Proceedings of the ABIS-94 GI Workshops `Adaptivität und Benutzermodellierung in interaktiven Software-Systemen', GMD, St. Augustin, 77-86, 1994.

Acquiring Primary and Secondary Assumptions in BGP-MS

Wolfgang Pohl and Alfred Kobsa
AG Wissensbasierte Informationssysteme
Universität Konstanz
Postfach 5560-D73, 78464 Konstanz
{pohl,kobsa}@inf-wiss.uni-konstanz.de

1 Introduction

User modeling components should include suitable mechanisms for acquiring user models, in addition to representation and management mechanisms. The methods developed to date can be divided into two groups: Those that extract *primary* assumptions about the user from his/her system input; and those that extract *secondary* (or *derivative*) assumptions from primary and other secondary assumptions. Secondary assumptions may be derived as soon as this is possible (e.g., immediate forward chaining as soon as new primary assumptions have been made), or their derivation may be deferred until they are needed (e.g., backward chaining triggered by a query to the user model).

In this paper we will describe the mechanisms that the user modeling shell system BGP-MS [KP94] offers to support the acquisition of user models. All these mechanisms can be seen as inference methods; some are able to draw very powerful inferences, while others execute more or less simple transformation steps, nevertheless providing useful services to the developer of a user modeling component.

After a short overview of how assumptions about the user are communicated to and represented in BGP-MS, we will describe how input of the application is transformed into primary assumptions about the user. Then the means that BGP-MS provides for interviewing the user and for drawing conclusions from the user's answers will be explained. The acquisition of primary assumptions from observed user actions, as well as stereotype activation which infers a whole set of secondary assumptions at a time, will be discussed. The final section will deal with inferences that are made within the individual user model.

2 Assumptions about the user in BGP-MS

BGP-MS accepts information from the application system regarding the user's beliefs and goals and stores it as *primary information* in its internal representations. Belief and goal descriptions that get communicated to BGP-MS have the form $\mathcal{M}p$, where $\mathcal{M}$ describes a belief/goal nesting and p the content of the belief or goal. The following syntax defines the *belief and goal description language* (BGDL) of BGP-MS. All expressions that can be formulated using this language are permissible at runtime.

```
<bel/goal-desc> ::=  (B S <belief/goal>)
<belief/goal>   ::=  p | (<modality> <agent> <belief/goal>) |
                        (<neg> (<modality> <agent> <belief/goal>))
<modality>      ::=  B | W
<neg>           ::=  NOT
<agent>         ::=  S | U | M
```

'B' and 'W' stand for the modal operators 'believe' and 'want', respectively. The subsequent symbol indicates whether the beliefs or goals are held by the system (S), the user (U), or whether they are mutually shared (M). The content of the belief or goal, *p*, must be expressible by a first-order formula (which can be augmented by the ι operator for referring to specific objects for which a proposition holds true).

Messages of the application that communicate observed beliefs and goals to BGP-MS are labeled `bgp-ms-tell`. An example is:

```
(bgp-ms-tell (B S (B U (all x (-> (inkjet-printer x) (printer x))))))
```
The user believes that inkjet printers are printers.

BGP-MS provides an integrated suite of knowledge representation mechanisms for representing its assumptions about the current user, its domain-specific user modeling knowledge, and optionally its general knowledge about the application domain. The outer representation layer consists of so-called *partitions*, which can be used to separate different types of assumptions, and may be ordered in hierarchies. For example, the "privately" held assumptions of the system about the user's beliefs about the application domain constitute an assumption type that can be associated with a specific partition (named SBUB for System Believes User Believes). Leaf partitions in a hierarchy (with all their directly or indirectly inherited contents) are called *views*. Within partitions, knowledge and assumptions can be represented in a conceptual representation scheme (SB-ONE, [Kob91]) as well as in first-order predicate calculus (FOPC).

This two-level representation permits one to express assumptions about the user's knowledge or goals with respect to the application domain, the system's knowledge about the application domain, and assumptions concerning application-relevant characteristics of user subgroups (the so-called *stereotypes*).

Fig. 1 shows an example of a simple partition hierarchy that contains partitions for several assumption types. SB (which includes, e.g., the domain knowledge of the system) and SBMBUB (mutual assumptions about user beliefs about the application domain) inherit from partition SBMB (mutual beliefs of system and user) since all mutually shared domain knowledge is known by the system and mutually known to be known by the user. Mutually known knowledge of the user, on the other hand, is known by the user, and known by the user to be known by the system to be known by the user, etc. Hence SBUB ("private" assumptions of the system about user beliefs), SBUBSBUB, etc. form subpartitions of SBMBUB. In this example we have only one partition concerning user goals (SBMBUW). DOS-PROGRAMMER and NETWORK-NOVICE are stereotype partitions that may be linked to SBUB at run-time.

3 Inferences based on observed beliefs and goals

The mapping of expressions of the BGDL to entries in the current user model is carried out by the BGP-MS function 'bgp-ms-tell'. The procedure is rather straightforward, as far as the modal part is

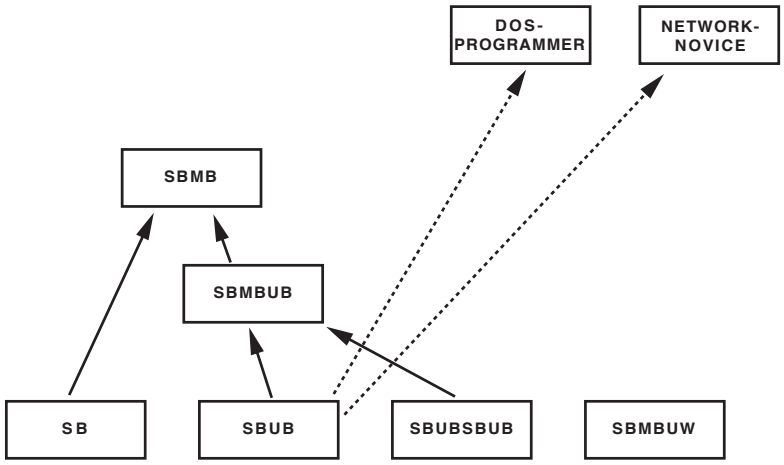

Figure 1: Domain knowledge, individual user model and stereotypes represented in a partition hierarchy

concerned. There exists a one-to-one correspondence between the modal sequence $\mathcal{M}$ and partition names; only the symbols that denote the modality and the agent have to be switched. For instance, when BGP-MS receives a message `(bgp-ms-tell (B S (B M (W U` *p*`))))`, it will enter the first-order expression *p* into the partition SBMBUW.

If the application developer decides to include both SB-ONE and FOPC into the knowledge representation of his user modeling system, the problem arises in which formalism *p* should be expressed. A knowledge categorization component, KN-TRANS [Win94], has therefore been developed which assigns *p* to the appropriate formalism. If SB-ONE is included in the representation and *p* expressible in SB-ONE, *p* will become translated into SB-ONE structures. Otherwise, it will be entered unchanged as a first-order expression. Hence the belief description in the message

```
(bgp-ms-tell (B S (B M (B U
  (all x (-> (inkjet-printer x) (printer x)))))))
```

will become translated into SB-ONE, resulting in a subsumption relation between the concepts 'printer' and 'inkjet-printer' in partition SBMBUB, while

```
(bgp-ms-tell (B S (B M (B U
  (all doc (-> (sent-to-printer doc)
               (exists p (printed-on doc p))))))))
```

results in '$\forall doc$ [sent-to-printer$(doc) \rightarrow \exists p$ printed-on(doc, p)]' being entered in partition SBMBUB.

The algorithm of the knowledge categorization component gives priority to expressing assumptions about the user in SB-ONE, if possible.

4 Inferences from user interviews

An interview consists of question blocks, each of which can consist of one or more questions. The questions of a question block are presented all at once to the user, and the user's answers are sent back to BGP-MS. The user may choose to skip individual questions, question blocks, or even the whole interview.

The definition of a question within a question block includes its name, the question text, the question type, and the conclusions part. The question type specifies the set of possible answers, which must be taken into account when the question is visualized. Possible question types include yes-no-questions, questions with a pre-defined set of answers (out of which one or more can be selected), and questions with a numeric range of answers. Finally, in the conclusions part, the developer can define the assumptions that should be derived from some specific answers. For this purpose, the BGDL must be used.

```
(define-interview-question
  :name network-experience
  :text "How often do you use networked computers?"
  :type (:select-one ("daily" "once per week" "rarely" "never"))
  :conclusions
    (((member answer '("rarely" "never"))
      (B S (network-novice *current-user*)))
     ((equal answer "daily")
      (B S (network-experienced *current-user*)))))
```

Figure 2: Definition of an interview question

Fig. 2 shows the definition of an interview question by which the user's degree of network experience is to be determined. The possible answers form a set containing "daily", "once per week", "rarely", and "never". If the answer is "rarely" or "never", the expression 'network-novice(*current-user*)' will be entered into partiton SB. If the answer is "daily", the expression 'network-experienced(*current-user*)' will be entered into SB. Otherwise, no new entries will be made.

The conclusions about the user are drawn as soon as the answers to a question block are returned to BGP-MS, and become entered into the user model. The interviewing component can take these assumptions into account both when it composes the next question block and when it draws the conclusions from new answers. BGP-MS therefore allows conditional expressions in the definition of the interview sequence and the conclusions parts of questions.

5 Inferences based on the user's actions

For acquiring assumptions about the user, acquisition heuristics have frequently been employed in the user modeling literature. These rules are often domain-specific, but there are also quite a lot of domain-independent heuristics, which normally describe the assumptions that should be made when the user carries out specific actions at the user interface [PKK94]. It is common to many of these heuristics that these assumptions can be regarded as prerequisites to the felicitous execution

of these actions. For example, the correct use of an object presumes that the user knows the object; hence he can be assumed to know the object if he uses it correctly.

This reminds one of the presupposition analysis method that has been applied for supporting the acquisition of a dialog partner model in natural-language dialog systems: A user utterance is analyzed with respect to the speech acts [Sea69] it verbalizes, and the so-called presuppositions are derived from each speech act, which must have been valid for the speaker to perform the acts correctly. This is especially interesting if these derivations can be made without regard to the content of the speech act, i.e. if they are only determined by its type (e.g. a question or an inform act).

In BGP-MS, we generalized the notion of natural-language speech acts to *dialog acts* that may occur in human-computer interaction via any kind of user interface, following other speech-act based approaches in this area [Win88, SS92]. The assumptions about the user that can be drawn from these dialog acts are interpreted as their presuppositions. As with speech acts, *dialog act types* that allow schematical derivations are needed for inferring presuppositions. A dialog act type is normally parametrized and can be associated with a set of *presupposition patterns*, which schematically describe the presuppositions of all instances of the dialog act type. After a set of such types along with their presupposition patterns has been defined, *dialog act analysis* can be performed: the presuppositions of an observed dialog act can be computed by suitably instantiating the presupposition patterns of its type.

We found that there are only few dialog act types that seem to be applicable in all possible interactive computer systems. Therefore BGP-MS offers the application developer not only a library of general dialog acts, but also the possibility to define new dialog acts that may serve as acquisition rules for an application.

As an example we show the definition of the REQUEST-EXPLANATION dialog act type, which should be applicable to all systems that provide explanations on demand.

```
(define-d-act :name REQUEST-EXPLANATION :parameters (topic)
              :presupp ((B S (B M (not (B U topic))))))
```

The REQUEST-EXPLANATION act type has one parameter, which is the topic to be explained by the application. If an instance of it occurs (announced to BGP-MS by a `d-act` message, it will henceforth be assumed that the user does not know this topic.

```
(d-act REQUEST-FOR-EXPLANATION ((:concept kernel)))
```

therefore results in

```
(bgp-ms-tell '(B S (B M (not (B U (:concept kernel)))))).
```

6 Activation and retraction of stereotypes

The stereotype management component of BGP-MS automatically activates the stereotypes that apply to the current user, and retracts active stereotypes that later turn out to be inappropriate. If the application developer decides to take advantage of this component, he has to define the activation and retraction conditions for each stereotype. Both refer to the current user model, but only take the most directly obtained (and hence probably most reliable) assumptions about the user into account.

These include beliefs and goals that were reported by the application, as well as assumptions made on the basis of the user's answers in the interview and the dialog acts he performed. Assumptions acquired by deductive inference processes or through the activation of other stereotypes are not considered.

BGP-MS offers a set of pre-defined condition schemes which the application developer can appropriately instantiate to define the activation and retraction conditions of the stereotypes in his application domain. Frequently used schemes include

- IFKNOWN *list*: is satisfied if the knowledge items contained in *list* are known to the user (i.e., are all contained in partition SBUB).
- IFUNKNOWN *list*: is satisfied if the knowledge items contained in *list* are all unknown to the user.

The application developer may define additional condition schemes in LISP. For defining the activation and retraction conditions of stereotypes, the schemes must be instantiated with appropriate knowledge items of the application domain (which may be any possible assumptions about the user). Instantiated schemes can be logically combined by the connectives AND, OR, and NOT, and also with any LISP code that returns a Boolean value.

```
(or (bgp-ms-ask (B S (network-novice *current-user*)))
    (IFUNKNOWN (:concept LAN) (:concept ethernet)
               (:concept file-server)))
```

Figure 3: Activation condition of the stereotype NETWORK-NOVICE

Fig. 3 shows an example of a stereotype activation condition that is associated with the NETWORK-NOVICE stereotype. It will be satisfied if the fact '(network-novice *current-user*)' is contained in partition SB, or if none of the concepts 'LAN', 'ethernet' and 'file-server' are contained in view SBUB. In this case, the partition SBUB will be connected to the stereotype partition NETWORK-NOVICE, and will automatically inherit its contents.

7 Inferences within the individual user model

The representation formalisms available in BGP-MS offer a variety of inference possibilities, which are carried out by two main inference processes. The representation system for conceptual knowledge, SB-ONE, offers numerous built-in inferences. For inferences in FOPC, the resolution-based theorem prover OTTER [McC94] is employed. Since the expressive power of SB-ONE is considerably weaker than that of FOPC, OTTER was chosen as the central processor for inferences. If desired by the application developer, OTTER may consult SB-ONE with regard to conceptual knowledge, if this seems promising. Normally, OTTER and SB-ONE inferences take place within views, so that partition inheritance is hard-wired into the inference process. But BGP-MS also provides means for performing view-connecting inferences. All general inference mechanisms described in this section can be driven by demand (e.g., queries of the application system) or by data (e.g., new observations about the user).

Inferences within Views Examples for built-in SB-ONE inferences include the automatic computation of the transitive closure of the subsumption relation: if the user knows that a laser printer is a PostScript printer, and that a PostScript printer is a printer, then he also knows that a laser printer is a printer (and possesses all properties of a printer). Another built-in inference process is classification, which positions a concept in the concept hierarchy based on its description by roles and value restrictions.

Inferences that are usually carried out on the basis of FOPC include, e.g., the execution of inference rules which the user presumably employs. Such inference rules normally cannot be represented in SB-ONE since (unlike FOPC) SB-ONE does not allow general implication. These inference rules are ascribed to the user and form part of his/her individual user model. Hence they need not be valid, but may be misconceptions and may be used to simulate the (potentially fallacious) reasoning of the user.

For example, assume that the NETWORK-NOVICE stereotype partition contains the following rule:

$$\forall doc, p \ \ [\text{printed}(doc) \wedge \text{standard-printer}(p) \rightarrow \ \text{printed-on}(doc, p)]$$

Since NETWORK-NOVICE has been activated, this rule is also visible in SBUB. Assume further that the fact 'printed(userdoc)' was entered into partition SBMB, and hence is visible in view SBUB. If SBMB also contains 'standard-printer(lw+)', the above rule can be used to infer that the user believes that 'userdoc' will be printed on printer 'lw+', which is represented by the derived fact 'printed-on(userdoc,lw+)' in view SBUB.

Since the division of view contents into logical and conceptual knowledge is hidden from the application through the automatic categorization of expressions of the belief and goal description language, it must also be guaranteed to the application that the communicated assumptions are all considered by the inference component, no matter whether they are represented in FOPC or SB-ONE. For this purpose, OTTER is currently being enhanced by a component that recognizes situations in its resolution process in which a query to the SB-ONE part of a view may be beneficial [Zim94].

Here is an example for an inference considering both logical and conceptual knowledge. Assume that we have a SB-ONE subsumption relation between PostScript (PS) printers and Apple laser-printers contained in view SB. Additionally, the system knows about the (logical) rule that ASCII documents cannot be printed on PostScript printers:

$$\forall doc, p \ \ [\text{ascii-document}(doc) \wedge \text{PS-printer}(p) \rightarrow \ \neg\text{printable-on}(doc, p)]$$

Moreover, assume that SBMB contains as mutual knowledge 'ascii-document(userdoc)' and 'apple-laser-printer(lw+)'. Since 'apple-laser-printer' is subsumed by 'PS-printer', SBMB also implicitly contains 'PS-printer(lw+)'. Since SB inherits from SBMB, all these facts are also contained in SB. Hence BGP-MS can infer in SB that

$$\neg\text{printable-on}(\text{user-doc}, \text{lw+})$$

View-connecting Inferences Although in many cases the *view-internal* mechanisms described above will meet the inferential needs of a user modeling component, we wanted to enable the application developer to formulate statements expressing relationships between *different* views. Relationships may exist between specific contents of views ("If the user believes A, then he will want B"), and between views in general ("Everything that is believed by the system is believed by

the user"). Here we will describe how both types of *view-connecting* knowledge can be represented and processed in BGP-MS.

The BGDL, which exactly comprises all assumptions about the user that can be represented within BGP-MS, forms a subset of a modal logic with indexed modal operators for beliefs and goals. The complete version of this logic, which includes the standard logical connectives, can be used to express relationships between specific contents of different views. For example,

$$\forall doc, p\ [B_S B_M W_U \text{ printed-on}(doc, p) \rightarrow\ B_S B_M B_U \text{ printable-on}(doc, p)] \tag{1}$$

represents a rule for inferring an assumption about a user belief from an assumption about a user goal: if the user is believed to want a document to be printed on a certain printer, then he is also assumed to believe that this is possible.

General relationships between views may be expressed by modal formula schemes, which (for our purposes) result from replacing FOPC parts in modal expressions by formula variables. For example,

$$(B_S B_M W_U\ \Phi \wedge B_S B_M B_U\ (\Phi \rightarrow \Psi)) \rightarrow B_S B_M W_U\ \Psi \tag{2}$$

with Φ and Ψ being formula variables represents the rule that users want any implication of their immediate goals if they know the implication relation.

Modal logic inferences can be performed using a first-order logic reasoner. To achieve this, a procedure is needed that translates modal expressions into FOPC whilst preserving their original semantics. The so-called *functional translation* [Ohl91] is a method that can be used to process standard modal formulas. In principle, also modal schemes can be integrated into the inference processes of BGP-MS: the algorithm SCAN [GO92] is able to translate modal schemes into first-order expressions representing the semantic behind the schemes. These methods will not be described here; for more details see [Poh93].

However, the use of these techniques for transforming modal expressions and schemes into predicate calculus has a severe consequence: their systematic application renders the partition mechanism superfluous, because everything with a modality must be translated in order to be considered in the inference process. Even expressions within views (which correspond to modal expressions within the scope of BGDL) are affected. From a representational point of view, this translation is possible, since our modal logic is a superset of BGDL expressions. On the other hand, since we would like to keep SB-ONE in BGP-MS as a specialized formalism for object class taxonomies with many "hardwired" inferences, partitions are still necessary, at least for the representation of SB-ONE in different views. The partition mechanism is also advantageous in cases where inferences are constrained to a single view, since then a view-internal OTTER or SB-ONE inference is more efficient.

Currently work is being carried out in the BGP-MS project which aims at reconciling the translation mechanisms to the partition paradigm. Our approach is based on the observation that basically the world paths that are associated with literals refer to partitions. This could be used to link literals of translated view-connecting formulas to literals contained in views, in order to make integrated inference steps possible.

Bidirectional inferences Inference processes within the individual user model can be executed in two different ways:

1. When answering queries of the application, BGP-MS can employ an integrated inference mechanism for backward reasoning, which verifies whether the queried item can be deduced from the current contents of the individual user model, thereby taking both view-internal formulas and SB-ONE structures as well as view-connecting knowledge into account. Inferred implicit assumptions become added to the user model in order to avoid the same reasoning steps in the future. The depth of the backward reasoning process is limited in order to restrain the number of inferred assumptions.

2. In addition, every input into BGP-MS can be examined whether it leads to new implicit assumptions about the user that can be derived in a forward-directed manner. In order to accomplish this, the BGP-MS inference engine can be started in a forward-reasoning mode. The inference depth of this process is also limited in order to avoid inadequate or irrelevant derivations with a high degree of implicitness. The results are likewise added to the current user model to possibly answer future questions of the application faster by avoiding question-driven backward reasoning. This forward-directed pre-computation of inferences is even inexpensive since it can be performed concurrently to the application.

8 Conclusion

In this paper we presented the capabilities of the user modeling shell system BGP-MS with respect to the acquisition of primary and secondary assumptions about the user, which are based on a variety of inference mechanisms. A detailed comparison of BGP-MS with other user modeling shells (namely [Fin89, BT94, Ver94, Kay94, Orw94]) is given in [KP94]. A first application of BGP-MS is KN-AHS [KMN94], a PC-based hypertext system that adapts the hypertext to the user's presumed familiarity with the technical concepts occurring in the text. These assumptions are made by BGP-MS based on the user's interaction with the hypertext, which becomes reported by KN-AHS.

Since user modeling is dynamic per se, a truth maintenance component is currently being developed for BGP-MS in order to support nonmonotonic inferences. BGP-MS does not allow vague assumptions (e.g., does not support probabilistic approaches), which may be regarded as a lack of the system. However, our approach is to provide more user modeling capabilities by providing greater expressive power to an application developer.

References

[BT94] G. Brajnik and C. Tasso. A shell for developing non-monotonic user modeling systems. *Int. J. Human-Computer Studies*, 40:31–62, 1994.

[Fin89] T. W. Finin. GUMS: A general user modeling shell. In A. Kobsa and W. Wahlster, editors, *User Models in Dialog Systems*, pages 411–430. Springer, Berlin, Heidelberg, 1989.

[GO92] D. Gabbay and H. J. Ohlbach. Quantifier elimination in second-order predicate logic. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning: Proc. of the Third International Conference (KR'92)*, pages 425–435. Kaufmann, San Mateo, CA, 1992.

[Kay94] J. Kay. The um toolkit for reusable, long term user models. Submitted to *User Modeling and User-Adapted Interaction*, 1994.

[KMN94] A. Kobsa, D. Müller, and A. Nill. KN-AHS: An adaptive hypertext client of the user modeling system BGP-MS. In *Proc. of the Fourth International Conference on User Modeling*, pages 99–105, Hyannis, MA, 1994.

[Kob91] A. Kobsa. Utilizing knowledge: The components of the SB-ONE knowledge representation workbench. In J. Sowa, editor, *Principles of Semantic Networks: Exploration in the Representation of Knowledge*. Morgan Kaufmann, San Mateo, CA, 1991.

[KP94] A. Kobsa and W. Pohl. The BGP-MS user modeling shell system. To appear in *User Modeling and User-Adapted Interaction*, 1994.

[McC94] W. W. McCune. Otter 3.0 reference manual and guide. Technical Report ANL-94/6, Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, IL, 1994.

[Ohl91] H.J. Ohlbach. Semantics-based translation methods for modal logics. *Journal of Logic and Computation*, 1(5):691–746, 1991.

[Orw94] J. Orwant. Heterogeneous learning in the Doppelgänger user modeling system. Submitted to *User Modeling and User-Adapted Interaction*, 1994.

[PKK94] W. Pohl, A. Kobsa, and O. Kutter. User model acquisition heuristics based on dialogue acts. WIS Report 6, WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, 1994.

[Poh93] W. Pohl. Viewübergreifendes Schließen in BGP-MS. In A. Kobsa and W. Pohl, editors, *Arbeitspapiere des Workshops "Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen"*. WIS Memo 7. WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, 1993.

[Sea69] J. R. Searle. *Speech Acts*. Cambridge University Press, 1969.

[SS92] S. Sitter and A. Stein. Modeling the illocutionary aspects of information-seeking dialogues. *Information Processing & Management*, 28(2):165–180, 1992.

[Ver94] H. Vergara. PROTUM: A prolog based tool for user modeling. WIS Memo 10, WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, 1994.

[Win88] T. Winograd. A language/action perspective on the design of cooperative work. *Human Computer Interaction*, 3(1):3–30, 1988.

[Win94] U. Winkler. Formalismentransformationen in KN-PART. Master's thesis, WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, 1994.

[Zim94] J. Zimmermann. Hybride Wissensrepräsentation in BGP-MS: Integration der Wissensverarbeitung von SB-ONE und OTTER. WIS Memo, WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, 1994.