In: Proc. of the European ACL Conference, Pisa, 1983.

CASE ROLE FILLING AS A SIDE EFFECT OF VISUAL SEARCH

Heinz Marburger
Research Unit for Information Science
and Artificial Intelligence
University of Hamburg
Mittelweg 179
D-2000 Hamburg 13, F.R. Germany

Wolfgang Wahlster
FB10 - Angewandte Mathematik
und Informatik
University of Saarbrücken
Im Stadtwald
D-6600 Saarbrücken 11, F.R. Germany

ABSTRACT

This paper addresses the problem of generating communicatively adequate extended responses in the absence of specific knowledge concerning the intensions of the questioner. We formulate and justify a heuristic for the selection of optional deep case slots not contained in the question as candidates for the additional information contained in an extended response. It is shown that, in a visually present domain of discourse, case role filling for the construction of an extended response can be regarded as a side effect of the visual search necessary to answer a question containing a locomotion verb. The paper describes the various representation constructions used in the German language dialog system HAM-ANS for dealing with the semantics of locomotion verbs and illustrates their use in generating extended responses. In particular, we outline the structure of the geometrical scene description, the representation of events in a logic-oriented semantic representation language, the case-frame lexicon and the representation of the referential semantics based on the Flavor system. The emphasis is on a detailed presentation of the application of object-oriented programming methods for coping with the semantics of locomotion verbs. The process of generating an extended response is illustrated by an extensively annotated trace.

1. INTRODUCTION

Frequently a questioner expects more than a direct, literal response although he must assume that the answerer is not informed about what particular information he is seeking. The questioner imputes to a cooperative dialogue partner the communicative competence to reply to a simple yes-no question like (1) with an extended response (cf. [12]. [11]) like (1a) rather than with a simple Yes.

(1) Are you going to travel this summer?
(1a) Yes, to Sicily.

In the absence of special information about the previous course of the dialog or the intentions of the questioner (the unmarked case) an answer like (1a) seems more appropriate than (1b) or (1c).

(1b) Yes, with an old school friend.
(1c) Yes, by plane.

Research on HAM-ANS is currently being supported by the German Ministry of Research and Technology (BMFT) under contract 08IT15038

Of course, there are numerous dialog situations in which (1b) or (1c) could be generated as a communicatively adequate response on the basis of a particular partner model. But it still must be asked why in dialogs of the type 'information supply' the unmarked response takes the form (1a) and not (1b) or (1c).

In this paper we will present the results of a computational study of this problem for the domain 'locomotion verbs' in dialogs based on a visually present world of discourse. This question is particularly important for the construction of cooperative dialog systems, since, in many applications, no explicit knowledge about the dialog goals of the questioner is available at the outset. If a system is nevertheless expected to 'over-answer', i.e. to volunteer information that has not specifically been requested, it must command a set of heuristic criteria for selecting the additional information that is to be verbalized [11].

It is noteworthy that the three additional points of information in (1a), (1b), (1c) correspond to filled deep case slots of the verb used in the question (GOAL, CO-AGENT and INSTRUMENT, respectively). This suggests that the unfilled optional case slots in the question are candidates for additional information. For a question like (2), in which all the deep case slots of 'break' are filled, only a direct response like (2a) is to be expected as a positive answer, while in (3), where only the obligatory deep case slots are filled, an extended response like (3a) can be expected.

(2) Did you break the window with your slingshotyesterday?
(2a) Yes.
(3) Did you break the window?
(3a) Yes, with my slingshot.

Since not every optional deep case of a given verb unspecified in the question is suitable for an unmarked extended response (e.g. (1a)-(1c)) we may define the problem more precisely by asking which of the deep case slots unspecified in the question are to be chosen as the unmarked values.

For our domain of investigation 'locomotion verbs' let us consider questions (4) and (5), which refer to a visually present world of discourse. In each case perceptual processes are assumed as a prerequisite for the answer.

(4) Which vehicle stopped?
(4a) The bus, on Hartungstreet.
(4b) The bus. because the driver stepped on the brake.
(5) Did the bus turn off?
(5a) Yes, from Hartungstreet onto Schlueterstreet.
(5b) Yes, together with the taxi cab.

The instantiation of the locative slot in answer (4a) and the source and goal slots in (5a) is predictable in contrast to the causative slot in (4b) and the co-agent slot in (5b). As examples (4) and (5) demonstrate, the same optional deep case slot is not always selected as the unmarked option. The choice is dependent upon the verb contained in the question. Moreover, (5a) shows that combinations of deep cases are possible in unmarked extended responses.

In the area under investigation here, the following heuristic can be employed to determine the selection of the deep case slots for an unmarked extended response: Select the deep case slots which contain the concepts necessary for the perceptual verification of the motion described by the verb.

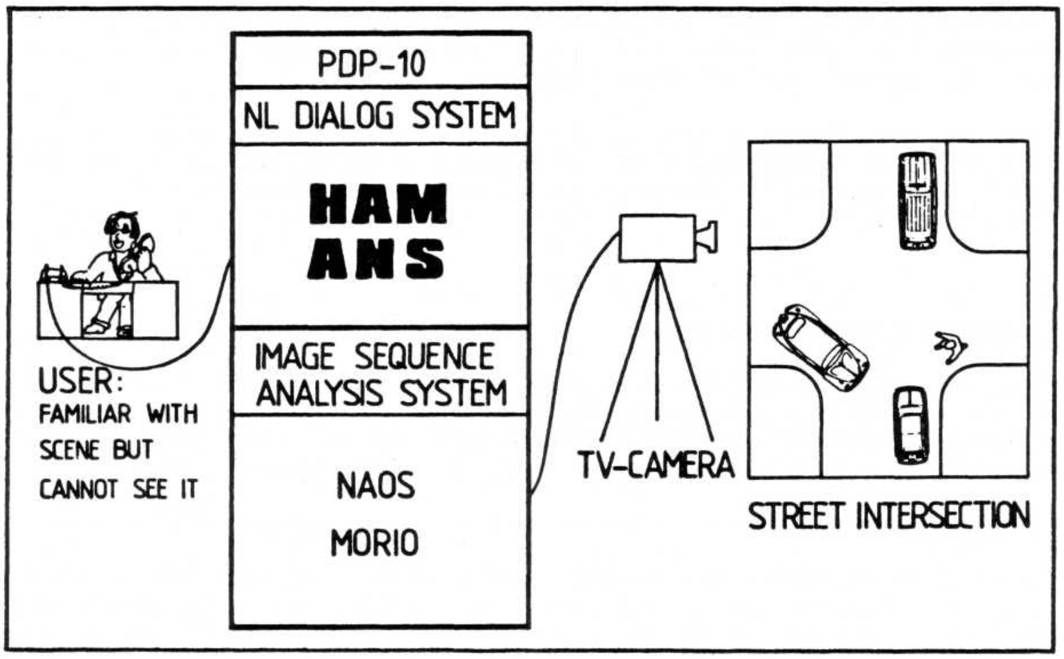


Fig. 1: Situational context of the dialog

In order to verify a stop-event it is necessary to determine the end point of the motion (cf. (4a)) but not the cause (cf. (4b)). For a turn-off event a change of direction between source and goal must be established (cf. (5a)). It is not essential to determine whether other objects make this change of direction at the same time (cf. (5b)).

Hence case role filling for the construction of an extended response can be regarded as a side effect of the visual search necessary to answer the question.

This also appears plausible when seen in the light of the beliefs that the questioner imputes to the answerer. The questioner believes that the answerer will fill in the case slots necessary for answering the question and that it is therefore unnecessary to explicitly mention these in the question. Additionally the questioner believes that the answerer believes that the questioner expects an extended reply and for this reason did not explicitly request the additional information. A cooperative dialog system fulfills this user expectation by applying the heuristic formulated above.

A prerequisite for the application of this heuristic is that the system have knowledge about which deep case slots are relevant for the verification of a movement. This prerequisite is not met by most natural language (NL) systems since they simply represent events in the domain of discourse in fully instantiated form using case frames, e.g. as part of a semantic net or frame hierarchy. In contrast, the German language dialog system HAM-ANS (Hamburg application-oriented natural language system) [6], which we have developed, can apply this heuristic because in addition to the case frame of each verb the system includes a representation of the referential semantics of predications associated with that verb which makes it possible to evaluate the visual input data for the movement in question.

The goal of this article is to elucidate the representation constructions for case frames and referential semantics of verbs of motion used in HAM-ANS and to illustrate their use in generating unmarked extended responses.

2. A SHORT OVERVIEW OF HAM-ANS

HAM-ANS is a large German natural language dialog system of both considerable depth and breadth which presently provides access to three different application classes, namely an expert system (hotel reservation situation), a database system (fishery data) and a scene analysis system (traffic scene).

The communicative situations the system handles are characterized as follows:

In the hotel reservation situation the system takes the role of a hotel manager, who tries to persuade the user to book a room. The caller is assumed to have the overall goal of determining whether the room offered meets his requirements. The system must attempt to recognize the user's specific desires concerning the room as they are revealed - usually indirectly - in his utterances and to make use of the various devices available in natural language that permit the room in question to be presented in a particularly favorable light (e.g. the generation of tendentious descriptions using hedged relative adjectives).

In the database application [8] HAM-ANS provides marine scientists with NL access to a fishery database, which contains data collected during international expeditions in the South Atlantic and the Indian Ocean. The complex structured data sets, which consist of cruise data concerning the vessels' voyages , krill biology data, relevant net hauls and oceanographic data, are stored in a relational database (76 relations, 11 MByte) with a PASCAL/R data base management system.

In the traffic domain we presuppose the following situational context: The system has observed a street intersection and supplies on the phone the user, who is familiar with the scene but cannot see it from his remote location, with information about the traffic at that intersection (cf. fig. 1). A geometrical description of the observed scene is provided by the image sequence analysis system NAOS/MORIO [9]. In this application our research efforts are concerned with the handling of time (tense and temporal adverbs) and the semantics of locomotion verbs (cf. fig. 2).

? How many trucks have you seen?
TWO.
? Which trucks have gone by?
A LIGHT-COLORED ONE ON HARTUNGSTREET.
? Did a yellow car and a green station wagon turn off earlier?
NO, A YELLOW CAR TURNED OFF HARTUNGSTREET ONTO SCHLUETERSTREET,
BUT THE GREEN STATION WAGON DID NOT TURN OFF FIRST.
? Were a young man and an old woman walking down Schlueterstreet
during this time?
YES, AN OLD WOMAN AND A YOUNG MAN.
? Did the green station wagon stop on Schlueterstreet?
YES.
? How many station wagons altogether have stopped?
ONE OF THE THREE ON SCHLUETERSTREET.
?What did I just ask you?
YOU ASKED: HOW MANY STATION WAGONS STOPPED.
?Did a station wagon drive off?
YES, ONE ON SCHLUETERSTREET.
? By the way, the light-colored truck: Did it drive down Schlueterstreet too?
YES.
? How many cars are there in all?
SIXTEEN.
? Is one of these cars blue?
YES. TWO OF THEM ARE KIND OF BLUE.
?Did a blue car drive down both streets?
YES, ONE BLUE CAR.
? So long!

Fig. 2: A transcript of an actual dialog (translated from German)

3. REPRESENTATION FORMALISMS FOR THE SEMANTICS OF LOCOMOTION VERBS

3.1. THE GEOMETRICAL SCENE DESCRIPTION

A basic requirement for answering questions about movements that have occurred in real sequences of scenes is an adequate representation of these sequences. Not only the shape, the centers of gravity, color, etc. of objects must be represented, but also the trajectories of moving objects.

This geometrical scene description consists of a combination of automatically generated outputs of the scene analysis processes (insofar as this is presently possible) and a number of manual augmentations.

The length in time of the scene under consideration is ca. 14 sec, which corresponds to ca. 360 single TV images. From these 360 images 72 snapshots are coded in a relational formalism, denoting which objects were observed, the shape of these objects, their current center of gravity and some other properties (e.g. color). The representation of the first snapshot contains information about all objects that are visible at that time. For the successive snapshots only changes with respect to the predecessors are recorded, i.e. objects and their descriptions are only entered if they have changed location or appeared in the scene. A trajectory of an object is determined by its different centers of gravity relative to an underlying coordinate system. In contrast to the real TV image sequence this representation is only 2 dimensional and thus provides a bird's-eye view of the scene.

3.2. THE REPRESENTATION LANGUAGES SURF AND DEEP

The logic-oriented semantic representation languages SURF and DEEP are the central representation formalisms used in HAM-ANS. These languages are designed to be declarative and easily extendable. SURF is the target language of the analysis components and source language for the generation components and thus as close as possible to NL utterances, whereas DEEP is better suited for the evaluation of utterances on the basis of the system's domain-specific knowledge sources.

Originally SURF and DEEP were designed to represent term and predicate structures which serve as a representation formalism for state descriptions occurring typically in the hotel reservation situation. For an adequate representation of the semantics of questions containing verbs, the definition of SURF and DEEP was augmented by meta-predicates for marking deep cases, tense and voice adapted from Fillmore's deep case theory [3]. Since events can be existentially quantified as in (6) or explicitly quantified as in (7)

(1) Did John fly to Hamburg?
(2) Did John fly to Hamburg three times last week?

SURF and DEEP provide a means of representing quantification of events. A special quantifier E-ACT denotes an existential quantification of events. Other quantifiers like those in (7) are currently not available but can easily be included. Examples of SURF and DEEP expressions are shown in the annotated example (cf. fig. 8) .

In this paper only some of the features of SURF and DEEP are discussed, see [6] for a more detailed description.

3.3. THE CASE-FRAME LEXICON

The case frames for verbs used in the system are stored in the case-frame lexicon [5]. Each entry in the word lexicon for a verb contains a pointer to its applicable case frame which describes the semantics of that verb in terms of case relations.

A case frame is represented as a combination of deep case descriptions specifying for each deep case its name, a marker, whether the deep case is obligatory (0) or optional (F), and the semantic restrictions which are required from a syntactic substructure to fill the deep case (cf. fig. 3).

This pointer technique permits the use of a specific case frame for several verbs during the analysis phase without predetermining a single process for these verbs during the evaluation of whole utterances. For verbs with different referential semantics, e.g. 'to accelerate' and 'to stop', a single case frame, namely that specifying an obligatory AGENT of type 'vehicle' and a optional LOCATIVE of type 'thoroughfare', is applied during the analysis phase.

Case frames are formulated in SURF so that the checking of the semantic restrictions can be accomplished by the inference rules usually applied during the evaluation of a complete utterance. The selectional restriction that, e.g., the NP 'a car' describe an object of the class of vehicles, and therefore be a possible candidate to fill the agent role of the verb 'to stop', can be verified because of the transitivity of the superset relation in the conceptual semantic net.

```
[rl-s: agent:
  [d-1: role-marker:     0
    restrictions:
    [lambda: x1 [af-a: ISA xl VEHICLE]]]
  objective:
  source:
  locative:
  [d-1: role-marker:     F
   restrictions:
   [lambda: x1 [af-a: ISA x1 THOROUGHFARE]]]
goal:
time:
path:
instrument :]
```

Fig. 3: Case frames for verbs of type 'to stop'

In the case-frame lexicon the case frames are not recorded in the form shown in fig. 3, but rather are represented as constructor calls for building a case frame according to the actual syntax definition of SURF. This guarantees that all possible modifications of SURF are immediately present in the case frames.

3.4. OBJECT-ORIENTED REPRESENTATION OF MOTION CONCEPTS

In object-oriented programming languages programming is more or less the activity of creating a world of entities called objects and of specifying a set of generic operations that can be performed on them. Objects can communicate with each other by sending and receiving messages. Essentially, running a program means that the object sends a message to an object (possibly to itself) which in turn sends a message etc., until the required task is fulfilled. An important benefit of the object-oriented style is that it lends itself to a particularly simple and lucid kind of modularity.

3.4.1. THE FLAVOR SYSTEM

The Flavor system [2] [13] is an implementation of the language features that support object-oriented programming. Two kinds of objects exist in a Flavor system, namely one called flavor and the other instance of a flavor. A flavor represents a generic object and an instance an individual realization of a generic object. It is possible to send messages to both kinds of objects. Flavors are organized in a directed graph called the flavor graph. There is one designated flavor, the vanilla flavor, which corresponds to the thing frame in FRL [10]. Since the heritage of information for each flavor is provided by the flavor graph, it is necessary to specify for each newly defined flavor its location in the graph by naming its direct predecessors (its superflavors). The information contained in a flavor is a combination of all the information inherited from its superflavors and the added information given by its own definition. The added information can also override, augment or modify the inherited information.

This is one dimension of the information contained in a flavor: owned or inherited. Another is the declarative/procedural distinction. The declarative knowledge of a flavor is stored in variables of different kinds whereas procedural knowledge is encoded in so-called methods.

One kind of variable - the instance variable - is used to give instances of the same generic object their individual information. The other kind - the class variable - is owned by a flavor, can be 'bequeathed' to other flavors, and accessed by any object in the flavor system. However, a flavor is only allowed to change a value of a class variable, if it owns this variable.

Methods are function definitions that implement the operations defined for each flavor. The combination of methods from different flavors is called mixing flavors.

In comparison with FRL the Flavor system has mainly three distinguishing features:

- The 'A kind of slot in FRL serves both for establishing an inheritance hierarchy and for connecting instances to superclasses, i.e. no clear distinction is made between generic frames and instances. On the other hand the flavor graph is built by specifying the superflavors for each flavor, instances are created by the make-instance-method.

- Because the distinction between generic frames and instances is not made in FRL there is also no distinction between instance variables and class variables. In the Flavor system the semantics of variables is more clearly defined in that instance variables can only be modified in instances and class variables can only be modified in flavors.

- Frames in FRL are passive data structures, whereas flavors can be (re-)activated, created and modified; they are autonomous; they are declarative and procedural at the same time and hence are entities which are better suited for as formalisms for representing common knowledge (cf. [2]).

Although the flavor system is a tool for the development of large software systems and not a knowledge representation language, it includes the basic concepts for the rapid design of specific knowledge representation formalisms. In contrast to a full-fledged knowledge representation language this approach requires some additional programming in the beginning, but it avoids any permanent overhead for features which are superfluous for the task at hand.

3.4.2. THE NOTION CONCEPT HIERARCHY

The Flavor system is used in HAM-ANS for representing a specialization hierarchy of motion concepts (cf. fig. 4). The root flavor of this hierarchy is the motion concept MOVE. Descendants in the tree, e.g. GOJY, TURN inherit the declarative and procedural information contained in their parents. Instance variables comprise information about the deep cases associated with the motion concept as well as information needed and extracted by methods. The methods are responsible for checking the referential semantics of the motion concepts. Instances of a flavor denote specific events in the domain of discourse that could be verified by the application of the methods.

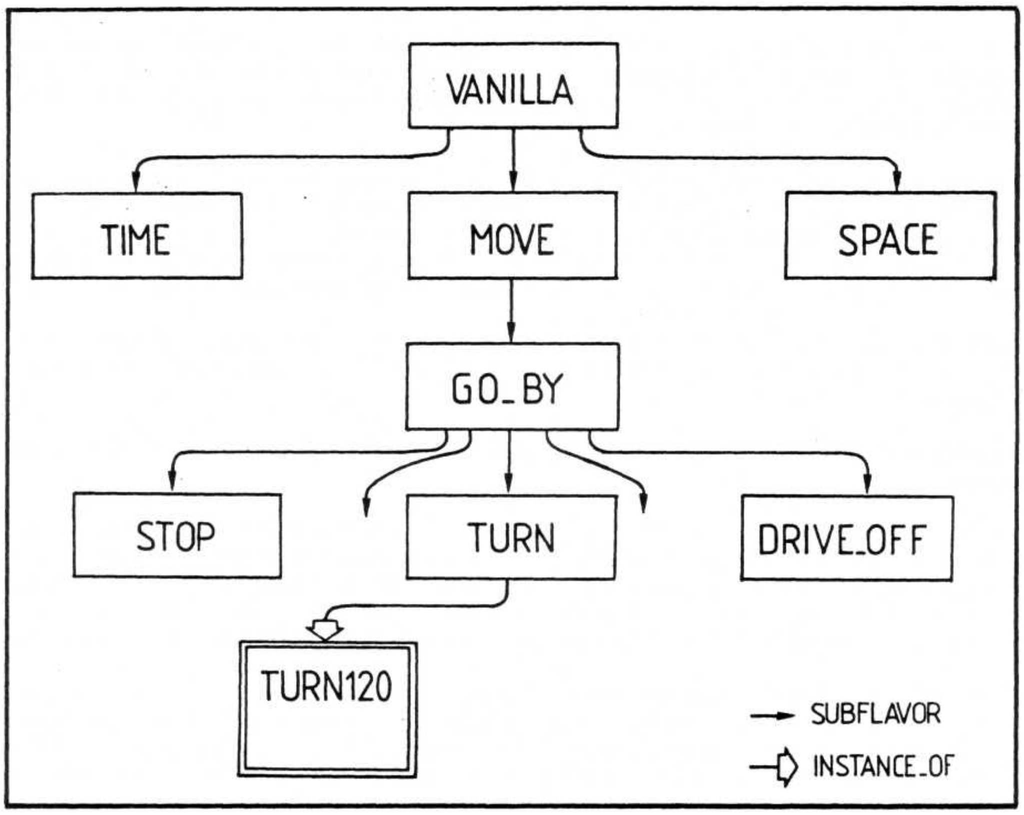


Fig. 4: The motion concept hierarchy

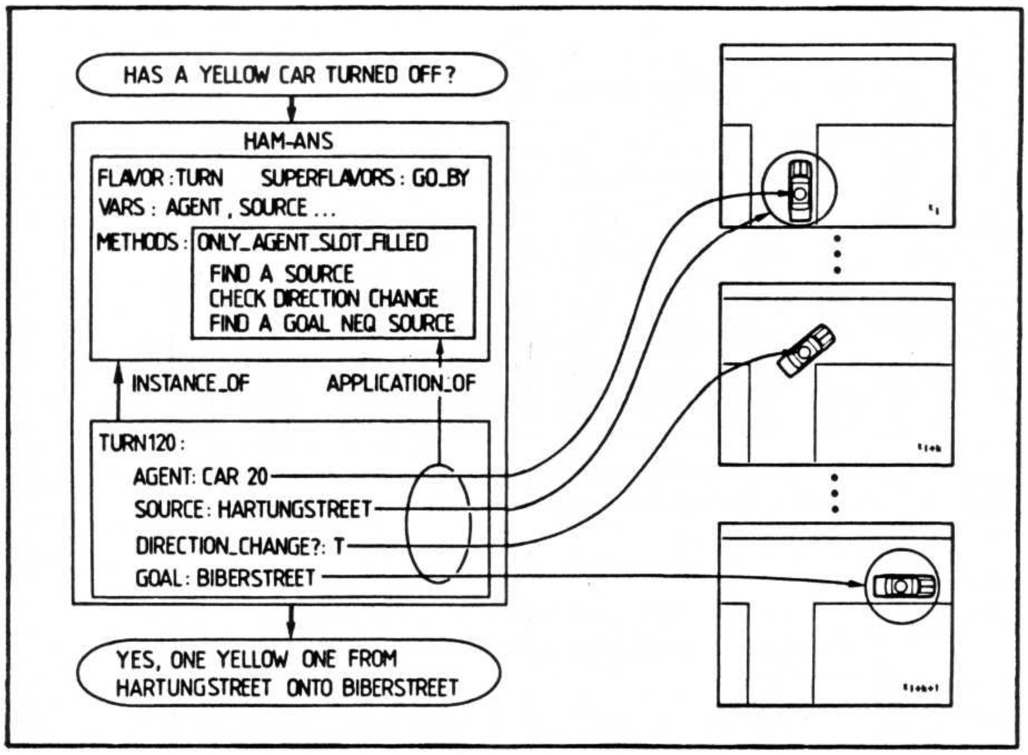


Fig. 5: Case slot filling as side effect of visual search

The methods of the additionally defined flavors TIME and SPACE are responsible for temporal and spatial computations. Instances of these flavors determine the temporal and spatial description of the actual scene: the length of the scene in time, the number of snapshots, the spatial extent, etc.

The task of checking the truth value of the proposition in a user's question is accomplished through message passing. These messages include: creating instances of motion concepts, e.g. TURN120, instantiating deep case slots specified in the question, and activating appropriate methods.

Let's now consider the example given in fig. 5 in more detail. Since only the AGENT was specified in the question, the selected method is ONLY^AGENT^SLOT^FILLED. After determining an interval of consideration this method calls further methods, namely FIND_A_SOURCE, DIRECTION_CHANGE and FIND_A_GOAL_NEQ_SOURCE. DIRECT1ON_CHANGE is a special method of the flavor TURN. The first and last methods are inherited (cf. fig. 6) from flavor GO_BY because they are also needed in that flavor for answering questions like: 'Has the yellow car driven

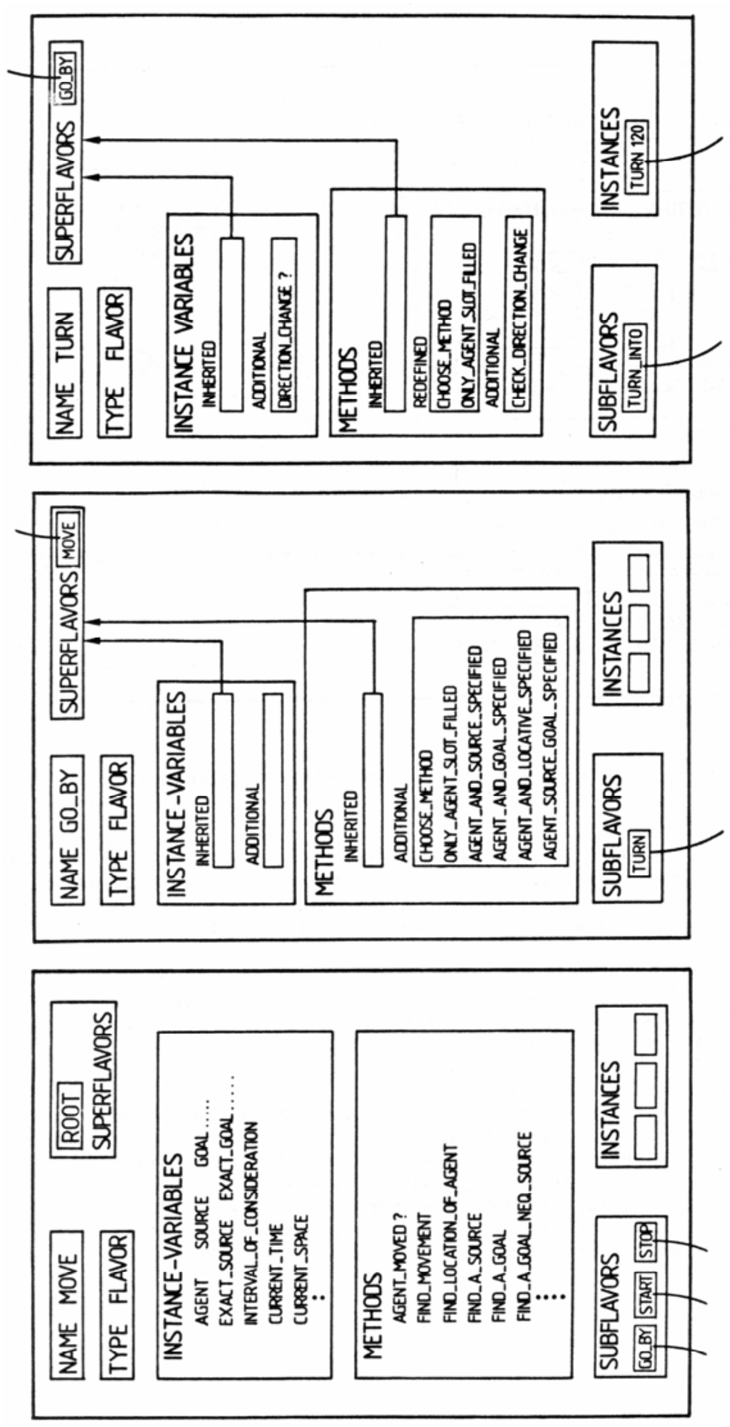
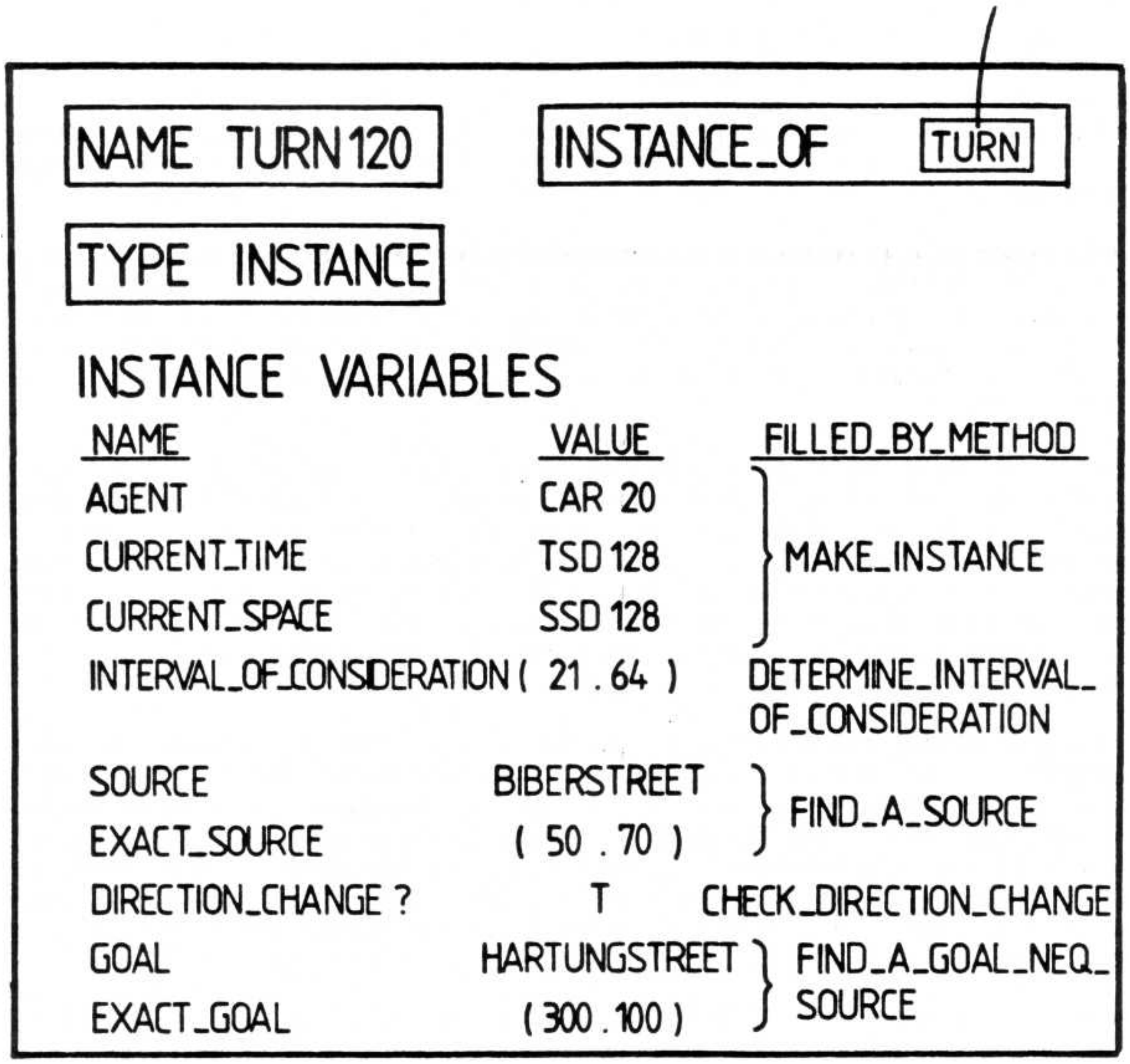


Fig. 6: Instance variables and methods in the motion concept hierarchy



NAME TURN120

INSTANCE_OF TURN

TYPE INSTANCE

INSTANCE VARIABLES

NAME	VALUE	FILLED_BY_METHOD
AGENT	CAR 20	MAKE_INSTANCE
CURRENT_TIME	TSD 128	
CURRENT_SPACE	SSD 128	
INTERVAL_OF_CONSIDERATION	(21 . 64)	DETERMINE_INTERVAL_OF_CONSIDERATION
SOURCE	BIBERSTREET	FIND_A_SOURCE
EXACT_SOURCE	(50 . 70)	
DIRECTION_CHANGE ?	T	CHECK_DIRECTION_CHANGE
GOAL	HARTUNGSTREET	FIND_A_GOAL_NEQ_SOURCE
EXACT_GOAL	(300 . 100)	

Fig.7: An instance of TURN

FIND_A_SOURCE identifies the first entry of the agent's trajectory in the interval of consideration and checks which of the objects of the static background these coordinates belong to. For thistest only those static objects are selected that satisfy the selectional restrictions for the source slot specified in the case-frame lexicon. If the test succeeds for an object, the name of this object is stored in the source slot. DIRECTION_CHANGE now follows the agent's trajectory looking for a significant change of direction. If this test is also positive, FIND_A_GOAL_NEQ_SOURCE is tried. This method searches for a point on the trajectory which is not inside the object identified in the source slot. If there is such a point, the same selectional check as for the source slot is executed for the possible goal object. The successful application of these methods yields a fully instantiated flavor instance, e.g. TURN120 (cf. fig. 7).

4. AN EXAMPLE OF THE PROCESSING OF AN UTTERANCE

The processing of a user's utterance may be illustrated by an example taken from the dialog in fig.2.

USER:	Which trucks have gone by?
HAM-ANS:	A YELLOW ONE ON HARTUNGSTREET.

The following discussion of some of the processing phases can best be understood if continual reference is made to fig. 8, which shows a traced version of the example.

The processing of a user's NL input starts with a rather elaborate lexical and morphological analysis - a process which on the one hand reduces single words to their canonical forms with their morphological and syntactic features (e.g. gender, person, number) and on the other hand recognizes syntagmatic groups of words and discontinuous verb constituents, transforming them according to predefined rules.

The generated structure - the preterminal string (not shown in fig. 8) - forms the input to the parser. The syntactic analysis consists of two different strategies, both of which use the same ATN-definitions of syntactic categories, e.g. for noun phrases and prepositional phrases. One of these strategies - always applied for sentences with copula verbs - uses a surface grammar to cope with word order variations. The other is a case-driven analysis strategy which is used for sentences containing verbs with an associated case frame.

Since in the example the verb 'to go by' has a case frame the second strategy is applied. After an access to the case-frame lexicon the case frame is constructed. This case frame is used to guide the parsing in the following manner: The algorithm first attempts to recognize those syntactic constituents that are possible candidates for a deep case marked obligatory, and then to recognize those constituents that are possible candidates for optional deep cases. When the input is completely consumed and all obligatory deep cases are filled the process ends.

The test for determining if a syntactic constituent is a possible candidate to fill a specific deep case is divided into a syntactic and a semantic check. The syntactic check requires, e.g., that in order to fill the agent role a constituent must contain the attribute 'nominative' (sentence in active voice) and that its number must correspond to that of the verb. The semantic check requires that the noun of the constituent fulfill the semantic restrictions specified for the specific deep case. This is accomplished through the building of a SURF expression for the constituent, the transformation of this expression into a DEEP expression, and the evaluation of the DEEP expression on the basis of the conceptual net.

In our example only the agent case is marked as obligatory and the noun phrase 'which trucks' fulfills both the syntactic and semantic requirements to fill this slot. Since no other syntactic constituents are encountered, the complete SURF representation is constructed.

The structure is normalized into a DEEP structure. One of the main tasks of this process is the determination of the scope of quantifiers. The algorithm used for this purpose is modelled after the one described by Hendrix [4]; it takes into account the relative strength of natural language quantifiers (e.g. 'a', 'both') and question operators (e.g. 'which', 'how many'). The strength is determined by a numeric value, which in some cases is modified by the degree of generality of the noun. E.g. the existential quantifier 'a' is weaker than the more specific quantifier 'both'.

Since, in the example discussed, the question operator 'which' is stronger than the existential quantifier for verbs 'E-ACT', the structure is rearranged.

```
? Which trucks have gone by?
    .
    .
    .
** Syntactic analysis

; ; Case frame

[rl-s: agent:
 [d-1: role-marker: 0
  restrictions:
  [lambda: xl [af-a: ISA xl VEHICLE]]]
 objective:
 source:
 [d-1: role-marker: F
  restrictions:
  [lambda: xl [af-a: ISA xl THOROUGHFARE]]]
 locative:
 [d-1: role-marker: F
  restrictions:
  [lambda: xl [af-a: ISA xl THOROUGHFARE]]]
 goal:
 [d-1: role-marker: F
  restrictions:
  [lambda: xl [af-a: ISA xl THOROUGHFARE]]]
 time:
 path:
 instrument:]

;; NPs:  [S-AGENT]

;; PPs:  [S-SOURCE S-LOCATIVE S-GOAL]

;; AGENT parsed

[lambda: x9
 [af-a: AGENT
  x9
  [t-s: [q-w: WHICH] [lambda: x8 [af-a: ISA x8 TRUCK]]]]]

;; SURF representation of input sentence

[af-d: EVENT
 [t-s: [q-qt: E-ACT] [lambda: xl0 [af-a: ACT xl0 GO_BY]]]
 [d-e: role-list:
  [rl-s: agent:
   [lambda: x9
    [af-a: AGENT
     x9
     [t-s: [q-w: WHICH] [lambda: x8 [af-a: ISA x8 TRUCK]]]]]
   objective:
   source:
   locative:
   goal:
   time:
   path:
   instrument:]
  mod:
  [d-m: tense:
```

Fig. 8: Annotated example interaction

```
   [lambda: xll [af-a: TENSE xll PERFJ]
   voice:
   [lambda: xl2 [af-a: VOICE xl2 ACTIVE]]]]]

** Normalisation: Transforming into DEEP representation

;; Extraction of a quantifier from an ATOMIC FORMULA

        [af-a: AGENT
         x9
         [t-q: [for: [q-w: WHICH] xl4] [af-a: ISA xl4 TRUCK]]]

                          |
                          |
                          V

        [f-d: [t-q: [for: [q-w: WHICH] xl4]
                    [af-a: ISA xl4 TRUCK]]
         [af-a: AGENT x9 xl4]]

;; DEEP structure

[f-d: [t-q: [for: [q-w: WHICH] xl4] [af-a: ISA xl4 TRUCK]]
 [f-d: [t-q: [for: [q-qt: E-ACT] xl3] [af-a: ACT xl3 GO_BY]]
  [f-e: role-list:
   [rl-d: agent:
    [af-a: AGENT x13 x14]
    objective:
    source:
    locative:
    goal:
    time:
    path:
    instrument:]
   mod:
   [f-m: tense: [af-a: TENSE x13 PERF] voice: [af-a: VOICE xl3 ACTIVE]]]
  ]]

** Evaluation

;; Evaluation of a formula with the quantifier

        [q-w: WHICH]

;; Evaluation of a formula with the quantifier

        [q-qt: E-ACT]

;; Object TRUCK1 has not moved during the entire scene

;; Evaluation of a formula with the quantifier

        [q-qt: E-ACT]

;; Testing of a partially instantiated case frame

[f-e: role-list:
 [rl-d: agent:
  [af-a: AGENT GO_BY TRUCK2]
  objective:
```

Fig. 8 [cont.]: Annotated example interaction

```
 source:
 locative:
 goal:
 time:
 path:
 instrument:]
mod:
[f-m: tense: [af-a: TENSE GO_BY PERF] voice: [af-a: VOICE GO_BY ACTIVE]]]

;; interval of consideration determined from tense [and adverb]

        [l . 64]

;; The object becomes visible between time points 56 and 65

;; The interval of consideration modified in accordance with object time is:

        [56 64]

;; Change determined between time points 56 and 57

;; Completed case frame

[f-e: role-list:
 [rl-d: agent:
  [af-a: AGENT GO_BY TRUCK2]
  objective:
  source:
  locative:
  [af-a: LOCATIVE GO_BY *0N HARTUNGSTREET]
  goal:
  time:
  path:
  instrument:]
mod:
[f-m. tense: [af-a: TENSE GO_BY PERF] voice: [af-a: VOICE GO_BY ACTIVE]]]

;; Verification of event was possible

;; Result of the Evaluation

[f-d: [t-q: [for: [q-s: [TRUCK2]] xl4] T]
 [f-d: [t-q: [for: [q-qt: E-ACT] xl3] [af-a: ACT xl3 GO_BY]]
  [f-e: role-list:
   [rl-d: agent:
    [af-a: AGENT xl3 x14]
    objective:
    source:
    locative:
    [af-a: LOCATIVE xl3 *ON HARTUNGSTREET]
    goal:
    time:
    path:
    instrument:]
   mod:
   [f-m: tense: [af-a: TENSE xl3 PERF] voice: laf-a: VOICE x13 ACTIVE]]]
  ]]
**  Inverse normalisation: Transforming into SURF representation
```

Fig. 8 [cont.]: Annotated example interaction

```
;; SURF representation of answer

[af-d: EVENT
 [t-s: [q-qt: E-ACT] [lambda: xl3 [af-a: ACT xl3 GO_BY]]]
 [d-e: role-list:
  [rl-s: agent:
   [lambda: xl3 laf-a: AGENT x13 [t-s: lq-s: [TRUCK2]] T]]]
   objective:
   source:
   locative:
   [lambda: xl3 [af-a: LOCATIVE xl3 *ON HARTUNGSTREET]]
   goal:
   time:
   path:
   instrument:]
 mod:
 [d-m: tense:
  [lambda: xl3 [af-a: TENSE xl3 PERFJ]
  voice:
  [lambda: xl3 [af-a: VOICE xl3 ACTIVE}]]]]

** Ellipsis generation

;; Elliptified SURF representation of answer

[rl-s: agent:
 [lambda: xO [af-a: AGENT xO [t-s: [q-s: [TRUCK2]] T]]]
 objective:
 source:
 locative:
 [lambda: xO [af-a: LOCATIVE xO *0N HARTUNGSTREET]]
 goal:
 time:
 path:
 instrument:]

** Verbalisation

** NP-Generation for TRUCK2

;; The generated NP for TRUCK2 is:

        [t-q: [for: [q-qt: A] x15] [f-o: AND laf-a: ISA xl5
        TRUCK] [af-a: REF xl5 LIGHT-COLORED]]]

;; Verbalized structure of answer

[SENTENCE [AGENT [NP [NP [N: SG] A LIGHT-COLORED [ELLIPSIS TRUCK]]]]
          [LOCATIVE [PP *0N [NP [N: SG] HARTUNGSTREET]]]]

** Surface transformations

A LIGHT-COLORED ONE ON HARTUNGSTREET.
```

Fig. 8 (cont.): Annotated example interaction

The task of evaluating a DEEP formula is governed by a generate and test strategy. Generate and test procedures can be viewed as being activated by pattern-directed invocation and differ from each other in that the generate procedures assign internal object identifiers to variables in DEEP formulas, while the test procedures yield two values, the first of which is either a fully instantiated formula equivalent to the input formula or a modified formula, and the second of which indicates the truth value of the input formula in the range [0,1]. In the interpretation phase these two processes interact in such a way that a test attempt activates generate procedures which in turn call test procedures and so on.

A closer look at our example shows that after the first test attempt has discovered a structure containing a variable - in this case the term representing the noun phrase 'which trucks' - a package of generate procedures is activated to produce the set of object identifiers denoting the referential set of objects that are trucks - here TRUCK1 and TRUCK2. The rest of the formula is then recursively sent to a test process with the variable 'x14' replaced by elements of the reference set for trucks one after; the other.

The next formula to be tested requires the generation of a set of instances of the type GO_BY. Since events are not represented in fully instantiated form but rather must be extracted from the geometrical scene description, a special set of procedures - the methods specified in the verb flavor hierarchy - is activated. (See section 3.4.2 for how this process functions.)

A verification of an event G0_BY is possible only for TRUCK2. The additional information extracted during the process of visual search - the specific location of the event - is recorded in the locative slot.

During the formation of the result of the evaluation, the system, guided by general heuristics, decides whether the additional detail will cause too great a complexity in the answer or not [11]. In this case the complexity is suitable and the location will be mentioned in the answer.

The word 'which' is defined as quantifier that causes a description of a set of objects to be returned (instead of a truth value). Thus the set of reference objects for which the proposition in question could be verified, i.e. TRUCK2. is substituted for the noun phrase 'which trucks'.

The resulting DEEP expression is transformed by the inverse normalization process into a SURF expression. In order to verbalize extended responses in a manner both informative and concise as possible, the ellipsis generation process elides those parts of the semantic representation of complete responses that are identical to the stored representation of the question [7].

The verbalization component produces a string of canonical words and their grammatical features using translation rules attached to the various categories of SURF expressions. A special subcomponent provides for the generation of noun phrases as descriptions of domain individuals, in our example TRUCK2. In this case the NP-generator decides not to generate a definite description since neither the system nor the user has already referred to TRUCK2 in the previous dialog and the existence of TRUCK2 as a moving object is not implied by the existential assumptions supplied by the a priori user model (cf. [7]). Instead, the indefinite NP 'a light-colored truck' is generated, using the property 'light-colored' as an initial characterization.

Finally the 'surface transformation' component [1] pronominalizes the noun 'truck' and yields a standard word order of the utterance and the correctly inflected forms of the canonical words.

5. CONCLUSION

We have attempted to show that case role filling for the construction of an unmarked extended response can be regarded as a side effect of the visual search necessary to answer questions referring to a visually present domain of discourse. A new method for the representation of the referential semantics associated with locomotion verbs has been presented in the framework of object-oriented programming based on the Flavor system. The approach presented has been useful in extending the communicative capabilities of the dialog system HAM-ANS as an interface to a vision system.

ACKNOWLEDGEMENTS

The authors would like to express their thanks to Henning Bergmann for typesetting, and Russell Block for editorial assistance and some useful comments on the draft version of the text.

REFERENCES

[1] BUSEMANN. S.: Problems involving the automatic generation of utterances in German. Memo ANS-8, Research Unit for Information Science and AI, Univ. of Hamburg, April 1982.

[2] DI PRIMIO, F., CHRISTALLER, T.: A poor man's flavor system. Working paper No. 47, ISSCO, Univ. de Geneve. 1983.

[3] FILLMORE, C. J.: The case for case. In: Bach, E., Harms, R. T. (eds.): Universals in linguistic theory. Holt, Rinehart & Winston, 1968, pp. 1-88.

[4] HENDRIX, G. G.: Semantic aspects of translation. In: Walker, D. E. (ed.): Understanding spoken language. New York, North-Holland, 1978, pp. 193-228.

[5] HOEPPNER, W.: ATN-Steuerung durch Kasusrahmen. In: Wahlster, W. (ed.): GWAI-82. Proc. 6th German Workshop on AI. Berlin: Springer, 1982, pp. 215-226.

[6] HOEPPNER, W., CHRISTALLER, TH., MARBURGER. H., MORIK, K.. NEBEL, B., O'LEARY, M., WAHLSTER, W.: Beyond domain independence: Experience with the development of a German language access system to highly diverse background systems. In: Proc. 8th IJCAI, Karlsruhe 1983, pp. 588-594.

[7] JAMESON, A., WAHLSTER, W.: User modelling in anaphora generation: Ellipsis and definite description. In: Proc. ECAI-82, Orsay 1982, pp. 222-227.

[8] MARBURGER. H., NEBEL. B.: Natuerlichsprachlicher Datenbankzugang mit HAM-ANS: Syntaktische Korrespondenz, natuerlichsprachliche Quantifizierung und semantisches Modell des Diskursbereichs. In: Kupka. I. (ed.): GI-13. Jahrestagung. (To appear)

[9] NEUMANN, B.: Towards natural language description of real- world image sequences. In: Nehmer, J. (ed.): GI-12. Jahrestagung. Berlin: Springer, 1982, pp. 349-358.

[10] ROBERTS. R.B., GOLDSTEIN. I.P.: The FRL manual. AI Memo 409. AI Lab.. MIT. Cambridge, 1977.

[11] WAHLSTER. W.. MARBURGER. H.. JAMESON. A.. BUSEMANN. S.: Over-answering yes-no questions: Extended responses in a NL interface to a vision system. In: Proc. 8th IJCAI. Karlsruhe 1983. pp. 643-646.

[12] WEBBER. B.. JOSHI. A.. MAYS. E., MCKEOWN, K.: Extended natural language database interaction. In: Int. J. Computers and Mathematics. Spring 1983.

[13] WEINREB, D.. MOON. D.: Lisp Machine Manual (4th ed.). MIT. 1981.