

LA TAO À GRENOBLE EN 1990

1980—90 : TAO du réviseur et TAO du traducteur

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

Introduction

Des recherches en TAO ont été menées à Grenoble depuis 1961, sous la direction du Professeur Vauquois, d'abord dans le cadre du CETA, laboratoire propre du CNRS, puis, à partir de 1971, dans le cadre du GETA, laboratoire universitaire associé au CNRS. À la fin de la première période, vers 1967-70, un premier système russe-français avait été développé, et testé sur plus de 300.000 mots de textes réels (articles scientifiques). Il s'agissait alors de *TAO du veilleur*, ou TA pure et dure, c'est-à-dire qu'on visait un processus purement automatique tel que le lecteur, supposé spécialiste du domaine, puisse accéder au contenu de l'original sans en connaître la langue.

À la suite d'un changement d'ordinateur (IBM 7044 à IBM 360/67), qui aurait demandé une conversion importante difficilement justifiable hors d'une perspective d'exploitation opérationnelle, ce système dut être abandonné, malgré sa qualité et son ampleur remarquables. Ce fut l'occasion de repartir sur des idées nouvelles, dans le contexte de la *TAO du réviseur*, dans laquelle on vise à produire automatiquement des traductions brutes, destinées à être révisées par un professionnel. Cette étape de révision manuelle devant naturellement se faire sur machine, ce fut aussi l'occasion d'aborder la *TAO du traducteur*, ou "traduction informatisée" (TI).

En 1978, une nouvelle méthodologie de programmation linguistique avait été proposée (approche transfert multiniveau, programmation heuristique), et commençait à être expérimentée sur divers couples de langues, dont principalement le russe-français, grâce aux éléments logiciels d'un environnement informatique de génération de systèmes de TAO multilingues, qui fut appelé "Ariane-78" quand sa première version complète fut disponible, début 1978. Ce nom fut choisi par référence à la déesse et à son fameux fil : il s'agissait de souligner que l'informatique, si elle est essentielle, doit être mise au service de non-informaticiens et leur permettre de travailler de façon autonome, grâce à des langages spécialisés adéquats (langages symboliques de règles), et à une interface interactive transparente.

Grâce au soutien du CNRS et de la DRET, un premier prototype "préopérationnel" russe-français fut ensuite développé et expérimenté en grandeur réelle, de début 1980 à fin 1986. En parallèle, un effort de valorisation fut entrepris. En 1981—82, le GETA fut le moteur du projet ESOPE de l'ADI, dont le but était de préparer un "projet mobilisateur de la filière électronique", le "PN-TAO" (projet national de TAO). De 1983 à 1987, le GETA y participa très activement, en augmentant les capacités et la fiabilité d'Ariane-78.4, et en étant au cœur de la spécification linguistique du système français-anglais pour l'aéronautique construit en coopération avec les partenaires industriels.

Depuis, des extensions considérables ont été apportées au système Ariane, dont la version actuelle, Ariane-G5.3, est ici brièvement décrite dans la première partie. Plusieurs outils logiciels complémentaires y ont été ajoutés, par le GETA ou par ses partenaires. Ils sont présentés dans la seconde partie. Enfin, on montre que les techniques de programmation linguistique en TAO sont en train de passer du stade de l'artisanat à celui d'un véritable génie logiciel. Pour donner une idée de ce que permet d'obtenir la TAO du réviseur moderne, et un aperçu sur son fonctionnement interne, nous terminerons en commentant quelques exemples réels de traductions brutes (avant révision) produites par le système B'VITAL/aéro/F-E fin 1988.

I. Ariane-G5

A. Principes généraux

Ariane-G5 est un générateur (G) de systèmes de TAO reposant sur cinq (5) langages spécialisés pour la programmation linguistique (LSPL). Chacun de ces langages est compilé, et les tables internes produites sont données en paramètres au "moteur" du langage.

Bien qu'il soit particulièrement adapté à l'approche transfert, il ne l'impose pas. À part les limites d'implémentation, la seule contrainte forte est que les structures de représentation des unités de traduction doivent être des arbres décorés.

La *sémantique intrinsèque* (terme emprunté à J. P. Desclés) est représentée dans ces langages, donc de façon linguistique. Si l'on veut écrire un système spécialisé à un sous-langage réduit et à un microdomaine, rien n'interdit de suivre la même voie que METEO [Chandioux & Guérard 81] et d'écrire des grammaires et des dictionnaires "sémantiques". Pour construire un système muni d'une sémantique extrinsèque ("ontologie" de l'univers de référence), il faudrait coupler Ariane-G5 à un "système expert correcteur", comme cela a été suggéré et prototypé dans [Gerber 84].

Par rapport à la quasi-totalité des systèmes existants, Ariane-G5 présente l'avantage que l'unité de traduction n'est pas réduite à la phrase, mais peut comporter plusieurs paragraphes (100 à 200 occurrences en général, soit près d'une page standard).

1. Environnement matériel et logiciel

Ariane-G5 tourne sous VMSP/CMS, sur gros ordinateurs (3090, 303X), sur mini (43XX, 937X), et sur gros micro (PC-AT/370, PS2-80/7437).

VMSP est un hyperviseur qui simule un ensemble de "machines virtuelles". Chaque machine virtuelle tourne sous un système d'exploitation propre. Par exemple, on peut faire tourner en même temps des machines virtuelles sous MVS/TSO, sous CMS, et sous AIX. Le sous-système RSCS permet d'organiser les machines virtuelles et les ressources réelles du système (périphériques) en réseau. CMS est un système d'exploitation interactif mono-utilisateur puissant, qui supporte un grand nombre de langages et d'outils de programmation, mais qui n'offre pas d'organisation hiérarchique des fichiers par répertoires et sous-répertoires. Il est en général utilisé pour le développement de logiciels ou pour des applications interactives.

La version actuelle d'Ariane-G5 représente environ 400.000 lignes sources, plus 30.000 lignes de messages par langue de dialogue¹. La partie exécutable réside sur un "minidisque" (disque virtuel) d'une machine virtuelle particulière, et occupe environ 10 Mo. Pour qu'une autre machine virtuelle puisse être une "machine utilisateur" (d'Ariane), il suffit d'effectuer une connexion logique (en "B/A") du minidisque Ariane à son minidisque ("A"). Ariane gère alors, sur le minidisque utilisateur, deux bases de données spécialisées, l'une contenant les logiciels et l'autre les textes (au maximum 3.000 langues sources et langues cibles, 1.000 "corpus", et 10.000 textes par corpus).

¹Ariane-78.4 avait deux versions parallèles, en français et en anglais. Ariane-G5 est programmé de façon totalement multilingue, mais la seule version complète des messages est actuellement en français.

La formation informatique minimale nécessaire pour utiliser Ariane-G5 consiste à apprendre les commandes élémentaires de début et de fin de session (login, logout), l'éditeur d'écran XEDIT, et, pour les développeurs de systèmes de TAO, l'organisation du moniteur interactif et les langages spécialisés.

2. Organisation logique

2.1. Étapes, phases et articulations du processus de traduction

La traduction d'une langue "source" vers une langue "cible" se fait en trois "étapes" successives : analyse, transfert et génération¹. Chaque étape est réalisée en au moins deux et au plus quatre "phases" successives, reliées ou non par des "articulations", qu'on peut en première approximation considérer comme de simples "changements de coordonnées". Les phases reçoivent un mnémonique à 2 lettres (ex: AM pour analyse morphologique), et les articulations un mnémonique à 4 lettres (ex: AMAS pour l'articulation AM-AS).

En analyse, on trouve successivement les phases :

AM	(analyse morphologique)	obligatoire,	écrite en ATEF ;
AX	(analyse expansive X)	facultative,	écrite en EXPANS ;
AY	(analyse expansive Y)	facultative,	écrite en EXPANS ;
AS	(analyse structurale)	obligatoire,	écrite en ROBRA.

En transfert, on trouve successivement les phases :

TL	(transfert lexical)	obligatoire,	écrite en EXPANS ;
TX	(transfert expansif X)	facultative,	écrite en EXPANS ;
TS	(transfert structural)	obligatoire,	écrite en ROBRA ;
TY	(transfert expansif Y)	facultative,	écrite en EXPANS.

En génération, on trouve successivement les phases :

GX	(génération expansive X)	facultative,	écrite en EXPANS ;
GS	(génération syntaxique)	obligatoire,	écrite en ROBRA ;
GY	(génération expansive Y)	facultative,	écrite en EXPANS ;
GM	(génération morphologique)	obligatoire,	écrite en SYGMOR.

L'ordre de ces phases à l'intérieur de chaque étape est imposé dans la version actuelle. Par conséquent, les "articulations" possibles, toutes écrites en TRACOMPL, sont AMAX, AMAY, AMAS, AXAY, AXAS, AYAS, ASTL, puis TLTX, TLTS, TXTS, TSTY, TSGX, TSGS, TYGX, TYGS, et enfin GXGS, GSGY, GSGM et GYGM. On n'a en fait besoin d'en écrire que si l'on veut composer deux phases provenant de linguiciels utilisant des jeux de variables hétérogènes.

Les opérations linguistiques effectuées dans chaque phase ne correspondent pas nécessairement à leurs noms de façon stricte. Par exemple, l'analyse morphologique peut être réalisée en AM, mais on peut aussi la distribuer en AM, AX, AY, et une fraction d'AS (par exemple, pour reconnaître la présence effective de tournures non connexes). En général, le transfert lexical est également réparti (au moins) entre les phases TL et TS, pour des raisons analogues. De même, la génération morphologique d'une langue comme l'arabe [Moneimne 89] peut avantagusement être répartie entre la fin de la phase GS et la phase GM.

¹On utilise ce terme plutôt que "synthèse", par analogie avec celui de "génération de code" en compilation.

2.2. Types de données essentiels et terminologie de base

À l'entrée et à la sortie du processus de traduction, l'unité de traduction est une simple *chaîne de caractères*. Les 256 caractères EBCDIC sont considérés comme atomiques (non structurés), et tous sont admis pour former les chaînes des langages spécialisés. Le blanc (X'40') est utilisé comme séparateur d'*occurrences*. Une unité de traduction est donc aussi une *suite d'occurrences*.

De la sortie d'AM à l'entrée de GM, l'unité de traduction est représentée par un arbre décoré. Chaque phase comporte une partie où le linguiste déclare le type de décoration, ou *jeu de variables* dans la terminologie d'Ariane-G5.

Une *variable élémentaire* (variable pour abrégé) est définie par un nom et un type (anonyme). Ce type est défini par un genre (exclusif, non-exclusif, arithmétique, lexical) et une liste de valeurs. Chaque variable a une valeur nulle dénotée par son nom suivi de "0".

- Une variable exclusive V définie par $V == (V1, V2, V3)$ prend ses valeurs dans $\{V0, V1, V2, V3\}$. On peut faire une analogie avec les types scalaires de Pascal, à ceci près que les identificateurs des valeurs sont ici locaux aux types.
- Une variable non-exclusive V définie par $V == (V1, V2, V3)$ prend ses valeurs dans les parties de $\{V1, V2, V3\}$. V0 dénote alors l'ensemble vide. On peut faire l'analogie avec les types ensembles de Pascal, avec la remarque précédente.
- Une variable arithmétique V définie par $V == (n)$, où n est un entier naturel non nul, prend ses valeurs dans :

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

- Il y a toujours une (et une seule) variable "lexicale", de nom UL, pour *unité lexicale*, qui est prédéclarée, et prend ses valeurs dans l'ensemble formé :
 - des valeurs prédéfinies " (UL0), 'ULTXT', 'ULFRA', 'ULSOL', 'ULOCC', 'ULMCP' ;
 - des valeurs introduites dans les composants linguistiques (essentiellement dans les dictionnaires) ;
 - des valeurs construites dynamiquement (par exemple pour les mots inconnus).

Une *décoration*, ou *masque de variables* en jargon Ariane-G5, est une combinaison de valeurs de toutes les variables du jeu de variables considéré, et est donc tout à fait analogue à une liste de propriétés en LISP.

On peut regrouper les variables de façon hiérarchique, le sommet de la hiérarchie étant prédéclaré jusqu'à un niveau dépendant du langage spécialisé. VAR dénote toujours le jeu de variables diminué de l'UL.

En ATEF, on distingue deux sous-ensembles, VARM et VARS, déclarés séparément, pour les variables dites "morphologiques" et "syntaxiques" (bien que, comme pour les phases, les linguistes ne respectent en général pas cette division, et ajoutent un bon nombre de variables de nature sémantique). VARM et VARS sont divisés en VAREM et VARNM, VARES et VARNS (exclusives et non-exclusives), car il n'y a pas de variables arithmétiques en ATEF.

En ROBRA et EXPANS, où les trois genres de variables sont possibles, le sommet de la hiérarchie est VAR (VARE π , VARN π , VARA π), où π est un caractère (redéfinissable dans DV) caractéristique de la phase. Par défaut, π vaut S, R, C, D, G pour AS, TL, TS, GS, GM, et X pour les autres phases EXPANS. Par exemple, en AS, pour raffiner VARE en variables syntaxiques et sémantiques, elles-mêmes divisées en propriétés et relations, on pourrait écrire :

-EXC- ** (mot-clé pour "exclusif").

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-exclusives.

...

Un *format* est un masque de variables constant auquel on a donné un nom, pour l'utiliser comme abréviation dans les dictionnaires et les grammaires. Un *arbre décoré* est un arbre orienté et ordonné dont chaque nœud porte une décoration.

2.3. Composants et variantes d'une phase

Comme dans la plupart des systèmes de TALN, les langages spécialisés sont organisés en *composants* physiquement distincts, pour des raisons de modularité et de taille. Les composants d'une phase forment un graphe acyclique de dépendance (pour la compilation).

- Une phase ATEF comporte deux composants de déclaration de variables (DVM, DVS), des formats morphologiques, syntaxiques et "généraux" (FTM, FTS, FTSG, ce dernier composant étant optionnel), 1 à 7 grammaires GR_i (1 ≤ i ≤ 7), 1 à 6 dictionnaires de "morphes" DIC_i (1 ≤ i ≤ 6), dont au moins un de "bases", et de 0 à 7 dictionnaires de tournures figées connexes DIC_i (7 ≤ i ≤ 14). FTM dépend de DVM, FTS de DVM et DVS, FTSG de FTS, les dictionnaires des formats, et les grammaires des dictionnaires.
- Une phase EXPANS comporte un composant de déclarations de variables (DV), un de procédures de condition et d'affectation sur les décorations (PROC), un de formats d'affectation (FAF), optionnellement un de "formats de conditions propres" (FCP), et de 1 à 7 dictionnaires (DIC_i). PROC, FAF et FCP dépendent de DV, et les DIC_i des précédents.
- Une phase ROBRA comporte DV, FAF, et de 1 à 7 grammaires (GR₁ à GR₇), avec les dépendances naturelles.
- Une phase SYGMOR comporte DV, FAF, PCP (procédures de conditions propres), GR_i (1 ≤ i ≤ 7), et DIC_i (1 ≤ i ≤ 13, un dictionnaire au moins étant accédé par l'UL), avec les mêmes dépendances qu'ATEF, sauf que les grammaires ne dépendent pas des dictionnaires.
- Une articulation TRACOMPL ne comporte qu'un composant, DV.

L'interface Ariane-G5 assure à tout instant la cohérence des tables internes en fonction de ces dépendances et des modifications effectuées par l'utilisateur.

Chaque phase peut donner lieu à des *variantes*, qu'on peut définir en fonction des types de textes à traduire.

- En ATEF et en SYGMOR, on choisit une des grammaires.
- En EXPANS, on choisit un sous-ensemble ordonné des dictionnaires, et le mode déterministe ou non du moteur.
- En ROBRA, on définit une liste d'au plus 14 grammaires à exécuter séquentiellement, la même pouvant apparaître plus d'une fois.

En combinant ces choix et le choix d'un chemin dans le graphe (de AM à GM pour une traduction), on obtient ainsi des *chaînes d'exécution* (pour la mise au point) et des *chaînes de production* (pour l'exploitation) qui sont elles aussi mémorisées et gérées par Ariane-G5.

3. Principes d'utilisation linguistique

Bien qu'Ariane-G5 ne propose ni n'impose aucune méthodologie de programmation linguistique, la plupart des utilisateurs suivent un certain nombre de principes, la plupart dus à B. Vauquois, qu'il peut être utile de mentionner ici brièvement.

3.1. Structures intermédiaires

Il est recommandé que l'analyse fournisse une *m-structure*, ou structure "multiniveau". La géométrie de l'arbre reflète l'organisation en groupes syntagmatiques, mais il s'agit d'un "arbre abstrait", dans lequel on ne retrouve pas nécessairement le texte de façon immédiate. Par exemple, il est commode de regrouper les constituants discontinus (ex: les garçons les ont tous vues), de "variabiliser" les négations, les auxiliaires, les articles, les prépositions fortement régies, certains modaux, etc. On arrive ainsi à des arbres beaucoup plus petits que des arbres "concrets" fournis directement par des grammaires hors-contexte étendues (GPSG ou autres). En principe, chaque nœud interne domine une feuille qui est le *gouverneur* du groupe ("head"), sauf si ce gouverneur est lui-même composé. Pour obtenir une structure de dépendances analogue à celles de l'école de Prague, il suffit, en première approximation, de "remonter" récursivement chaque gouverneur à la place de son père.

Les propriétés comme les relations sont codées dans les décorations des nœuds. Par exemple, un nœud portant comme valeur de fonction syntaxique attribut d'objet (SF=ATROBJ) est l'attribut du groupe dominé par celui de ses frères qui porte SF=OBJ1. Il y a donc deux niveaux syntaxiques, celui des classes (syntaxiques et syntagmatiques, X et X en notation chomskyenne), et celui des fonctions. Pour traduire vers des langues qui ne sont pas extrêmement proches sans devoir écrire de gros transferts de structures, on conseille de rajouter deux niveaux, logiques et sémantiques.

Le niveau logique (variable RL) donne les positions des arguments des prédicats linguistiques. ARG0 dénote le sujet logique (le plus souvent l'acteur) du prédicat gouverneur du même groupe, ARG1 son objet logique (en général le patient, sauf dans des constructions ergatives comme "la branche casse"), et ARG2 son 3ème argument. Par exemple, "la construction de la maison" et "construire la maison" auront une structure identique à ce niveau, le groupe "(de) la maison" étant ARG1.

TRL10 est mis à la place de ARG1 si le prédicat est attributif (être, sembler, paraître...), et TRL21 est mis à la place de ARG2 si le prédicat attribue ARG2 à ARG1 (considérer ARG1 comme TRL21). On note ainsi que la relation ne lie pas le nœud au gouverneur (le prédicat), mais à un autre argument. De même, on code souvent dans une autre variable (comme RLI, pour "relation logique inverse") la liaison entre les arguments dans les constructions avec contrôle. Par exemple, dans "je lui demande de venir", le groupe "de venir" est ARG1 de "demande", et porte RLI=00 si c'est l'ARG0 (moi) qui vient et RLI=02 si c'est l'ARG2 (lui).

On utilise enfin, essentiellement sur les circonstants (compléments faiblement régis) la relation sémantique (RS), qui correspond grosso modo aux "cas profonds" (localisation, origine, but, accompagnement, manière, qualification, mesure, cause, concession...). En pratique, RL et RS sont complémentaires, car il est extrêmement difficile (même à la main) d'affecter des RS aux arguments de façon fiable, et on ne peut correctement traduire les circonstants que si leurs RS sont connues.

À cet égard, le fameux problème de la traduction des prépositions est souvent mal posé. S'il s'agit d'un argument, c'est toute la construction qu'on traduit en bloc (prédicat+arguments, par exemple "parler de qc à qn", "compter avec qn"). S'il s'agit d'un circonstant, c'est la RS, éventuellement particularisée par la préposition (ou son absence). Ainsi, dans "venir par Lyon" et "venir via Lyon", le circonstant portera RS=LOC, SEM=ESPACE et SLOC=QUA (localisation dans l'espace, mouvement à travers qc), ce qui permet de traduire correctement "par" ("by" et non "with", "through" ou "out of"). Conserver la préposition permet aussi de traduire de façon plus exacte dans une langue comme l'anglais, qui en a aussi deux pour ce sens ("by", "via").

On procède également à plusieurs niveaux pour les variables d'actualisation, comme le nombre (morphologique et logique), le temps (tense/time), etc. L'ordre du texte est le plus possible reflété dans la structure. En effet, il donne des informations importantes qu'on sait mal formaliser, comme l'articulation thématique et l'emphase. Cela évite d'avoir à le noter explicitement dans une variable tactique.

Si on ne la regarde qu'aux niveaux "profonds", la m-structure d'une phrase peut représenter toute une famille de phrases de sens équivalents. Si on la prend en totalité, elle ne doit correspondre qu'à une seule phrase, à des variantes orthographiques près (comme cuillère/cuillère, paye/paie).

Outre ces différents niveaux de description linguistique, on encode aussi dans la m-structure produite par l'analyse les ambiguïtés non résolues, et les doutes sur des parties de construction, de façon à éviter une explosion combinatoire, et à pouvoir à la fois avertir le réviseur et essayer de transférer les ambiguïtés qui se conservent en traduction (ex: la conquête des Gaulois).

Le but du transfert est d'effectuer la traduction lexicale, et quelques adaptations de la structure destinées à proposer au générateur une structure cohérente avec le système linguistique de la langue cible, dite *g-structure*, ou structure génératrice. En principe, le générateur considère que la structure qu'il reçoit est sous-spécifiée par rapport aux niveaux de surface, et il les recalcule.

Ainsi, la première étape logique de la génération consiste à choisir une paraphrase du sens exprimé par la *g-structure*, en produisant la m-structure de la traduction à obtenir. La seconde consiste à produire un arbre de surface (arbre "concret" ou *s-structure*), en créant des nœuds pour les articles, les auxiliaires, les éléments de négation, les ponctuations, en divisant ou regroupant des phrases si nécessaire, etc. La troisième étape est la génération morphologique, qui, à partir de la liste des feuilles, fabrique les occurrences du texte final.

3.2. Organisation des dictionnaires

La notion d'unité lexicale est très utile pour l'étape de génération. Elle permet de représenter de façon compacte des familles dérivationnelles. Les dictionnaires usuels modernes utilisent une notion analogue. En analyse, cette notion permet de diminuer la taille des dictionnaires, et surtout de pouvoir traiter de façon systématique les néologismes obtenus par des dérivations productives.

Du point de vue linguistique, on est naturellement conduit à regrouper, par exemple, *démarrer*, *démarrreur*, *démarrage*, *démarrable*,... dans l'UL *démarrer-V*, les dérivations retenues ayant les 3 aspects sémantique, syntaxique et morphologique (ex: nom d'agent ou d'instrument en -eur), dans l'ordre d'importance. Du point de vue pratique, on sépare souvent d'une telle famille le nom d'agent ou d'instrument, car la dérivation en question ne peut être utilisée pour produire des paraphrases correctes en traduction ("démarrreur" → "ce qui fait démarrer" ?), et car cela permet de séparer l'indexage purement terminologique de ces termes de l'indexage plus complexe de toute une famille de déverbaux.

Selon la classe syntaxique du lemme principal (source des dérivations), on distingue les UL verbales, nominales et adjectivales. Il y a bien sûr des UL réduites à un seul terme (mots outils, adverbes non dérivés comme *là*, *ici*...).

En Ariane-78, on était contraint de représenter toute l'information lexicale dans les dictionnaires d'AM. En Ariane-G5, on a introduit des phases d'expansion lexicale qui permettent de répartir l'information. Par exemple, on peut suggérer d'utiliser AM pour passer des morphes (bases, affixes, désinences...) aux lemmes, AX pour passer aux UL, et AY pour traiter les tournures non figées ou non connexes (ex: verbes à particules séparables en allemand).

3.3. Organisation des grammaires

En ATEF et SYGMOR, c'est assez simple, quoiqu'il faille résister à la tentation de trop utiliser les fonctions heuristiques d'ATEF.

En ROBRA, le type de programmation est très différent selon qu'il s'agit d'analyse, de transfert ou de génération. En analyse, on commence par travailler en parallèle sur toute l'unité de traduction pour procéder à une normalisation de l'arbre (mots composés, ambiguïtés immédiatement solubles, regroupement de tournures prédites et trouvées, dates, noms propres, etc.). Ensuite, on provoque un appel récursif d'un sous-système transformationnel sur chaque phrase, puis éventuellement sur chaque sous-groupe sûr et assez gros, pour que la stratégie (parcours du graphe de contrôle) soit bien dirigée par les données. À la fin de l'analyse, on revient à un traitement de tout l'arbre, ce qui permet par exemple de tenter de trouver les référents des pronoms pour lesquels on n'en a pas trouvé à l'intérieur de la phrase où ils apparaissent.

En transfert, c'est assez simple. On traite les traductions de groupes complexes, on vérifie éventuellement les conditions contextuelles codées dans des sous-arbres représentant des traductions multiples, pour réduire les polysémies, puis on ajuste la g-structure, en y codant éventuellement des conseils ou des ordres au générateur en vue de produire telle ou telle forme syntaxique.

En génération, on travaille en descente récursive, pour produire la m-structure, puis pour commencer à construire la s-structure. On termine en général en traitant en parallèle l'affectation finale des variables d'actualisation de surface (accords, concordances), ce qui peut amener à modifier légèrement la géométrie (insertion d'auxiliaires avec traitement des clitiques). Il faut faire attention à ne pas utiliser de grammaires à la fois récursives et itératives, ce qui provoque de nombreux et coûteux appels de grammaires inutiles.

B. L'interface interactive

Une fois sous Ariane-G5, on peut :

- travailler sur les composants linguistiques (phases, articulations), dans les sous-environnements PRAM,... PRGM, PRAMAX,... PRGYGM (création, modification, compilation, listage...).
- travailler sur les textes (PRTXT), avec de multiples possibilités.
- travailler sur les chaînes d'exécution, et sur les chaînes de production.
- exécuter tout ou partie du processus de traduction pour le mettre au point (chaque phase reçoit des paramètres de trace et de sortie du résultat), à l'aide des chaînes d'exécution disponibles.
- produire des traductions brutes en mode d'exploitation, en utilisant une chaîne de production (pas de traces, paramètres uniquement pour la sortie du résultat final).
- effectuer la révision de traductions brutes (en multifenêtrage sous XEDIT, avec possibilité de passer sous THAM, qui offre des possibilités supplémentaires). Il est prévu de pouvoir également réviser les arbres produits par chaque phase, sous XEDIT, avec possibilité d'appeler TTEDIT, l'analogue de THAM pour les arbres.
- effectuer des traitements sur plusieurs phases à la fois (compilation, listage, effacement, duplication...).
- obtenir de nombreuses informations sur les objets gérés par Ariane (liste des langues sources et cibles, des corpus, liens entre langues sources et cibles, états de compilation...).
- lire ou modifier les valeurs des paramètres globaux (mode de dialogue, langues source et cible courantes, corpus courant...).

À tout moment, une aide en ligne est disponible, le plus souvent à deux niveaux de détail. Ce gestionnaire interactif est l'œuvre de M. Quézel-Ambrunaz [60]. Il représente environ 50.000 lignes d'EXEC2 et 100.000 lignes d'ASM370.

C. ATEF, un langage pour l'analyse morphologique

ATEF a été conçu en 1971 par J. Chauché [Chauché 75], qui a écrit le moteur, tandis que P. Guillaume et M. Quézel-Ambrunaz écrivaient les compilateurs des différents composants. Depuis, un certain nombre d'extensions ont été apportées, mais le modèle algorithmique sous-jacent n'a pas varié. Il s'agit en effet d'un outil très satisfaisant.

Le système traite successivement chaque occurrence du texte, en examinant *a priori* toutes les analyses possibles (mode non-déterministe total avec rétrogression). La forme courante est notée C. Un résultat d'analyse est une décoration ou une suite de décorations (cas des mots composés). Chaque étape d'une analyse particulière consiste à choisir un des dictionnaires ouverts, à y trouver un article dont la clé, ou *morphe*, soit préfixe (ou suffixe, en mode droite-gauche) de ce qui reste à analyser (noté A), qui est diminué d'autant, et à appliquer l'une des règles associées au format morphologique de l'article en question.

Les règles peuvent comporter des conditions portant sur l'état courant (C), sur les chaînes C et A, sur les résultats partiels produits par l'analyse en cours (PS1 à PS9) en cas de mot composé, et aussi sur les quatre occurrences précédentes (de P1 à P4) et sur leurs résultats d'analyse. Une forme particulière de condition consiste à donner une liste de "sous-règles" et à exiger que l'une au moins s'applique (comme une sous-règle peut avoir des sous-règles, on travaille en mode non-déterministe unaire avec rétrogression). Il est enfin possible de mettre en réserve une condition sur l'analyse de l'occurrence suivante.

Il existe trois types d'actions : affectation de valeurs au masque C, transformation de ce qui reste à segmenter (chaîne A), et appel à des fonctions spéciales. Ces fonctions permettent de :

- contrôler la rétrogression (backtrack) incorporée, en élaguant l'arbre des choix (fonctions FINAL, ARRET, ARD, ARF, STOP) ou en ouvrant et fermant les dictionnaires (par affectation de la variable obligatoire DICT) ;
- produire un résultat partiel à partir de l'état courant C (fonction SOL) ;
- transformer C ou A en une UL, tout en réduisant A à " (fonctions TRANS et TRANSA) ;
- décider qu'une borne de phrase est atteinte (fonction INIT).

En cas d'occurrence non reconnue ("mot inconnu"), c'est-à-dire si aucune analyse n'arrive à réduire A à " tout en produisant un état courant C muni d'une UL non vide, le système recommence l'analyse en attachant à l'occurrence le format morphologique obligatoire MODINC, qui doit en particulier appeler la règle obligatoire MOTINC, dite "règle du mot inconnu". Comme cette règle peut appeler des sous-règles et comme ce format peut appeler d'autres règles, on peut construire une vraie sous-grammaire du mot inconnu, et mettre en place des stratégies élaborées.

Au fur et à mesure du traitement, l'automate construit un graphe (quadrichrome arrière) dont les nœuds sont les masques (listes de masques pour les mots composés) associés aux solutions trouvées, et où les arcs indiquent la compatibilité des analyses entre elles. Le graphe final est ensuite transformé dans la forme souhaitée. Actuellement, la sortie en Q-graphes n'est plus disponible, et la sortie en graphe monochrome avant comme la sortie en arbre "avec homophrases" (tous les chemins du graphe quadrichrome sont présentés séparément) ne sont pas utilisées.

La sortie standard d'ATEF est un arbre "sans homophrases". La racine correspond à tout le texte, et porte UL='ULTXT'. Ses fils correspondent aux phrases (déterminées par la grammaire) et portent UL='ULFRA'. Sous chacun d'eux, on trouve les 'ULOCC' correspondant aux occurrences (mots ou tournures figées connexes). Sous chaque 'ULOCC', on trouve les différents résultats d'analyse morphologique de l'occurrence correspondante. Chaque résultat est, soit un masque de variables (un nœud), soit un sous-arbre de racine 'ULMCP' (mot composé) dominant les masques correspondant aux différentes parties reconnues dans ce mot.

D. ROBRA, un langage pour la transformation d'arbres décorés

ROBRA [Boitet, Guillaume & Quézel-Ambrunaz 78] est un langage d'écriture de systèmes transformationnels agissant sur des arbres décorés. C'est le successeur du langage CETA [Chauché 75]. De nombreuses extensions y ont été apportées, la sémantique a été précisée dans certains cas, et le moteur a été totalement respécifié et réécrit.

Un système transformationnel (ST) est défini par un graphe de contrôle (GC), un ensemble de grammaires transformationnelles (GT) et un ensemble de règles (RP pour "règles de production"). Une GT est un ensemble ordonné de règles. Un GC est un graphe dont les nœuds portent chacun une GT ou un symbole de sortie (&NUL) et dont les arcs portent des conditions d'arbres. Notons ici qu'un composant "grammaire" (GRi) d'une phase ROBRA contient en fait un système transformationnel, qui peut être constitué d'un grand GC avec des dizaines de GT.

Pour exécuter un ST sur un arbre objet (AO), on utilise le GC comme structure de contrôle non déterministe (unaire avec rétrogression) : partant d'un nœud initial, on cherche le premier chemin licite menant à une sortie. Sur ce chemin, on exécute les grammaires contenues dans les nœuds, et, pour pouvoir traverser un arc, il faut que l'AO courant vérifie la condition qu'il porte.

L'exécution d'une GT est faite d'une application élémentaire en mode unitaire (U), ou de plusieurs en mode itératif (E pour "exhaustif"). Dans une application élémentaire, on applique en parallèle le plus possible de règles de la GT, d'où un mécanisme de résolution de conflits. Une application élémentaire n'est terminée que lorsque les éventuels appels récursifs de sous-grammaires (SGT) ou de sous-systèmes (SST) provoqués par l'application de certaines règles de la GT sont terminés.

Un système d'interdictions (règles marquées, points bloqués) permet de tester statiquement la décidabilité du ST : le compilateur peut prévenir l'utilisateur des risques d'indécidabilité (boucle dans le GC, mode "libre" dans une GT itérative, contrainte de récursion non respectée, etc.).

Les schémas qui apparaissent en partie gauche de règles ont une très grande puissance d'expression. Pour chaque nœud, on peut indiquer si l'on recherche ses fils le plus à gauche ou le plus à droite possible, dans l'ordre ou dans le désordre. On peut rechercher des nœuds à des profondeurs inconnues à l'avance au moyen de "nœuds généralisés". Enfin, il s'agit de règles sous-contexte, la racine du schéma (RS) pouvant ne pas être confondue avec la racine de la transformation effective (RT). Tout ce qui n'est pas dominé par la RT forme le contexte, ou "chapeau". La RT peut être active ou contextuelle. Tout ce qu'elle domine est actif.

La notion de réécriture parallèle en ROBRA est assez forte, puisqu'on peut avoir un parallélisme "normal" (RT disposées sur des nœuds distincts d'une coupe de l'AO), "vertical" (une RT peut en dominer une autre), et "horizontal" (plusieurs RT contextuelles et au plus une RT active peuvent s'instancier sur le même nœud de l'AO).

Enfin, on peut écrire en partie droite de règle des affectations conditionnelles des variables extrêmement complexes, ce qui contribue à faire de ROBRA un outil extrêmement puissant.

ROBRA est bien un modèle de substitution, même si, dans l'implémentation, l'exécution élémentaire d'une grammaire transformationnelle se fait par transduction d'un arbre en entrée vers un arbre en sortie. C'est pourquoi le type de décoration est nécessairement conservé.

E. EXPANS, un langage pour l'expansion et le transfert lexical

EXPANS [Guillaume 1989b] est fondé sur un modèle de transduction d'arbre décoré. Les arbres d'entrée et de sortie peuvent être sur deux jeux de variables différents. Chaque nœud produit un sous-arbre dans l'arbre image. Ce sous-arbre est déterminé par consultation des dictionnaires, dans leur ordre de priorité, selon l'UL portée par le nœud. Une action par défaut est toujours prévue.

Un article de dictionnaire contient comme clef une valeur d'UL, et comme contenu une liste de triplets <condition,image,affectations>. Les conditions portent sur le nœud de l'arbre origine et éventuellement sur ses voisins immédiats (père, frères gauche et droit). L'image exprime la géométrie du sous-arbre à produire, et les affectations permettent de calculer les valeurs des variables des nœuds du sous-arbre produit en fonction de celles des nœuds sources accessibles.

Au niveau d'un dictionnaire, si l'UL du nœud source est trouvée, on recherche le premier triplet dont la condition est vérifiée (la dernière condition doit être vide, c'est-à-dire identiquement vraie), et on produit le sous-arbre correspondant. Sinon, il y a échec.

En mode déterministe, on parcourt les dictionnaires dans leur ordre de priorité jusqu'à obtenir un succès. En mode non-déterministe, on parcourt tous les dictionnaires, dans l'ordre, et on produit comme image un sous-arbre construit en enracinant les sous-arbres produits par les différents dictionnaires sous un nouveau nœud. Ce mode permet par exemple de ne pas "cacher" la traduction usuelle d'un mot qui a aussi une traduction différente dans un domaine particulier dont le dictionnaire serait prioritaire.

F. SYGMOR, un langage pour la génération morphologique

SYGMOR est fondé sur un modèle de transducteur d'états finis déterministe. Sa première version a été conçue par B. Thouin et programmée par D. Jaeger. [Guillaume 1989a] présente les améliorations et extensions qu'il y a apportées ces dernières années. SYGMOR prend en entrée une suite de décorations et produit en sortie une chaîne de caractères. On dispose d'un contexte réduit aux décorations courante (C) et précédente (P), et on travaille sur deux chaînes, la chaîne de travail (T) et la chaîne en sortie (S).

La grammaire est de structure assez simple. Chaque règle comporte des conditions, des actions et des "règles suivantes". Les actions consistent essentiellement à écrire à droite, à gauche ou au milieu (dernier point de concaténation) de T une chaîne littérale ou le résultat de la recherche dans l'un des dictionnaires à partir de la valeur de l'une des variables de C (les dictionnaires accédés par les UL donnent les bases, les autres les affixes). On peut aussi modifier C, et "rappeler" S dans T (en la concaténant à gauche et en l'effaçant).

Pour chaque décoration, SYGMOR recherche la première règle applicable, et l'applique. Il exécute alors dans l'ordre les règles suivantes, sans tenir compte de leurs parties "règles suivantes". La liste peut comporter des règles facultatives. Si une règle obligatoire échoue, on revient à l'état initial et on applique, si elle est présente, la règle MOTINC, et sinon l'action d'erreur (vider S, puis S:=T). On continue en prenant en compte la partie "règles suivantes" de la dernière règle appliquée.

G. TRACOMPL, un langage pour la transformation de décorations ("articulations")

TRACOMPL [Guillaume 1989b] est le sous-langage d'écriture de tous les composants DV. On en a fait un langage autonome pour pouvoir écrire des "articulations". Il s'agit de transformer une décoration d'un Jeu1 vers un Jeu2. Pour cela, on procède en deux étapes :

- On décrit d'abord le Jeu2 et ce que l'on doit connaître du Jeu1 pour réaliser la transformation. Les noms des variables présentes dans les deux sont préfixés par "\$", et ceux des variables présentes dans Jeu1 et non reprises dans Jeu2 par "\$\$". Les autres sont considérées comme nouvelles.
- On complète cela en écrivant (partie CVAR) une action conditionnelle qui peut tester les variables de la décoration source en Jeu1, en Jeu2 après le "reformatage" décrit par la partie précédente, et en Jeu2 dans leur état courant. À l'aide de cette action, on peut par exemple transformer une variable à 2 valeurs en une variable à 3 valeurs, ou inversement.

II. Outils associés à Ariane-G5

A. Aides à la construction de systèmes de TAO

1. ATLAS, un système d'aide à l'indexage dans les dictionnaires codés

Il s'agit d'un langage d'écriture de "cartes d'indexage", spécifié et implémenté par D. Bachut [Bachut & Verastegui 84]. Il a été utilisé pour produire de nombreux manuels d'indexage du système russe-français. On décrit un graphe acyclique dont les nœuds internes portent des questions, les arcs les réponses possibles, et les feuilles les résultats auxquels on parvient (le plus souvent, des noms de formats ou de procédures).

Ce graphe peut être dessiné, pour fabriquer des manuels sur papier, ou être utilisé dynamiquement, pour créer des enchaînements de menus à l'écran dans une fenêtre, et envoyer les résultats aux endroits appropriés dans une seconde fenêtre où défile le dictionnaire à construire ou à compléter.

2. VISULEX, un outil de visualisation synthétique des informations lexicales contenues dans un linguiciel écrit en Ariane

VISULEX a été produit par [Bachut & Verastegui 84], dans le cadre du projet ESOPE de l'ADI. Ce système permet de visualiser tout ou partie des informations lexicales d'un système écrit en Ariane-78 ou en Ariane-G5, à deux niveaux (codes et commentaires), ce qui évite aux linguistes d'avoir à chercher dans de multiples fichiers.

3. BDTAO, un gestionnaire de bases de données lexicales pour la TAO

Il s'agit d'un système de gestion de bases de données lexicales spécifiques à la TAO, mais non à Ariane, conçu et développé par D. Bachut et R. Gerber de B'VITAL. Les formats et les dictionnaires d'analyse et de génération d'une langue sont construits automatiquement par BDTAO à partir de la sous-base monolingue concernant cette langue. Ainsi, on évite d'avoir à indexer plusieurs fois le même terme dans plusieurs dictionnaires. Pour le transfert, il y a une sous-base par couple de langues.

On distingue le dictionnaire "noyau", qui fait partie du système grammatical de la langue et est codé directement, le dictionnaire général, et les dictionnaires terminologiques. En transfert, la partie générale est beaucoup plus complexe que la partie terminologique, et la construction des dictionnaires Ariane correspondants n'est pas encore automatisée (en attendant, on se sert donc de manuels d'indexage usuels, et on indexe directement dans les dictionnaires Ariane).

4. TTEDIT, un éditeur transformationnel d'arbres décorés

Pour développer indépendamment des analyseurs, des transferts et des générateurs, il faut disposer de jeux d'essais. [Durand 88] a développé pour cela TTEDIT, qui se présente comme un éditeur d'arbres (simplement étiquetés ou décorés), dont l'originalité est que ses opérations de base sont des transformations de sous-arbres, comme en ROBRA, et non des manipulations directes sur les nœuds ou les arcs. Cela permet de travailler sur de gros arbres comme on travaille sur des textes avec des éditeurs munis de fonctions de recherche et remplacement fortement paramétrées. TTEDIT est complètement intégré à l'éditeur standard de textes XEDIT. Comme en XEDIT, on peut écrire des "macros", mais il s'agit ici de grammaires transformationnelles analogues à celles de ROBRA.

[Guilbaud 88] a utilisé TTEDIT pour, à partir d'un arbre produit par l'analyseur du français de B'VITAL et transformé par un transfert réalisé pour l'occasion (en Ariane), réaliser les transformations complémentaires *ad hoc* permettant de produire une structure interface d'analyse conforme à la "législation linguistique" du projet Eurotra et... aux sentiments personnels des responsables pour chaque cas particulier non prévu par cette "législation".

C. Outils textuels

1. THAM

THAM signifie "Traduction Humaine Aidée par la Machine". Programmé en EXEC2/XEDIT, THAM fonctionne comme une macro de XEDIT [Bachut & Verastegui 84]. Supposons que l'utilisateur d'Ariane soit en train de réviser une traduction automatique. Par une simple pression sur une clé, il met en route THAM, qui lui permet d'accéder à un dictionnaire "main gauche" modifiable dynamiquement, en sus du texte révisé, de la traduction brute et de l'original. THAM peut aussi être utilisé de façon autonome.

Il ne s'agit pas là d'un "poste du traducteur/réviseur" complet, comme ceux offerts par certaines firmes (Weidner, Alps...), mais d'une extension utile à XEDIT. En fait, pour une utilisation industrielle, il est bien préférable de réviser les traductions sur un PC, sur un MacIntosh, ou, solution de luxe, sur un Documenteur™ de Xerox.

Plutôt que de développer nous-mêmes des environnements complets pour traducteur/réviseur, ce qui est un travail très lourd, nous avons préféré coopérer avec d'autres équipes sur certains aspects de leurs outils (adaptation en cours du poste SISKEP anglais-malais de l'USM au français-malais, développement d'un outil d'intégration extensible de lemmatiseurs et de dictionnaires, O_DI_LE, en collaboration avec l'équipe TRILAN du LGI, auteur du système PILAF, et avec WinSoft, auteur de l'outil dictionnaire WinTool™). Avec O_DI_LE [Tomasino 90], on pourra sélectionner sous un traitement de textes, un tableur ou un SGBD du commerce une chaîne comme "tiendrons compte" et accéder à un dictionnaire par la clé "tenir compte", automatiquement générée à partir du résultat du lemmatiseur obtenu par adaptation de PILAF.

2. LT, un langage d'écriture de transcripateurs

LT [Lepage 86] permet d'écrire rapidement des transcripateurs de textes. Par exemple, on utilise dans le système russe-français une transcription des textes russes dans un sous-ensemble du jeu de caractères de PL/I (majuscules, chiffres et quelques signes spéciaux). Un transcripateur écrit en LT permet d'obtenir les textes en cyrillique (non supporté par la configuration disponible), avec majuscules, minuscules et formatage minimal, sur une imprimante ASCII (S700) munie de fontes cyrilliques et connectée au 4361 via un PCI.

Le modèle abstrait est un transducteur d'états finis à deux bandes. La bande d'entrée est munie de deux têtes de lecture. La première ne peut qu'avancer, et la seconde, qui sert de "regard en avant", peut avancer ou être rappelée à la position de la première. La bande de sortie est munie d'une tête d'écriture. Les états sont structurés : un état complet est la combinaison d'un état élémentaire et d'un ensemble de valeurs d'un certain nombre de variables.

Écrire un transcripateur consiste à déclarer les variables et leurs types (par exemple, fonte, langue, longueur du regard en avant...), puis à décrire le graphe du système de transition, avec un nœud par état élémentaire. Les arcs portent les transitions, qui sont de classiques règles de production. Il y a un certain nombre de fonctions prédéfinies, ainsi que des possibilités de paramétrage et de factorisation, pour éviter d'avoir à écrire un trop grand nombre de règles.

3. SCRIBERE/SCRIBM, une extension au formateur SCRIPT/DCF

SCRIBERE permet de décrire le contenu et la structure logique d'un document au moyen de macros écrites en SCRIPT/DCF d'IBM. Cet outil offre des possibilités du genre de celles de SGML. Il a été développé par D. Bachut et N. Verastegui pour suppléer à l'absence de GML, décidément trop onéreux. On en a profité pour prendre en compte certains aspects de nature linguistique (langue courante, transcription courante, etc.). SCRIBM a été écrit par R. Zajac à partir de SCRIBERE pour l'enrichir sous différents aspects (notes, matrices...).

Les problèmes de représentation et de traitement de textes multilingues ne se découvrent que petit à petit, mais sont considérables. Nous espérons, en participant à l'action internationale TEI sur l'encodage des textes, trouver des solutions réellement satisfaisantes, et utilisables dans des systèmes de TAO multilingue.

III. Vers un véritable génie logiciel

A. Bref historique

1. Le prototype préopérationnel russe-français (1980—87)

C'est en développant le système russe-français depuis le stade de maquette à celui de réel prototype, utilisé de façon "préopérationnelle" (flux de textes envoyés régulièrement et à renvoyer traduits sous certains délais, dictionnaires à améliorer en liaison avec un spécialiste de technique et de traduction) que nous avons vu la nécessité de développer des outils d'aide à la programmation linguistique comme ATLAS et VISULEX. Dès le début des années 80, il est devenu évident que ce type de programmation pourrait et devrait être comparé à la programmation de gros systèmes informatiques. À titre d'exemple, Ariane-78 a demandé environ 30 hxa de travail, et le système russe-français environ 80.

Ce prototype, réalisé par N. Nédobejkine et ses collaborateurs, n'est pas extrêmement spécialisé à un certain type de textes. Son vocabulaire, d'environ 30.000 termes (mots simples, mots composés, tournures), contient 5 à 6.000 termes généraux, le reste se partageant entre différents domaines scientifiques et techniques (sciences de l'espace et de la Terre, métallurgie, aéronautique, linguistique...). Au départ, on ne savait pas clairement s'il s'agirait de TAO du veilleur ou de TAO du réviseur. Il s'est avéré que la qualité obtenue était assez bonne pour qu'on puisse réviser une page en environ 1/4h, un temps tout à fait comparable à celui de la révision d'une traduction brute humaine — bien sûr, les fautes ne sont pas les mêmes, et on peut utiliser la puissance de l'éditeur pour accélérer le travail. Cependant, pour ce couple de langues, la seule utilisation réaliste en France viserait la veille scientifique. Encore faudrait-il avoir accès aux textes sur support magnétique, par bandes ou par réseau.

2. Les analyseurs de l'anglais et les maquettes de traduction (1983—89)

La méthodologie de construction de systèmes de TAO en Ariane doit beaucoup à diverses coopérations à travers lesquelles B. Vauquois a pu essayer diverses méthodes et dégager les principes évoqués plus haut. Cela a commencé, dès 73-74, par des travaux sur l'analyse du français, en liaison avec le SFB/100 de Saarbrücken (J. Weissenborn, E. Stegentritt), puis du portugais (P. Daun Fraga). Les méthodes mises au point sur le français furent reprises et améliorées sur le portugais, puis de nouveau reprises et affinées sur le français, dans le cadre d'une étude sur du français-anglais pour les Télécoms (Vauquois, Guilbaud, Dymetman) en 81-82.

L'analyse de l'anglais devint ensuite le point commun de plusieurs études faites en coopération. La plus importante mena à un prototype anglais-malais [Tong 86]. D'autres aboutirent à des maquettes vers le chinois [Yang 81], le thaï, et enfin l'arabe [Moneimne 89]. Cette analyse fut aussi le point de départ des travaux sur un "analyseur standard" de l'anglais entrepris durant le PN-TAO et poursuivis quelque temps ensuite.

L'analyse de l'anglais étant plus délicate que celle du français, la nécessité d'une spécification "statique" des grammaires "dynamiques" d'analyse et de génération apparut à B. Vauquois dès 1980. [Chappuy 83] et [Vauquois & Chappuy 85] en présentent les aspects formels et pratiques. Une méthodologie de spécification et d'implémentation d'analyseurs et de générateurs à partir de grammaires statiques fut définie durant le projet ESOPE de l'ADI, à l'occasion de la construction d'une maquette pédagogique anglais-français (BEX-FEX).

3. Le système opérationnel français-anglais de B'VITAL

Le travail linguistique sur le français-anglais a été poursuivi par la petite société B'VITAL, fondée lors du PN-TAO, et continue aujourd'hui, dans le cadre d'une action du Ministère de l'Industrie, au niveau de SITE, la plus grosse société européenne de documentation et de traduction technique, dont B'VITAL est devenue filiale. La grammaire statique du français comptait en 1986 environ 150 planches de construction et 450 planches de désambiguïsation (règles de préférence). L'AS comporte aujourd'hui plus de 20.000 lignes sources (en ROBRA). Les dictionnaires comportent près de 20.000 termes, plus de la moitié appartenant au vocabulaire terminologique.

B. Grammaires statiques et bases lexicales

Au départ, les grammaires statiques étaient sur papier. [Yan 87] présente un premier type d'informatisation, sous forme d'un environnement sur MacIntosh, qui intègre divers outils disponibles pour mettre la même information sur machine.

[Zaharin 86] avait travaillé dans sa thèse sur la sémantique formelle des grammaires statiques, et avait proposé quelques améliorations. Depuis, il a produit avec son équipe SaGE, une application sur MacIntosh qui permet de calculer sur une grammaire statique. On n'a pas encore une spécification exécutable, mais SaGE peut fabriquer automatiquement un analyseur et un générateur à partir d'une telle grammaire, selon les principes généraux vus plus haut. Il ne s'agit encore que de travaux préliminaires (en particulier, les systèmes transformationnels produits sont loin d'être optimisés), mais encourageants.

Enfin, il serait hautement souhaitable de développer des bases lexicales non plus spécifiques à la TAO, mais à usages multiples. Diverses études ont été entreprises [Boitet & Nédobejkine 86], allant jusqu'au prototypage sur une base trilingue en télécommunications (français-anglais-japonais). De nombreux problèmes se posent encore au niveau de la définition du contenu linguistique de telles bases, de leur structure logique, et de leur implémentation, aucun type de SGBD du commerce ne semblant bien adapté. Nous participons à plusieurs projets dans ce domaine, notamment à l'action "lexiques" du PRC-CHM (G. Perennou).

C. Exemples commentés de traductions français-anglais

Ces exemples sont tirés de traductions brutes produites par le système B'VITAL/aéro/F-E lors de démonstrations à TEC-88.

Après essai, s'assurer du fonctionnement correct de l'ensemble raccord

After test, check that the coupling assembly works correctly.

On remarque ici le passage d'un groupe nominal prépositionnel "du fonctionnement correct" à un groupe verbal, "that... works correctly", avec pour corollaire le passage de l'adjectif "correct" à l'adverbe "correctly". Ces transformations ne sont pas effectuées au transfert. C'est la première étape de la génération syntaxique qui, à partir de la "g-structure" (structure génératrice), considérée comme sous-spécifiée relativement aux fonctions syntaxiques et aux classes syntagmatiques et morphosyntaxiques, recalcule ces niveaux en fonction de l'objectif initial (ici, construire une phrase verbale), à partir des niveaux plus profonds (relations logiques à l'intérieur du cadre prédicatif strict, relations sémantiques pour les compléments circonstanciels).

Grâce à la notion d'unité lexicale, le générateur sait, sans avoir besoin de consulter un dictionnaire, quels lemmes contient la famille dérivationnelle considérée, ce qui conditionne les paraphrases possibles. Ici, "fonctionnement" a été ramené à l'UL "fonctionner-V", traduit par "work-V", qui porte la potentialité de dérivation vers un nom d'action. C'est donc simplement l'ordre de préférence des règles de choix des catégories syntagmatiques qui provoque la construction d'une subordonnée plutôt que d'un groupe nominal ("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” est ici un verbe support, et “porter une date” est traduit par “to write a date” et non par “to carry a date”, grâce à un test effectué en transfert lexical sur les traits syntaxiques et sémantiques de l’argument 1 de “porter” (l’objet logique).

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” est traduit par le verbe simple “to drain”, grâce à la notion d’unité lexicale, et à l’organisation du transfert lexical. “Vidange” est ramené à “vidanger-V”, et cette UL donne en traduction un arbre dans lequel on code la possibilité de la présence d’un verbe support du genre de “effectuer”, “faire”, etc., verbe dont la traduction sera effacée au cours du transfert structural.

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” est reconnu comme un prédicat composé, avoir-but-V(x0,x1), qui est traduit par use-V(x1,x0), avec conversion d’arguments, ce qui explique la génération d’un passif.

Enduire légèrement le joint neuf de liquide d’utilisation.

Slightly coat the new joint with operating fluid.

La traduction des prépositions est toujours délicate. Il faut savoir si elles introduisent des arguments ou des circonstants. Ici, “enduire-V” est un prédicat à 3 arguments (qn enduit qn/qc de qc), le troisième étant introduit par “de”. L’analyseur préfère compléter le cadre argumentaire, et l’introducteur de cet argument pour “coat-V” est “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.

La préposition “jusqu’à” introduit ici deux circonstants. Ce qu’on traduit en fait, c’est la relation sémantique (ici, RS=LOC avec SEM=TEMPS et SLOC=QUA), précisée par la préposition et par les traits sémantiques du gouverneur (“head”) du groupe, ici PROCESSUS pour “allumer-V”, et du prédicat (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.

Aucune transformation explicite n’est effectuée. La proposition infinitive est rendue par une subordonnée par le simple fonctionnement du générateur, expliqué plus haut. Comme l’argument 0 (sujet logique) n’est pas exprimé, on génère un passif. Il s’agit d’une préférence de style, et on pourrait aussi bien générer “until one obtains a pressure of 9 bars”, ou “until obtaining...”, comme dans l’exemple suivant.

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.

Ici, la construction préférée pour la conjonction “before” est le gérondif. D’autre part, la préposition “à” introduit l’argument 1. Dans la m-structure produite par l’analyse, elle peut fort bien avoir été supprimée. “With” est contenu dans le cadre de valence de “proceed-V” pour la même position argumentaire, et est fabriqué par le générateur.

Conclusion

Nous espérons avoir donné ici un aperçu des outils informatiques et des méthodes linguistiques développés à Grenoble en vue de construire des systèmes de TAO du réviseur. Bien que les aspects de TAO du traducteur (poste de travail informatisé du traducteur ou du réviseur) aient été à peine abordés, il s’agit d’un complément indispensable aux outils de traduction automatique, et de nombreux industriels y travaillent. C’est d’ailleurs pourquoi cet aspect a été relativement peu développé au GETA.

Pour l’avenir, les progrès essentiels en TAO du réviseur devraient venir du côté du génie logiciel. Cependant, le développement de nouvelles recherches en vue de traduire des textes bruités (lecture optique ou dialogues parlés) commence à faire émerger de nouvelles idées algorithmiques et devraient provoquer l’apparition de nouveaux langages spécialisés.

Enfin, la grande diffusion de micro-ordinateurs de plus en plus puissants permet d’envisager l’idée de TA personnelle, fondée sur le couplage des méthodes que nous venons de voir avec un dialogue entre le système et l’auteur du document à traduire. Il faudra pour cela imaginer une architecture logicielle tout à fait nouvelle, et sans doute revoir une bonne partie des méthodes linguistiques, en particulier en ce qui concerne la désambiguïsation, puisque la postédition directe par un professionnel (le réviseur) devra être remplacée par une prédiction indirecte (par l’auteur).

-0-0-0-0-0-0-0-0-0-

Références

- [1] ABOU 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: ingénierie 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., QUÉZEL-AMBRUNAZ M. (1978) *Manipulation d'arborescences et parallélisme: le système ROBRA*. Proc. COLING-78, Bergen.
- [21] BOITET Ch., GUILLAUME P., QUÉZEL-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., QUÉZEL-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, Taipei, 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., GUÉRARD 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] CHAUCHÉ 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 français, 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. (1989) *Ariane-G5 - Extensions apportées au langage SYGMOR*. GETA, janvier 1989, 6 p. (Ariane-G5 version 3)
- [42] GUILLAUME P. (1989) *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 Correspondanciennes 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) *ODJLE, 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 automatisée 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.

-0-0-0-0-0-0-0-0-0-