THE <C,A>,T FRAMEWORK IN EUROTRA : A THEORETICALLY COMMITTED NOTATION FOR MT

D.J. Arnold: University of Essex, Colchester, Essex, CO4 3SQ, UK.
S.Krauwer: University of Utrecht, Trans 14, 3512 JK Utrecht, NL.
M.Rosner: ISSCO, 54, Route des Acacias, 1227 Geneva, Switzerland.
L. des Tombe: University of Utrecht, Trans 14, 3512 JK Utrecht, NL.
G.B. Varile: Commission of the European Communities, L-2920 Luxembourg.

Abstract.

This paper describes a model for MT, developed within the Eurotra MT project, based on the idea of compositional translation, by describing a basic, experimental notation which embodies the idea. The introduction provides background, section 1 introduces the basic ideas and the notation, and section 2 discusses some of the theoretical and practical implications of the model, including some concrete extensions, and some more speculative discussion.

0. Introduction, aims and background.

As Kay (1985) has emphasised, machine translation today is always experimental in nature. We think a number of things follow from this, among them the need for clear and rather strong theoretical principles which can be treated as hypotheses for testing. The idea is, of course, that such testing is revealing irrespective of the confirmation or disconfirmation of the hypotheses. Furthermore, especially where projects of considerable size are concerned, clear and explicitely stated theoretical principles are necessary for the common understanding of the problem.

We assume that it is possible to distinguish a number of different levels of description for MT theories (programmes, systems, etc), where in general, the relation between levels is that lower levels are motivated, or evaluated with respect to higher levels. The aim of this stratification is to introduce a kind of modularity, so that it is possible to preserve stability while responding to changing perceptions of 'the MT problem'. We distinguish the following levels of description:

Mu0: A set of executable programs, and descriptions of a set of languages and the relations between them.
Mu1: A set of substantive theories of representation, and a set of languages in which linguistic descriptions are expressed.
Mu2: The basic theory of translation, general theory of (linguistic) representation & computational apparatus.
Mu3: The basic principles, aims, goals, characteristic assumptions, the 'spirit' of the enterprise.

The purpose of the paper is to discuss the Mu2 level, concentrating on the basic theory or model of translation. The approach will be to describe a family of abstract, special purpose 'MT machines' by describing the syntax and semantics of a very basic notation for MT. We will make some assumptions about representation and user languages for concreteness, but they will be simplified, and unrealistic in the main. Next we examine the model of translation by making explicit some of the theoretical commitments implicit in the notation, discussing some its attractions, and weaknesses, and sketching some possible remedies for the latter. Section 1 will present the notation, with some relevant background, section 2 will discuss the commitment the notation makes.

1. The <C,A>T model of translation.

The ideas described in this section were first presented at the the Colgate Conference on Methodological Issues in MT (Arnold et al, Des Tombe et al, 1985). A key idea in what follows is an interpretation of the idea that translation is a 'compositional' process. To our knowledge, the first application of this idea in MT is in the work of Jan Landsbergen in the Rosetta project (cf Landsbergen 1984).

The fundamental problem of MT is to find a notation, with an associated interpreter, for describing the translation relations between texts in different languages in a 'natural' way. Since it seems impossible to provide such a notation for relating texts to texts directly, the standard response is to decompose the problem 'horizontally' into a sequence of steps, as in :

(1) TL_S --- RL_1 --- RL_2 --- RL_3 ... RL_n --- TL_T

where TL_S and TL_T are source and target text languages, and the RL_i are representation languages (or levels of representation, as they are often called) of some sort. (Notice that in what follows we will systematically use the term 'language' ambiguously for both natural languages and representation languages). Given such a picture one naturally thinks of the languages that are input and output of transfer, but for the purpose of this discussion they could be any pair of representation languages at all. What is crucial to this discussion is the assumption that MT characteristically involves more than one representation language. This point is worth stressing. The following discussion will be couched in terms of a representational theory (Mu1) that involves several specific levels of representation. We believe that this is motivated, but the interest of the general model in no way depends on the existence of these levels.

Given this, one is obviously lead to consider (a) the nature of the representation languages, and (b) the nature of the relations between the representation languages themselves, and between the representation languages and the text languages.

We will not discuss the nature of the representation languages here in any detail (see Arnold et al, January 1985, for detailed discussion), but it seems important that they should be:

(i) Differentiating: This is required if the system is to preserve whatever properties are preserved under 'correct translation'. If two (unambiguous) texts are not translation equivalents i.e. if they differ with respect to these properties, then the representation languages must be rich enough to provide different representations for them.

(ii) Learnable (specific and independently definable): By 'learnable' we mean simply that it must be possible for linguists (who must state the relations between a language and its neighbours, ultimately in the form of an executable description) to be able to understand that relation: for any given text they should be able to determine the appropriate representation, and vice versa: the intuitive semantics of the representation language should be accessible to them. This normally means that the representational theory should be rather specific and constrained. It also means that an independent specification of the language should be available. To take the most problematic case, consider the writer of the generation component in a multilingual MT system, where the only definition of the input representation language is that provided by the transfer components themselves. The task seems clearly impossible.

(iii) Simply relatable: This is the most straightforward, and the most commonly appealed to requirement for the adequacy of representation languages. It must at least be easier to relate the representation languages than it is to relate the text languages. It is this requirement that usually rules out natural languages (e.g. Latin) as representation languages, and motivates representation languages which are highly structured.

We will not be concerned with particular representation languages that have been proposed in Eurotra but, for the purposes of exposition, it is worth a brief description of the kind of representation languages (levels) we will assume in what follows. We will assume there are three such languages (apart from levels such as actual, and normalised text):

(a) A surface constituent/morphological structure level of a rather standard kind, allowing more than one x-bar projection of the major lexical categories (see Arnold et al, January 1985). This level is usually called ECS (Eurotra Constituent Structure).

(b) A level which represents syntactic relations or dependency, and information about syntactic category. This level involves structures where each (non-coordinate) construction contains a primitive item which is the head, or governor ('gov') of the other elements of the construction (this amounts to a stipulation that there can be no more than one projection of lexical categories: X^1 syntax, in effect). This level is usually called ERS (Eurotra Relational Structure).

(c) A level which represents semantic relations/dependency, and thus contains an indication of semantic relations (case or theta roles). Like the syntactic relational level, we will take this to be an X^1 level. It is intended to abstract away from surface syntactic phenomena that are not relevant to translation, and to re-interpret some syntactic characteristics semantically (e.g. replacing tense marking by an indication of time reference). This level is normally called Interface Structure (IS), since it provides an interface between analysis and transfer, and transfer and generation components.

Turning now to the relations between these languages, as regards the text<->representation language relation, there is very little to say in an MT context that does not arise elsewhere in computational linguistics, where parsing and generation are two of the major areas of research. It is the relation between the representation languages that concerns MT specifically. We think the following three conditions are important:

(i) compositionality
(ii) directness (primitiveness, 'one-shot-ness')
(iii) simplicity (statable in a simple way)

Idioms apart, the translation of complex expressions is normally based in some systematic way on the translations of their parts -- normal translation is in some sense 'compositional'. Of course, it is only because of this that the translation relation is productive at all, and if one is looking for principles, it seems reasonable to require something like this of the relation between the representational languages also. The following gives a slightly more precise sense to this idea:

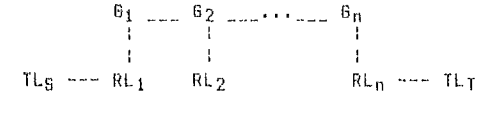
> Translation is compositional when the translation of a complex expression is some (reasonably straightforward) function of the translation of the basic expressions it contains, plus the translation of their mode of combination.

Condition (ii) 'directness' or 'primitiveness' is simply intended to exclude illicit representational levels between those officially sanctioned: what it says is that the RL_i-RL_j relation must be direct, or primitive, and cannot be mediated by other undeclared representational levels. Such a condition helps to maintain clarity (and learnability), and gives content to the other conditions.

A more formal reconstruction of these ideas (along the lines suggested in Montague (1974)) might be as follows.

We begin by defining compositional translation as a relation between 'grammars' (generative devices) specifying languages, rather than languages directly, thus making the RL_i-RL_j relationship parasitic on the relation between the corresponding 'grammars'. Instead of (1), we are thinking of a picture like (2), where a 'vertical' dimension has been added

(2)



Compositional translation of two representation languages RL_i and RL_j is then defined by a pair of relations T and T^{-1} ('translators') between the G_i and G_j (generative devices, grammars) specifying (generating, enumerating) RL_i and RL_j.

We take a generative device G to be a pair, $\langle C,A \rangle$ where C is a finite set of constructors ('rules' defining the class of complex expressions), and A is a finite set of atoms (basic expressions).

We say that translation from L_i to L_j is strictly compositional if there is mapping T from $G_i=\langle C_i,A_i\rangle$ to $G_j=\langle C_j,A_j\rangle$ such that:

(i) T maps A_i into A_j, and
(ii) there is a mapping t from C_i into C_j such that if $u = \langle c : u_1,\ldots,u_n\rangle$, then $T(u) = \langle tc : T(u_1),\ldots,T(u_n)\rangle$

As will become apparent, these definitions imply a very restricive theory of translation, one which is much too strong to be usable. However, before discussing its inadequacy, we will make the ideas involved more concrete by describing a notation for Gs and Ts which is strongly committed to these ideas, and by discussing a very simple example of the use of the notation. The description is rough and not precisely formalised, but should give an idea of the issues involved.

Notation for constructors and atoms:

```
atom ::= (name, feature description)
constructor::= (name,feature description)[argspec*]
argspec::= (name, feature description)
```

(In fact, atoms are simply constructors with arity 0, but we will preserve the intuitive distinction here). The feature theory we assume here is extremely simple: a feature description is a set of attribute-value pairs. The 'name' is just a distinguished feature representing the intuitive linguistic basis of the language being described (thus, it might be a syntactic category, a syntactic relation, or a semantic relation as appropriate). Notice that this name need not be unique. Each constructor has in addition a unique abbreviatory constructor name which is used by the T-rules.

The language L generated by a G is a set of well-formed object (wfo's) such that:

Every atom is a wfo; and if c_n is a constructor of arity n, and each of $u_1, \ldots, u_n$ is a wfo, then $c_n : u_1, \ldots, u_n$ is a wfo.

This very simple syntax for Gs will lead to over-generation -- for example, it will allow np constructors with two arguments as a wfo, even if the first argument is a verb, and the second is a preposition. For this reason we supplement the purely syntactic description with a semantics based on applying constructors to arguments. We will thus normally be concerned only with the subset wfos that are also constructs, in the following sense:

Every atom is a construct; a constructor applied to some arguments yields a construct providing the arguments unify with the appropriate argspecs of the constructor.

Examples (for a constituent structure language):

```
atoms: a_example = (example, (cat=n, num=sing))
       a_this =    (this, (cat=det, num=sing))
constructor:
   Cnp= (np, (num=X, per=3)) [(_, (cat=det, num=X))
                              (_, (cat=n, num=X))]
construct    Cnp : a_this a_example =
   (np, (num=sing, per=3))
                [(this, (cat=det, num=sing)),
                 (example, (cat=n, num=sing))]
```

The syntax and semantics of Ts is roughly as follows. Syntactically, a T-rule is of the form: lhs ==> rhs, where lhs and rhs are atoms or constructors of source and target language Gs respectively. For example, the following might be T-rules relating a level based on syntactic relations with one based on semantic relations for the atoms corresponding to the verbs like and hit. (They are both assumed to assign subject and object relations to their dependents syntactically. They are assumed to assign respectively, experiencer and patient, and agent and patient to their semantic dependents. Since the leading linguistic idea at both levels is relational, and the nature of the relation cannot be determined for constructs in isolation, the name feature of atoms and constructors is 'blank' at these levels).

```
(3)  (_,(word=like, cat=v, frame=subj-obj)) ==>
                (_, (word=like, cat=v, frame=exp-pat))
(4)  (_,(word=hit, cat=v, frame=subj-obj)) ==>
                (_, (word=hit, cat=v, frame=agent-pat))
```

The semantics of this is that all source language atoms which unify with the lhs are translated to all target language atoms which unify with the rhs.

The following might be a constructor to constructor T-rule for the same two levels :

(5) $c_{subj\text{-}obj}$ ==> $c_{exp\text{-}pat}$

meaning that any source language construction built by applying $c_{subj\text{-}obj}$ to some arguments $u_1, \ldots u_n$ is translated by applying $c_{exp\text{-}pat}$ to the translations of $u_1, \ldots, u_n$

This syntax and semantics for T-rules implements the idea of strict compositionality defined above.

This model is elegant, but inadequate, given the way natural languages appear to be. What strict compositionality requires is at least a rather strong homomorphism between the languages related by a T. It is easy to find examples where this looks implausible.

For example, consider the common need to re-order members of a construction in translation; or the need to eliminate 'formal' items which are a part of constructions in one language (one level) but not in another (perhaps re-expressing some information they carry as part of a feature), as in (6); or the kind of simple structural change involved in going from a level which has both S and VP constructions to one which has verb, subject, and object as members of a single construction (7); or the need to re-analyse an item which is part of one construction in one language, as part of another construction in translation (8).

```
(6)  [S for jules to understand it ] ==>
          [S[-finite] jules understand it ]
(7)  [S  jules [VP hit sandy ]] ==>
          [S hit jules sandy ]
(8)  [ rely [PP on sandy ]] ==>
          [ rely-on [NP sandy ]]
```

Of course, one could easily vary assumptions about representations so that these examples disappear, but other examples conflicting with the new assumptions will be just as easy to find. Notice that though,

for simplicity, we have chosen examples that are close to the surface of one language (English), there are many examples of this kind between languages:

(9) [jules zwemt graag] ==>[jules$_i$ likes[e$_i$ to swim]]
'jules swims likingly'
(10) [give[a hand to X]] ==> [(helpen ( )),X]
(11) [$_n$ apple seller] ==> [$_{np}$ vendeur [$_{pp}$ de pommes]]

'Lexical holes' such as English exhibits with respect to Dutch (English has no adverb 'likingly' to translate graag), and idioms such as give a hand will normally give rise to the need for non-strictly compositional translations, for obvious reasons. As a more general example, it is often the case that what is expressed lexically or syntactically in one language is expressed morphologically in another. Thus, modality is often expressed by inflection in Romance languages, and by combinations of separate lexical items in Germanic languages, and correspondences between compounds in Germanic languages, and syntactic constructions in Romance languages (as in (11)) are very common. Treating this kind of thing will certainly lead to non-strictly compositional translations somewhere.

The solution to this problem adopted by Landsbergen (1984) in Rosetta is to 'tune' the Gs to each other, thereby ensuring that they are homomorphic, and that something close to strict compositionality can be preserved. (In fact, Landsbergen requires the translation relation to be symmetric, so the grammars turn out to be isomorphic). This preserves the elegance of the model, but at the expense of the elegance of the linguistic theories and descriptions (the Mu1 and Mu0), which become extremely complex, and potentially unusable. For example, it requires give a hand to and helpen, and graag, and like to to be treated alike. Providing a systematic and general characterisation of a theory of representation which allows this seems highly problematic. What one expects is that the representational theory will become unlearnable in the sense described above. A second objection to this approach is the obvious one that it eliminates the modularity that is potentially available with this model (each G can be thought of as a module, e.g.). This reduces its attractiveness from a developmental point of view, particularly where multi-lingual MT involving large numbers of languages, or wide coverage (hence collaboration of large numbers of individuals) is envisaged.

For these reasons, we have preferred to explore an alternative approach, which involves allowing some relaxations of strict compositionality. The following three relaxations have been proposed :

(i) To allow variables on either side of T rules :

(12) e.g. c27 [1, 2, 3] ==> c38 [2, 3]

with the meaning: translate any expression formed by applying c27 to three arguments by an expression formed by applying c38 to the translation of the second and third arguments. This relaxation allows for re-orderings, deletions, and reduplications by T-rules, and seems an entirely natural extension of strict compositionality.

(ii) To allow functions made up of constructors, atoms, and variables of the appropriate Gs on either side of T-rules, e.g.

(13) e.g. c$_S$ [1, c$_{vp}$ [2, 3]] ==> c$_{subj/obj}$ [2, 1, 3]

Notice that since the output of such a translation rule is still an expression in the target language (i.e. an expression built by applying target G constructors to target constructs), this relaxation still yields 'one shot' translation.

(iii) To allow the choice of the target constructor (function) to be dependent on properties of the arguments involved. For example, one does not want all [v pp] constructions to be treated like rely on in (8), and the exceptional translation behaviour of idioms, and constructions involving lexical holes is clearly dependent on the presence of particular properties within constructions (e.g. the presence of particular lexical items):

(14) e.g. c35 [1, 2/f1=v1, 3] ==> c46 [1, 2, 3/f2=v2]

with the meaning: translate constructions formed by applying c35 to three arguments by constructions formed by applying c46 to their translations, providing the second argument of c35 unifies with a feature description where the attribute f1 has the value v1, and the translation of the third argument unifies with a feature description where the attribute f2 has the value v2.

Though there is no provision for wild notational devices such as path variables, these relaxations greatly increase the power of the notation, to an extent which is problematic, given our methodology. We would naturally like to impose restrictions, so that we can preserve the idea that in compositional translation the translation of a whole is some 'reasonably straightforward' function of the translation of its parts. One possibility is to impose special restrictions (or alternatively restrict some relaxations) to certain translators (e.g. one would like the transfer translators to be as near as possible restricted to some kind of atom-atom translator). More generally, one might require that at most one side of a T-rule be a function (in the sense of relaxation (b)), or to require that context sensitive T-rules may only refer to attributes of particular arguments (e.g. attributes of the heads of constructions, perhaps, or to only allow them to test for the presence of particular lexical items among their arguments). There are interesting methodological and empirical problems involved in trying to find appropriate restrictions, but we will not pursue them here, since (as will appear in the following section), the notation is still restrictive enough for there to be a theoretical commitment which deserves discussion.

2. The theoretical commitments of the model.

The attractiveness of our model as a framework for practical and theoretical MT derives from its modularity and its orderliness in the main. Practically, it ensures that translation proceeds via a series of representations which are described explicitly, and which therefore have to be capable of systematic description, and it ensures that the language generated by applying a sequence of translators is always a subset of a language that has been explicitly described. It thus comes as close as possible to

excluding 'hybrid' representations, and ensuring that representations languages will be 'learnable'. Moreover, the separation of Gs and Ts, and the use of a semantics based on unification provides a high degree of declarativeness, and the homogeneity and uniformity of the model may be of practical benefit. The separation of Gs and Ts also provides a high degree of modularity, so, e.g. different Gs can in principle be developed in parallel, and the effects of modifications may often be localised to one G and the adjacent Ts. This is developmentally attractive.

Methodologically and theoretically the model is attractive in a number of ways. The complexity of T-rules provides a very simple and effective evaluation metric against which to judge competing proposals about representational levels (so it is relatively easy to find arguments why there should or should not be intermediate representations of a certain sort). And it provides a level of abstraction at which linguists and implementers can communicate easily. However, perhaps the most important advantage of the model is that it decomposes the 'problem of MT', and provides a framework for investigating some interesting and apparently manageable sub-problems. Some of these are discussed in a preliminary way here.

This notation, and hence the model that it instantiates, in effect provides a context free grammar notation augmented by a simple feature theory based on unification, and (via the T rules) the capacity for certain transformations. We have no demonstration of the weak genenerative capacity of the notation, but one suspects it is at least as powerful as the notations of LFG (Kaplan & Bresnan 1982), or FUG (Kay 1985). Taking full advantage of relaxations (a)-(c) below may well yield Turing machine capacity. While this makes it likely that the notation provides <u>some</u> treatment of all translationally relevant phenomena, it is still rather restricted as regards descriptive or expressive capacity, and there is no guarantee that the treatment will be 'natural', appropriate, or even practically usable.

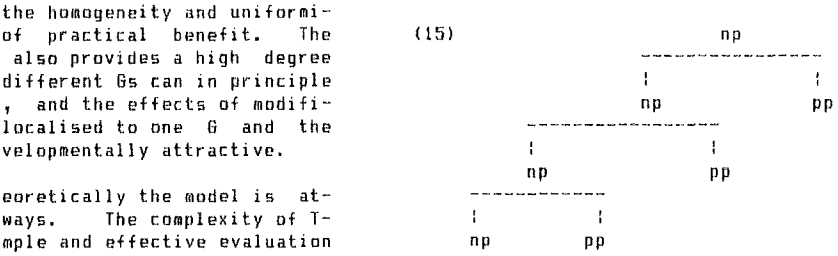
One approach to the issue of usability is the provision of user friendly abbreviations (e.g.), and it is fairly easy to imagine some conventions which would take this basic (Mu2) notation and make it more usable as a programming language for linguists (i.e. a Mu1 'user language').

Some of the major modifications to the model which have been proposed include:

(a) The introduction of special versions of constructor application in place of unification, for example, in the treatment of co-ordinate constructions. The properties of co-ordinate constructions are partly determined by the fact that they inherit the common features of their elements, so the feature description of the construction should be the <u>generalisation</u> (roughly self consistent intersection of the feature descriptions) of the elements, rather than their unification.

(b) The introduction of Kleene star to avoid deeply recursive structures in the treatment of constructions which allow arbitrary numbers of arguments (e.g. most constructions can include an indefinitely large number of PP modifiers). Since the syntax of Gs requires specific reference to the arity of constructors, the obvious way of dealing with this phenomena in the basic notation is to have recursive constructors (e.g. a constructor that combines an np and a pp to form an np), yielding structures such as:

(15)



This treatment is not obviously incorrect, but it is not necessarily the most intuitively satisfactory treatment either, and it can have the undesriable effect of burying the lexical heads of constructions arbitrarily far down inside them.

(c) A closely related point is that the model described is committed to representation languages where members of constructions are strictly ordered ($[c_{32}$: a,b] is a different object from $[c_{32}$: b,a] -- e.g. the latter may fail to unify to give aconstruct, while the former succeeds). This may not always be very natural, especially where relational languages are concerned: since the elements of constructions are distinguished by their roles, they do not also need to be distinguished positionally.

A number of modifications along these lines are being discussed in the project. They are not unproblematic (or even obviously correct), for example, (b) and (c) above both suggest that constructors be treated as operations on sets, rather than lists of arguments. Apart from changing the formal nature of constructors, a problem will arise in going from unordered representation languages to ones which are ordered, motivating an extension to the T-rule notation. Nevertheless, they seem to within a reasonable distance of (and hence compatible with) the essentials of the basic model.

A consequence of the CFG basis of the model is that constructs are always hierarchical objects similar to tree structures (each application of a constructor yields a new level of structure, intuitively). The model is most naturally applied in the description of linguistic phenomena that can be thought of hierarchically, and in translating between languages that capture such phenomena. Thus, it is naturally applied in the description of phrase and relational structures (though cf above), and given the unification based semantics, in dealing with phenomena such as agreement between members of constructions.

Moreover, though the 'naturalness' of the treatment is perhaps more questionable, it provides interesting, and apparently workable accounts of a number of phenomena that are not obviously hierarchical. For example: it is reasonably easy to see how the relaxations of strict compositionality allow a treatment of functional control and unbounded syntactic dependencies (Arnold et al, 1985 sketches a crude, but straightforward treatment exploiting the possibility of having functions composed of target G constructors and variables in T-rules).

One can also envisage a treatment of pronominal reference (which is naturally thought of in terms of co-indexation across trees, rather than in hierarchical terms) within this model along the following lines. Suppose every construct is assigned a unique index, and every non-atomic construct is augmented by two lists (which we will refer to jointly as the a/a lists):

(i) an antecedent list, of the indices of the potential antecedents the construct contains;
(ii) an anaphor list, of the indices of the potentially anaphoric items the construct contains.

We will say that a member of the antecedent list and a member of the anaphor list are 'compatible' providing they do not differ with respect to the relevant inherent linguistic properties (such as number and gender). Every time a constructor is applied to some arguments, the a/a lists of the arguments are inspected:

(i) if two arguments have compatible items on their lists, then an indication that these two items are bound to each other is added to the construct that results;
(ii) the indices of the arguments, and members of the arguments' a/a lists are used to form the relevant a/a lists of the construct.
Apart from testing for inherent properties of antecedents and anaphors, structural conditions can be imposed, e.g. the c-command condition can be imposed by allowing members of antecedent lists to be transfered 'upwards' to only one construct. This seems to provide a basic method for expressing all the antecedent-anaphor relations, in so far as they are grammatically determined, at least.

It is a natural consequence of the CFG basis of the notation that, under this treatment of antecedent-anaphor relations, objects do not themselves contain an indication of their antecedents (or anaphors). Instead, this information is present in the construct that contains them. For the same reasons, the notation involves an interesting commitment in some kinds of context senstive translation.

Consider, for example, the translation of the verb know into French, which (for simplicity) we will take to be savoir if there is a sentential complement, and connaitre if there is a nominal complement. i.e. in standard rewrite notation:

(16) a. know ~> connaitre /__ np-obj
b. know ~> savoir /__ s-obj

Our notation has no direct analogue to this sort of statement: the context sensitivity has to be taken care of in the translation of the construction containing the verb. Suppose c_{32} is the English constructor which builds the IS representation of Jules knows S, and suppose that English-French transfer contains the following T-rules:

(17) a. know ==> connaitre
b. know ==> savoir
c. c_{32} ==> c_{f36}

Then c_{32}: know, Jules, S will produce both of:

(18) (i) c_{f36}: connaitre Jules S'

(ii) c_{f36}: savoir Jules S'

and what one expects is that unification in (i) (or some later translation of (i)) will fail to produce a construct (e.g. because c_{f36} checks the syntactic category of its third argument). This seems a very natural account for cases such as these, where the target G contains the information for making the right choice. But one cannot expect this always to be so (e.g. where the information required is only part of the source language), and in such cases context sensitive T-rules will be required (cf relaxation (c) above). It is clear that this notation is committed to such cases being less common, at least.

As already noted, this model provides for a high degree of modularity in principle. In fact, the degree of modularity is rather extreme: not only are individual Gs modules, but individual constructors and atoms are modules also. Though the use of a feature theory allows some generalisations to be captured, the degree of modularity means that many generalisations that hold 'horizontally' (across languages), and 'vertically' (within languages) are missed.

The most obvious case of horizontal generalisations are 'invariances' and default translations across languages. For example, one does not expect the value of the attribute which identifies individual lexical items to change normally during analysis and generation, and the simplest and most restrictive view of transfer would be that only this attribute changes. Similarly, one knows that syntactic subjects normally correspond to semantic agents, and vice versa.

Within individual languages some capacity to capture generalisations across constructors is a prerequisite of some of the modifications mentioned above (in particular, CF type treatments of unbounded dependencies depend on some such mechanism if massive redundancy is to be avoided). More generally, one would like to be able to state conventions (about e.g. the percolation of attributes from heads of constructions to constructions) once and for all, rather than having to state them separately in each constructor, and there are various defaults which could make construction of Gs easier (e.g. the default case is that verbs have regular morphology).

Here we will briefly describe a fairly simple extension of the basic notation which is capable of dealing with these phenomena: injection rules, which describe (relations between) classes of constructs by stating partial descriptions of (pairs of) constructs. The following are examples:

(19) (_, {cat=v, morph-form=regular })
(20) (_, {cat=X, ...})[(gov, {cat=X, ...})]
(21) (_, {cat=s, tense=v1}) [] ==>
(_, {time=v2}) []

The idea being that such rules can be used to 'inject' generalisations into existing constructors, atoms, or T-rules. The normal problem with such 'meta' devices as these is controlling their interaction. A very simple way of avoiding this would be to adopt the following semantics: applying an injection rule I to a rule (atom, constructor, or T-rule) R succeeds if I unifies with R, in which case the

unification replaces R. This semantics means that injection rules cannot affect the cardinality of the rule set, but it greatly simplifies the form and content of the rules that must be written, and provides a perspicuous way of stating certain generalisations.

As stated, injection rule (19) is intended to unify with all atoms that have cat=v, and to add in the information that they have regular morphology. Notice that this injection rule will fail to apply to any atom that already has a different specification for morph-form, so there is a straightforward way of treating exceptions such as irregular verb morphology. (20) is a very simple example of a percolation injection. When applied to the constructors of a relational level G, it will ensure that the category of the head (gov) of a construct is percolated to the construction. Again, exceptions can be stipulated in individual constructors. (21) is intended to state a correspondence between tense=v1, and time=v2, and will inject this relationship into all T-rules that translate sentences.

It is appropriate to end by mentioning the most obvious open questions, since they suggest the direction which future work should take.

(i) We have not yet investigated the implications of the model for robustness, and while the model has been set up so that T-rules should be reversible to a large extent, we have insufficient practical experience with it to know how far this potential can be exploited.

(ii) Perhaps the most obvious theoretical commitment is that the notation is linguistic in nature, designed for representing linguistic knowledge (it would not be a very natural method for representing more general 'real world' knowledge). We think this is appropriate in MT, which is fundamentally about relating linguistic objects. However, as many examples indicate, there is an important role for general knowledge in MT, and this must be accommodated somewhere. No doubt various compromises are possible, and there is certainly room in the model for such quasi-linguistic entities as semantic features, but taking the model seriously involves rejecting knowledge representation languages as levels per se. The role of general knowledge representations cannot therefore be a step in translation, and can only be to provide a method of choosing between alternative representations at linguistic levels.

(iii) This leads directly to another point: the framework provides a number of ways of coping with non-determinism (filtering by target Gs, context sensitive T-rules, e.g.), but there is no method for the explicit comparison of competing representations (e.g. as in 'preference semantics' Wilks (1978)), and to provide such a method seems beyond the scope of the model we have described. Practically, it is not clear whether this is a problem or not, however, an approach which is consistent with the general spirit of the model might be to define a number of 'choice' levels, at which choices between alternatives would be made (IS is the obvious candidate). We would require G and T rules to be set up so that all alternative representations at these levels would be translationally equivalent, so that choice could be arbitrary.

(iv) We have assumed an extremely simple feature theory (e.g. we have not allowed attributes to take features as values). It is clear that a more sophisticated theory is desirable, and some work has been done in this direction within the model. However, what is not clear is how the extra descriptive power of an extended feature theory affects the 'pragmatics' of the model -- the way the model should be used for linguistic description.

Investigation of this model is still at an early stage, and much of the above is speculative or schematic. However, despite its preliminary status, we feel the approach described here is promising, and we hope we have said enough to show why we feel it is worthy of attention.

More information on the Eurotra project can be found in e.g. King & Perschke (1982) and King et al (1985).

References

D.J.Arnold, L.Jaspaert & L.des Tombe "ELS-3: Eurotra Linguistic Specifications (January 1985)" Eurotra Report ETL-5, DGXIII, CEC Luxembourg.

D.J.Arnold, L.Jaspaert, R.Johnson, S.Krauwer, M.Rosner, L. des Tombe, G.B. Varile, & S. Warwick "A MU1 View of the (C,A), T Framework in Eurotra" in Proceedings of the Conference on Theoretical & Methodological Issues in Machine Translation of Natural Languages, Colgate Univ., Hamilton, N.Y. 1985, pp. 1-14.

R.Kaplan & J. Bresnan "Lexical Functional Grammar: a Formal System for Grammatical Representation" in The Mental Representation of Grammatical Relations, J. Bresnan (ed), MIT Press, 1982.

M.Kay "Functional Unification Grammar: a Formalism for MT" in Proceedings of COLING 84 Stanford, California, 1984, pp. 75-78.

M. King, R.L.Johnson & L.des Tombe "Eurotra: a Multilingual system under Development" in Computational Linguistics, 11:2-3, 1985, pp. 155-169.

M.King & S.Perschke "Eurotra & its Objectives" in Multilingua, 1:1, 1982, pp.27-32.

J.Landsbergen "Isomorphic Grammars and their Use in the Rosetta Translation System" to appear in Machine Translation: the State of the Art, M.King (ed), Edinburgh Univ. Press.

R.Montague Formal Philosophy, Yale University Press, 1974.

L. des Tombe, D.J.Arnold, L.Jaspaert, R.Johnson, S.Krauwer, M.Rosner, G.B. Varile, & S. Warwick "A Preliminary Linguistic Framework for Eurotra (June 1985)" in Proceedings of the Conference on Theoretical & Methodological Issues in Machine Translation of Natural Languages, Colgate Univ., Hamilton, N.Y. 1985, pp. 283-288.

Y.Wilks "Making Preferences More Active" in AI Journal, 11:3, 1978, pp. 197223.