Heloise — A reengineering of Ariane-G5 SLLPs for application to π-languages

Vincent Berment[1] *Christian Boitet*[2]
(1) INaLCO, Place du Maréchal de Lattre de Tassigny - 75775 Paris cedex 16
(2) GETALP, LIG-campus, 385, avenue de la Bibliothèque - 38041 Grenoble cedex 9
Vincent.Berment@imag.fr, Christian.Boitet@imag.fr

ABSTRACT

Heloise is a reengineering of the specialised languages for linguistic programming (SLLPs) of Ariane-G5 running both Linux and Windows. Heloise makes the core of Ariane-G5 available to anyone willing to develop "expert" (i.e. relying on linguistic expertise) operational machine translation (MT) systems in that framework, used with success since the 80's to build many prototypes and a few systems of the "multilevel transfer" and "interlingua" architecture. This initiative is part of the movement to reduce the digital divide by providing easily understandable tools that allow the development of lingware for poorly-resourced languages (π-languages). This paper shows how Heloise can contribute to the democratisation of quality MT, describes some technical aspects of it, and provides elements of comparison with Ariane-G5.

KEYWORDS: machine translation, specialised languages for linguistic programming, SLLP, MT lingware, online lingware building, collaborative lingware building, Ariane-G5, Ariane-Y, Heloise, under-resourced languages

TITRE ET RÉSUMÉ EN FRANÇAIS

Héloïse — une réingénierie des LSPL d'Ariane-G5 pour application aux langues-π

Héloïse est une réingénierie des langages spécialisés (LSPL) d'Ariane-G5 tournant sous Linux et Windows. Héloïse rend le cœur d'Ariane-G5 accessible à toute personne désirant réaliser par elle-même des systèmes de traduction automatique (TA) experts (s'appuyant sur une expertise linguistique) opérationnels dans cet environnement, qui a été utilisé avec succès depuis les années 80 pour construire de nombreux prototypes et quelques systèmes adoptant une architecture de "transfert multiniveau" et d'"interlingua". Cette démarche s'inscrit dans le mouvement visant réduire la fracture numérique par la mise à disposition d'outils facilement appropriables, et permettant de développer des linguiciels pour des langues peu dotées (langues-π). Cet article montre comment Héloïse peut participer à la démocratisation de la TA de qualité, en décrit quelques aspects techniques, et donne des éléments de comparaison avec Ariane-G5.

MOTS-CLÉS EN FRANÇAIS : traduction automatique, langages spécialisés pour la programmation linguistique, LSPL, linguiciels de TA, construction collaborative de linguiciels, Ariane-G5, Ariane-Y, Heloise, langues peu dotées.

1 Héloïse : des compilateurs de LSPL et des interfaces utilisateurs

1.1 Compilateurs

Fonctionnellement, Héloïse offre un service équivalent aux compilateurs d'Ariane-G5 : il permet de transformer des linguiciels en programmes exécutables. Contrairement à Ariane-G5, Héloïse produit cependant un exécutable qui est l'image des linguiciels, ne recourant pas à des interpréteurs, supposés trop lents, bien que pouvant présenter des avantages comme la facilité de réaliser un débogueur symbolique. Les compilateurs d'Héloïse réalisent successivement :

- un arbre d'analyse à partir des différentes parties des linguiciels (déclarations de variables, formats, dictionnaires, grammaires, procédures...),
- une structure d'interprétation à partir de chacun de ces arbres ou directement une base de données dans le cas des dictionnaires,
- du code C++ réalisant la part d'algorithme associé,
- la compilation du code de la phase et son édition de liens.

Héloïse fait appel à deux bibliothèques ouvertes :

- `saint-jean` (Claude Del Vigna), un générateur d'analyseurs syntaxiques dans le formalisme duquel ont été écrits les analyseurs des LSPL et des arbres décorés,
- `sqlite` (http://www.sqlite.org/), un gestionnaire de bases de données léger, utilisé pour les dictionnaires.

Les traces d'exécution d'Héloïse et d'Ariane-G5 sont strictement équivalentes, sans amélioration ni modification des fonctionnalités d'Ariane-G5. Les compilateurs d'Héloïse présentent cependant quelques limitations par rapport à ceux d'Ariane-G5 ainsi que quelques différences.

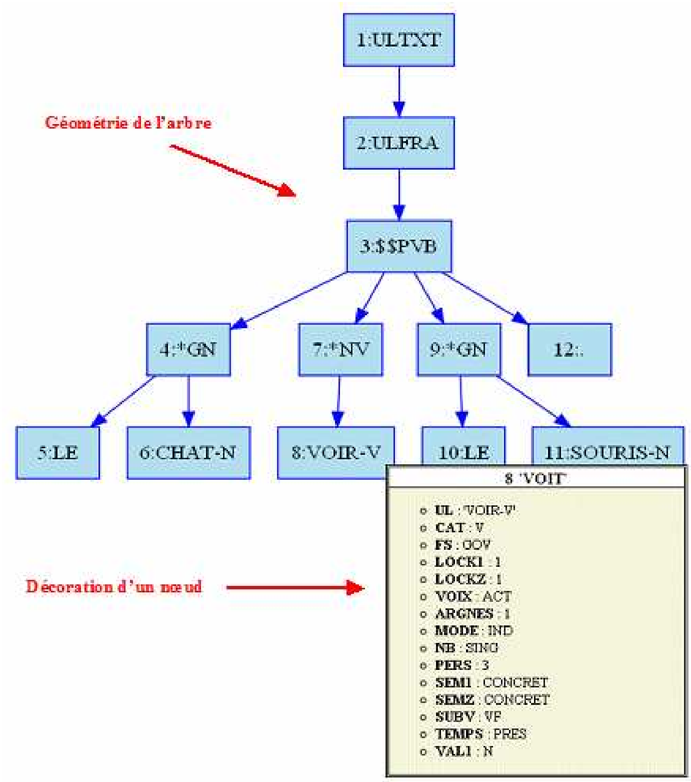
- Il y a quatre compilateurs, dans Héloïse, et non dix (REFORM/TRACOMPL y est traité comme un cas particulier d'EXPANS qui n'a que les fichiers spécifiquement TRACOMPL ; voir [Boitet et al, 1985] et [Guillaume, 1989]).
- Le LSPL ATEF d'Héloïse est limité aux sorties sans homographes (voir [Boitet, 1982]).
- L'ensemble constitué des quatre compilateurs, de la bibliothèque HCL et du moniteur Win32 est écrit en C++, l'interface Web est écrite en HTML, AJAX et PHP, et fait appel à des petits programmes C/C++ pour le calcul des arbres affichés en SVG.
- Les compilateurs et les programmes compilés fonctionnent à la fois sous Windows et sous Linux.
- Le traitement des erreurs est (provisoirement) assez limité dans Héloïse.

1.2 Interfaces utilisateurs

L'interface utilisateur est simplifiée (mais plus moderne) par rapport au moniteur Ariane. Il existe deux versions d'Héloïse : une application Win32 et un service Web.

L'application Win32 a été développée sous Visual C++. En haut à gauche de l'application, un champ de saisie est destiné au texte à traduire et un autre à la traduction. Dans la partie droite, un premier onglet concerne la traduction et permet, en particulier, de visualiser les phases à exécuter. Les autres onglets sont spécifiques aux différentes phases et permettent de gérer les fichiers, lancer les compilations et visualiser les résultats et les traces. Le couple de langues est choisi dans une liste déroulante située dans le bas de l'application.

L’interface Web est agencée différemment de l’interface Windows mais offre les mêmes fonctionnalités. On retrouve la gestion de la traduction et les deux champs de saisie dans la partie droite, la sélection du couple de langues en haut à gauche, et la gestion des phases (compilation, traces…) en bas à gauche. Un affichage graphique des arbres de sortie au format SVG est aussi disponible dans cette interface. La copie d’écran ci-dessous montre l’arbre (technologie SVG) obtenu par Héloïse avec le linguiciel FR3 d’analyse du français pour la phrase « *Le chat voit la souris.* ». L’encadré, qui s’affiche quand la souris passe sur un nœud, présente les décorations portées par le nœud (ici, le nœud correspondant au mot « *voit* »).



8 'VOIT'

- **UL** : 'VOIR-V'
- **CAT** : V
- **FS** : GOV
- **LOCK1** : 1
- **LOCK2** : 1
- **VOIX** : ACT
- **ARGNES** : 1
- **MODE** : IND
- **NB** : SING
- **PERS** : 3
- **SEM1** : CONCRET
- **SEM2** : CONCRET
- **SUBV** : VF
- **TEMPS** : PRES
- **VAL1** : N

FIGURE 1 – Visualisation de la géométrie et des décorations dans Héloïse

2 Introduction

Ariane-G5 is a generator of machine translation systems developed and improved by the GETA group[1] during the years 1970 and 1980. This framework, despite the numerous publications and cooperative projects that made it widely known, remains of difficult access because of the "mainframe" environment under which it runs (zVM/CMS on z390). Ariane-G5 can be accessed either natively through a 3270 terminal emulator or using CASH, a portable "meta-environment" (written in Revolution) which contains the source files (lingware, corpus), and which communicates with Ariane-G5 that performs all the treatments (compilations and executions of "translation chains").

Heloise is a reengineering of compilers and "engines" of Ariane-G5's Specialized Languages for Linguistic Programming (SLLPs), running both Linux and Windows. The aim of its author when he developed this new version of Ariane-G5 SLLPs, was to make this system available to anyone wishing to design his own operational expert MT system (i.e. an MT system relying on linguistic expertise, as opposed to systems based on statistical properties of languages). This approach is part of the movement aiming at reducing the digital divide through the provision of tools, usable by non-specialists, and enabling them to develop their own language services.

This article shows how Ariane-G5 can significantly contribute to the democratization of quality machine translation and to its usability to under-resourced languages (π-languages), and especially to under-resourced languages pairs (not only pairs of π-languages!). It then describes some technical aspects of Heloise and provides a comparison with Ariane-G5 as well as with Ariane-Y, another software developed within the GÉTALP and also deriving from Ariane-G5. The need for complementary tools in addition to the SLLPs is discussed in the conclusion.

3 Ariane-G5

3.1 General principles

Ariane-G5 is a generator of machine translation systems. It uses an expert approach (including a description of the languages handled) and the generated systems are generally based on a multilevel transfer linguistic architecture, and developed using a heuristic programming approach. It has also been used for "abstract pivot" approaches (IF semantico-pragmatic formulas for speech MT in the CSTAR and Nespole! projects in 1995-2003, and UNL linguistic-semantic graphs since 1997).

Ariane-G5 relies on five Specialized Languages for Linguistic Programming (SLLPs) operating on decorated trees. Each of these languages is compiled and the internal tables produced are given as parameters to the "engines" of the languages. The specificity of an SLLP is that it offers high-level data structures (decorated trees or graphs, grammars, dictionaries) and high-level control structures (1-ary or N-ary non-determinism, pattern-matching in trees, guarded iteration).

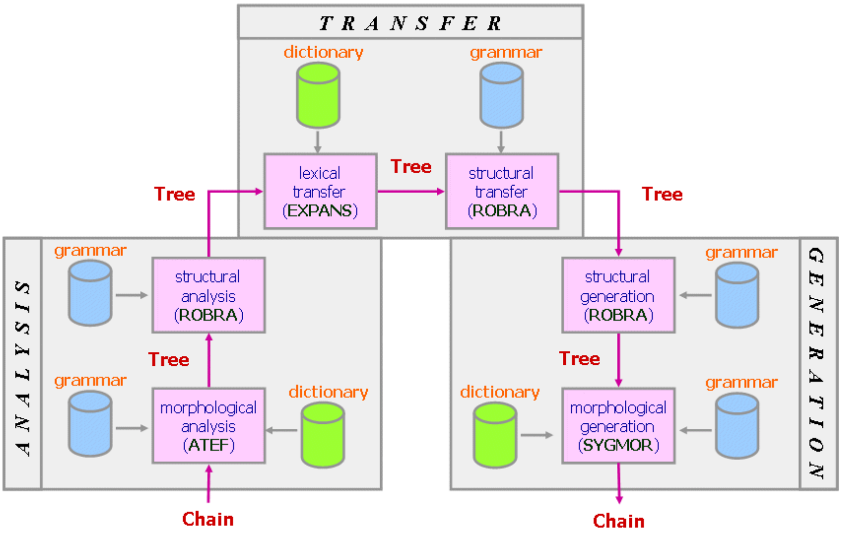

TRANSFER
dictionary
grammar
lexical transfer (EXPANS)
Tree
structural transfer (ROBRA)
Tree
Tree
ANALYSIS
grammar
structural analysis (ROBRA)
Tree
grammar
morphological analysis (ATEF)
dictionary
Chain
GENERATION
grammar
structural generation (ROBRA)
Tree
dictionary
morphological generation (SYGMOR)
grammar
Chain

FIGURE 2 – Ariane: analysis, transfer et generation.

3.2 Programming languages for L-developers

From the beginning, Ariane has offered high-level programming languages — specialized languages for linguistic programming (LSPL) — thus simplifying the work of the L-developers:

- ATEF (string-to-tree transformations) allows writing morphological and morphosyntactic analyzers producing decorated trees encoding the remaining ambiguities, with the possibility of treating inflectional derivational and compositional morphology, connected uninflected idioms, and of performing sophisticated treatment of "unknown words".
- ROBRA (tree-to-tree transformations) allows writing transformational systems operating on decorated trees. It offers parallel rewriting, guarded iteration and recursion, and 1-ary non-determinism (by backtracking) at the level of the control graph.
- TRANSF/EXPANS (tree-to-tree transformations) allows writing transformation phases of lexical items, where one node can be transformed into a large subtree;
- REFORM/TRACOMP [3] (tree-to-tree transformations) can perform conversions of decoration sets between phases;
- SYGMOR (tree-to-string transformations) allows writing morphological and morphotactic generators.

[3] REFORM/TRACOMPL, which is a sub-language of the ROBRA, TRANSF/EXPANS, and SYGMOR SLLPs, only appears implicitly in figure 1.

These languages were designed in order to free the L-developers of the need to use conventional programming languages to write the rules and dictionaries.

3.3 Architecture principles and limitations of Ariane-G5

Although it received many improvements from Pierre Guillaume and Maurice Quézel-Ambrunaz between 1985 and 2003, Ariane-G5 retains many historical limitations. These limitations are mainly:

- the size of decorations (limited to 32 bytes, 2 for the lexical unit (UL) and 2 for the form), and therefore the number of lexical units active at the same time must be less than 65,536 (32,768 "static" and as many "dynamic"), and the size of the source text of a translation unit must be less than 64K or 46.8 full standard pages[4], or 11,700 words),
- the length of an occurrence (up to 256 characters);
- the length of a UL[5] (maximum 34 characters, because of a syntax in tabular format, which is inconvenient to take as UL the UWs of the UNL project: they are replaced by small subtrees);
- the maximum number of nodes in a tree (64K), more than enough for a standard linguistic tree (there are 2.5 to 3 nodes per word), but not for multiple parse trees (in the LIDIA mockup of interactive disambiguation MT, there can be 400 nodes per word).

The new developments that are Ariane-Y ([Nguyen, 2009], pp. 70—74, 93—100) and Heloise free the user of these limitations.

3.4 The usefulness of a reengineering of Ariane-G5 for π-languages

Until the arrival of operational statistical solutions (Pharaoh, Moses, Joshuah, and especially Google, which reached an advanced stage of maturity), the cost and difficulty of development of machine translation systems restricted to about twenty — the main languages the world — the number of languages benefitting from this type of service, i.e. the languages that allowed a rapid return on investment.

This situation lasted until about 2007. Since then, things have considerably progressed: for example, more than 60 languages are already available as source and as target language on Google Translation (http://translate.google.fr). However, the quality of translations obtained remains low for the reasons cited in [Boitet, 2008]. This paper shows in particular that the systems obtained by an expert approach are unavoidable — and also more economical — to open under-resourced languages (π-languages) and under-resourced language pairs (π-pairs of languages) to quality automated translation systems.

To democratize machine translation quality, it is also desirable to have efficient and open solutions. To limit the cost of building translation systems, generators must allow non-specialists — spontaneous groups (often diasporas) collaborating over the Internet, linguists having lexical resources... — to develop themselves a significant portion of an initial system and then to enrich and to maintain it on their own.

[4] A "standard page" has 1,400 alphabetical characters / 250 words in French or in English, or 400 characters in Chinese, Japanese or Korean.

[5] UL = Lexical Unit (French acronym), often not a lemma, but a symbol denoting a whole derivational family.

Besides its scientific interest, Ariane-G5 has several interesting features that make it a serious candidate for this purpose:

1. It is a generic generator of machine translation systems.

- It has been used to make mockups and prototypes for language pairs including a wide variety of European and Asian languages, in all about ten languages (Arabic, Chinese, English, French, German, Japanese, Malay, Portuguese, Russian, Thai), thus proving its usability for the most varied languages and language pairs.
- It takes as input linguistic resources described in specialized languages accessible to non-programmers.

2. In September 2010, the LIG has put under the BSD license[6] the existing lingware base, greatly facilitating the implementation of new systems[7].
3. Systems produced by Ariane-G5 are all made of three steps — analysis, transfer and generation — exchanging clearly defined hybrid multi-level structures. Accordingly:

- the analysis step only depends on the source language,
- the generation step only depends on the target language,
- that choice of language architecture allows to:

  - reuse totally the analyzers and generators made for a language (2*N if there are N languages),
  - limit to the transfer the effort to build a new language pair[8].

From the point of view of the democratization of machine translation quality, Ariane-G5 also has several disadvantages.

- The generated systems do not run natively on operating systems that are the most common: Windows, MacOS and Linux.
- No generated system yet reached or approached the level of commercial products (e.g., the transfer dictionary (dictionary of lexical units (UL)) of the Russian-French system, which is one of the largest, does not exceed 12,000 UL, the equivalent of 40,000 lemmas). The capacity of Ariane-G5 to go full scale (dictionaries of more than 1 million UL...) remains to be demonstrated.
- Lingware programming remains difficult and requires close collaboration between L-developers (linguists who develop grammars and dictionaries), and Ariane experts ([Vauquois, 1979] indicated that the correct composition for a team developing a grammar is one computer scientist and three linguists).
- The native monitor (human-machine interface via an IBM 3270 terminal) no longer meets current standards, and the meta-CASH environment, although very friendly and portable, designed by E. Blanc, can only be used if a zVM system administrator creates a virtual machine for each potential developer. Therefore, the only L-developments made for the

[6] See http://en.wikipedia.org/wiki/BSD_licenses.

[7] These lingware modules will soon be available for download.

[8] Contrary to a widespread but erroneous idea, the number of transfers for getting N(N-1) translation systems between N languages is not necessarily quadratic. For example, if we take as a "pivot" the linguistic trees of one of the N languages and write the 2(N-1) transfers between this language and the others, then 2(N-1) translations will be done with only one transfer and (N-1)(N-2) with two transfers. We can also use as pivot "UNL trees" equivalent to the "UNL graphs" (abstract structures, eventually under-specified, of English utterances), so all the translations will then be made with a double lexical and structural transfer.

last 15 years are limited to lingware written for GÉTA projects (CSTAR and Nespole! for speech translation, 1995-2003, and UNL enconverter/deconverter for French, since 1997).

- Ariane is a system designed in the 1970s so its designers have almost all left the laboratory. The L-developer community created during the ESOPE project (active between 1982 and 1992) fell apart when B'VITAL was purchased by SITE/Eurolang, thus complicating recruiting and training new L-developers.

The development of Heloise (and of course of Ariane-Y) had as main goals to transmit specific skill of Ariane and to create a PC version of its SLLPs. Work is underway to further facilitate the task of the L-developers.

3.5 Ariane-Y

The Ariane-Y project is a response to this need for reengineering of Ariane-G5. Started in 2000, this project driven by GETALP aims to achieve high performance version (removal of limitations), free (LGPL license), open (user-friendly interface) and incremental (dictionaries can change during treatment) of Ariane-G5. The software, having recently received additional funding from the ANR (Traouïéro project), should be available within one year. This project is described in [Nguyen, 2009].

4 Heloise, a reengineering of Ariane-G5

Heloise is essentially a reengineering of the Ariane-G5 compilers, except that the code generated by Heloise is directly executable code, when Ariane-G5 generates an intermediate compact code interpreted by “engines”. The development of Heloise [9] was performed in the technical perspective described in [Berment, 2004]. The objective defended in this PhD thesis is to provide non-computer specialists, with an amount of training as limited as possible, the possibility to enrich the language services offered by standard host platforms. For Heloise, the targeted service is machine translation and the host platform is, for example, a commercial word processor (Microsoft Word, Open Office ...) or an Internet browser. The linguistic complements in figure 3 are then machine translation systems, produced by Heloise for a given language pair.

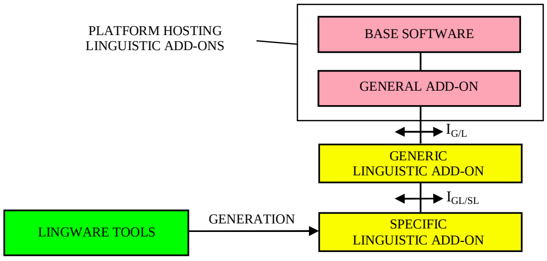

PLATFORM HOSTING LINGUISTIC ADD-ONS
BASE SOFTWARE
GENERAL ADD-ON
$I_{G/L}$
GENERIC LINGUISTIC ADD-ON
$I_{GL/SL}$
LINGWARE TOOLS
GENERATION
SPECIFIC LINGUISTIC ADD-ON

FIGURE 3 – Modular organization of the developments

Please refer to section 1 (in French) for technical details about Heloise, especially for a description of its SLLP compilers and of its users’ interfaces.

[9] This development, done in 2009, followed a theoretical study ([Del Vigna et al]).

5 Comparisons and performances

5.1 Comparison between Heloise, Ariane-G5 and Ariane-Y

The table below provides some comparisons between Heloise, Ariane-G5 and Ariane-Y.

	Ariane-G5	**Ariane-Y**	**Heloise**
Number of SLLPs	5	5	4
Programming languages used to implement the SLLPs			
SLLP Compilers	ASM360, PL360, PL/I, EXEC/XEDIT, REXX	C/C++ et REXX	C/C++
SLLP engines or « interpreters »	ASM360, PL360	C/C++	No interpreter (executable code)
Techniques used for SLLPs			
Implementation architecture	Compilation directly producing loadable bytecode	Compiler calling a « loader » producing a bytecode	Double compilation: SLLP $\rightarrow$ C++, C++ $\rightarrow$ executable code
Development of SLLP compilers	Direct writing in ASM360 and PL/I (only SYGMOR)	`ANTLR`	`saint-jean` (Claude Del Vigna)
Standard compilers used	ASM360, PL360, PL/I	`gcc/g++`	`Visual C++` (Win32) `gcc/g++` (Linux)
Internal management of dictionaries	Compressed form allowing a binary search	`TabFich` (infinite tables) and AVL (almost balanced trees)	`sqlite`
Human-machine interfaces			
HMI	Command line and parameters files editor (IBM 3270) Hypertext stacks (CASH/RunRev for PC)	Web demo interface[13]	Web (Linux) Application (Windows)

TABLE 1 – Elements of comparison between Ariane-G5, Ariane-Y and Heloise.

[13] Several interfaces are planned (command line and graphical dialogs like in CASH).

5.2 Performance

5.2.1 Execution time

In morphological analysis of French (FR3 lingware), the analysis of 30 pages of text (Word pages in Times New Roman 12 points, 15,858 words, generating a tree of 37,729 nodes in AM output) takes approximately 42 seconds (the tests give at least 40,326 ms and a maximum of 43,402 ms) and 431,653 database accesses only contribute for ~5.7 seconds (exact figures: min. 5,672 ms, max. 5,906 ms), about 14%.

As a comparison, the morphological analysis of the same 30 pages by SYGFRAN, the analyser of French created by Jacques Chauché (LIRMM), takes about 2 seconds on the site http://www.lirmm.fr/~chauche/ExempleAnl.html, at the « *liste lemmes* » item. Ariane-G5, for the same text, takes 23 seconds, measure that must be weighted against the lower power of the H30 machine (42 to 126 times less).

The table below (Table 2) provides the computation times used by the server (a PC under Linux 2.6, CentOS release 5.3) for analysing a text of 480 words in French (from Victor Hugo's "*Les Misérables*", as were the 30 pages mentioned before). Several trials were done to study the dispersion of the results, which is finally rather low (< 2.5 % of the total).

Phase	Time (trial 1)	Time (trial 2)	Time (trial 3)
AM	1 813 ms	1 686 ms	1 679 ms
AS 1	1 845 ms	1 524 ms	1 485 ms
AS 2	10 770 ms	10 468 ms	10 475 ms
AS 3	1 814 ms	1 744 ms	1 722 ms
AS 4	39 029 ms	39 253 ms	39 675 ms
AS 5	962 ms	984 ms	977 ms
Total	**57 033 ms**	**55 659 ms**	**56 013 ms**

TABLE 2 – Computation times for analysing a text of 480 words.

The linearity with the data was evaluated with a longer text (~5 Word pages in Times New Roman 12 points, 2,880 words, thus 6 times longer). The parse trees contain 6,499 nodes at the output of AM and 5,629 nodes at the output of AS5. The results are summarized below (Table 3):

Phase	Time (480 words)	Time (2,880 words)	Ratio (x 6)
AM	1 813 ms	10 344 ms	x 5,7
AS 1	1 845 ms	26 998 ms	x 14,6
AS 2	10 770 ms	76 851 ms	x 7
AS 3	1 814 ms	23 581 ms	x 13
AS 4	39 029 ms	247 300 ms	x 6,3
AS 5	962 ms	22 698 ms	x 24
Total	**57 033 ms**	**407 772 ms**	**x 7**

TABLE 3 – Relationship between text size and computation time.

The order of magnitude is still the same, but important disparities can be noted in three ROBRA phases: AS1, AS3 and more especially AS5.

5.2.2 Data size

With the technique used in Heloise, the generated code (formats, dictionaries…) can reach sizes that dramatically exceed the compilers capacities (g++, Microsoft), thus making a problem for scalability. The strategy selected in Heloise to solve this problem is a "divide and conquer" approach: the code to compile is split into several files of controlled size (for example, a file can contain the image of 1,000 formats).

Note that this problem does not exist in Ariane-G5, which compilers can process up to 8,000 formats and 30,000 lexical entries dictionaries in one time. As for Ariane-Y, ANTLR allows to process files of more than one million lines.

Conclusion and future work

Several works and research axes are under way around Heloise, including:

- the study of tools even more "appropriable" and easily understandable by L-developers and of a simpler development platform (graphical programming...);
- the evaluation of the effort needed for creating a system for a new language pair with at least one of them under-resourced;
- the study of performances and of translation quality improvement, including, for example, the addition of new SLLPs (dependency analyzer, Q-systems...), statistical treatments or the introduction of learning techniques.

Conclusion

The development of Heloise was the occasion to evaluate the behaviour and performances of Ariane-G5 outside its "mainframe" environment. The result appeared to be good enough to decide to make Heloise being used for developing operational MT systems. However, it remains for that aim to:

- develop more user-friendly tools (graphical programming, Q-systems...) such that non-specialists can easily become L-developers and develop new lingware by themselves, in particular for π-couples of languages,
- demonstrate Heloise capacity (so also Ariane-G5 and Ariane-Y) to produce operational systems with similar performances to current commercial systems (dictionaries of several millions of entries, low translation delays, good translation quality…).

A modified version of Ariane-G5 could also be developed, as that was foreseen with the Ariane-X project. This could for example consist in adding a dependency analyser or some statistical processing. The use of the HCL library would ease that development.

Other ideas for democratizing quality machine translation remain to be explored. They can be technological (use of translation memories [Boitet, 1999], translation through a UNL pivot [Boitet, 2002]…), legal (license policy) or methodological (collaborative project, involvement o the diasporas, software reuse... [Berment, 2004]).

References

Bachut D., Le projet EUROLANG : *une nouvelle perspective pour les outils d'aide à la traduction*, Actes de TALN 1994, journées du PRC-CHM, Université de Marseille, 7-8 avril 1994.

Bachut D., Verastegui N., *Software tools for the environment of a computer aided translation system*, COLING-1984, Stanford University, pages 330 à 333, 2-6 juillet 1984.

Berment V., *Méthodes pour informatiser les langues et les groupes de langues peu dotés,* Thèse de doctorat, Grenoble, 18 mai 2004.

Boitet C., *Le point sur Ariane-78 début 1982 (DSE-1), vol. 1, partie 1, le logiciel,* rapport de la convention ADI n° 81/423, avril 1982.

Boitet C., Guillaume P., Quézel-Ambrunaz M., *A case study in software evolution: from Ariane-78.4 to Ariane-85*, Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages, Colgate University, Hamilton, New York, 14-16 août 1985.

Boitet C., *Current machine translation systems developed with GETA's methodology and software tools,* conférence Translating and the Computer 8, 13-14 novembre 1986.

Boitet C., *La TAO à Grenoble en 1990, 1980-90 : TAO du réviseur et TAO du traducteur*, partie des supports de l'école d'été de Lannion organisée en 1990 par le LATL et le CNET, 1990.

Boitet C., *A research perspective on how to democratize machine translation and translation aids aiming at high quality final output*, MT Summit VII, Kent Ridge Digital Labs, Singapour, pages 125 à 133, 13-17 septembre 1999.

Boitet C., *A roadmap for MT: four « keys » to handle more languages, for all kinds of tasks, while making it possible to improve quality (on demand),* International Conference on Universal Knowledge and Language (ICUKL 2002), Goa, 25-29 novembre 2002.

Boitet C., *Les architectures linguistiques et computationnelles en traduction automatique sont indépendantes,* TALN 2008, Avignon, 9-13 juin 2008.

Del Vigna C., Berment V., Boitet C., *La notion d'occurrence de formes de forêt (orientée et ordonnée) dans le langage ROBRA pour la traduction automatique, Approches algébrique, logique et algorithmique*, Journée thématique ATALA sur la traduction automatique, ENST Paris, 1er décembre 2007.

Guillaume P., Ariane-G5 : *Les langages spécialisés TRACOMPL et EXPANS,* document GÉTA, juin 1989.

Guilbaud J.-P., Ariane-G5 : *Environnement de développement et d'exécution de systèmes (linguiciels) de traduction automatique*, Journée du GDR I3 co-organisée avec l'ATALA, Paris, novembre 1999.

Nguyen H.-T., *Des systèmes de TA homogènes aux systèmes de TAO hétérogènes*, Thèse de doctorat, Grenoble, 18 décembre 2009.

Vauquois B., *Aspects of mechanical translation in 1979,* Conference for Japan IBM Scientific program, juillet 1979.

Vauquois B., *Computer aided translation and the Arabic language,* First Arab school on science and technology, Rabat, octobre 1983.