11
SUSY

11.1 Background

Research on MT at the Universität des Saarlandes in Saarbrücken, Germany began in the mid 1960s with the development of a parser for German. A more substantial project began in 1967, just at the time when elsewhere MT research was coming to a stop as a result of the ALPAC report (section 1.3). This explored the possibility of adapting the Systran system for Russian–German translation, but the attempt failed and the group started work on its own prototype. In 1972 the project combined with other activities on data processing and mathematical linguistics to form the Sonderforschungsbereich 100 Elektronische Sprachforschung (SFB-100) ('Special Research Group 100: Electronic Language Research'), funded primarily by the Deutsche Forschungsgesellschaft ('German Research Association', a government body). The Russian–German prototype was the starting point for research on the multilingual system known as SUSY (*S*aarbrücker *Ü*bersetzungs*sy*stem 'Saarbrücken Translation System'). The languages involved have been German, Russian, English, and French (also briefly, Esperanto); the emphasis throughout has been the analysis and generation of German.

Over a period of 20 years, SUSY underwent many changes in response to developments in the field. There are consequently problems of presenting the system. This chapter describes basically the mature version of the SUSY-I system of the mid 1980s, programmed in Fortran, and implemented until 1981 on a Telefunken TR440, and thereafter on a Siemens computer. The later SUSY-II demonstrated further developments, incorporating insights from the earlier version and showing substantial influences from the Eurotra project (Chapter 14) with

which many researchers at Saarbrücken were heavily involved. Indeed, work on SUSY came to an end in 1986, because in effect the research had merged with the work on Eurotra.

11.2 Basic system design

SUSY is basically a **transfer** system with monolingual analysis and synthesis (generation) phases and a bilingual phase in which lexical and structural transfer takes place. The structures that are computed are essentially dependency tree structures, a reflection of the predominance of the Valency grammar approach among German linguists (section 2.9.5). The input can be optionally pre-edited. There is no interactive post-editing. Due to the incorporation of 'fail-soft' modules, SUSY always produces some sort of output.

The system is highly **modular** in one major respect: the translation process is broken down into separate sub-processes. The modules are applied in strict sequence. In general, analysis and synthesis modules are language-specific, while transfer modules are designed for specific language pairs. However there are also, in the version we describe, some modules which are not independent of particular pairs, especially in analysis. In this respect, SUSY is not quite a pure multilingual transfer system, where, it will be recalled (Chapter 4), the analysis module for a source language should combine with any target languages in the system. Close inspection of some SUSY analysis modules reveals that they do take account of target languages. This explains why the different versions of SUSY reported in the literature often diverge slightly in their details. While the principle of multilinguality may be undermined as a result, the modularity of SUSY must be seen as a positive feature. It permits revisions and changes in some parts of the analysis and synthesis processes while retaining other parts from previous versions, both in order to incorporate new linguistic ideas and approaches, and to adapt the analysis modules of one language pair for use with a different language pair.

In another respect, SUSY does not score highly. There is generally no clear separation of linguistic and algorithmic aspects, although again there have been differences between versions. The early SUSY system had effectively no such separation in the sense that, although highly modularised, the various modules were directly programmed in Fortran. The major innovation of the later SUSY-II version was the use of a rule-writing formalism.

The basic stages of processing in SUSY are as in Figure 11.1.

The substages (or 'operators' in the SUSY terminology), with their titles, are as follows. Text input (LESEN) may be preceded by an optional stage of pre-editing. Dictionary look-up (WOBUSU) and its associated morphological analysis is followed by homograph resolution (DIHOM) and then by various levels of syntactic parsing: SEGMENT identifies clauses and phrases and a tentative dependency structure, NOMA identifies nominal groups and VERBA verbal groups. The whole sentence receives a complete parse by KOMA (complement analysis), and problems of lexical and structural ambiguity are dealt with by the semantic disambiguation module SEDAM. The single stage of transfer (TRANSFER) involves both lexical and structural transfer.

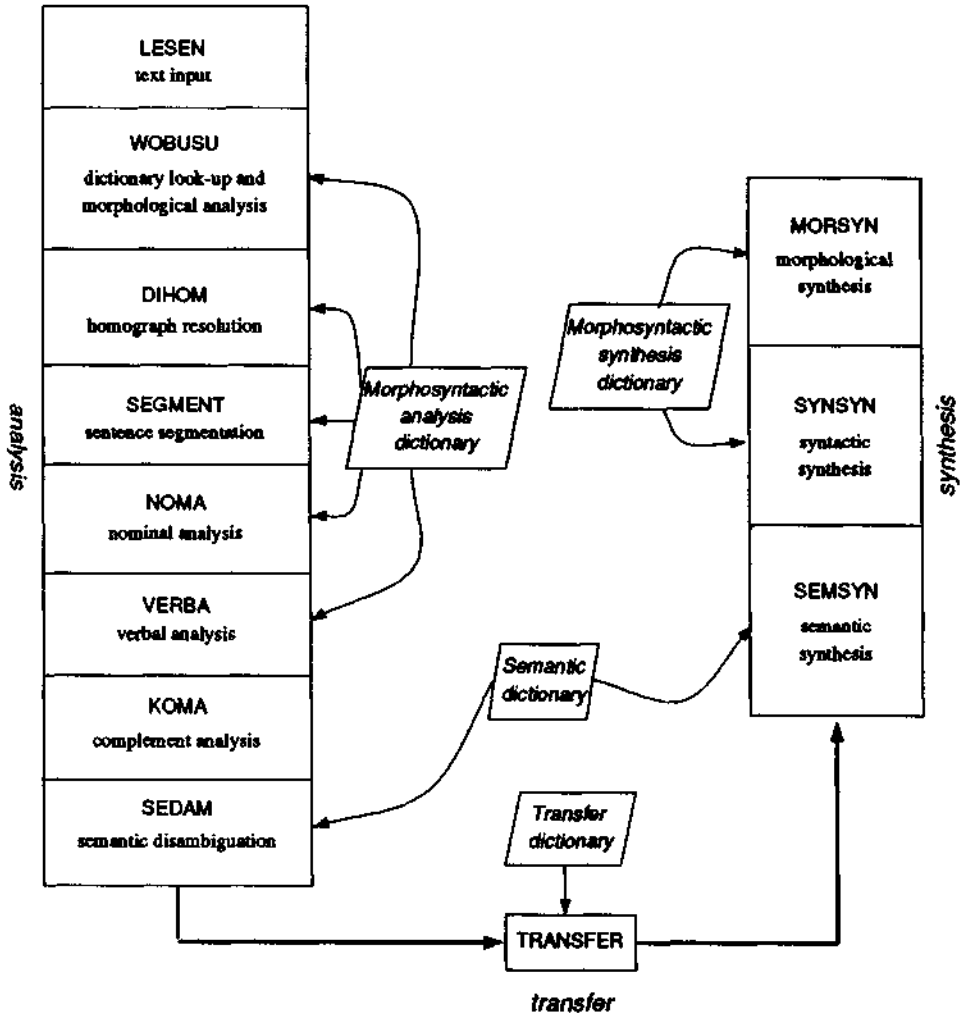


Figure 11.1 Translation process in SUSY

The target language is generated by passing through the stages of semantic synthesis (SEMSYN), syntactic synthesis (SYNSYN) and morphological synthesis (MORSYN).

There are separate monolingual dictionaries for source and target languages, containing essentially morphological and syntactic data for lexical items. Each language has separate dictionaries for analysis and synthesis; the largest are the German analysis dictionary with 140,000 entries, and the German synthesis dictionary with 14,000 entries; others are much smaller. Semantic processes involve 'semantic dictionaries' for source and target languages; in this case the same monolingual dictionary is used for both analysis and synthesis, the largest being again for German (75,000 entries), with others much smaller. Semantic dictionaries include semantic features and syntactic and semantic conditions for polysemes, in particular for prepositions. The bilingual transfer dictionaries

comprise basically lexical equivalences with some information on structural contexts. Their relatively small sizes (only English–German more than 10,000 entries) reflect the experimental nature of the SUSY project.

We describe the operation of the modules after first outlining the type of data structure found in SUSY, the pre-editing options and the mechanism for ensuring some output if any component fails (the RESCUE operator).

11.3 Data structure

As in any modular system, the data structure is very important, since it ensures communication between modules. We can consider it from two points of view, the computational and the linguistic. From the computational point of view, the data structure of SUSY is similar to that found in Systran; it is motivated by programming convenience, in the SUSY case the programming language being Fortran. SUSY uses a data structure where there is one data record per word, and these are stored in a sequential file. The 'word record' is divided into 'cells', each of one or more bytes, corresponding to specific information about the word. The translation process consists of successively rewriting the file, changing the values in appropriate cells.

From a linguistic point of view, the data structure is a dependency tree structure, in which the governor–dependent relationships for each word are identified. In a noun phrase such as *the very big system*, the noun *system* is the governor with immediate dependents *the* and *big*, and the adjective *big* having *very* as its dependent. At the sentence level, the main verb is taken to be the overall governor, with its complements (subject, object, etc.) and modifiers (circumstantial adverbials) as dependents. The dependency information is augmented by indications of the types of dependents, e.g. subject or object, modifier or determiner, and so on.

Although both Systran and SUSY use rather crude low-level data structures, the linguistic interpretation is more sophisticated in SUSY. Specifically, some of the cells are used as 'pointers', permitting the sequential file structure to be used for representing tree structures, in the following way. Each word record is numbered consecutively, according to the position in the sentence of the word it represents. Each byte in the word record is capable of storing a single character: a letter or integer. The first cell might consist of 20 bytes, say, and is used to store the string itself. The next five cells might be one byte each, and contain grammatical information from the dictionary (e.g. category N=noun, V=verb, etc., and if a noun whether S[ingular] or P[lural], or if a verb its person, number and tense, etc.). The next cell might be 30 bytes containing the stem (root) form and morphological information, e.g. that the stem allows only certain endings, and so on. (The sizes and nature of the cells suggested here are for illustrative purposes, and do not necessarily reflect accurately the exact structure of SUSY word records.)

We assume, again for the purpose of illustration, that bytes numbered 75 and 76 are used to store pointer information: in byte 75 will appear the record number of the governor of the current record, and in byte 76 a code indicating what sort

of dependent it is (e.g. S[ubject], O[bject], M[odifier] and so on). In this way we might have the data structure shown in Figure 11.2 for the sentence (1):

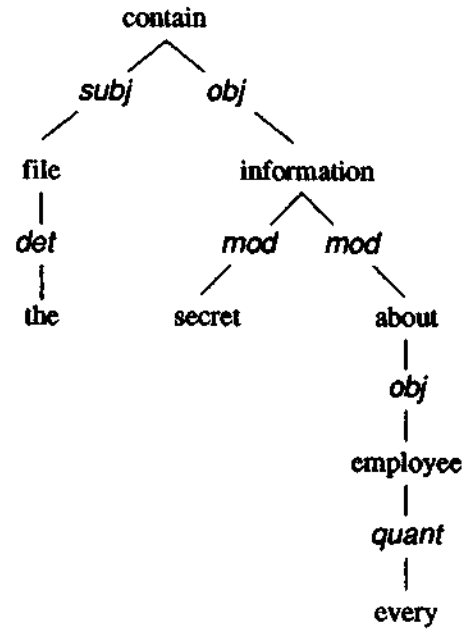
(1) The file contains secret information about every employee.

Record no.				
1	THE	D	THE	2D
2	FILE	NS	FILE	3S
3	CONTAINS	V3SP	CONTAIN	0
4	SECRET	A	SECRET	5M
5	INFORMATION	NS	INFORM	3O
6	ABOUT	P	ABOUT	5M
7	EVERY	Q	EVERY	8Q
8	EMPLOYEE	NS	EMPLOY	6O

Figure 11.2 Data structure (simulated)

The information in bytes 75 and 76 effectively represents the following dependency tree (2):

(2)



Although the data structure may be rather crude computationally, it can be seen that by using this pointer mechanism it can be used in a linguistically quite

sophisticated way. There are, however, problems when alternative analyses are possible (e.g. caused by category ambiguities, as we shall illustrate). The creation of alternative tree structures means that there have to be copies of word record files differing perhaps only in the details of one or two codes.

11.4 Pre-editing and the fail-soft RESCUE operator

SUSY provides the option of pre-editing source texts by inserting special codes to aid analysis. Pre-editing permits the enhancement of the system's performance, but is not an essential feature of it. The analysis processes look out for these codes but are also able to operate without them. Examples of some of the SUSY codes have been given in section 8.3.1. When we look in more detail at the SUSY analysis stages, we will see the effect of codes in different modules.

A characteristic and innovative feature of SUSY is the explicit incorporation of the **fail-soft** operator RESCUE, which comes into operation whenever something has gone wrong with any of the modules. Each module includes consistency checks to see that the input and output conform to what is expected. In other words, there are minimum entry requirements for each module. Within each module there is the expectation that certain computations will have been successful and a new result produced. If the expectation is not fulfilled (because the rules are too strict, or the input is very unusual) then the appropriate RESCUE operator is triggered: a mechanism which attempts to complete the work of a module by relaxing or ignoring the constraints which prevented the production of some output.

In this respect, SUSY is a 'try anything' system. Whatever the input it will be able to produce some sort of translation, even if no more than a partial word-for-word version. Moreover, the triggering of this special procedure means that the system itself is 'aware' that the result may be of a low quality and can indicate the fact this when producing output.

11.5 Analysis

11.5.1 Text input and dictionary look-up

The first module, called LESEN ('read'), is a stage of non-linguistic preprocessing. LESEN reads the text into the initial data structure, divides the text into sentences, and assigns each word an identifying number. It also normalises the text, which means taking account of typographical information (notably, for German, whether the word begins with a capital letter). For each word, a 'word record' is created, as described above (section 11.3) and including any pre-editing information.

The output of LESEN is passed to the next module, called WOBUSU (*Wörterbuchsuche* 'dictionary look-up'). Only words marked during pre-editing as proper names, e.g. *Tom=* are ignored. There are three monolingual dictionaries in SUSY: a high-frequency dictionary of grammatical words, a stem dictionary which includes all irregular forms and all irregular stems, and an idiom dictionary. The dictionary entry gives the stem or the full form, a lemma (or lexical item)

and the corresponding morphological and syntactic information. It also gives, depending on the category, certain paradigmatic information. For example, for a noun the information concerns its gender, plural form, prepositional valency (e.g. *Lust* AUF *Kuchen* 'desire for cakes', *Vertrag* MIT *Guinea* 'treaty with Guinea'), and so on. For a verb, the dictionary indicates whether it includes a separable prefix, what inflectional paradigms it belongs to, which verb type it belongs to (e.g. full, auxiliary, modal), whether it combines with *zu*, which form of passive is available (none, personal, impersonal, or both), whether the past participle can be used as an adjective, what case its reflexive pronoun takes, if appropriate, which auxiliary verb (*haben* or *sein*) is found in compound tenses, and what its valency frame is (including up to three prepositional objects).

11.5.2 Morphological analysis

The WOBUSU module also incorporates morphological analysis, which is broken down into two sub-modules, INFLECTION and COMPOUND.

The first module, INFLECTION, deals with single words and attempts to give an analysis in terms of stem + affix. It examines all possible segmentations of words and filters out those which are not plausible combinations of stems and affixes, i.e. where the proposed stem or affix form does not exist. For example, the word *Speichern* would be segmented as follows:

(3) SPEICHERN+0 SPEICHER+N SPEICHE+RN
SPEICH+ERN SPEIC+HERN SPEI+CHERN
SPE+ICHERN SP+EICHERN S+PEICHERN

When the stems (the parts before +) are looked up in the dictionary, only four are found to be plausible: namely *SPEICHERN* (the infinitive of the verb *speichern* 'to store', *SPEICHER* (the stem form of the noun *Speicher* 'store', or the stem form of the verb *speichern*), and *SPEICHE* (the stem of the noun *Speiche* 'spoke' of a wheel). Of these four possibilities, it can immediately rule out *SPEICHE+RN* since *+RN* is not a possible ending. Each of the remaining solutions is for the moment equally plausible, and the alternatives are stored in the word record, namely

(4) SPEICHERN+0 infinitive of *speichern*
SPEICHER+N dative plural of noun *Speicher*
SPEICHER+N inflected form of verb *speichern*

The resolution of this ambiguity is left to a later module. In this example there are only three alternatives, but multiple output from INFLECTION is quite frequent in German, which as we have seen (section 5.1) has a number of highly ambiguous endings out of context.

If INFLECTION fails to provide at least one solution, then the COMPOUND module is activated. This deals with compound words and derivations, which are a major problem for a language like German which permits very free and productive derivation and compounding. Whereas in English, compounding is an essentially syntactic phenomenon in that it concerns identifiable words used in sequence, in German it is a morphological problem. Individual elements of compounds are joined together to make single 'words', which are rarely listed as such in dictionaries.

The COMPOUND module works in a similar manner to INFLECTION, but segmentation is more complex: not simply identifying stem + affix, but recognising patterns of multiple affixes and roots:

(P (Z)) R (S)
(P) (G) R (S)
(P) R ((F) R)^n (S) $n \geq 1$

where brackets indicate optionality, the index n repetition, and

P = prefix
S = suffix
R = root
F = compounding unit *e*, *es*, *s*, *n*
G = pre-morpheme *ge*
Z = morpheme *zu*

For example (5).

(5) *herauszubringen* ('to bring out') = *HERAUS+ZU+BRINGEN* (P Z R)
ausgebildete ('educated'+inflection) = *AUS+GE+BILDET+E* (P G R S)
Geburtstagsgeschenke ('birthday presents') =
GEBURT+S+TAG+S+GESCHENK+E (R F R F R S)

If both INFLECTION and COMPOUND fail to suggest an analysis, the word is treated as 'unknown'. However, because German is highly inflected, it is often possible to guess the syntactic category of unknown words from their endings. In this way, the results of either module which provide plausible combinations of endings and unverified stems can be passed on to further stages. For example, the ending *-te* usually indicates a finite verb, as in *pflegte* ('cared for'), though it might not (e.g. *Kette* 'chain').

The results of INFLECTION and COMPOUND are passed to a final sub-module of WOBUSU, which deals with the identification of fixed phrases. The operation of morphological analysis before the recognition of fixed phrases allows SUSY to treat semi-fixed idioms which are nevertheless susceptible to grammatical inflection, e.g. noun phrases which take case endings, idiomatic verbs which are inflected for tense and agreement, and so on.

11.5.3 Homograph disambiguation

In the data structure resulting from WOBUSU is a sequence of words which have been looked up in the dictionary, many of which are ambiguous in category. The next phase DIHOM (*Di*sambiguierung von *Hom*ographen 'homograph disambiguation') attempts to resolve these ambiguities. The technique used is based on compatibility and distributional information, and is therefore more like 'stochastic parsing' models than the more traditional grammar-based approaches.

DIHOM consists of three sub-processes. Two general routines dealing with impossible sequences and with the ranking of possible ones are preceded by routines for the treatment of 'special cases'. These are specific items or classes of items which require a particular strategy for disambiguation. One special case is the German word *bis*, which can be a subordinating conjunction, a coordinating

conjunction, an adverb or a preposition. It can be treated as a special case because this particular combination of category ambiguities is unique to this word. Other examples are non-contiguous conjunctions such as *weder ... noch* ('neither ... nor'), *um ... willen* ('for ... sake'), *not only ... but also*, and so on.

The second sub-process is called INHIBIT, which in some accounts precedes the routines for special cases. This looks for and eliminates impossible sequences of categories. In German the sequence determiner + finite verb is impossible; therefore, if a determiner is followed by a word which could in isolation be interpreted as a finite verb (e.g. *das Verlangen* 'the demand'), this reading would be eliminated. A further example from German is that a finite verb form may not follow immediately after a preposition (**mit machte*); thus *vererbten* in *aus vererbten Grundstücken* cannot be a finite verb ('[they] inherited') but must be an adjective (derived form the verb), thus meaning 'from hereditary lands'.

The third and chief process in DIHOM is the calculation of 'weightings' for the remaining ambiguities based on tables of probabilities and compatibilities. 'Probabilities' measure the likelihood of two categories occurring in sequence: it is a purely statistical measure which results in not a single solution but a ranked list of solutions. For example, if a noun–adjective ambiguity follows a determiner then it is more likely to be a noun, but the adjective possibility is still accepted, with a lower score. 'Compatibility' is a similar measure which takes account of mutual probabilities, i.e. when there are two ambiguous words occurring together the probability of the solution for one depends on its compatibility with the solution for the other. Compatibility testing also looks for relationships between words over longer distances: for example, if there is a relative pronoun then there should be a finite verb somewhere to its right. The final step is to combine weightings for individual words in the calculation of a ranked order of probabilities for the word sequences.

It is easy to see how the fail-soft mechanism (section 11.4) can take advantage of this approach: if in later modules it is found that the preferred homograph disambiguation was wrong (because it does not permit the analysis to proceed) then it is a relatively simple matter to backtrack and try the next ranked solution.

The probability and compatibility scores were initially estimated by linguists and then adjusted over a long period in response to incorrect translations. The alternative of basing these measures on large-scale corpus analysis (cf. section 18.2) does not seem to have been used.

11.5.4 Phrasal analysis

The next module is called SEGMENT, and its purpose is to identify clause boundaries within the sentence, i.e. segment the sentence into main clause, subordinate clauses, parenthetical clauses, and so on. Clause boundaries are identified by looking for punctuation marks (for which conventions are stricter and hence more reliable in German), for subordinating conjunctions and for relative pronouns. A major difficulty is the recognition of the scope of conjunctions, e.g. whether *und* ('and') coordinates two nouns, or two noun phrases, or two verb phrases, or whatever. This module also includes a kind of validity checking, which looks for obligatory

elements within clauses. For example, the main clause in a sentence should have a finite verb, an infinitival clause should contain an infinitive verb form, and so on.

A different segmentation routine was written for English (PHRASEG) and French based more closely on the analysis of surface structure, particularly the positions of nouns and finite verbs. It was introduced because the punctuation conventions of English and French differ so markedly from those for German (see section 2.3).

The output of SEGMENT (or PHRASEG) is passed to two stages of phrasal analysis dealing with noun groups and verb groups. The two modules are NOMA (*Nom*inal*a*nalyse) and VERBA (*Verbal*analyse) respectively, though some descriptions of SUSY have different names for these modules.

NOMA operates on the segments identified by the previous module, and consists of twenty or more sub-processes dealing with the internal structure of noun phrases. These sub-processes are each devoted to specific analysis tasks such as apposition, numerals, adjectival groups, embedded structures and so on. Some are based on the identification of 'markers' (such as punctuation or specific words, e.g. *for instance*, as signs of apposition); others are based on semantic features to establish valency relations. The main sub-process, however, is the one which deals with the recognition of simple groups such as preposition + determiner + noun, determiner + adjectival group + noun, etc. The procedure is based on lists of permissible sequences (rather as in COMPOUND above), with plausibilities based on congruences of case, gender and number as appropriate. The 'longest match' principle is applied, i.e. the longer sequences of categories are tested first.

NOMA is followed by a similar process of verb group analysis, VERBA. Just as NOMA treats different types of noun structure, VERBA deals with compound verb groups, modal auxiliaries and so on. This is a major task for German which has relatively few inflected tenses (e.g. in comparison with French), but allows many combinations of auxiliaries and modal verbs, e.g. (6); as the translation indicates, there are similar problems in English.

(6) *Die Aufgabe hätte getan werden sollen.*
'The task ought to have been done'

11.5.5 Structural analysis

The output from NOMA and VERBA provides the building blocks for the next stage of the analysis, namely the structural or complement analysis carried out by the KOMA module ('*Kom*plement*a*nalyse'). The purpose of KOMA is to determine the valency structure for each clause. For example, KOMA determines which of the noun groups identified by NOMA should fill the argument (or 'complement') slots for the verb identified by VERBA in a verbal clause. In the case of complex noun groups, its task is to establish the internal valency and argument relations. Verb arguments include nominal groups, prepositional phrases, infinitival and subordinate clauses. Arguments of nouns are adjectival groups, plus prepositional phrases and certain types of subordinate clause, e.g. *daß*-clauses as in (7a) or infinitivals as in (7b).

(7a) *die Idee, daß er kommen soll* 'the idea that he should come'

(7b) *sein Versprechen, pünktlich zu kommen* 'his promise to come on time'

KOMA works by first attempting to fill up the valency slots appropriate for the main word in the clause, e.g. subject and object for a finite verb. This is achieved by checking for compatibility in terms of syntactic case and number, and, to a certain extent, by checking for simple semantic feature co-occurrence restrictions. Remaining elements are analysed as adverbials or appositionals, though again there is some checking for semantic compatibility (e.g. adverb of direction with a verb of movement).

The module also attempts to reconstruct elliptical structures, e.g. from (8a) to (8b).

(8a) *Der Bauer lacht und singt.*

(8b) *Der Bauer lacht und der Bauer singt.*
'The farmer laughs and [the farmer] sings'

Similarly, for English, KOMA tries to find the 'deep' subjects in sentences like (8c) and (8d).

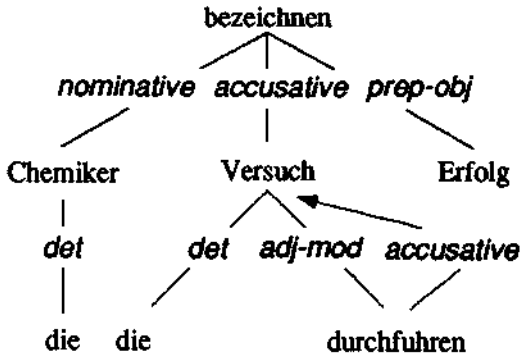
(8c) John persuaded him to go home.

(8d) John promised him to go home.

As a consequence of the analysis in terms of valencies, passive constructions are represented in active forms, as this example (9) of the output from KOMA illustrates.

(9a) *Die durchgeführten Versuche werden von dem Chemiker als Erfolg bezeichnet.*
'The experiments carried out are declared a success by the chemist'

(9b)



11.5.6 Semantic disambiguation

The final module in the analysis phase is called SEDAM (*Se*mantische *Dis*am*b*iguierung). Its procedures are based primarily on semantic features assigned to nouns and some pronouns in the 'semantic dictionaries' and on various types of semantic rules attached to entries. Semantic features are of two kinds: those taken from a restricted set of presumed universals (human, abstract, animate, etc.) and those from a language-specific set of features developed initially for noun groups (geographical location, vehicle, profession, animal, plant, etc.). The first

set is hierarchically structured, so that superordinate features are not entered in dictionaries but are created automatically as appropriate. The rule sets attached to entries are invoked for changes, deletions and insertions in structures, and are controlled by syntactic or semantic specifications of contexts.

The purpose of SEDAM is to allocate an interpretation to syntactic structures which are semantically ambiguous or vague. For example, a noun referring to a process may be modified in a number of ways which are not made explicit: a modifying noun might refer to the agent of the process (e.g. *human understanding*), or to the object (*satellite launching*), or to the method (*computer simulation*), or to the location (*inner-city deprivation*), and so on. Such ambiguities are resolved by the application of rules expressing semantic preferences.

Another type of semantic disambiguation which SEDAM attempts is the distinction of homographs of the same category, i.e. words which up to this point can be correctly assigned to a single syntactic category, but for which there are two (or more) distinct meanings and uses. For example, *trade* can refer to an occupation or profession, or to commerce in general. The first sense may be isolated by the presence of a possessive pronoun (*his trade*) or preceding preposition (*by trade*); the second may be excluded if the noun is plural. Most distinctions are, however, based on co-occurrence restrictions applied to semantic features, e.g. in recognising the various uses of *raise* ('cultivate', 'express', 'produce', 'bring up', etc.). Evidently, these distinctions are in part motivated by the need to provide the information necessary for lexical transfer into particular languages; it is not clear to what extent the SUSY researchers regard them as applicable more widely.

Included in SEDAM, also with a view to transfer problems, are routines for identifying 'semi-fixed' expressions, e.g. the phrase *take into consideration*, in structures where other components may interrupt (10).

(10) The Commission took the plan into consideration.

The basic task is to interpret the verb + prepositional phrase as a single unit with *Commission* and *plan* as its subject and object arguments respectively.

Prepositions are dealt with by a special routine which examines valency relations and the semantic features of arguments and which creates 'artificial words', in effect interlingual elements, which are left unchanged during the subsequent transfer processes.

11.6 Transfer and synthesis

TRANSFER is a sequence of processes, using the bilingual dictionary to replace source language lexical forms by their target language equivalents. Normally, the corresponding dependency structure is preserved, though exceptionally this can be altered by TRANSFER. This module also includes a sub-process which attempts to translate words which WOBUSU identified as unknown. The main part of TRANSFER is the simple translation of lexical units which is made on a word-by-word basis, without looking at the surrounding context. There are some special purpose sub-modules within TRANSFER that deal with particular cases such as the translation of negation, the treatment of surface case relations within noun groups, and so on. For certain language pairs, additional TRANSFER modules are needed, for example

to deal with the lack of determiners when translating from Russian as the source language.

The task of SYNTHESE is to construct target language sentences on the basis of the output from TRANSFER. It consists of three sub-processes dealing with semantic, syntactic and morphological generation.

SEMSYN (*Sem*antische *Syn*these) has the task of generating idioms or 'semi-fixed' constructions where appropriate, or of producing the correct prepositions for verbs and nouns (e.g. translating *to* with geographical names: *in die Schweiz, zu Berlin, nach Europa*), and of translating 'artificial words' into target lexical items. Artificial words are lexical units formed during analysis and include disambiguated homographs (e.g. *Schloß*$_1$ 'lock', *Schloß*$_2$ 'castle') and discontinuous multi-word lexical items (e.g. *Landwirtschaft betreiben* as the translation of *to farm*, compound prepositions such as *away from*, English phrasal verbs such as *take down*, and so on).

SYNSYN (*Syn*taktische *Syn*these) takes the output of SEMSYN and produces a sequence of lexical items (i.e. word stems) with the associated morphological information necessary for morphological synthesis. Part of the task of this module is to determine the correct surface word order, as well as the surface case endings of nouns and various agreement phenomena, depending on the target language concerned (e.g. dative plural ending in German, subject–verb agreements in many languages, and number–gender agreement in French noun groups).

There are a number of interesting sub-processes within SYNSYN. One called SGKOMP deals with compounds and derivations. For example, when translating the French phrase *système de traduction* ('translation system') into German it creates a compound noun *Übersetzungssystem* rather than the noun phrase *System von* [or *für*] *Übersetzung*. Another module SNOADJ determines which of various possible realizations for certain modifier–noun combinations should be chosen, e.g. whether *Israeli lemons* or *lemons from Israel* should be output.

One of the more complex sub-processes of SYNSYN is SPORE, which deals with the synthesis of pronouns and possessive adjectives. In German, for example, pronouns must agree in grammatical gender with their antecedent nouns. In (11), *it* refers to *file* (which is feminine in German *Datei*) and not *system* (which is neuter), and so should be translated as *sie* and not *es*.

(11) If the system deletes a file, the user can recover it by ...

There is a similar problem with German possessive adjectives, which must likewise show grammatical gender concord with antecedents, often conflicting with natural gender (12).

(12) *Die Gemeinschaft und ihre Mitglieder*
'The community and its members'
Lit.: The community and her members

Some attempt is also made to deal with anaphora across sentence boundaries using a system for weighting potential antecedents according to their positions and roles in preceding sentences.

The process of translation is completed with the MORSYN (*Mor*phologische *Syn*these) module, the aim of which is to convert the stem + morphological

feature units into target text strings. This is sometimes relatively straightforward, though with some languages (especially German) morphological forms are highly dependent on surrounding context; for example, German adjective endings depend not only on the case and gender of the noun which the adjective modifies, but also on what sort of determiner the noun group has: *ein guter Mann*, *der gute Mann*. Similarly, both English and French have cases of morphological alternation which is contextual rather than grammatical (French *du* replacing *de le*, elisions before vowels *l'*, *d'*, choice of English *a* or *an*). Finally, MORSYN also deals with capitalization and punctuation, both particularly important for German output.

11.7 Conclusion

In most essentials, SUSY is an example of a transfer-based system, but in some respects it is a hybrid. From the computational point of view it exhibits a basically first generation type of architecture similar to that found in Systran (Chapter 10). Linguistic routines are written not in some higher-level rule-writing formalism but directly in the low-level programming language Fortran. Weak points of SUSY are consequently the programming environment and the attendant primitive data structure and simplistic mode of processing, in particular the single non-branching sequence of modules.

Nevertheless, the linguists working on SUSY have shown that it is still possible to maintain a consistent linguistic approach, though of course the details of their implementations are only accessible to readers familiar with low-level programming practices. It must also be said that the abundant use of acronyms — only partially revealed in this discussion — can be distracting or even confusing if in English-language descriptions they differ from those in German accounts.

One characteristic of SUSY's hybrid 'direct' ancestry is the emphasis in the first stages of analysis on non-linguistic methods. It is not until the SEGMENT procedures that there is anything which would be called 'parsing' in the traditional (computational linguistics) sense. Many problems are resolved on the basis of *ad hoc* rules which search for particular categories or even particular words in as yet unstructured strings, or which weigh the probabilities and compatibilities of sequences of categories irrespective of structural relations.

It may be noted also that analysis procedures are not entirely monolingual. Target language oriented analysis is found particularly in SEMSYN. It is doubtful, for example, that a strictly monolingual analysis of English would identify so many different meanings of *raise* if it were not to be translated into German where choices have to be made between *vorbringen*, *züchten*, *anbauen*, *aufwerfen*, *erzeugen*, *aufbringen*, *anheben*, etc. In a 'true' transfer-based system these selection problems would be handled by bilingual lexical transfer routines.

In general, the translation quality of SUSY is comparable to that of other contemporary systems, and indeed SUSY has been used in several practical experiments. In a collaborative project SUSY was linked with Kyoto University's TITRAN system to translate titles of documents between German and Japanese, using English as a 'switching language'. A less successful experiment was the use of SUSY by the translation service of the German Bundessprachenamt ('Federal

Language Office'), which faltered due to an unwillingness to accept lower quality translations. By contrast, a successful automatic indexing experiment made use of the German analysis modules to derive standardised (root) forms of words in abstracts and full texts together with some indicators of structural relationships.

Despite these partial successes, the developers of SUSY recognised that the original system had its limitations, and started working on an essentially new project SUSY-II. The intention was to develop a linguistic rule-writing formalism, a simplification of the analysis stages (reduction to three basic processes), a more satisfactory treatment of structural ambiguity, and the introduction of preferential rule application. Some of these new approaches were explored by other researchers at Saarbrücken engaged on a parallel MT project for French–German translation (ASCOF). However, these plans for SUSY-II were in turn overtaken by events, and in 1986 work on MT in SFB-100 was terminated, with efforts at Saarbrücken concentrating on the German contribution to Eurotra.

11.8 Sources and further reading

The information for this chapter is derived from numerous sources in English and German. In particular, there is a large set of reports and working papers published by SFB-100 itself, under the general title *Linguistische Arbeiten*, in two series (the second starting in 1982). For those able to read German, and interested in very precise details of the implementation of SUSY, some of these may be worth seeking out.

The most thorough English-language description of SUSY is Maas (1987), though also worth consultation are Maas (1977), Luckhardt (1982), and Hutchins (1986:233–9), the latter an outsider's viewpoint. Useful references in German include Maas (1978) and Blatt *et al.* (1985), and from the *Linguistische Arbeiten* reports, Luckhardt (1976) and Luckhardt and Maas (1983).

The German–Japanese collaboration is described by von Ammon and Wessoly (1984/5); the Bundessprachenamt experiment is described at its outset in Wilms (1981); the automatic indexing project is described in Kroupa and Zimmermann (1987); a comparison of SUSY and SUSY-II can be found in Maas (1981) and in Luckhardt (1985), while Maas (1984) describes SUSY-II in detail. The ASCOF project is described in Biewer *et al.* (1985).