satisfy. The rule states that a domain expert will eventually reveal any relevant aspect of the universe of discourse. This corresponds with a quantified version of the cognitive requirements D1 and D2 that are assumed from a domain expert ([4]).

The expressiveness requirement requires the grammar $\mathsf{G}(S)$, derived from sample sentences S, to be sufficiently expressive. This means that the grammar should be sufficiently expressive to handle simple extensions of S. The more sample sentences have been provided, the less likely it will therefore be that an new sample sentence will lead to a major revision of the derived grammar $\mathsf{G}(S)$. The expressiveness requirement for a modeling method now may be formulated as:

$$\forall_{\epsilon_1} \exists_{\epsilon_2} \left[\Delta(\sigma, S) < \epsilon_2 \Rightarrow \Delta(G(S), G(S+\sigma)) < \epsilon_1\right] \text{ if } |S| \rightarrow \infty$$

This is referred to as the *weak law of modeling*, and is a requirement the system analyst is supposed to satisfy. From the weak law of elicitation and the weak law of modeling, we can easily prove the *strong law of information modeling.*

$$P(\Delta(G(S), G(S+\sigma)) < \epsilon) \rightarrow 1 \text{ if } |S| \rightarrow \infty$$

So the resulting grammar can be made as stable as required by a sufficiently long dialog. The resulting grammar is the grammar that satisfies Theorem 1.

In the remainder of this paper, we argue that the ORM style of modeling satisfies this condition.

3 An example: base fact-oriented

We assume that sentences can be entered in a format that can be converted into index expressions. Basically, we make a distinction between two variants of the controlled language, which we refer to as the internal and the external variant. The external format is a close as possible to natural language, with the restriction that it can be converted by pure elementary linguistic techniques into the associated internal language. The reason for this choice is that during this structuring process of the input sentences, no modeling decisions are to be taken.

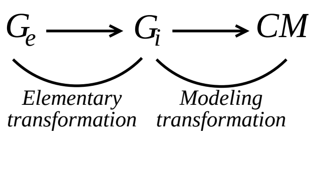

Fig. 3. Language levels

As an example, we use index expressions as underlying internal language. The rationale is that index expressions are well suited to distinguish the underlying predicate and to provide the involved agents and the role in which they are involved. For example, the sentence *person Smith visits country Italy* would be converted into the index expression:

> Parse (*person Smith visits country Italy*) = visit **agens** (person **being** Smith) **patiens** (country **being** Italy)

We will not be concerned with the process of parsing sentences and transforming them into index expressions, but simply assume that sentences are provided in this format. The reverse process is performed by the beautify operator.

> Beautify (visit **agens** (person **being** Smith) **patiens** (country **being** Italy)) = *person Smith visits country Italy*

Furthermore, we also will not focus on the process of beautifying an index expression into the format of the external controlled language. In this paper, we will rather start from the internal controlled language, and assume that input sentences are provided in parsed format.

3.1 Dialog actions

The intention of the modeling dialog is to produce a high quality domain description that is agreed upon by the participants. In figure **??** we see the interaction displayed in this simple type of dialog. We assume the knowledge transfer between the actors in the modeling dialog is performed using the following *dialog actions*:

Propose(s) The domain expert analyst offers sentence s.
Ask(s) The system analyst offers sentence s for validation.
Accept(s) The domain expert accepts sentence s.
Reject(s) The domain expert accepts sentence s.

The controlled language grammar (see figure 1) describes the format of the sentences. This format is discussed later.

3.2 A sample dialog

In this section we discuss a sample session that starts with:

> s_1 : Propose(person Smith visits country Italy)

The sentence *person Smith visits country Italy* will be converted into the following index expression:

> visit **agens** (person **being** Smith) **patiens** (country **being** Italy)

The header of the index expression represents the predicate of the sentence. The subtrees represent the various agents involved in this predicate. The labels mark their roles. Special roles are **agens** (indicating the agens of the sentence) and **patiens** (indication the object of the sentence). Other objects will have a particle that clarifies their role.

The participant *person* **being** *Smith* is interpreted as an instance of an object type referred to as *person*. The remainder of this expression (in this case **being** *Smith*) is a unique reference to an object of the UoD.

From this we derive a general format of parsed sentences. We will describe this as a format for index expressions as we assume they are produced by the parser. This format is the underlying assumption of the system analyst for the structure of parsed sentences (we use the AGFL format to describe grammar rules, see [10] for more details on AGFL):

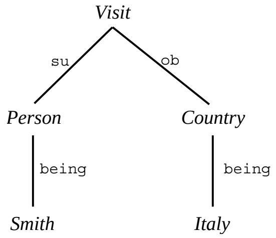

Fig. 4. The index expression

index expression :: predicate, roles.
roles :: role; role, roles.
role :: connector (agent)
agent :: constant agent; compound agent.
constant agent :: string.
compound agent :: type, roles.
predicate :: type.

Note that the index expression itself is seen as an instance of the type that is named by its verb.

We assume that any proposed sentence contains at least one part being variable. Such a part is required to be a *compound agent* if we want to make a clear distinction between concrete and abstract objects. In our example the system analyst thus has the following options:

1. both agens and patiens are parameterizable
2. only agens is parameterizable
3. only patiens is parameterizable

We assume a system analyst to have a maximal generalization attitude. As a consequence, the system analyst will derive the following conceptual rules from this first sentence:

1. there is a sentence type *visit*:
   visit :: visit **agens** person **patiens** country.

   The roles in this sentence type may be addressed by deriving, using simple linguistic techniques, elementary role names to describe the predicate from the point of view of both participants. In this case, the following roles names will be automatically added:
   visiting
   visited by
2. there is an object type *person*:
   person :: string.
   We also record the concrete instance provided, and add the following rule:

person :: ”Smith”.

3. there is an object type *country*:

   country :: string.

   We also record the concrete instance provided, and add the following rule:

   country :: ”Italy”.

After the introduction of these rules, the domain ontology has been extended. The domain language (as described by Lisa-D, [11]) has grown to allow the new sentence formats. This basically is the feedback loop as described in figure 1. For example, the question ***visited by Person Smith*** would lead to the answer

Country Italy

Uncertainty of the analyst In terms of ORM modeling, we still have a problem. According to our structure, the object type *person* is instantiated with the value 'Smith'. The standard way of addressing a person thus is by means of a value. As there is no further explanation of how this value relates to the object, we have to interpret *person* as a so-called label type. The intention of ORM modeling is to make a clear separation between the abstract and concrete objects. Concrete objects can be communicated, and are used to describe abstract objects. For example, the person in our example is uniquely identified by the name *Smith.* It is the assumption of ORM that instances are addressed in sample sentences by their so-called standard name.

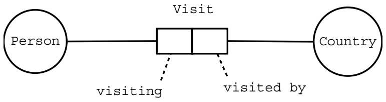

Fig. 5. After processing first sentence

A consequence of this separation between abstract and concrete is that any structural aspects have to be positioned at the abstract level. In this case, our intention clearly is to see *person* as an entity type. So we have to clarify the relation between this entity type and the identifying value. For example, our sentence could be rephrased as follows:

person with surname Smith visits country with name Italy

leading to the following index expression:

visit **agens** (person with (surname **being** Smith)) **patiens** (country with (name **being** Italy))

This expression is graphically displayed in figure 6. However, it is up to the domain expert to reformulate the example sentence in this format. As long as this has not been done, the system analyst may assume such a labeling relation to exist. And as long as this assumption has not been falsified by some sentence from the domain expert, the analyst may keep to this assumption.

However, in the ultimate model there is no place for any assumptions. During the modeling process, the analyst will at some point decide that this assumption has to be resolved. This is performed by the system analyst by some appropriate question, to be answered by the domain expert.

So at any point during the modeling process, the analyst will have a conceptual model in progress, and a set of assumptions that still have to be validated. The analyst will choose the assumption that seems to be the most urgent to be validated by the domain expert. If the domain expert confirms the assumption, then this assumption can be removed. If falsified, then the system analyst will reconsider the modeling decisions taken sofar, and construct an up-to-date conceptual model in progress.

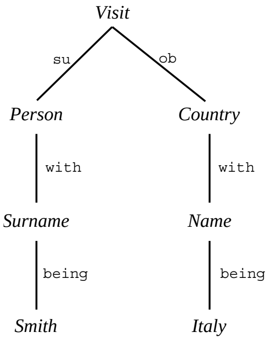

Fig. 6. The modified index expression

Processing the second sentence At this point, we assume the uncertainty of the system analyst of the status of both *person* and *country* has been resolved according to the described format. Typical uncertainties still remaining are wether or not uniqueness and total roles have to be added to the fact type. The analyst could have the strategy of assuming the following constraints as shown in figure 7. Next the domain expert enters the following sentence:

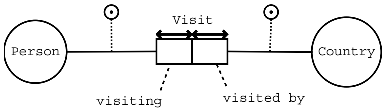

Fig. 7. Constraints assumed

s_2 : Propose(visit **agens** (person with (surname **being** Smith))
patiens (country with (name **being** Greece)))

The system analyst has no difficulties to parse this sentence according to the grammar sofar. However, the analyst will also conclude that an assumption has been falsified by this sentence. This leads to the adapted schema from figure 8.

Generalization and specialization In order to give an idea of the kind of modeling decisions to be taken, we assume the next sentence to be entered is:

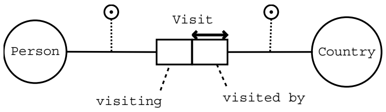

Fig. 8. After dropping a constraint

s_3 : Propose(visit **agens** (student **being** Baker) **patiens** (country with (name **being** Greece)))

Note that this sentence is different from the first two sentences. It is easily verified that:

$$\Delta(s_3, \{s_1, s_2\}) = \frac{2+1}{4+3} = 0.43$$

At this point, the analyst has an example of multiple object types that seem to play the **agens** role. There are two options to handle this situation.

1. both *person* and *student* belong to the same specialization hierarchy.
2. both *person* and *student* belong to the same generalization hierarchy.

The decision depends on the identification of *person* and *student*. If they have the same identification, then they are part of the same subtype hierarchy. In this case probably *person* will be the pater familias, but there is no way for the analyst to derive this from the modeling dialog sofar. On the other hand, if *person* and *student* would have a different identification, then there probably is a generalization hierarchy, and a common generalization for the **agens** role in fact type *Visit*.

The need to clarify the identification status of *student* has a high priority.

4 Conclusion and further research

In this paper we have shown a formal approach to information modeling, and demonstrated our ideas with a very small sample session. Future research will be directed to build larger sample sessions, and to develop tools to analyze such sessions automatically. It might also be interesting to look for mechanisms to compare the merites of different modeling strategies.

References

1. Taylor, R.S.: Question-negotiation and information seeking in libraries. College and Research Libraries (1968) 178–194
2. Bosman, S., Weide, T.v.d.: A case for incorporating vague representations in formal information modeling. In: Conferentie Informatiewetenschap 2003, TU Eindhoven, The Netherlands (2003)
3. Smithson, M.: Ignorance and uncertainty: emerging paradigms. Springer Verlag (1989)
4. Frederiks, P., Weide, T.v.d.: Information modeling: the process and the required competencies of its participants. In Meziane, F., Métais, E., eds.: 9th International Conference on Applications of Natural Language to Information Systems (NLDB 2004). Volume 3136 of Lecture Notes in Computer Science., Springer-Verlag, Berlin, Germany, EU (2004) 123–134 To appear in Data and Knowledge Engineering.
5. Frederiks, P., Weide, T.v.d.: Deriving and paraphrasing information grammars using object-oriented analysis models. Acta Informatica **38** (2002) 437–88
6. Bosman, S., Weide, T.v.d.: Assistance for the domain modeling dialog. Technical report, Radboud University, Nijmegen, The Netherlands (2004)
7. Weide, T.v.d., van Bommel, P.: Measuring the incremental information value of documents. Information Sciences (2004) In Press.
8. Shanzhen Yi, Bo Huang, W.T.C.: Xml application schema matching using similarity measure and relaxation labeling. Information Sciences **169** (2005) 27–46
9. Bruza, P., Weide, T.v.d.: Stratified Hypermedia Structures for Information Disclosure. The Computer Journal **35** (1992) 208–220
10. Koster, C., Oltmans, E.: Proceedings of the first AGFL Workshop. Technical Report CSI-R9604, Computing Science Institute, University of Nijmegen, Nijmegen, The Netherlands (1996)
11. Hofstede, A.t., Proper, H., Weide, T.v.d.: Exploiting Fact Verbalisation in Conceptual Information Modelling. Information Systems **22** (1997) 349–385