Software and lingware engineering in recent (1980—90) classical MT : Ariane-G5 and BV/aero/F-E

Christian BOITET
GETA, Institut IMAG
(UJF & CNRS)
BP 53X, 38041 Grenoble Cedex, France

Introduction

Research in MT have been pursued at Grenoble since 1961. At the end of the first period, around 1967-70, a first Russian-French system had been developed, and tested on more than 400,000 running words of real texts (scientific articles). The aim was *MT for the watcher,* or "pure" MT, where a fully automatic process produces unrevised "rough" translations good enough that the reader, supposed to be a specialist of the domain, can access the content of the original without knowing the source language. Such translations are often judged to be very bad by professional translators and very good by users.

Due to a change of computer system (IBM 7044 à IBM 360/67), which would have necessitated an important conversion hardly justifiable without a perspective of immediate operational use, this system was abandoned, despite its remarkable quality and coverage. That was the occasion to start exploring new ideas in the context of *MT for the revisor,* where the aim is to automatically produce "raw" translations destined to be revised by a professional in order to get final results of professional quality. As the step of human revision must naturally be performed on a computer, *MT for the translator* (machine aids for human translation and revision) was also studied.

In 1978, a new methodology for linguistic programming had been formulated by B. Vauquois (multilevel transfer approach, heuristic programming), and began to be experimented on various language pairs, with a large effort on Russian-French, using the elements of a computer environment for the generation of multilingual MT systems, which was called "Ariane-78" at the beginning of 1978, when its first complete version became available. This name was chosen in reference to the goddess and her famous thread, to underline the idea that computer science, even if essential, must be put to the service of linguists and lexicographers who are not computer scientists and enable them to work autonomously, thanks to specialized languages (symbolic, rule-based languages) and to a transparent interactive user interface.

Thanks to support from CNRS and DRET, a first "preoperational" Russian-French system was developed and experimented in real size, from the beginning of 1980 to the end of 1986. In parallel, an effort in technological transfer towards industry was undertaken, in particular in the framework of the Machine-Aided-Translation National Project (CAT-NP, or PN-TAO in French, 1983-87). GETA augmented considerably the power and reliability of Ariane-78.4, and was at the heart of the linguistic specification of the French-English system for aircraft manuals built in cooperation with industrial partners.

Since then, the Ariane system has been considerably extended. Its current version, Ariane-G5, is briefly described in the first section. Various complementary software tools have been constructed, by GETA or its partners. They are reviewed in the second section. Finally, we show that linguistic programming techniques in MT are changing from the stage of workmanship to that of real engineering. In order to give an idea of what modern MT for the revisor can produce and a concrete overview of its internal functioning, the third section ends with a commentary of several raw (unrevised) translations produced by B'VITAL's BV/aero/F-E system at the end of 1988 (current results are even better, but could not be made available in time).

I. Ariane-G5

A. General principles

Ariane-G5 is a generator (G) of MT systems based on five (5) specialized languages for linguistic programming. Each such language is compiled. The internal structures produced by its compiler are used as parameters by its "engine".

Although Ariane-G5 is particularly well adapted to the transfer approach, it does not impose it. Apart of the implementation limits, the only strong constraint is that the structures representing the units of translation be decorated trees.

Intrinsic semantics (a term borrowed from J. P. Desclés) is represented in these languages, hence in a linguistic fashion. If one wants to write a system specialized to a restricted sub-language and to a microdomain, it is possible to use the same technique as for METEO [Chandioux & Guérard 81] and to write "semantic" grammar and dictionaries. In order to construct a system equipped with *extrinsic semantics* ("ontology" of the universe of reference), it would be necessary to couple Ariane-G5 with an "expert corrector system", as suggested and prototyped in [Gerber 84].

As opposed to almost all existing systems, Ariane-G5 presents the advantage that the unit of translation is not restricted to the sentence, but may contain several paragraphs (in general, 100 to 200 occurrences, that is almost a standard page).

1. Harware and software environment

Ariane-G5 runs under VMSP/CMS, on mainframes (3090, 303X), minis (43XX, 937X), and on big micros (PC-AT/370, PS2-80/7437).

VMSP is a hypervisor which simulates a set of "virtual machines". Each virtual machine runs under its own operating system. For example, it is possible to let virtual machines run at the same time under MVS/TSO, CMS, and AIX. The RSCS subsystem enables to organize the virtual machines and the real resources (peripherals) as a network. CMS is a powerful interactive single user operating system, which supports a large number of programming languages and tools, but does not offer a hierarchical file organization *à la* Unix or MS-DOS (release 6 is rumoured to have it). It is generally used for software development or for interactive applications.

The current version of Ariane-G5 represents about 400,000 source lines, plus 30,000 lines of messages for each dialogue language[1]. The executable part resides on a "minidisk" (virtual disk) of a particular virtual machine, and takes about 10 Mb. To enable another virtual machine to be a "user machine" (of Ariane), one simply performs a logical connection (as "B/A") of Ariane's minidisk to the machine's minidisk ("A"). From then on, Ariane manages on the user minidisk two specialized data bases, one containing the lingware and the other the texts (with a maximum of 3,000 source and target languages, 1,000 "corpora" and 10,000 texts per corpus).

The minimal computer background necessary to use Ariane-G5 consists in learning the elementary commands for beginning and ending a VMSP session (login, logout), the XEDIT screen editor, and, for the developers of MT systems, the organization of the interactive monitor and the specialized languages.

---

[1] Ariane-78.4 had two parallel versions, in French and in English. Ariane-G5 is programmed in a totally multilingual fashion, but only the French version of the messages is complete at the time of writing.

2. Logical organisation

2.1. Steps, phases and articulations of the translation process

Translation from a "source" language into a "target" language is performed in three successive "steps" : analysis, transfer and generation[1]. Each step is realized in at least two and at most four successive "phases", possibly linked together by "articulations", which may be considered in first approximation as simple "coordinate changes". Each phase is identified by a two-letter mnemonic (e.g. AM for morphological analysis — analyse morphologique in French), and each articulation by a four-letter mnemonic (e.g. AMAS for the AM-AS articulation).

In analysis, the successive phases are :

AM	(morphological analysis)	obligatory,	written in ATEF ;
AX	(expansive analysis X)	optional,	written in EXPANS ;
AY	(expansive analysis Y)	optional,	written in EXPANS ;
AS	(structural analysis)	obligatory,	written in ROBRA.

In transfer, the successive phases are :

TL	(lexical transfer)	obligatory,	written in EXPANS ;
TX	(expansive transfer X)	optional,	written in EXPANS ;
TS	(structural transfer)	obligatory,	written in ROBRA ;
TY	(expansive transfer Y)	optional,	written in EXPANS.

In generation, the successive phases are :

GX	(expansive generation X)	optional,	written in EXPANS ;
GS	(syntactic generation)	obligatory,	written in ROBRA ;
GY	(expansive generation Y)	optional,	written in EXPANS ;
GM	(morphological generation)	obligatory,	written in SYGMOR.

In the current version, the order of these phases within each step is fixed. Hence, the possible "articulations", all written in TRACOMPL, are AMAX, AMAY, AMAS, AXAY, AXAS, AYAS, ASTL, then TLTX, TLTS, TXTS, TSTY, TSGX, TSGS, TYGX, TYGS, and finally GXGS, GSGY, GSGM and GYGM. As a matter of fact, one needs to write articulations only for composing two phases taken from lingware components using heterogeneous "sets of variables" (see below).

The linguistic operations performed in each phase don't necessarily correspond to their names in a strict manner. For example, morphological analysis may be realized in AM, but it is also possible to distribute it between AM, AX, AY and a fraction of AS (for example, to test for the occurrence of "predicted" possible non connex idioms). In general, lexical transfer is also distributed between (at least) TL and TS, for analogous reasons. Similarly, morphological generation of a language such as Arabic [Moneimne 89] may advantageously be distributed between the end of GS and GM.

[1]This term is used rather than "synthesis", by analogy with that of "generation" in compiler construction.

2.2. Essential data types and basic terminology

At the input and output sides of the translation process, the unit of translation is a simple *string of characters*. The 256 EBCDIC characters may be used in the specialized languages to build strings, and all are considered to be atomic (e.g., "é", "É" and "ê" are not known to be variants of "e"). The blank (X'40') is used as separator of *occurrences*. A translation unit, then, is also a *sequence of occurrences*.

From the output of AM to the input of GM, a unit of translation is represented by a decorated tree. Each phase contains a part where the linguist declares the decoration type, or *set of variables* in Ariane-G5 terminology.

an *elementary variable* ("variable" in short) is defined by a name and an (anonymous) type. This type is defined by a *kind* (exclusive, non-exclusive, arithmetical, lexical) and a *list of values*. Each variable has a null value, denoted by its name followed by "0".

- An exclusive variable V defined by V == (V1, V2, V3) takes its values in {V0, V1, V2, V3}. There is an analogy with the scalar types of Pascal, but, here, value identifiers are local to the types.
- A non-exclusive variable V defined by V == (V1, V2, V3) takes its values in the power set of {V1, V2, V3}. V0 denotes the empty set. There is an analogy with the set types of Pascal, the preceding remark being still valid.
- An arithmetical variable V defined by V == (n), where n is a non null natural integer, takes its values in :

$$\left[-2^{\lceil \log_2(n) \rceil}, 2^{\lceil \log_2(n) \rceil}-1\right]$$

- There is always one (and only one) "lexical" variable, named "UL", for *lexical unit* (unité lexicale), which is predefined and takes its values in the set consisting of :
  - the predefined values " (UL0), 'ULTXT', 'ULFRA', 'ULSOL', 'ULOCC', 'ULMCP' ;
  - the values introduced in the lingware components (essentially in the dictionaries) ;
  - the values constructed dynamically (for example to handle unknown words).

A *decoration,* or *mask of variable* in Ariane-G5 jargon, is a combination of values for all the variables of the considered set, very much analogous to a property list in LISP.

It is possible to group variables in a hierarchical fashion, the top of the hierarchy being predeclared until a level depending of the specialized language. VAR always denotes the set of variables minus the UL variable.

In ATEF, two subsets, VARM and VARS, are distinguished and declared separately, for the said "morphological" and "syntactic" variables (although, as for the phases, the linguists don't in general respect that division and add a number of variables of semantic nature). VARM and VARS are further subdivided into VAREM and VARNM, VARES and VARNS (exclusive and non-exclusive), as there are no arithmetical variables in ATEF.

In ROBRA and EXPANS, where the three kinds of variables are possible, the top of the hierarchy is VAR (VAREπ, VARNπ, VARAπ), where π is a character (redefinable in DV) characteristic of the phase. By default, π is set to S, R, C, D, G for AS, TL, TS, GS, GM, and to X for the other EXPANS phases. For example, in AS, in order to refine VARE into syntactic and semantic variables, themselves divided into properties and relations, one might write :

```
-EXC-        ** (key-word for "exclusive").

   VSYNTE ==    (PSYNTE (CAT (N, V, A, R, S...), K (PHVB, PHINF, GV, GN, GA))
                ,RSYNTE (FS (SUJ, OBJ1, OBJ2, EPIT, CIRC...)).

   VSEME ==     (PSEME (PREDIC (ETAT, ACTION, PROC), MATIERE (DISC, CONT))
                ,RSEME (RL (ARG0, ARG1, ARG2, ARG01, ARG02, TRL10...))).

-NEX-        ** non-exclusive.
   ...
```

A *format* is a constant mask of variable to which a name has been given, in order to use it as abbreviation in dictionaries and grammars. A *decorated tree* is an oriented and ordered tree where each node bears a decoration.

2.3. Components and variants of a phase

As in most NLP systems, the specialized languages are organized in physically distinct *components,* for reasons of modularity and size. The components of a phase form an acyclic dependency graph (known by the compiler).

- An ATEF phase contains two components of variables declaration (DVM, DVS), "morphological", "syntactic" and "general" formats (FTM, FTS, FTSG, the last one being optional), 1 to 7 grammars GRi ($1 \leq i \leq 7$), 1 to 6 dictionaries of "morphs" DICi ($1 \leq i \leq 6$), at least one of them being of "bases" (morphs with lexical references), and from 0 to 7 dictionaries of fixed connex idioms, DICi ($7 \leq i \leq 14$). FTM depends on DVM, FTS on DVM and DVS, FTSG on FTS, the dictionaries on the formats, and the grammars on the dictionaries.
- An EXPANS phase contains a component of variables declaration (DV), one of "condition and assignment procedures" on decorations (PROC), one of "assignment formats" (FAF), optionnally one of "proper condition formats" (FCP), and from 1 to 7 dictionaries (DICi). PROC, FAF and FCP depend on DV, and the DICi on the preceding.
- A ROBRA phase contains DV, FAF, and from 1 to 7 grammars (GR1 to GR7), with the natural dependencies.
- A SYGMOR phase contains DV, FAF, PCP ("proper condition procedures"), GRi ($1 \leq i \leq 7$), and DICi ($1 \leq i \leq 13$, at least one dictionary being accessed by the UL), with the same dependencies as in ATEF, with the exception that grammars don't depend on dictionaries.
- A TRACOMPL articulation contains only one component, DV.

The Ariane-G5 environment ensures at all times the coherency of the internal tables as a function of these dependencies and of the modifications made by the user (linguist).

Each phase may give rise to *variants,* which may be defined according to the types of texts to be translated.

- In ATEF and SYGMOR, one chooses one of the grammars.
- In EXPANS, one chooses an ordered subset of the dictionaires, and the mode (deterministic or not) of the engine.
- In ROBRA, one defines a list of a most 14 grammars to be executed sequentially, the same one being allowed to occur more than once.

By combining these choices and the choice of a path in the graph of phases (from AM to GM for a translation), one obtains *execution lines* (for debugging) and *production lines* (for cranking out translations) which are also memorized and managed by Ariane-G5.

3. Principles of linguistic use

Although Ariane-G5 does not propose or impose any methodology of linguistic programming, most of its users follow a certain number of principles, mainly due to B. Vauquois, which it may be useful to mention briefly at this point.

3.1. Intermediate structures

It is recommended that analysis delivers an *m-structure,* or "multilevel" structure. The geometry of the tree reflects the organization in syntagmatic groups, but the structure is an "abstract tree", from which the text may not necessarily be reconstructed in an immediate way. For example, it is convenient to regroup discontinuous constituent (e.g. "les garçons les ont tous vues" for "the kids$_i$ have all$_i$ seen them$_j$"), to "variabilize" negations, auxiliaries, articles, strongly governed prepositions, certain modals, etc., thus obtaining trees considerably smaller than the "concrete" ("surface") trees provided by direct application of extended context-free grammars (GPSG or others). In principle, each internal node dominates a leave which is the *governor* of the group (from "gouverneur" in French, usually "head" in English), unless the governor is itself a compound. In order to get a dependency structure analogous to those of the Prague school, it is enough, as a first approximation, to recursively "send up" each governor to replace its mother node.

Properties and relations are coded in the decorations attached to the nodes. For example, a node having "attribute of object" as value of the syntactic function (SF=ATROBJ) is the attribute of the group dominated by its (unique) sister node having SF=OBJ1. Hence, there are two syntactic levels, that of classes (morphosyntactic and syntagmatic, $\underline{X}$ et $\underline{\underline{X}}$ in Chomskyan notation), and that of functions. To translate into languages which are not extremely near to the source language without having to write large structural transfers, it is advisable to add two more levels, logical and semantic.

The logical level (RL variable, for "relation logique") gives the positions of arguments of linguistic predicates. ARG0 denotes the logical subject (most often the actor) of the predicate "governor" of the same group, ARG1 denotes its logical object (in general the patient, but not in ergative constructs such as "the twig breaks"), and ARG2 denotes its third argument. The numbering is such that ARG1 corresponds to OBJ1 in standard active constructs, but that is purely a convention. For example, "the building of the house" and "to build the house" have identical structures at that level, the group "(of) the house" being ARG1.

TRL10 is used in place of ARG1 if the predicate attributes ARG1 to ARG0 ("to be", "to seem", "to appear"...), and TRL21 in place of ARG2 if the predicate attributes ARG2 to ARG1 ("to consider ARG1 as TRL21"). That is a way to indicate that the relation does not link the node with the governor (the predicate), but with another argument. Similarly, one often uses another variable (such as RLI, for "inverse logical relation") to code the link between arguments in control constructs. For example, in "I ask him to come", the group "to come" is ARG1 of "ask", and bears RLI=00 if ARG0 (I) is coming and RLI=02 if ARG2 (he) is coming.

Finally, we use, mainly on circumstancials (weakly governed complements), the semantic relation (RS), which *grosso modo* corresponds to the "deep case" (localization, origin, goal, accompaniment, manner, qualification, measure, cause, concession...). In practice, RL and RS are complementary, because it is extremely difficult (even manually) to assign RS to arguments in a reliable manner, and circumstancials can be correctly translated only if their RS are known.

In this respect, the famous problem of the translation of prepositions is often no well stated. If an argument is concerned, the whole construct (predicate+arguments, e.g. "to talk about sth. with sb.", "to count for sb., sth.") should be translated as a block. If a circumstantial is concerned, the RS, possibly particularized by the preposition (or its absence) should be translated. For instance, in "to come by Lyon" and "to come via Lyon", the circumstantial should bear RS=LOC, SEM=SPACE and SLOC=QUA (localization in space, movement through sth.), thus allowing for exact translation of "by" ("par" and not "près de", "à côté de", "devant", "de", "d'après", "suivant", "à"...). Keeping the preposition also allows to translate more exactly into a language like French, which also has two prepositions for this sense ("par", "via").

Several levels are similarly used for the actualization variables, such as number (morphological and logical), time vs. tense, etc. The order of the text is reflected as much as possible in the structure. As a matter of fact, order gives important informations which are not well formalized, such as thematic articulation and emphasis. That dispenses from coding it explicitly in a tactical variable.

If one considers the m-structure of a sentence only at the "deep" levels, it can be thought to represent a whole family of sentences of equivalent meanings. If it is considered at all levels, it should correspond to only one sentence, notwithstanding spelling variants (such as disc/disk, program/programme, or corpuses/corpora).

Besides these various levels of linguistic description, one also encodes in the m-structures produced by analysis unresolved ambiguities and doubts on parts of the construction, in order to avoid a combinatory explosion, and to be able to warn the revisor and at the same time to try to transfer those ambiguities which persist in translation (e.g. "the conquest of the Romans").

The aim of transfer is to perform lexical translation, and some adaptations of the structure aiming at delivering to the generator a structure coherent with the linguistic system of the target language. That structure is called *g-structure,* for "generating structure". In principle, the generator considers that the g-structure it receives is under-specified with respect to the surface levels, and recomputes them.

Hence, the first logical step of the generation consists in selecting a paraphrase of the meaning expressed by the g-structure by producing the m-structure of the translation to be produced. The second step consists in producing a surface tree ("concrete" tree, or *s-structure),* by creating nodes for articles, auxiliaries, negation elements, punctuations, by dividing or merging sentences if necessary, etc. The third step is the morphological generation which, starting from the sequence of the leaves, constructs the occurrences of the final text.

3.2. Organization of dictionaries

The notion of lexical unit is very useful for the generation step. It allows to represent derivational families in a compact way. Modern dictionaries use a similar notion. In analysis, that notion allows to reduce the size of the dictionaries, and to handle in a systematic way neologisms obtained by productive derivations.

From the linguistic point of view, one is naturally led to regroup, for example, *to heat, heater, heating, heatable,...* in the UL *heat-V,* the derivations used having the three aspects of semantics, syntax and semantics (e.g. agent or instrument noun ending in -er), in order of importance. From the practical point of view, one often separates the agent or instument noun from such a family, because the derivation in question cannot be used to produce paraphrases correct in translation ("heater" —> "something which heats" ?), and because that allows to separate the purely terminological indexing of those terms from the more complex indexing of whole families of deverbals.

According to the syntactic class of the principal lemma (source of the derivations), one distinguishes verbal, nominal and adjectival ULs. Of course, there are ULs reduced to only one term (function words, non-derived adverbs such as *there, here...).*

In Ariane-78, it was necessary to represent all lexical information in the AM dictionaries. In Araine-G5, lexical expansion phases have been introduced to give the possibility of distributing the information. A possible organization is to use AM to go from morphs (roots, affixes, endings...) to lemmas, AX to go to the ULs, and AY to handle non fixed or non connex idioms (e.g. verbs with separable particles in German).

3.3. Organisation of grammars

In ATEF and SYGMOR, organizing the grammars is quite simple, although one should resist the temptation to overuse ATEF heuristic functions.

In ROBRA, the programming technique is quite different according to whether analysi ., transfer or generation is concerned. Analysis usually begins by working in parallel on the whole unit of translation to normalize the tree (compound words, resolution of immediately solvable ambiguities, grouping of non connex idioms, dates, proper nouns, etc.). Then, a sentence-specific transformational sub-system is recursively called on each sentence. Other sub-systems may be called on groups (clauses, phrases). That enables the strategy (traversal of the control graph) to be directed by the data. The end of analysis usually consists in processing again the whole tree. For example, that allows to try to find referents of pronouns for which none has been found inside the sentences where they occur.

In transfer, things are quite simple. One handles the translations of complex groups, tests the context conditions coded in the subtrees representing multiple translations, in order to reduce polysemies, and adjusts the g-structure, thereby possibly annotating it with advices or orders to the generator, in order to trigger the production of particular syntactic forms (e.g. to get the reflexive or the passive for stylistic reasons).

Generation uses recursive descent to produce the m-structure, and then to begin construction of the s-structure. Parallel processing is usually used to assign final values to surface actualization variables (to propagate agreement constraints), which may lead to slight modifications of the geometry (insertion of auxiliaries and clitics). On should be cautious not to use grammars at the same time iteratively and recursively, because that leads to numerous and costly useless grammar calls.

B. The interactive interface

Under Ariane-G5, it is possible to :

- work on the linguistic components (phases, articulations) in the subenvironments PRAM,... PRGM, PRAMAX,... PRGYGM (creation, modification, compilation, listing...).
- work on the texts (PRTXT), with numerous facilities.
- work on the "execution lines" and on the "production lines".
- execute all or part of the translation process for debugging purposes (each phase receives parameters for tracing and for outputting the result), using one available "execution line".
- crank out translations (no trace, only final output parameters), using an available "production line".
- revise raw translations (in multiwindow mode under XEDIT, with the possibility to switch to THAM, which offers additional facilities). It is also planned to allow linguists to revise the trees produced by each phase, under XEDIT, with the possibility to switch to TTEDIT, the analog of THAM for trees.
- perform actions on several phases at the same time (compilation, listing, erasing, duplication...).
- obtain various informations on the objects managed by Ariane (list of source and target languages, of corpuses, links between source and target languages, compilation status...).
- read or modify the global parameters (dialogue mode, current source and target languages, current corpus...).

An on-line help is available, for the most part at two levels of detail. That interactive monitor is described in [Quézel-Ambrunaz 90a]. It represents about 50,000 lines of EXEC2 and 100,000 of ASM370.

C. ATEF, a langage for morphological analysis

ATEF was designed in 1971 by J. Chauché [Chauché 75], who wrote the engine, while P. Guillaume and M. Quézel-Ambrunaz wrote the compilers of the different components. Since then, ATEF has undergone numerous extensions, but the underlying algorithmic model has not varied. As a matter of fact, it is a very satisfactory tool.

The system successively handles each occurrence of the text, examining *a priori* all possible analyses (non-deterministic total mode with backtrack). The current occurrence is named C. An analysis result is a decoration or a sequence of decorations (in the case of compound words). Each step of a particular analysis consists in choosing one of the open dictionaries, in finding there an item which key, the *morph,* is a prefix (or a suffix, in right-to-left mode) of what remains to be analyzed (noted A), which is reduced accordingly, and in applying one of the rules associated with the morphological format of the considered item.

The rules may contain conditions bearing on the current state (decoration C), on the strings C and A, on the partial results produced by the current analysis (PS1 to PS9) in case of o compound word, and also on the four preceding occurrences (from P1 to P4) and on the results of their analysis. A particular form of condition consists in giving a list of "sub-rules" and in asking that at least one of them applies (as a sub-rule may itself have sub-rules, that happens in non-deterministic unary mode with backtrack). It is finally possible to store a condition on the analysis of the following occurrence.

There exist three types of action : assignment of values to mask C, transformation of what remains to be segmented (string A), and call of special functions. These functions allow to :

- control the built-in backtrack by pruning the choice tree (FINAL, ARRET, ARD, ARF, STOP) or by opening and closing dictionaries (through assignment of the obligatory non-exclusive morphological variable DICT) ;
- produce a partial result from the current state C (SOL) ;
- transform C or A into a UL, thereby reducing A to the empty string " (TRANS, TRANSA) ;
- decide that a sentence boundary has been reached (INIT).

If an occurrence is not recognized ("unknown word"), that is, if no analysis succeeds in reducing A to " while producing a current state C having a non-null UL, the system starts analysis again, after having attached to the occurrence the obligatory morphological format MODINC, which must in particular call the obligatory rule MOTINC ("rule of the unknown word"). As that rule may call sub-rules, and as that format may call other rules, it is possible to construct a true "grammar of the unknown word", and to program sophisticated strategies for analyzing unknown words.

During processing, the automaton (ATEF "engine") constructs a ("4-colour backward") graph where the nodes are the masks (or lists of masks for the compound words) associated with the solutions found, and the edges indicate compatibility between analyses (at distance 1 to 4). The final graph is finally transformed into the desired form. The Q-graph output is now no more available, and two other output forms, a 1-colour forward graph and a tree "with homosentences" (presenting all paths in the 4-colour graph separately) are no more used.

The standard output of ATEF is a tree "without homosentences", which encodes ambiguities. Its root corresponds to the whole text and bears UL='ULTXT'. Its daughters correspond to the sentences (determined by the grammar) and bear UL='ULFRA'. Under each of them are nodes with UL='ULOCC', which correspond to the occurrences (words or fixed connex idioms). Under each 'ULOCC' are the different results of the morphological analysis of the corresponding occurrence. Each result is either a mask of variables (a node) or a sub-tree with root having UL='ULMCP' (compound word) dominating the masks corresponding to the different parts recognized in the word.

D. ROBRA, a langage for transforming decorated trees

ROBRA [Boitet, Guillaume & Quézel-Ambrunaz 78] is a language for writing transformational systems working on decorated trees. It is the successor of the CETA language [Chauché 75]. Numerous extensions have been introduced, the semantics has been made more precise, and the engine has been totally respecified and rewritten.

A transformational system (ST) is defined by a control graph (GC), a set of transformational grammars (GT) and a set of rules (RP for "production rules"). A GT is an ordered set of rules. A GC is a graph where each node bears a GT or the exit symbol (&NUL) and the edges bear tree conditions. Note that each "grammar" component GRi of a ROBRA phase actually countains a whole transformational system, possibly consisting of a large GC with dozens of GTs.

To execute a ST on an object tree (AO), ROBRA uses the GC as non-deterministic (unary with backtrack) control structure : starting from an initial node, it looks for the first valid path leading to an exit. On this path, it executes the grammars countained in the nodes, and traverses an edge only if the current AO verifies its condition.

The execution of a GT consists in one elementary application in unitary (U) mode, or of several in iterated mode (E for "exhaustive"). In an elementary application, as many rules as possible are applied in parallel (according to the modes of the GT), which necessitates a mechanism for conflict resolution. An elementary application ends only after the recursive calls of sub-grammars (SGT) or sub-systems (SST) possibly triggered by the application of certain rules have been completed.

A system of interdictions (rules are marked, nodes are blocked) allows to statically test the ST for decidability : the compiler may warn the user of the risks of undecidability (loops in the GC, "free" mode in an iterative GT, constraint on recursive calls not satisfied, etc.).

The schemas which appear in left-hand sides of rules have a very great expressive power. For each node, it is possible to indicate whether its daughters are to be looked for in leftmost or rightmost positions, in order or disorder (free permutations). It is possible to look for nodes at unspecified dephths by using "generalized nodes".

Finally, the rules may be context-sensitive, the root of the schema (RS) being possibly different of the root of the effective transformation (RT). What is not dominated by the RT belongs to the context, or "hat". The RT may itself be active or contextual. What it dominates belongs to the active part.

The notion of parallel rewriting in ROBRA is quite strong, as parallelism may be "normal" (RT located on distinct nodes of a cut of the AO), "vertical" (a RT may dominate another), and "horizontal" (several contextual RT and at most one active RT may be instantiated on the same node of the AO).

Finally, it is possible to write extremely complex conditional assignments of variables in the right-hand side of rules, which contributes to make ROBRA an extremely powerful tool.

ROBRA is really a production system of the substitution type, even if, in the current implementation, elementary application of a transformational grammar is done by transduction of an input tree into an output tree (both represented linearly). For that reason, the decoration type is necessarily preserved by a transformation system.

E. EXPANS, a langage for lexical expansion and transfer

EXPANS is based on a model of transduction of decorated trees [Guillaume 89b]. The decoration types of the input and output trees may be different. Each node is transformed into a sub-tree in the output tree. That sub-tree is determined by consultation of the dictionaries, in their order of priority, through the UL beared by the node. A default action is always foreseen. A dictionary item has an UL value as key, and a list of triples <condition/image/assignments> as content. The conditions concern the node of the input tree and possibly its immediate neighbours (mother, left and right sisters). The image describes the geometry of the sub-tree to be produced, and the assignments allow to compute the values of the variables on the nodes of the sub-tree from those of the accessible input nodes.

At the level of a dictionary, if the UL of the node is found, EXPANS looks for the first triple which condition is verified (the last condition must be empty, that is, identically true), and the corresponding sub-tree is produced. Otherwise, this dictionary fails. In deterministic mode, dictionaries are searched in their order of priority until a success is obtained. In non-deterministic mode, all dictionaries are searched, in order, and the image produced is a sub-tree constructed by rooting the sub-trees produced by the dictonaries under a new node. That mode allows for example not to "hide" the usual translation of a word which has also a different translation in a particular domain which dictionary has been given higher priority.

F. SYGMOR, a langage for morphological generation

SYGMOR is based on a model of finite-state deterministic transducer, which first version was designed by B. Thouin and programmed by D. Jaeger. [Guillaume 89a] describes extensions and amendments he contributed to it in recent years. SYGMOR takes as input a sequence of decorations and produces as output a string of characters. A context reduced to the current (C) and preceding (P) decorations is available, and two strings are used, the "working string" (T, for "chaîne de travail"), and the output buffer (S, for "chaîne en sortie").

The grammar has a quite simple structure. Each rule is made of three parts : condition, actions and "subsequent rules". Actions consist essentially in writing to the right, to the left or in the middle (last point of concatenation) of T a literal string or the result of accessing one of the dictionaries through the value of one of the variables of C (bases, affixes). Again, dictionaries items are triples (<condition/format/string>), the conditions being evaluated on C. It is also possible to modify C, and to "recall" S to T (by concatenating it to the left and emptying it).

For each decoration, SYGMOR looks for the first applicable rule and applies it. It then applies the subsequent rules in order, without taking their subsequent rules into account. A subsequent rule may be obligatory or optional. If an obligatory rule fails, SYGMOR goes back to the initial state and applies the rule MOTINC, if present, and otherwise the default error action (empty S, then do S:=T). Processing continues by taking into account the subsequent rules of the last rule applied, until an empty list of subsequent rules is reached.

G. TRACOMPL, a langage for transforming decorations ("articulations")

TRACOMPL [Guillaume 89b] is the sub-language used to write all DV components. It has been made into an autonomous language in Ariane-G5 to allow for writing "articulations". The goal is to transform decorations of a Set1 into a Set2. For that, we proceed in two steps :

- First, one describes Set2 and what should be known of Set1 in order to perform the transformation. The names of the variables present in both sets are prefixed by "$" and those of variables present in Set1 but not transmitted to Set2 by "$$". The others (not prefixed) are considered to be new.
- On completes that by writing (CVAR part) a conditional action which can test the variables of the input decoration in Set1, in Set2 after the "reformatting" described by the preceding part, and in Set2 in their current state (during execution of the action). Thanks to that, it is for example possible to transform a variable with 2 values into one with 3, and conversely.

II. Tools associated with Ariane-G5

A. Helps to construct MT systems

1. ATLAS, a system to help indexing in coded dictionaries

ATLAS is a language for writing "indexing charts", designed and implemented by D. Bachut [Bachut & Verastegui 84]. It has been used to produce numerous indexing manuals of the Russian-French system. One describes an acyclic graph where the internal nodes bear questions, the edges possible answers, and the leaves the results attained (usually, names of formats or of procedures).

That graph may be drawn, to produce paper manuals, or be used dynamically on screen, to create menus in a window, and send the results to the appropriate places in a second window showing the dictionary to be constructed or modified.

2. VISULEX, a tool for the synthetic visualization of lexical informations contained in a lingware written in Ariane

VISULEX has been produced by [Bachut & Verastegui 84] in the framework of the ESOPE project of ADI. This system allows to visualize all or part of the lexical informations contained in a system written in Ariane-78 or Ariane-G5, at two levels (codes and comments), which frees linguists from having to search many files at the same time.

3. BDTAO, a lexical data base management system for MT

BDTAO is a lexical data base management system specifically designed for MT, but not for Ariane, by D. Bachut and R. Gerber of B'VITAL. Ariane analysis and generation formats and dictionaries for a language are constructed automatically by BDTAO from the monolingual sub-base concerning that language. This way, the necessity to code the same term several times in different dictionaries. For transfer, there is one sub-base for each language pair is avoided.

There is a distinction between the "kernel" dictionary, which belongs to the grammatical system of the language and is directly coded, the general dictionary, and the terminological dictionaries. In transfer, the general part is much more complex than the terminological part, and the construction of the corresponding Ariane dictionaries is not yet fully automatized (in the meantime, usual indexing manuals are used to index directly in Ariane TL dictionaries).

4. TTEDIT, a transformational editor of decorated trees

In order to develop independently analyzers, transfers and generators, test data must be available. [Durand 88] has developed TTEDIT, which is a tree editor (trees may bear simple labels or complex decorations). Its originality is that its basic operations are sub-tree transformations, as in ROBRA, and not direct manipulations on the nodes and edges. That allows one to work on large trees as one works on texts using editors equipped with highly parametrized search and replace facilities. TTEDIT is completely integrated with the standard text editor XEDIT. As in XEDIT, it is possible to write "macros", which are in fact transformational grammars analogous to those of ROBRA.

Starting from a tree produced by B'VITAL's analyzer of French and transformed by a transfer written for the occasion (in Ariane), [Guilbaud 88] has used TTEDIT in order to realize additional *ad hoc* transformations leading to an analysis interface structure conforming to the "linguistic legislation" of the Eurotra project and... to the personal feelings of people in charge for each particular case not foreseen by that "legislation" (another argument in favor in real specifications).

C. Tools for handling texts

1. THAM

THAM stands for "Machine-Aided Human Translation" ("Traduction Humaine Aidée par la Machine"). Programmed in EXEC2/XEDIT, THAM works as an XEDIT macro [Bachut & Verastegui 84]. Suppose that an Ariane user is revising a raw machine translation. By simply hitting a key, he starts THAM, which allows him to access a "natural" dictionary, dymamically modifiable, together with the revised text, the raw translation and the source text. THAM may also be used in stand-alone mode.

This tool is by far not a full "translator/revisor workstation", as those offered by certain firms (Weidner, ALps...), but only a useful extension to XEDIT. As a matter of fact, for an industrial use, it is far more preferable to revise the translations on a PC, a MacIntosh, or, if luxury is permitted, on a Xerox Star™ machine.

Rather than to develop ourselves full environments for translator/revisor, which is very heavy work, we have preferred to cooperate with other teams on certain aspects of their tools.

2. LT, a language for writing transcriptors

LT [Lepage 86] allows to write quickly transcriptors of texts. For example, the Russian-French system uses a transcription of the Russian texts in a subset of the PL/I character set (upper case roman letters, digits, usual punctuation marks, and some special signs). A transcriptor written in LT allows to produce the texts in cyrillic (not available on the available minicomputer configuration), with upper case, low case, and minimal formatting, on an ASCII printer (S700) equipped with cyrillic fonts and connected to the mini 4361 through a PCI.

The abstract model here is a finite-state transducer with two tapes. The input tape has two reading heads. The first one can only go forward, while the second one is used as a "look-ahead" and may go forward or be "recalled" to the position of the first one. The output tape has one writing head. The states are structured : a full state is the combination of an elementary state and a set of values of a certain number of variables.

Writing a transcriptor consists in declaring the variables and their types (e.g. font, language, length of the look-ahead...), and then in describing the graph of the transition system, with one node for each elementary state. The edges bear the transitions, which are classical production rules. There are a number of predefined functions, as well as facilities for parametrizing and factorizing, to avoid writing a too large number of rules.

3. SCRIBERE/SCRIBM, an extension to SCRIPT/DCF

SCRIBERE allows to describe the textual content and the logical structure of a document by using macos written in IBM's SCRIPT/DCF formatter. That tool offers possibilities inspired by SGML. It has been developed by D. Bachut and N. Verastegui to compensate for the unavaibility of GML, too expensive. The circumstance has been used to take into account certain linguistic aspects present in texts (current language, current transcription, etc.). SCRIBM has been written by R. Zajac starting from SCRIBERE to enrich it (footnotes, matrices, references, indexes...).

The problems of representation and processing of multilingual texts appear slowly, but are considerable. We hope that many researchers will join the TEI (Text Encoding Initiative) and eventually come up with really satisfactory solutions, usable in multilingual MT systems.

III. Towards real lingware engineering

A. Short history

1. The preoperational Russian-French prototype (1980—87)

The development of the Russian-French system from the stage of mockup to that of a real prototype, used in a "preoperational" setting (a flow of texts was regularly sent, translations had to be sent back before a certain delay, dictionaries had to be improved in relation with a specialist of technique and translation) has shown us the necessity to develop tools to help linguistic programming in Ariane, such as ATLAS or VISULEX. Since the beginning of the eighties, it has become evident that this type of programming could and should be compared with the programming of large software systems, and attacked accordingly. As an example, the Ariane-78 software has cost about 30 manxyears of work, and the Russian-French lingware about 80.

This prototype, realized by N. Nédobejkine and his collaborators, is not strongly specialized to a certain type of texts. Its vocabulary, of about 30,000 terms (simple and compound terms, idioms), contains 5 to 6,000 general terms, the rest being distributed between various scientific and technical domains (space and earth sciences, metallurgy, aeronautics, linguistics...). At the beginning, it was not clear whether MT for the watcher or MT for the revisor was aimed at. It appeared that the quality obtained was good enough to allow for revision of a page in about 1/4 of an hour, a time very much comparable to that of the revision of a human raw translation — of course, the errors are not the same, and the power of the editor may be used to speed up work. However, for that language pair, the only realistic use in France would concern scientific information gathering. But that would necessitate access to the texts in computer-readable form, on tape or through a network.

2. The analyzers of English and the translation mockups (1983—89)

The methodology for constructing MT systems in Ariane owes much to various cooperations through which B. Vauquois has been able to try various methods and come up with the above-mentioned principles. That began as soon as 73-74 with studies on the analysis of French, in relation with SFB/100 at Saarbrücken (J. Weissenborn, E. Stegentritt), and then of Portuguese (P. Daun Fraga). The methods developed for French were taken over and improved for Portuguese, and again reused and improved on French, in the context of a feasibility study on French-English for the French Telecoms (Vauquois, Guilbaud, Dymetman) in 81-82.

The analysis of English became then the common point of several studies made in cooperation. The most important led to an Englis-Malay laboratory prototype [Tong 86]. Others led to mockups into Chinese [Yang 81], Thai, and finally Arabic [Moneimne 89]. That analysis was also used as a starting point for a work on a "standard analyzer" of English undertaken during the MAT-NP and pursued for some time afterwards at B'VITAL.

As machine analysis of English is more delicate than that of French, the necessity of a "static" specification of the "dynamic" analysis and generation grammars appeared to B. Vauquois as soon as 1980. [Chappuy 83] and [Vauquois & Chappuy 85] present its formal and practical aspects. A methodology for specifying and implementing analyzers and generators from static grammars was defined and experimented during the ESOPE project of ADI on the development of a pedagogical English-French mockup (BEX-FEX).

3. B'VITAL's operational French-English system (BV/aero/F-E

The linguistic work on French-English has been pursued by the small firm B'VITAL, founded during the MAT-NP, and continues today in the framework of an action of the French Ministery of Industry, at the level of SITE, the largest European firm in technical documentation and translation, of which B'VITAL has become a subsidiary. In 1986, the static grammar of French consisted in about 150 "construct boards" ("boards" are 2-dimensional representations of correspondence rules) and 450 "disambiguation boards" (preference rules). AS is today more than 20,000 lines long (in ROBRA). The dictionaries contain about 20,000 terms, of which more than half are terminological.

B. Static grammars and lexical data bases

Static grammars were first prepared on paper. [Yan 87] presents a first computer environment to handle them, which is realized on the MacIntosh by integrating an array of commercially available tools.

In his thesis, [Zaharin 86] studied the formal semantics of static grammars and proposed some improvements. Since then, he and his team have produced SaGE, a MacIntosh application which allows to comput on a static grammar. That is not yet an executable specification, but SaGE can automatically produce an analyzer and a generator (in ROBRA) from such a grammar, following the general strategic principles explained above. This work is preliminary (in particular, the transformational systems produced are far from being optimized), but quite encouraging.

Finally, it would be highly desirable to develop lexical data bases, no more specific to MT, but aiming at multiple uses. Various studies have been undertaken [Boitet & Nédobejkine 86], up to the prototyping on a 3-lingual base in telecommunications (French-English-Japanese). Various problems are still posed at the level of the definition of the linguistic content of such bases, of their logical structure, and of their implementation, no commercial DBMS beeing really adequate.

C. Commented examples of French-English translations

These examples have been selected from raw translations produced by B'VITAL 's BV/aero/F-E system during demonstrations at TEC-88.

Après essai, s'assurer du fonctionnement correct de l'ensemble raccord	*After test, check that the coupling assembly works correctly.*

Note here the passage from a nominal (prepositional) phrase "du fonctionnement correct" to a verbal clause, "that... works correctly", with as corollary the passage from the adjective "correct" to the adverb "correctly". These transformations are not performed during transfer. It is the first step of syntactic generation which, starting from the "g-structure" (generating structure), considered as under-specified with respect to the syntactic functions and the syntagmatic and morphosyntactic classes, recomputes these levels depending on the initial goal (here, to construct a verbal phrase), using the deeper levels (logical relations inside the strict predicative frame, semantic relations for the circumstancial complements).

Thanks to the notion of lexical unit, the generator knows, without having to consult a dictionary, which lemmas are contained in the considered derivational family, which constrains the possible paraphrases. Here, "fonctionnement" has been reduced to the UL "fonctionner-V", translated as "work-V", which bears the potentiality of deriving an action noun. It is then simply the order of preference between the rules controlling the choice of syntagmatic categories which triggers the construction of a subordinate clause rather than of a nominal phrase ("the correct working of the coupling assembly").

Porter sur celle-ci la date de la dernière réception ou révision.	*Write on this one the date of the last reception or of service.*

"Porter" is a support verb here, and "porter une date" is translated as "to write a date" and not as "to carry a date", thanks to a test performed at lexical transfer on the syntactic and semantic features of the second argument (ARG1) of "porter " (its logical object).

Effectuer la vidange générale et la purge du carburant (voir chapitre 12).	*Drain in a general manner and bleed fuel (see chapter 12).*

"Effectuer la vidange" is translated here by the simple verb "to drain", thanks to the notion of lexical unit, and to the organization of the lexical transfer. "Vidange" is reduced to "vidanger-V", and that UL gives in translation a tree in which the possibility of the presence of a support verb such as "effectuer", "faire", etc. is coded. The translation of the support verb, if present, will be erased by the structural transfer.

Le bouchon a pour but d'assurer la protection d'un raccord auto-obturable lorsque celui-ci n'est pas utilisé au sol ou en vol.	*The trap is used for carrying out the self-sealing coupling protection when this one is not used at the ground or in flight.*

"Avoir pour but" is recognized as a compound predicate, "avoir-but-V(x0,x1)", which is translated as "se-V(x1,x0)", with a conversion of arguments. That explains the generation of a passive.

Enduire légèrement le joint neuf de liquide d'utilisation.	*Slightly coat the new joint with operating fluid.*

The translation of prepositions is always delicate. It is necessary to know whether they introduce arguments or circumstants. "enduire-V" is a predicate with 3 arguments (qn enduit qn/qc de qc), the third one being introduced by "de". The analyzer prefers to fill the argument frame, and the introductor of the corresponding argument of "coat-V" is "with".

Ouvrir progressivement le robinet (3), appliquer une pression jusqu'à 1,5 bar jusqu'à l'allumage du voyant lumineux DS2 et l'extinction du voyant DS1.	*Gradually open tap (3), apply a pressure up to 1,5 bar until the light DS2 switching on ((ignition)) and the signal lamp DS1 extinction.*

The preposition "jusqu'à" introduces here two circumstancials. What is really translated is the semantic relation (here, RS=LOC with SEM=TEMPS and SLOC=QUA), refined by the preposition and by the semantic features of the "governor" (head) of the group, here PROCESS for "allumer-V", and af the predicate ("appliquer-V").

Ouvrir progressivement le robinet (3) jusqu'à obtenir une pression de 9 bars.	*Gradually open tap (3) until a pressure of 9 bars is obtained.*

No explicit transformation is performed. The infinitive clause is rendered by a subordinate clause simply through the normal functioning of the generator, as explained above. As argument 0 (logical subject) is not expressed, a passive is generated. That is only a matter of stylistic preference. It would be equally possible to generate "until one obtains a pressure of 9 bars", or "until obtaining...", as in the following example.

Procéder à la dépose des panneaux.	*Remove the panels.*
IMPORTANT : avant de déposer ou de reposer le panneau central intrados de voilure, il est nécessaire de procéder à certaines modifications.	*IMPORTANT : before removing or reinstalling the lower central wing panel, it is necessary to proceed with some modifications.*

Here, the construct preferred for the conjunction "before" is the gerundive. On the other hand, the preposition "à" introduces argument 1. In the m-structure produced by analysis, it may well have been suppressed. "With" is contained in the valency frame of "proceed-V" for the same argument position, and is introduced by the generator.

Conclusion

We have given here only an overview of the computer tools and linguistic methods developed at Grenoble for building MT systems for revisors. Although the aspects of computer aids to translators and revisors have been mentioned just in passing, they are an indispensable complement to "real" MT, and many industrial firms are working in that direction.

In the future, decisive advances in MT for the revisor should come from the lingware engineering side. However, new research aiming at translating noisy texts (optical character reading or spoken dialogues) begins to lead to new algorithmic ideas and should trigger the creation of new specialized languages.

Finally, the large diffusion of increasingly powerful micros allows to envisage the idea of Personal MT, based on coupling the methods presented above with a dialogue between the system and the author of the document to be translated. For that, it will be necessary to imagine a completely new software architecture, and to reshape a good part of the linguistic methods, in particular what concerns disambiguation, because direct postedition by a professional (the revisor) would have to be replaced by indirect preedition (by the author).

-o-o-o-o-o-o-o-o-o-o-o-

References

[1] ABBOU A., éd. (1988) *Traduction Assistée par Ordinateur.* Actes du séminaire international sur la TAO et dossiers complémentaires, Observatoire des Industries de la Langue, Paris, mars 1988, 234 p.

[2] B'VITAL (1988) *Translation Examples.* Support de démonstrations à TEC-88, Grenoble.

[3] BACHUT D., VERASTEGUI N. (1984) *Software tools for the environment of a computer-aided translation system.* Proc. of COLING-84, ACL, 330-334, Stanford.

[4] BOITET Ch., rédacteur. (1982) *"DSE-1"— Le point sur ARIANE-78 début 1982.* Contrat ADI/CAP-Sogeti/Champollion, 3 vol.

[5] BOITET Ch. (1985) *Traduction (assistée) par Ordinateur: ingéniérie logicielle et linguicielle.* Colloque RF&IA, AFCET, Grenoble.

[6] BOITET Ch. (1986a) *The French National MT-Project: technical organization and translation results of CALLIOPE-AERO.* IBM Conf. on Translation Mechanization, Copenhagen.

[7] BOITET Ch. (1986b) *Current Machine Translation systems developed with GETA's methodology and software tools.* ASLIB Conf. London.

[8] BOITET Ch. (1987a) *Research and development on MT and related techniques at Grenoble University (GETA).* Revised from Boitet (1984), in "Machine Translation today: the state of the art", Proc. third Lugano Tutorial, 2–7 April 1984, M. King, ed., Edinburgh University Press, 1987, 133–153.

[9] BOITET Ch. (1987b) *Current projects at GETA on or about Machine Translation.* Proc. II World Basque Congress, San Sebastian, 7–11 Sept. 1987, 28p.

[10] BOITET Ch. (1988a) *Software and lingware engineering in modern M(A)T systems.* in "Handbook for Machine Translation", Batori, ed., Niemeyer, 1988.

[11] BOITET Ch. (1988b) *Dictionnaires intégrés multiusages et multicibles (DIMM) : une première expérience.* Colloque sur l'histoire de la terminologie, Institut Libre Marie Haps, Bruxelles, 25—26 mars 1988, à paraître, 3 p.

[12] BOITET Ch. (1988c) *L'apport de Bernard Vauquois à la traduction automatique et au traitement automatique des langues naturelles.* Colloque sur l'Histoire de l'Informatique en France. Grenoble, 3—5 mai 1988, vol. 2, p. 63—82.

[13] BOITET Ch. (1988d) *Hybrid Pivots using m-structures for multilingual Transfer-Based MT Systems.* Meeting of the Japanese Institute of Electronics, Information and Communication Engineers, June 1988, NLC88-3, 17—22.

[14] BOITET Ch. (1988e) *Bernard VAUQUOIS' Contribution to the Theory and Practice of building MT Systems : a historical perspective.* Second International Conference on Theoretical and Methodological Issues in the Machine Translation of Natural Languages, Pittsburgh, June 1988, 18 p.

[15] BOITET Ch. (1988f) *PROs and CONs of the pivot and transfer approaches in multilingual Machine Translation.* New Directions in Machine Translation. BSO congress, Budapest, August 1988, 13 p.

[16] BOITET Ch. (1988g) *Representation and Computation of Units of Translation for Machine Interpretation of Spoken Texts.* TR-I-0035 ATR, Osaka & GETA, August 1988, 41 p. et in "The Czech Journal of AI", 1989.

[17] BOITET Ch. (1988h).*Record of Six Work Sessions on Concepts, Methods and Tools from Existing Running Real-Size MT Systems.*
TR-I-0044 ATR, Osaka & GETA, October 1988, 68 p.

[18] BOITET Ch. Éd. (1988i) *BERNARD VAUQUOIS et la TAO, vingt-cinq ans de Traduction Automatique, ANALECTES.* Grenoble, 1988, 700 p. (diffusé à partir du 1/1/89)

[19] BOITET Ch., GERBER R. (1984) *Expert systems and other new techniques in MT.* Proc. COLING-84, ACL, 468-471, Stanford.

[20] BOITET Ch., GUILLAUME P., QUEZEL-AMBRUNAZ M. (1978) *Manipulation d'arborescences et parallélisme: le système ROBRA.* Proc. COLING-78, Bergen.

[21] BOITET Ch., GUILLAUME P., QUEZEL-AMBRUNAZ M (1982) *ARIANE-78, an integrated environment for automated translation and human revision.* Proc. COLING-82, North-Holland, Ling. series 47, 19-27, Prague.

[22] BOITET Ch., GUILLAUME P., QUEZEL-AMBRUNAZ M. (1985) *A case study in software evolution: from ARIANE-78 to ARIANE-85.* Proc. of the Conf. on theoretical and methodological issues in Machine Translation of natural languages, 27-58, Colgate Univ., Hamilton, N.Y.

[23] BOITET Ch., NEDOBEJKINE N. (1981) *Recent developments in Russian-French Machine Translation at Grenoble.* Linguistics 19(3/4), 199-271.

[24] BOITET Ch., NEDOBEJKINE N. (1983) *Illustration sur le développement d'un atelier de traduction automatisée.* Colloque "l'informatique au service de la linguistique", Univ. de Metz.

[25] BOITET Ch., NEDOBEJKINE N. (1986) *Toward integrated dictionaries for M(a)T: motivations and linguistic organization.* Proc. COLING-86, IKS, 423-428, Bonn.

[26] BOITET Ch., TCHEOU F. X. (1990) *Un codage phonético-structural des caractères chinois dans les textes chinois.* RR. IMAG-GETA n° 001, juin 1990, 8 p. Soumis à ROCLing-III, Taipeh, septembre 1990.

[27] BOITET Ch., ZAHARIN Y. (1988) *Representation Trees and String-Tree Correspondences.* COLING-88, Budapest, 22-27 August 1988, 6 p.

[28] CHANDIOUX J., GUERARD M.F. (1981) *METEO: un système à l'épreuve du temps.* Meta **26**(1), 17–22.

[29] CHAPON L., DJAMEI M., DJOHARIAN P., MORENO F. (1989) *Etude et révision du système PILAF. Portage vers le Mac. Réalisation d'un lemmatiseur.* Projet DESS-IDC, juin 1989, 120 p.

[30] CHAPPUY S. (1983) *Formalisation de la description des niveaux d'interprétation des langues naturelles.* Thèse, Grenoble.

[31] CHAUCHE J. (1975) *Les langages ATEF et CETA.* AJCL, microfiche **17**, 21—39.

[32] DURAND J. C. (1988) *TTEDIT : Un éditeur transformationnel d'arbres.* Thèse de Doctorat, UJF (Grenoble 1), mars 1988.

[33] COLMERAUER A. (1970) *Les systèmes-Q, un formalisme pour analyser et synthétiser des phrases sur ordinateur*. TAUM, Univ. de Montréal.

[34] FENG Z. W. (1981) *Mémoire pour une tentative de traduction multilingue du chinois en frrançais, anglais, japonais, russe et allemand*. Doc. GETA, Grenoble, 40p. + annexes.

[35] GERBER R. (1984) *Etude des possibilités de coopération entre un système fondé sur des techniques de compréhension implicite (système logico-syntaxique) et un système fondé sur des techniques de compréhension explicite (système expert)*. Thèse, Grenoble.

[36] GUILBAUD J.-Ph. (1984) *Principles and results of a German-French MT system*. Lugano tutorial on Machine Translation.

[37] GUILBAUD J.-Ph. (1986) *Variables et catégories grammaticales dans un modèle ARIANE*. Proc. COLING-86, IKS, 405–407, Bonn.

[38] GUILBAUD J.-Ph. (1988) *Projet Eurotra : Réalisation en ARIANE d'un transfert des structures produites par l'analyseur du français de B'VITAL vers des structures interfaces IS EUROTRA*. GETA, juin 1988, 205 p.

[39] GUILBAUD J.-Ph. (1989) *Quelques aspects de la représentation et du calcul linguistique au GETA. Groupes nominaux et pronoms. Application à l'allemand*. Doc. GETA, juillet 1989, 32 p. (préparé pour un exposé présenté au Collège de France le 29/11/88 dans le cadre du séminaire "Analyse assistée par ordinateur — les conditions de l'anaphorisation").

[40] GUILBAUD J.-Ph. (1990) *Méthodes et représentations linguistiques en Ariane-G5. Étude de l'étape d'analyse et introduction à l'étape de transfert*. GETA, juin 1990, 25 p.

[41] GUILLAUME P. (1989a) *Ariane-G5 - Extensions apportées au langage SYGMOR*. GETA, janvier 1989, 6 p. (Ariane-G5 version 3)

[42] GUILLAUME P. (1989b) *Ariane-G5 - Les Langages Spécialisés TRACOMPL et EXPANS* . GETA, septembre 1989, 93 p. (Ariane-G5 version 3)

[43] HUTCHINS W. J. (1982) *The evolution of machine translation systems*. In: Practical experience of machine translation, Proc. ASLIB-81 Conf., London, 5–6 Nov. 1981, Lawson, ed, North Holland, 21–37.

[44] HUTCHINS W. J. (1986) *Machine Translation : Past, Present, Future*. Ellis Horwood, John Wiley & sons, Chichester, England, 382 p.

[45] ISABELLE P., BOURBEAU L. (1984) *TAUM-AVIATION: its technical features and some experimental results*. Computational Linguistics, **11**:1, 18-27.

[46] JEIDA (1989) *A Japanese view of Machine Translation in light of the considerations and recommendations reported by ALPAC, USA*. Japanese Electronic Industry Development Association, Tokyo, 1989, 197 p.

[47] KITTREDGE R. (1983) *Sublanguage — Specific Computer Aids to Translation — a survey of the most promising application areas*. Contract n° 2-5273, Université de Montréal et Bureau des Traductions, mars 1983, 95 p.

[48] KITTREDGE R. (1986) *Analyzing Language in Restricted Domains : Sublanguage Description and Processing*. Grishman R. & Kittredge R., eds., Lawrence Erlbaum, Hillsdale, New-Jersey, 1986.

[49] LAWSON V. (1982), ed *Practical experience of Machine Translation*. Proc. ASLIB-81 Conf., London, 5-6 Nov. 1981, North-Holland.

[50] LAWSON V. (1985), ed *Tools for the trade, Translating and the computer 5*. Proc. ASLIB-81 Conf., London, 10-11 Nov. 1983.

[51] LEPAGE Y. (1986) *A language for transcriptions*. Proc. COLING-86, IKS, 402-404, Bonn.

[52] LEPAGE Y. (1988) *Ambiguities and second generation MT systems*. First European Conference on Information Technology for Organisational Systems. Athens, May 1988, 16—20, 6 p.

[53] LEPAGE Y. (1989) *Un système de Grammaires Correspondancielles d'Identification* Thèse, Université Joseph Fourier, Grenoble, juin 1989, 184 p.

[54] MELBY A. (1982) *Multilevel translation aids in a distributed system*. Proc. COLING-82, North-Holland, Ling. series 47, 215-220, Prague.

[55] MONEIMNE (1989) W. *TAO vers l'arabe. Spécification d'une génération standard de l'arabe. Réalisation d'un prototype anglais-arabe à partir d'un analyseur existant.* Thèse, UJF, Grenoble, juin 1989, 159 p. + annexes.

[56] NAKAMURA J-i., TSUJII J-i., NAGAO M. (1984) *Grammar writing system (GRADE) of Mu-Machine Translation Project and its characteristics.* Proc. COLING-84, ACL, Stanford.

[57] NEDOBEJKINE N. (1990) *Représentation du lexique dans la théorie linguistique du GETA.* GETA, juin 1990. Soumis aux Journées du PRC-CHM, Toulouse, janvier 1991, 42 p.

[58] NIRENBURG & al. (1989) *KBMT-89 Project Report.* Center for Machine Translation, Carnegie Mellon University, Pittsburgh, February 1989, 286 p.

[59] PUERTA M.-C. (1988) *Analyseur standard de l'anglais (ASA).* Rapport de DEA, USMG, GETA, septembre 1988, 135 p.

[60] QUEZEL-AMBRUNAZ M. (1990a) *Ariane-G5 v.3 - Le moniteur.* GETA, juin 1990, 206 p. (version 3)

[61] QUEZEL-AMBRUNAZ M. (1990b) *Ariane-G5 v.3. Transfert des composants linguistiques : d'une machine vers une bande et d'une machine vers une autre machine.* GETA, juin 1990, 9 p. (Ariane-G5 version 3)

[62] SAKAMOTO Y., SATOH M., ISHIKAWA T. (1984) *Lexicon features for Japanese syntactic analysis in MU-project-JE.* Proc. COLING-84, ACL, Stanford.

[63] SLOCUM J. (1984) *METAL: the LRC Machine Translation system.* Lugano tutorial on Machine Translation.

[64] STEWART G. (1975) *Manuel du langage REZO.* TAUM, Univ. de Montréal.

[65] TCHEOU F., GU Y. (1988) *First results of a French-Chinese machine translation prototype in ARIANE using a French operational analyser.* GETA, décembre 1988, 8 p.

[66] TOMASINO I. (1990) *$O_D I_L E$, Outil d'Intégration Extensible de Dictionnaires et de Lemmatiseurs.* GETA, mai 1990. Soumis aux Journées du PRC-CHM, Toulouse, janvier 1991, 15 p.

[67] TONG L. C. (1986) *English-Malay translation system: a laboratory prototype.* Proc. COLING-86, IKS, 639-642, Bonn.

[68] VAUQUOIS B. (1978) *Description de la structure intermédiaire.* Réunion aux CE, Luxembourg, avril 1978, 27 p.

[69] VAUQUOIS B. (1979) *Aspects of mechanical translation in 1979.* Conference for Japan IBM Scientific Program, July 1979, 52 p.

[70] VAUQUOIS B. (1980a) *La traduction automatique.* In : Les Sciences du Langage en France au 20ème siècle. Articles recueillis par B. Pottier, Paris, SELAF, 1980, 799–824.

[71] VAUQUOIS B. (1980b) *Etude de la validité du formalisme choisi pour représenter la structure linguistique interface.* Rapport de contrat CEE, avril 1980, 81 p.

[72] VAUQUOIS B. (1980c) *L'informatique au service de la traduction.* Journées d'Etudes du Festival International du Son Haute-Fidélité, Stéréophonie, Paris, 1980, et In : META, Journal des Traducteurs, Montréal, **26**/1, 9–17, mars 1981.

[73] VAUQUOIS B. (1981a) *Etude de typologie de textes. Ambiguïtés lexicales dans les textes de la revue Electronics.* Research report, TELECOM, juillet 1981, 50 p.

[74] VAUQUOIS B. (1981b) *Traduction assistée par ordinateur. Formation de spécialistes. Préparation du transfert technologique.* Projet ESOPE, Contrat ADI, mai 1981, 25 p.

[75] VAUQUOIS B. (1983) *Automatic computer aided translation and the arabic languages.* First Arab School on Science and Technology. Applied Arabic Linguistics and Signal and Information Processing, Rabat, October 1983, 21 p.

[76] VAUQUOIS B. (1984) *The organization of an automated translation system for multilingual translations at GETA.* IBM Europe Institute on Natural Language Processing, Davos, July 1984, 41 p.

[77] VAUQUOIS B. (1985) *The approach of GETA to automatic translation. Comparison with some other methods.* International Symposium on Mechanical Translation. Riyadh, March 1985, 67 p.

[78] VAUQUOIS B. (1988) *BERNARD VAUQUOIS et la TAO, vingt-cinq ans de Traduction Automatique, ANALECTES.* Ch. BOITET, éd. Association Champollion & GETA, Grenoble, 1988, 700 p.

[79] VAUQUOIS B., BOITET Ch. (1984) *Etudes sur les travaux en TAO portant sur des langues asiatiques (japonais, chinois, malais, thaï) et étude des problèmes liés au traitement de textes utilisant de grands jeux de caractères (idéogrammes).* R. R. n° 83/739, Ass. Champollion/ADI, juin 1984, 81 p.

[80] VAUQUOIS B., BOITET Ch. (1985) *Automated translation at Grenoble University.* In : Computational Linguistics, **11**/1, 28–36, Jan.-March 1985.

[81] VAUQUOIS B., BOITET Ch., DAUN FRAGA P., GUILBAUD J.P., NEDOBEJKINE N. (1982) *Définition d'une méthode de travail d'équipe linguistique.* Projet ESOPE, Contrat ADI, novembre 1982, 94 p.

[82] VAUQUOIS B. CHAPPUY S. (1985) *Static Grammars. A formalism for the description of linguistic models.* Proc. 1st Int. Conf. on Theoretical and Methodological Problems in the Machine Translation of Natural Languages, Colgate University, Hamilton, N.Y., Aug. 1985.

[83] VAUQUOIS B., DAUN FRAGA P (1982) *Etude de faisabilité d'un système de traduction automtaisée anglais-français.* Research report, TELECOM, février 1982, 22 p.

[84] WHEELER P. (1986) *The LOGOS translation system.* Proc. of IAI-MT, 20-33, IAI-EUROTRA-D, Saarbrücken.

[85] WITKAM T. (1987) *Interlingual MT – an industrial initiative.* Proc. of the MTS, 135–140, Hakone.

[86] ZAHARIN Y. (1986a) *Strategies and heuristics in the analysis of a natural language in Machine Translation.* Ph.D. thesis, Universiti Sains Malaysia, Penang (research conducted in cooperation with GETA, Grenoble).

[87] ZAHARIN Y. (1986b) *Strategies and heuristics in the analysis of a natural language in Machine Translation.* Proc. COLING-86, IKS, 136–139, Bonn.

[88] ZAJAC R. (1986b) *SCSL: a linguistic specification language for MT.* Proc. COLING-86, IKS, 393-398, Bonn.

[89] YANG P. (1981) *Un essai sur la génération du chinois.* Doc. GETA, Grenoble, 30p. + annexes.

[90] YAN Y. F. (1987) *Vers une ingénierie de la production de linguiciels. Spécification et réalisation d'un prototype de poste de travail linguistique pour la spécification de correspondances structurales.* Thèse, Univ. de Grenoble.

-o-o-o-o-o-o-o-o-o-o-