Managing information at linguistic interfaces*

Johan Bos and **C.J. Rupp**
Computerlinguistik
Universität des Saarlandes
D-66041 Saarbrücken
{bos,cj}@coli.uni-sb.de

Bianka Buschbeck-Wolf and **Michael Dorna**
Institut für Maschinelle Sprachverarbeitung (IMS)
Universität Stuttgart
D-70174 Stuttgart
{bianka,michl}@ims.uni-stuttgart.de

Abstract

A large spoken dialogue translation system imposes both engineering and linguistic constraints on the way in which linguistic information is communicated between modules. We describe the design and use of interface terms, whose formal, functional and communicative role has been tested in a sequence of integrated systems and which have proven adequate to these constraints.

1 Introduction

This paper describes the development and usage of interface representations for linguistic information within the Verb*mobil* project: a large distributed project for speech-to-speech translation of negotiation dialogues, between English, German and Japanese. We take as our reference point the Verb*mobil* Research Prototype (Bub et al., 1997), but this is only one of a sequence of fully integrated running systems. The functional, formal and communicative role of interface terms within these systems, once instigated, has already been maintained through changes in the overall architecture and constitution of the Verb*mobil* consortium.

There are two aspects to our story:

- the practical and software engineering constraints imposed by the distributed development of a large natural language system.
- the linguistic requirements of the translation task at the heart of the system.

* This work was funded by the German Federal Ministry of Education, Science, Research and Technology (BMBF) in the framework of the Verb*mobil* project under grant 01 IV 701 R4 and 01 IV 701 N3. The responsibility for this article lies with the authors. Thanks to our many Verb*mobil* colleagues who helped to design and develop the results presented here, and also to the anonymous reviewers for giving useful hints to improve this paper.

The prominence of the engineering requirements is further heightened by the fact that we are dealing with a spoken dialogue system which must strive towards real time interactions.

We proceed by describing the requirements of the Verb*mobil* Research Prototype, the actual contents of the Verb*mobil* Interface Term (henceforth VIT), the semantic formalism encoded within VITs and processing aspects. We conclude that VITs fulfill the joint goals of a functional interface term and an adequate linguistic representation within a single data structure.

1.1 Modularity

We are concerned here with the interface representations exchanged between modules that make use of traditional linguistic concepts. Figure 1 shows a simplified form of the Verb*mobil* Research Prototype architecture[1], the modules in the grey shaded area make use of VITs as linguistic interface terms and are loosely termed the "linguistic" modules of the system, in contrast to the other modules shown which are chiefly concerned with the processing of acoustic signals. The linguistic design criteria for VITs derive mainly from the syntactic and semantic analysis module, labelled *SynSem*, the generation and the transfer modules.

One point that should be made at the outset is that these linguistic modules really are modules, rather than, say, organisational constructs in some larger constraints system. In practice, these modules are developed at different sites, may be implemented in different programming languages or use different internal linguistic formalisms, and, indeed, there may be interchangeable modules for the same function. This level of modularity, in itself, provides sufficient motivation for a common interface represen-

[1] In that we have excluded the modules that employ alternative techniques and express no interest in linguistic information.

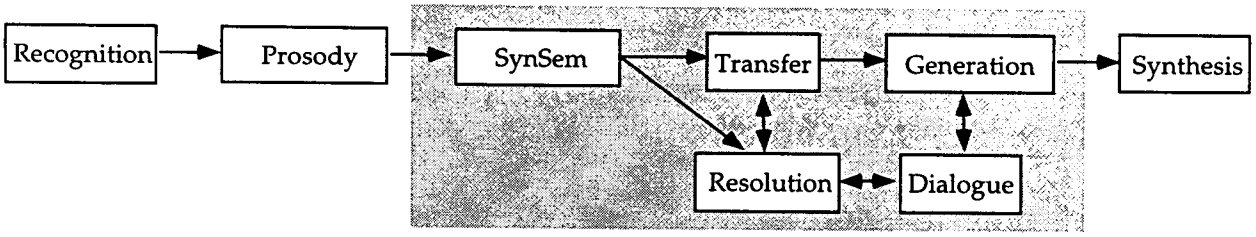

Figure 1: A Simplified diagram of the Verb*mobil* Research Prototype Architecture

tation among the linguistic modules, allowing the definition of a module's functionality in terms of its I/O behaviour, but also providing a theory independent *lingua franca* for discussions which may involve both linguists and computer scientists. The Verb*mobil* community is actually large enough to require such off-line constructs, too.

1.2 Encoding of Linguistic Information

A key question in the design of an interface language is what information must be carried and to what purpose. The primary definition criterion within the linguistic modules has been the translation task. The actual translation operation is performed in the transfer module as a mapping between semantic representations of the source and target languages, see (Dorna and Emele, 1996). However, the information requirements of this process are flexible, since information from various levels of analysis are used in disambiguation within the transfer module, including prosody and dialogue structure.

To a large extent the information requirements divide into two parts:

- the expressive adequacy of the semantic representation;
- representing other types of linguistic information so as to meet the disambiguation requirements with the minimum of redundancy.

The design of the semantic representations encoded within VITs has been guided by an ongoing movement in representational semantic formalisms which takes as a starting point the fact that certain key features of a purely logical semantics are not fully defined in natural language utterances and typically play no part in translation operations. This has been most clearly demonstrated for cases where quantifier scope ambiguities are exactly preserved under translation. The response to these observations is termed *underspecification* and various such underspecified formalisms have been defined. In one sense underspecification is, in itself, a form of information management, in that only the information that is actually present in an utterance is represented, further disambiguation being derived from the context. In the absence of such contextual information further specificity can only be based on guesswork.

While the management of information in the VIT semantics consists of leaving unsaid what cannot be adequately specified, the amount of information and also the type of information in the other partitions of the VIT (see Section 2.1) has been determined by the simple principle of providing information on justified demand. The information provided is also quite varied but the unifying property is that the requirements are defined by the specific needs of transfer, in distinguishing cases that are truly ambiguous under translation or need to be resolved. For example:

(1) Geht das *bei ihnen*?

  a. Is it possible *for you*?

  b. Is it possible *at your place*?

In (1), the German preposition *bei* displays an ambiguity between the experiencer reading (1a) and the spatial interpretation (1b). The resolution of this ambiguity requires in the first instance three pieces of information: the type of the verb predicate, the sort of the internal argument of *bei* and the sort of the subject. This, in turn, requires the resolution of the reference of the anaphor *das*, where morphosyntactic constraints come into play. If the referent has the sort `time` then the experiencer reading (1a) can be selected. This is the more usual result in the Verb*mobil* context. Should the referent

Slot Name	Description
VIT ID	combines a unique tag for the turn segment described by the current VIT and the word lattice path used in its linguistic analysis;
Index	a triple consisting of the entry points for traversing the VIT representation;
Conditions	labelled conditions describing the possibly underspecified semantic content of an utterance;
Constraints	scope and grouping constraints, e.g. used for underspecified quantifier and operator scope representation;
Sorts	sortal specifications for instance variables introduced in labelled conditions;
Discourse	additional semantic and pragmatic information, e.g. discourse roles for individual instance;
Syntax	morpho-syntactic features, e.g. number and gender of individual instances;
Tense and Aspect	morpho-syntactic tense combined with aspect and sentence mood information, e.g. used for computing surface tense;
Prosody	prosodic information such as accenting and sentence mood.

Table 1: A list of VIT slots.

be sortally specified as a situation, further information will be required to determine the dialogue stage, i.e. whether the time of appointment is being negotiated or its place. Only in the latter case is the spatial reading (1b) relevant.

(2) Dann würde das *doch* gehen

a. Then, it WOULD be possible, *after all.*

b. It would be possible, *wouldn't it?*

Consider the discourse particle *doch* in (2) which can be disambiguated with prosodic information. When *doch* is stressed and the utterance has falling intonation, it functions as a pointer to a previous dialogue stage. Something that was impossible before turned out to be feasible at the utterance time. Then, *doch* is translated into *after all* and the auxiliary takes over the accent (2a) [2]. If (2) has a rising intonation and the particle is not stressed, it signals the speaker's expectation of the hearer's approving response. In English, this meaning is conveyed by a question tag (2b). Lieske et al. (1997) provide a more detailed account of the use of prosodic information in Verb*mobil*.

In addition to the information that is explicitly represented in the specified fields of a VIT, including the surface word order that can be inferred from the segment identification, and the resolution of underspecified ambiguities in context, transfer might require further information, such as domain-specific world knowledge, speech act or discourse stage information. This information can be obtained on demand from the resolution component (see Figure 1). This flexible approach to the information required for transfer is termed *cascaded disambiguation* (Buschbeck-Wolf, 1997) and is balanced against the fact that each level of escalation implies a correspondingly greater share of the permissible runtime.

[2]We indicate prosodic accent with SMALL CAPITALS.

2 The Verb*mobil* Interface Term

The VIT encodes various pieces of information produced and used in the linguistic modules. The content of a VIT corresponds to a single segment (or utterance) in a dialog turn. This partitioning of turns enables the linguistic components to work incrementally.

2.1 Multiple Levels of Information

A VIT is a record-like data structure whose fields are filled with semantic, scopal, sortal, morpho-syntactic, prosodic, discourse and other information (see Table 1). These slots can be seen as analysis layers collecting different types of linguistic information that is produced by several modules. The information within and between the layers is linked together using constant symbols, called "labels", "instances" and "holes". These constants could be interpreted as skolemized logical variables which each denote a node in a graph. Besides purely linguistic information, a VIT contains a unique segment identifier that encodes the time span of the analyzed speech input, the analyzed path of the original word lattice, the producer of the VIT, which language is represented, etc. This identifier is used,

for example, to synchronize the processing of analyses from different parsers. For processing aspects of VITs see Section 3.

A concrete example of a VIT is given in Figure 2 in a Prolog notation where the slots are also marked. This example is further discussed in Section 2.2.

2.2 VIT Semantics

The core semantic content of the VIT is contained in the two slots: `Conditions` and `Constraints`. The conditions represent the predicates of the semantic content and the constraints the semantic dependency structure over those predicates. This partitioning between semantic content and semantic structure is modelled on the kind of representational metalanguage employed in UDRS semantics (Reyle, 1993) to express underspecification. The semantic representation is, thus, a metalanguage expression containing metavariables, termed *labels*, that may be assigned to object language constructs. Moreover, such a metalanguage is *minimally recursive* [3], in that recursive structure is expunged from the surface level by the use of metavariables over the recursive constituents of the object language.

In UDRSs quantifier dependencies and other scope information are underspecified because the constraints provide incomplete information about the assignment of object language structures to labels. However, a constraint set may be monotonically extended to provide a complete resolution. VIT semantics follows a similar strategy but somewhat extends the expressivity of the metalanguage.

There are two constructs in the VIT semantic metalanguage which provide for an extension in expressivity relative to UDRSs. These have both been adopted from immediate precursors within the project, such as Bos et al. (1996), and further refined. The first of these is the mechanism of *holes* and *plugging* which originates in the *hole semantics* of Bos (1996). This requires a distinction between two types of metavariable employed: labels and holes. Labels denote the instantiated structures, primarily the individual predicates. Holes, on the other hand, mark the underspecified argument positions of propositional arguments and scope domains. Resolution consists of an assignment of la-

[3] We owe the term *minimal recursion* to Copestake et al. (1995), but the mechanism they describe was already in use in UDRSs.

```
vit(
% Segment ID
    vitID(sid(116,a,ge,2,181,2,ge,y,syntaxger),
% WHG String
          [word(jedes,24,[15]),
           word(treffen,25,[19]),
           word(mit,26,[112]),
           word(ihnen,27,[114]),
           word(hat,28,[11]),
           word(ein,29,[117]),
           word(interessantes,30,[121]),
           word(thema,31,[122])]),
% Index
     index(12,11,i3),
% Conditions
     [decl(12,h23),
      jed(15,i6,18,h7),
      treffen(19,i6),
      mit(112,i6,i13),
      pron(114,i13),
      haben(11,i3),
      arg3(11,i3,i6),
      arg2(11,i3,i16),
      ein(117,i16,120,h19),
      interessant(121,i16),
      thema(122,i16)],
% Constraints
     [in_g(122,120),
      in_g(121,120),
      in_g(114,18),
      in_g(112,18),
      in_g(19,18),
      leq(15,h23),
      leq(11,h7),
      leq(117,h23),
      leq(11,h19)],
% Sorts
     [s_sort(i6,meeting_sit),
      s_sort(i13,human),
      s_sort(i16,info_content)],
% Discourse
     [prontype(i13,he,std),
      dir(112,no)],
% Syntax
     [num(i6,sg),
      pers(i6,3),
      pers(i13,1),
      num(i13,pl),
      num(i16,sg),
      pers(i16,3)],
% Tense and Aspect
     [ta_mood(i3,ind),
      ta_tense(i3,pres),
      ta_perf(i3,nonperf)],
% Prosody
     [pros_accent(15),
      pros_accent(121),
      pros_mood(12,decl)]
   )
```

Figure 2: A Verb*mobil* Interface Term

bels to holes, a *plugging*. In general, the constraint set contains partial constraints on such a plugging, expressed by the "less than or equal" relation (`leq`) but no actual equations. `leq(L,H)` should be interpreted as a disjunction that the label is either subordinate to the hole or equal to it. Alternatively, this relation can be seen as imposing a partial ordering. A valid plugging must observe all such constraints.

The other extension in expressivity was the introduction of additional constraints between labels. These form another type of abstraction away from logical structure, in that conjunctive structure is also represented by constraints. The purpose of this further abstraction is to allow lexical predicates to be linked into additional structures, beyond that required for a logical semantics. For example, focus structure may group predicates that belong to different scope domains. More immediately, prosodic information, expressed at the lexical level, can be linked into the VIT via lexical predicates. The constraints expressing conjunctive structure relate basic predicate labels and *group labels* which correspond to a conjunctive structure in the object language, such as a DRS, intuitively a single box. Grouping constraints take the form `in_g(L,G)`, where the "in group" relation (`in_g`) denotes that the label `L` is a member of the group `G`. The content of a group is, thus, defined by the set of such grouping constraints.

$$g = \bigwedge_{i=1}^{n} l_i \text{ such that } \texttt{in\_g}(l_i, g) \in \texttt{Constraints}$$

These two forms of abstraction from logical structure have a tendency to expand the printed form of a VIT, relative to a recursive term structure, but in one respect this is clearly a more compact representation, since lexical predicates occur only once but may engage in various types of structure.

2.3 An Example

Figure 2 shows the VIT produced by the analysis of the German sentence: *Jedes Treffen mit Ihnen hat ein interessantes Thema* ("Every meeting with you has an interesting topic"). The instances which correspond to object language variables are represented by the sequence {`i1, i2, ...`}, holes by {`h1, h2, ...`} and labels, including group and predicate labels, by {`l1, l2, ...`}. The base label of a predicate appears in its first argument position. The predicates `haben`, `arg2` and `arg3` share the same label because they form the representation of a single predication, in so-called neo-Davidsonian nota-

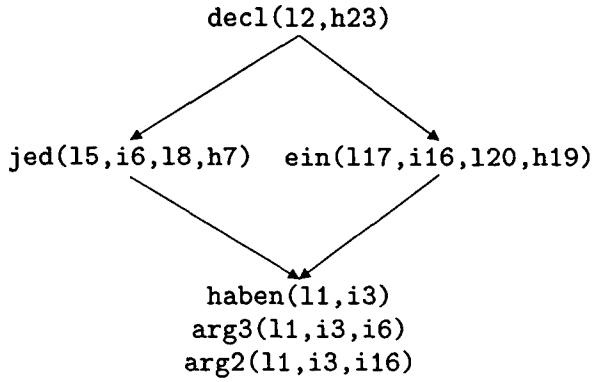

Figure 3: A Graphical Representation of the Scoping Constraints

tion (e.g. (Parsons, 1991)). The two groups `l20` and `l8` form the restrictions of the existential quantifier, `ein`, and the universal, `jed`, respectively. Two of the scoping constraints place the quantifiers' labels below the top hole, the argument of the mood operator (`decl`). The other two link the quantifiers respective scopes to the bottom label, in this case the main verb, but no constraints are imposed on the relative scope of the quantifiers. The whole structure is best viewed as a (partial) subordination hierarchy, as in Figure 3.

A complete resolution would result from an assignment of the labels {`l1, l5, l17`} to the three holes {`h23, h19, h7`}. Taking into account the implicit constraint that any argument to a predicate is automatically subordinate to its label, there are in fact only two possibilities: the pluggings `p1` and `p2` given below,

	h23	h19	h7
p1	l5	l1	l17
p2	l17	l5	l1

corresponding to the two relative scopings of the quantifiers:

$\forall$x(TREFFEN(x)&MIT(x,z)$\rightarrow\exists$y(THEMA(y) &INTERESSANT(y)&HABEN(x,y)))

and

$\exists y(\text{THEMA}(y)\&\text{INTERESSANT}(y)\&$
$\forall x(\text{TREFFEN}(x)\&\text{MIT}(x,z)\rightarrow\text{HABEN}(x,y)))$

A full resolution is, however, rarely necessary. Where transfer requires more specific scoping constraints these can be provided in a pairwise fashion, based on default heuristics from word order or prosodic cues, or at the cost of an additional call to the resolution component.

3 VIT Processing

3.1 Lists as Data Structures

Almost all fields in a VIT record are represented as lists.[4] In general, list elements do not introduce any further recursive embedding, i.e. the elements are like fixed records with fields containing constants. The non-recursive nature of these list elements makes the access and manipulation of information (e.g. addition, deletion, refinement, etc.) very convenient. Moreover, the number of list elements can be kept small by distributing information over different slots according to its linguistic nature (see Section 3.2 below) or the producer of the specific kind of data (see Section 2.1). The kind of structuring adopted and the relative shortness of the lists make for rapid access to and operations on VITs.

It is significant that efficient information access is dependent not only on an appropriate data structure, but also on the representational formalism implemented in the individual data items. This property has been presented as an independent motivation for minimally recursive representations from the Machine Translation point of view (Copestake et al., 1995), and has been most thoroughly explored in the context of the substitution operations required for transfer. We believe we have taken this argument to its logical conclusion, in implementing a non-recursive semantic metalanguage in an appropriate data structure. This, in itself, provides sufficient motivation for opting for such representations rather than, say, feature structures or the recursive QLFs (Alshawi et al., 1991) of CLE (Alshawi, 1992).

3.2 ADT Package

In general, linguistic analysis components are very sensitive to changes in input data caused by modifications of analyses or by increasing coverage. Obviously, there is a need for some kind of robustness at the interface level, especially in large distributed software projects like Verb*mobil* with parallel development of different components. Therefore, components that communicate with each other should abstract over data types used at their interfaces. This is really a further projection of standard software engineering practice into the implementation of linguistic modules.

In this spirit, all access to and manipulation of the information in a VIT is mediated by an abstract data type (ADT) package (Dorna, 1996). The ADT package can be used to build a new VIT, to fill it with information, to copy and delete information within a VIT, to check the contents (see Section 3.3 below), to get specific information, to print a VIT, etc. To give an example of abstraction, there is no need to know where specific information is stored for later lookup. This is done by the ADT package that manages the adding of a piece of information to the appropriate slot. This means that the external treatment of the VIT as an interface term is entirely independent of the internal implementation and data structure within any of the modules and vice versa.[5]

3.3 Consistency Checking

As a side effect of adopting an extensive ADT package we were able to provide a variety of checking and quality control functions. They are especially useful at interfaces between linguistic modules to check format and content errors. At the format checking level language-specific on-line dictionaries are used to ensure compatibility between the components. A content checker is used to test language-independent structural properties, such as missing or wrongly bound variables, missing or inconsistent information, and cyclicity.

As far as we are aware, this is the first time that the results of linguistic components dealing with semantics can be systematically checked at module interfaces. It has been shown that this form of testing is well-suited for error detection in components with rapidly growing linguistic coverage. It is worth noting that the source language lexical coverage in the Verb*mobil* Research Prototype is around 2500 words, rising to 10K at the end of the second phase[6]. Furthermore, the complex information produced by

[4] In typical AI languages, such as Lisp and Prolog, lists are built-in, and they can be ported easily to other programming languages.

[5] The kind of data structure used by the communication architecture (Amtrup and Benra, 1996) is, similarly, transparent to the modules.

[6] In the year 2000.

linguistic components even makes automatic output control necessary.

The same checking can be used to define a quality rating, e.g. for correctness, interpretability, etc. of the content of a VIT. Such results are much better and more productive in improving a system than common, purely quantitative, measures based on failure or success rates.

4 Conclusion

We have described the interface terms used to carry linguistic information in the Verb*mobil* system. The amount and type of information that they carry are adapted to the various constraints that arose during the distributed development of such a large system over a periods of several years. The VIT format has been integrated into several running systems, including the most successful, the Verb*mobil* Research Prototype. Subsequent modifications in the second phase of Verb*mobil* have been possible in a systematic and monotonic fashion. Similarly, the adaption of several new syntactic and semantic analysis modules has not presented any major or unseen problems. Among the tasks currently under development is the use of VIT representations in the generation of dialogue protocols, in the language of each participant.

We feel that VITs have proved themselves adequate to the considerable system requirements through their continued, efffective use. We also find it reassuring that, despite the priority given to the engineering requirement, we have been able to embed within this interface language representations that are, at least, equivalent to the state of the art in Computational Semantics.

References

Hiyan Alshawi, David M. Carter, Björn Gambäck, and Manny Rayner. 1991. Translation by Quasi Logical Form Transfer. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL'91)*, pages 161–168, Berkeley, CA.

Hiyan Alshawi, editor. 1992. *The Core Language Engine*. ACL-MIT Press Series in Natural Languages Processing. MIT Press, Cambridge, MA.

Jan W. Amtrup and Jörg Benra. 1996. Communication in Large distributed AI Systems for Natural Language Processing. In *Proceedings of the 16th International Conference on Computational Linguistics (Coling'96)*, pages 35–40, Copenhagen, Denmark.

J. Bos, B. Gambäck, C. Lieske, Y. Mori, M. Pinkal, and K. Worm. 1996. Compositional Semantics in Verbmobil. In *Proceedings of the 16th International Conference on Computational Linguistics (Coling'96)*, pages 131–136, Copenhagen, Denmark.

Johan Bos. 1996. Predicate Logic Unplugged. In Paul Dekker and Martin Stokhof, editors, *Proceedings of the Tenth Amsterdam Colloquium*, pages 133–143, ILLC/Department of Philosophy, University of Amsterdam.

T. Bub, W. Wahlster, and A. Waibel. 1997. Verb*mobil*: The Combination of Deep and Shallow Processing for Spontaneous Speech Translation. In *Proceedings of the 22nd International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97)*, Munich, Germany, April.

Bianka Buschbeck-Wolf. 1997. Resolution on Demand. Verb*mobil* Report 196, IMS, Universität Stuttgart, Germany.

A. Copestake, D. Flickinger, R. Malouf, S. Riehemann, and I. Sag. 1995. Translation using Minimal Recursion Semantics. In *Proceedings of the 6th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI'95)*, Leuven, Belgium.

Michael Dorna and Martin C. Emele. 1996. Semantic-based Transfer. In *Proceedings of the 16th International Conference on Computational Linguistics (Coling'96)*, Copenhagen, Denmark.

Michael Dorna. 1996. The ADT-Package for the Verb*mobil* Interface Term. Verb*mobil* Report 104, IMS, Universität Stuttgart, Germany.

C. Lieske, J. Bos, B. Gambäck M. Emele, and C.J. Rupp. 1997. Giving prosody a meaning. In *Proceedings of the 5th European Conference on Speech Communication and Technology (EuroSpeech'97)*, Rhodes, Greece, September.

T. Parsons. 1991. *Events in the Semantics of English*. MIT Press, Cambridge, Mass.

Uwe Reyle. 1993. Dealing with Ambiguities by Underspecification: Construction, Representation and Deduction. *Journal of Semantics*, 10(2):123–179.