The Rosetta Translation System

Margreet Joyce Sanders
Philips Research Laboratories
Eindhoven - the Netherlands

1 Introduction

Successful communication very much depends on a language known to all parties involved. In case there is no language shared by all, translation must take place. The ideal for communication between people on a computer network would be to have a fully automatic translation system, available to all participants, so that everyone can type a message in his or her own mother tongue, while the addressee almost immediately receives a translated version on the screen. Such a translation system should have command over two kinds of knowledge: all *linguistic aspects* of the languages to be translated (including both grammars and dictionaries), and *knowledge of the world* (also including knowledge of the context in which the conversation takes place), to disambiguate the many ambiguities always present in natural language. Perhaps one day such a system may be developed, but at present the latter aspect, knowledge of the world, cannot be formalized well enough to be incorporated into a translation system. Fortunately, during computer translation in a networking environment the computer can communicate with the sender of the message and may ask this sender for help in disambiguating anything linguistics cannot solve. What such an 'interactive' system crucially needs is a way to find *all possible* interpretations of a given sentence. On the one hand, it is necessary to decide which constructions and interpretations are syntactically or semantically impossible, so that the sender of the message need not be pestered with too many questions, and on the other hand, the system may not overlook possible interpretations or it will not ask enough questions to fully disambiguate the given message.

What the Rosetta project aims at is to develop a translation system of the sort I have just described. The current phase of the project, Rosetta3, will be completed early in 1989, and encompasses an extensive specification of the syntax of the languages Rosetta deals with, i.e. Dutch, English and

Spanish. In this paper, the five linguistic translation principles which are used in the Rosetta system will be presented, on the basis of a step-by-step explanation of how the system works. Starting with a source language sentence, a translation is derived in six steps. These six steps, with the names of the intermediate results, are presented in Fig. 1, and will be explained in more detail below. Some indication will be given of the way in which Rosetta's five principles, being the Principles of Explicit Grammars, One Grammar, Interlinguality, Compositionality and Isomorphy, take care of a number of problems that occur during translation by means of a computer.

The overview given here is brief and informal. For a more extensive description of the structure of the Rosetta translation system, the reader is referred to the articles mentioned at the end of this paper.

The Rosetta team working on this project consists of both linguists and information scientists, under supervision of the leader of the project at the Philips Research Laboratories, Prof. Jan Landsbergen.

2 Morphology

The starting point for any translation is recognition of the strings of input characters as word forms of the source language. This is done in a *morphological analysis* component. Throughout this paper, the details of the Rosetta system will be explained using this one example sentence:

Does he love flowers?

For a simple sentence like this, the morphology recognizes a form like *does* as derived from the noun *doe* (plural) or from the verb-stem *do*, in which case it must be a third person singular. This morphological decomposition takes place in two steps: first, segmentation rules blindly remove any substring which might be an inflectional or derivational affix, then lexical rules check whether the resulting stem exists in the source language and can take the affix found in the segmentation phase. A successful combination of stem and affix is representend in a socalled 'Lexical S-tree'. The basic, non-inflected forms are all in a large dictionary, together with attributes indicating their morphological and syntactic behaviour. For English, the attributes of the verb *love* would specify that it has a regular past tense and participle, that the verb is a main verb, that it is not reflexive (like 'to perjure *oneself*'), that it does not take an obligatory preposition (like 'to talk *to* sb.'), etc.

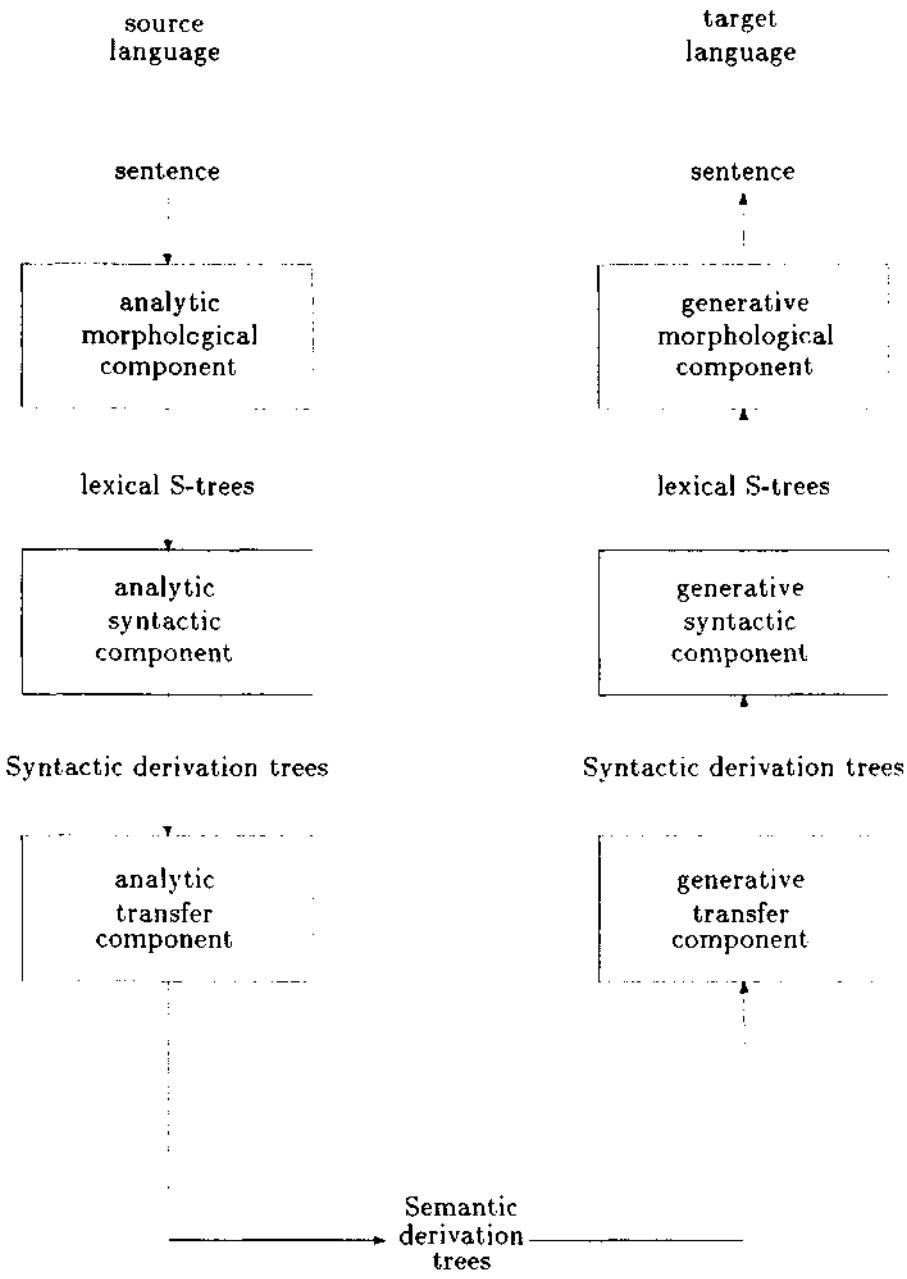

Figure 1: *Global design of the Rosetta system*

3 Syntax

One must realise that word-by-word translation, based on morphology alone, usually does not lead to an acceptable result. For the example sentence given above many translations would be possible, since the strings *does*, *love* and *flowers* are quite ambiguous, all being verb forms and nouns (*does* = noun pl, verb 3rd ps sg; *love* = noun, verb infinitive, present pl. and sg; *flowers* = noun pl, verb 3rd ps sg). One translation of the sentence into Dutch might be *Doet hij liefde bloeit?* (Does-he-love(n)-flowers(v)), which is quite wrong and ridiculous. Correct choice of verb vs. noun might produce *Doet hij van bloemen houden?* (Does-he-of-flowers-love), which is still wrong. Of course, what is needed is a *grammar* of the source language, i.e. an explicit description of the syntax and the semantics of the language saying in what way or ways words in a sentence belong together to make up the structure of the sentence and its meaning, and which structures occur in this specific language. For our example sentence, the grammar would have to find out that we are dealing with a question, that there is a verb (love) with an auxiliary verb (do), a subject (he) and an object (flowers), and that the tense of the question is present tense. The grammar devised for Rosetta is called the *M-grammar*. Syntax and semantics have been separated in two components, as could be seen in Fig. 1, and the syntactic component again exists of two parts. In analysis, these two parts are called the *Surface Parser* and the *M-Parser*, and they will be explained below.

3.1 Surface Parser

Once the sentence has passed the analytical morphological component, the resulting lexical S-trees first go to the *Surface Parser*. This parser computes a set of tentative structures covering all the words in the sentence, first combining them in larger units and then forming a set of so-called 'Surface trees', in which the syntactic relations between the elements in the sentence are indicated. The Surface Parser recognizes that in the example sentence mentioned it makes no sense for the forms *does* and *love* to be analysed as nouns. A very simplified example of how the Surface Parser deals with the example sentence is given in Fig. 2. The node names are given in capitals, the relation names are in small letters. The Surface Parser may come up with a number of solutions, some of which will appear to be impossible later on. The existence of a separate Surface Parser is motivated by the way in which the second part of the syntactic component, *M-parser*, is organized.

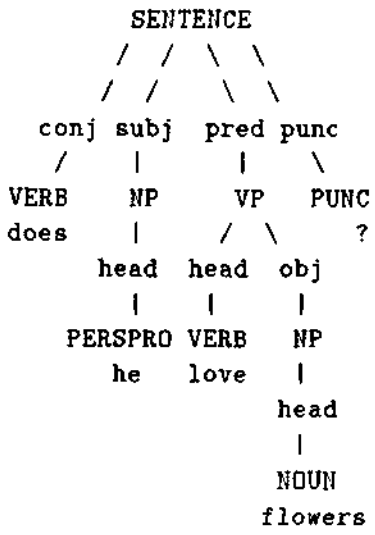

Figure 2: *Simplified Result of the Surface Parser*

In analysis, this component takes as its input a full Surface tree, with all the nodes and relation names specified as was indicated in Fig. 2.

3.2 M-Parser

In *M-parser*, the syntactically correct structures are selected, and at the same time the foundation is laid for determining the *meaning* of the structure. M-parser consists of a large number of syntactic rules, all having transformational power. These rules may change the nodes or relation names in the syntactic structure, or modify the values of attributes of specific nodes. A distinction is made between rules and transformations. *Rules* (in the strict sense) are syntactic operations that have a meaning, or at least that convey information which is relevant for translation. *Transformations* are meaningless and usually only serve to adapt the syntactic structure to the demands of the language under consideration. Transformations do not convey information that must be remembered in translation. The order of application of all the rules and transformations is determined by *control expressions*, which say which rules and transformations may or must follow which other

rules. Many paths are possible, but the rules are written in such a way that only the successful trees remain. Continuing with the example sentence, a very simplified explanation is presented of how M-parser works. Note that only the bare essentials are given here.

First, the characteristics of the sentence as a whole are considered. It is clear that we are dealing with a question here, and since a question means something else than a statement, this observation should be remembered in translation. Let's call the syntactic rule which indicates the question aspect and removes the question mark **RQuestion** (*Does he love flowers)*. Now, we may safely invert the order of subject and operator, transforming the sentence structure into that of an ordinary declarative sentence (note it still has the auxiliary *does*). The inversion itself is not relevant for translation, and we call the transformation which undoes it **TInvert**. We now have a structure to which we can apply the rules for normal statements (*He does love flowers*).

We proceed by identifying the tense of the operator as present tense; a meaningful rule **RPresent** must be applied (*He does love flowers*). As part of the check on whether the structure proposed by the surface parser is syntactically correct, it is also determined whether subject and operator agree in person and number. This is again a syntactic requirement of English, which in itself carries no meaning. In our example sentence, the agreement demands are satisfied, and the operator can be changed to a verb stem now, to provide an even more generalized sentence structure. All this is done by the transformation **TAgree** (*He do love flowers*). The next step is that we recognize that an auxiliary has been used. For questions, the auxiliary *do* has become quite meaningless, since we already determined that a meaningful rule RQuestion should apply, and we may safely remove the auxiliary with a transformation **TDoAux**. Once this rule has applied, we are left with only a few elements: *He love flowers*.

The next step is to replace the subject and the object by abstract variables. This is done in two **Substitution Rules**. The two noun phrases extracted from the sentence are further reduced to a personal pronoun and a noun by Noun Phrase Rules, which I here call **RNP1** (for the bare NP consisting of the personal pronoun *he*) and **RNPGen** (for the generic NP *flowers*). We now proceed by deciding whether the structure we have left (a specific verb with two abstract variables functioning as subject and as object: *x1 love x2*) is acceptable. First, the structure is compared with the verbpatterns the verb *love* can take. This means that we have to check whether *love* takes a direct object. This is a purely language dependent

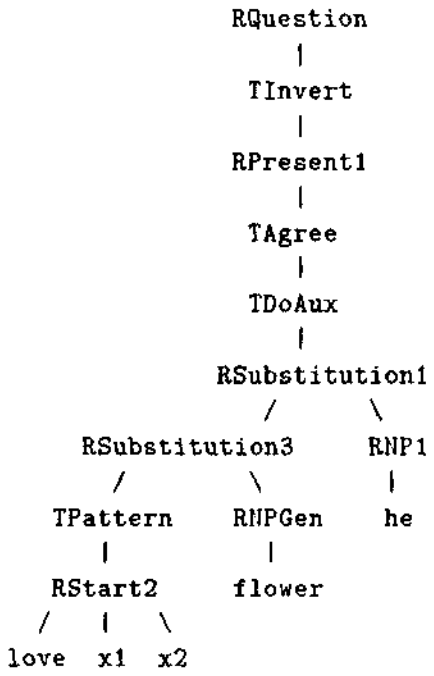

Figure 3: *Simplified Syntactic Derivation Tree of Source Language, including the Transformations applied*

check (in another language, the relation might well be different), and thus it is performed by a transformation **TPattern**. Since everything fits, the relation name 'direct object' may now be replaced by the neutral term 'argument'. In the final rule, **RStart1**, it is decided that the verb *love* may indeed take a subject and one other argument, and then we are finished with the main structure. The structure in which the rules and transformations that were applied during the derivation are represented is called a *syntactic derivation tree*. As a summary, the full derivation tree is presented in Fig. 3.

4 Semantics

The next point to explain is how and in what sense the Rosetta grammars associate meaning with a sentence. The rules written for the M-grammar are based on the *Compositionality Principle*, which says that the meaning of an expression is a function of the meaning of its parts. This is a principle borrowed from the framework of Montague Grammar (hence the *M* in the name of our grammars). Only an informal explanation will be given here of how and to what extent this principle is used in M-grammar. A sentence consists of an ordered string of words, most of which correspond with a certain meaning. During analysis of the sentence, the words appear to have been combined by applying rules and transformations of the language in question. When all the grammar rules used in building the sentence have been traced, the result is a syntactic derivation tree, showing the rules (meaningful) and transformations (meaningless) in their order of application, and the basic expressions (in our example *he, love, flower*) on which they operated. In Rosetta, this syntactic derivation tree is used to determine a meaning representation. This step is performed in the *Analytic Transfer*, which forms the link between the syntactic representation and the Intermediate Language (IL).

Starting from the syntactic derivation tree, every *rule* in this tree is mapped onto an IL meaning rule, and every basic expression is mapped onto a representation of the meaning of this basic expression. The *transformations* from the syntactic derivation tree have no IL representation, since they only served to check whether a syntactic structure was compatible with the specifications of the specific source language, but carried no meaning in themselves. The meaningless words in the sentence (as the auxiliary *do* in the example given above) have no corresponding basic expression in the syntactic derivation tree at all, so they do not have an IL representation either. The resulting representation in the Intermediate Language is called a *semantic derivation tree*, and it closely resembles the syntactic derivation tree. The Intermediate Language representation is given in Fig. 4.

Both the meaning rules and the basic expressions they operate on are represented by unique names. The Intermediate Language used in Rosetta contains only three kinds of elements: there are rule names, names for basic expressions, and trees in which these are combined. So the meaning of a sentence or a basic expression is not spelled out fully, but simply represented by a unique name. As long as this unique name can be interpreted by all languages the system deals with, there is no need to be more specific.

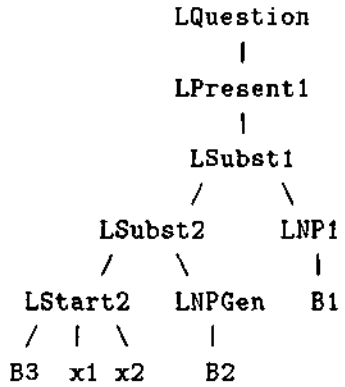

Figure 4: *Simplified Semantic Derivation Tree*

Rosetta's choice for an Intermediate Language instead of direct transfer of the structural representation found in the M-grammar to the target language, is called the principle of *Interlinguality*.

It is important to realize that a sentence is always translated into an interlingual expression first, and only then will be input to the grammar of the target language. To a certain extent, the Rosetta grammars are language-specific, without any direct reference to structures occurring in other languages. Some reservation on the language-specificity will be made below. Also, questions for disambiguation asked to the sender of the message are based on the meaning representation used in the Intermediate Language, not on any specific target language.

It should also be noted that in Rosetta3, the sentence is considered the basic unit for translation. This working strategy is partly caused by the fact that at present, it cannot yet be satisfactorily formalized how the meaning of expressions is determined by the text in which they occur. Also, it seems more logical to explore the full capacity of sentence-based grammar first, and only then proceed to the more intricate field of text semantics.

5 Isomorphic Grammars

One final step remains: to get from the Intermediate Language to the target language. The third principle formulated by Rosetta is the *Principle of*

Explicit Grammars. This principle says that not only for the source language but also for the target language a full description of the grammar must be used. In generation, the grammar of the target language must define explicitly how elements of the target language must be combined to form the structures and meaning needed for translation of a given sentence. Of course, since the organization of the interlingual meaning representation lacks any direct reference to overt syntactic structures, Rosetta must have an explicit grammar for generation. However, there are independent reasons for preferring an explicit grammar for generation, for instance the fact that only an explicit grammar can guarantee that no syntactically ill-formed structures will be produced in the target language.

A separate point concerns the decision whether each language must have two grammars, one for its analysis and one for its generation, or only one grammar. In Rosetta, a clear choice is made for the one grammar option, and a *One Grammar Principle* has been formulated, which is also called the *Reversibility Principle*. Analysis and generation are covered by the same grammar rules, working either one way or the other, i.e. combining basic elements to form a structure or breaking an existing structure down to its basic elements. Now, the fifth and most characteristic principle of the Rosetta system comes to the fore: the principle of *Isomorphy*.

In Rosetta, it is claimed that two sentences are translations of each other if they have the same semantic derivation tree and, hence, isomorphic syntactic derivation trees. That means that from an IL tree, a new (generative) syntactic derivation tree is made for the target language which, as far as the meaningful rules are concerned, must have the same geometry as the analytic syntactic derivation tree that produced the IL-tree, but which contains rule names and basic expressions specific to the target language. This is done in the *Generative Transfer*. The principle of Isomorphy means, in practice, that two sentences are translations of each other if they are derived from the same basic meanings in the same way. To specify which of the transformations needed in the target language may occur between which meaningful rules, the same control expressions are used as those functioning in analysis of the target language. In generation, the ordering of the meaningful rules themselves is fixed by the syntactic derivation tree, and only the transformations must be given a place. A simplified representation of a successful generative derivation path may look something like Fig. 5. Note that Fig. 5 is not a derivation tree in the proper sense, since it is not known in advance which of the many transformations will be successful. Only when *M-generator* has worked its way upward from the bottom to the

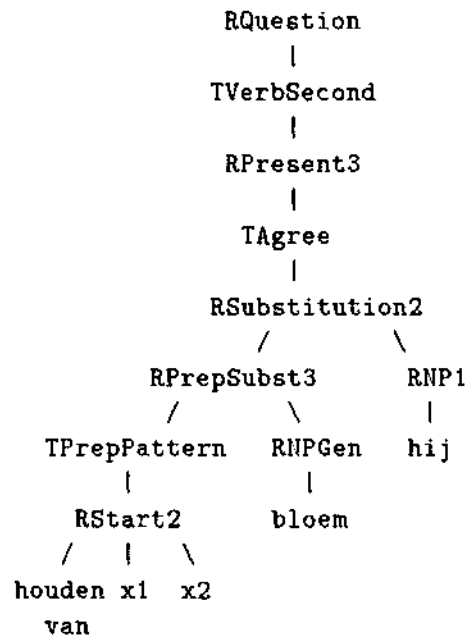

Figure 5: *Simplified Representation of Successful Derivation Path of Target Language Sentence*

top of the tree, executing all the rules and trying all the transformations, can the representation as given here be determined. M-generator is the generative 'version' of M-parser, i.e. the same set of rules and transformations, but working the other way around, as was explained when introducing the Reversibility Principle. When M-generator has finished, all the words needed for a correct translation of the input sentence are in their correct surface order.

Finally, the generative counterpart of the Surface Parser, called *Leaves*, picks off all the words from the tree, and passes them on to the generative morphology, which specifies how the third person of the verb *houden (van)* (love) is spelled, etc. The translation process has been completed.

For this isomorphic approach to work, the M-grammars of the languages involved have to be attuned to each other. It must be possible to express the 'meaning' of the syntactic rules occurring in one language (e.g. the ques-

tion aspect, the tense aspect, and substitution) in all other languages too. although the rules may work on quite different syntactic structures, and all languages must have basic expressions corresponding to each other, although these expressions may have different syntactic categories. This causes the Rosetta grammars to be different than they would have been if they were fully language-specific, since phenomena with semantic relevance occurring in one language must always be taken into account in all other languages, even if these do not show any overt sign of the phenomenon.

6 Structural Mismatches

As was explained above, Rosetta does not preserve the original syntactic structure of a sentence. Therefore, it is relatively easy to translate words of different syntactic categories into each other, or to produce a syntactic structure in the target language that is quite different from the one in the source language. Let me give an example of both phenomena.

A simple example of a rule that works on somewhat different syntactic structures in Spanish and English but expresses the same meaning is the formulation of a generic noun phrase (NP). In English, one says: *He loves flowers*, without any article, to express that someone likes a certain (generic) set of objects. Since such an English bare NP, without any article, expresses an idea of genericity, it is covered by a special Noun Phrase Rule for generics, which I here call RNPGen. An NP with an article, as in the sentence *He loves* the *flowers*, would be covered by quite another NP rule, dealing with definite NPs. The English RNPGen is mapped onto an IL rule LNPGen. In Spanish, this rule is translated into a noun phrase rule producing a Spanish generic NP, but here a definite article is needed to express the same fact: *Le gustan* las *flores* (Him-please-the-flowers). Therefore, the Spanish rule for generic Noun Phrases will introduce the article *las*.

In the translation of the example sentence discussed above (*Does he love flowers?*) an even simpler structural difference occurred between source and target language: in English, *love* takes a noun phrase as direct object, while the structure in Dutch had to provide for a prepositional object (houden *van*). Although the actual syntactic structures are different, this difference is caused by language specific demands (verbpatterns) only, and has no consequences for translation.

An often quoted example of the second phenomenon, where basic expressions have different syntactic categories, is the pair *like* (verb) - *graag*

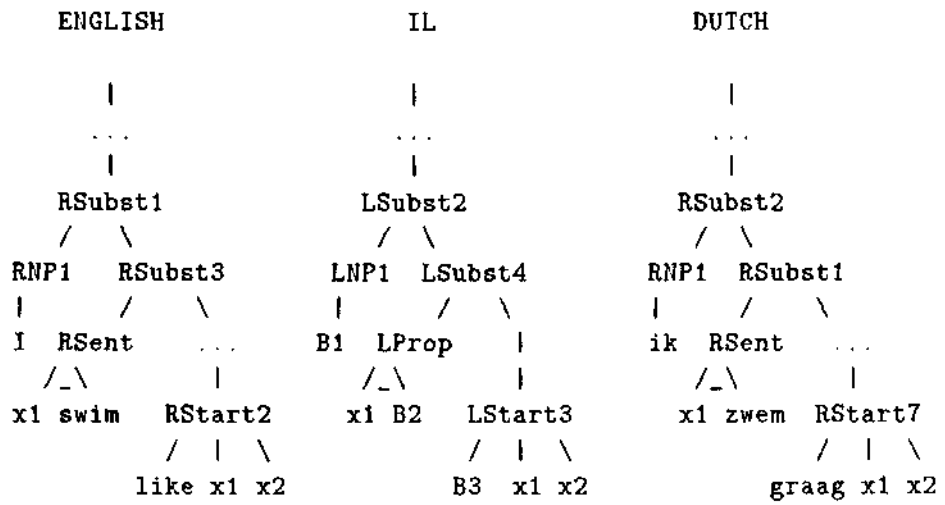

Figure 6: *Simplified Derivation Trees for the* Graag - Like *Case*

(adverb) in English and Dutch. The sentences *I like to swim* and *Ik zwem graag* (I-swim-gladly) are considered translations of each other. In Fig. 6, a very simplified model is presented of the relevant part of the two syntactic derivation trees and the IL-tree, to illustrate how Rosetta deals with this. The English rules are all quite normal, **RSubst3** replacing a sentential complement with a to-infinitive by a variable (*x1 like x2*). The interlingual meaning rule, **LSubst4**, is mapped onto all kinds of Dutch rules substituting a sentential or propositional element for a variable. One of these rules is **RSubst1**, which happens to be a special rule. In generation it takes a propositional structure with an adverb as its main element (*x1 graag x2*) and transforms it into a sentence once another sentence (*x1 zwem*) is substituted into it to replace variable *x2*, resulting (after a few more transformations) in a sentence with the elements *x1 zwem graag*. Notice once more that the syntactic derivation trees themselves do not reveal anything about the syntactic structures present in the two languages.

7 Summary

In this paper, an overview was presented of how the Rosetta translation system works. While tracing the translation process for one example sentence, the main principles used in Rosetta were explained, being the principles of Explicit Grammars, One Grammar (or Reversibility), Interlinguality, Compositionality, and Isomorphy. These principles allow the development of a translation system which recognizes all interpretations of a sentence that are possible given the description of the grammar of the specific language. The Rosetta team will complete a research prototype of the translation system, Rosetta3, early in 1989. In the next version of the system, a medium scale application-oriented version which hopefully will be ready in 1991, full use will be made of the interactive facility of the system, so that the sender of a message to be translated will be asked for information to solve the ambiguities present in the sentence.

Acknowledgements
The author wishes to express her thanks to all those who commented on earlier versions of this paper, and thus contributed to a clearer and more coherent report of the Rosetta translation system.

Publications on Rosetta
For the interested reader, a short list is given of introductory articles on the Rosetta Translation System. Further references can be obtained through the author.

Appelo, L., C. Fellinger and J. Landsbergen, 'Subgrammars, Rule Classes and Control in the Rosetta Translation System', Philips Research M.S. 14.131, *Proceedings of European ACL Conference*, Copenhagen, 1987, pp. 118-133

Appelo, L. and J. Landsbergen, 'The Machine Translation Project Rosetta', Philips Research M.S. 13.801, *Proceedings First International Conference on State of the Art in Machine Translation*, Saarbrücken, 1986, pp. 34-51

De Jong, F. and L. Appelo, 'Synonymy and Translation', Philips Research M.S. 14.269, *Proceedings of the 6th Amsterdam Colloquium*, Amsterdam, 1987 (a Dutch translation will appear in *De Spektator*, 1988)

Landsbergen, J., 'Isomorphic grammars and their use in the Rosetta translation system', Philips Research M.S. 12.950. Paper presented at the Tutorial on Machine Translation, Lugano, 1984. In: M. King (ed.), *Machine Translation the State of the Art*, Edinburg University Press, 1987

Leermakers, R. and J. Rous, 'The Translation Method of Rosetta', Philips Research M.S. 13.701, *Computers and Translation*, 1986, Vol. 1, nr. 3, pp. 169-183