The Next Step: Moving To An Integrated MT System For High-Volume Environments

Walter K. Hartmann
Lernout & Hauspie
Speech and Language Services Division
USA

Abstract:

Within the realm of small to medium-sized translation companies, the demands placed on MT in a high-volume production environment plagued with extremely demanding turn-around times and cost pressures are quite different from most other uses of MT. With the help of an analysis of a typical project the author shows the need for MT to become an integrated part of a translation application which will reduce the amount of extraneous processes to a minimum. In conclusion, a system is proposed which will streamline all the ancillary processes in order conform to customers' turn-around demands without jeopardizing post-editing quality.

1. Introduction

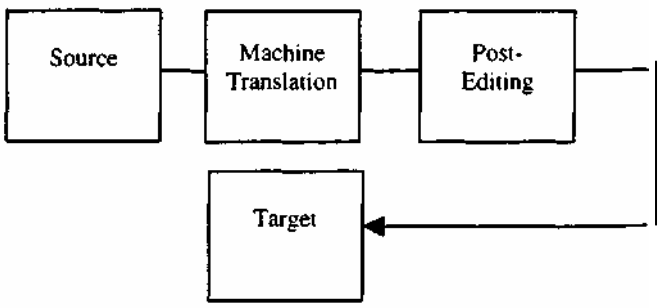
With the advent of higher-end PC-based machine translation systems, small and medium-sized translation companies could finally consider machine translation a viable service to offer their clients. Before this point, mainframe or Unix-based systems on the one hand imposed a high demand for hardware and up-front development expenditures, unaffordable to most companies. On the other hand, low-cost and low-end systems did not present a realistic alternative to traditional manual translation.

The PC based higher-end systems, in contrast, delivered the needed quality to develop MT plus post-editing into a viable business solution. At the same time they were not placing too many restrictions on the use beyond those inherited from the larger systems, such as the preference for interactive workstation use and, consequently, single-file processing.

The main benefit of the availability of PC-based MT systems for the translation business such as mine was at the time was that it was now possible to market services to larger potential clients, those with higher demands for speedy and cost-efficient services. The services we could offer with our own MT system propelled us into being able to offer translation of high-volume service literature and other technical manuals quickly, efficiently and consistently.[1]

And, after spending many years extolling to clients the virtues of MT, we were suddenly faced with a demand that it deliver what we were promising. Faster turn-around, higher throughput and lower prices are now demanded in dimensions I had never imagined. In today's fierce climate of commercial translation, MT has moved from a singular, monolithic software application to an integrated piece, albeit the centerpiece, of the translation puzzle.[2]

What should be, and is often considered, a simple process:



turns out to be much more complicated.

The many processes that prepare source materials for processing by the MT program, and the actions required to transform the MT output into the desired target files are the other pieces, and it is a great challenge to put these together in a high volume production environment. Once a project consists of several thousand files containing over one and one half million words, with a production schedule that is limited to around 90 days, the academic contemplation of machine translation, its approaches and limits, fades quickly to make room for a hectic work environment.

And it is from this kind of near-chaotic production scenario that I want to discuss the needs of production houses such as the firm for which I work. It may surprise you that very little will be mentioned about translation quality, actual processing speed and the other topics often discussed in this forum. We have, by now, taken the more pragmatic view that the attainable MT quality has to suffice for our production needs. Naturally, we would welcome any and all improvements in output quality, they are, however, not tantamount to our work.

More to my point, I will propose that MT take a more efficient place in the production chain, surrounded by integrated tools that streamline the process. While some of the MT products currently on the market have already integrated some of the modules proposed in this paper, none of them currently offers the full suite of tools in one package under one unified user interface. The conclusion of my presentation, therefore, will outline the specifics of such an integrated product; that is, my dream system.

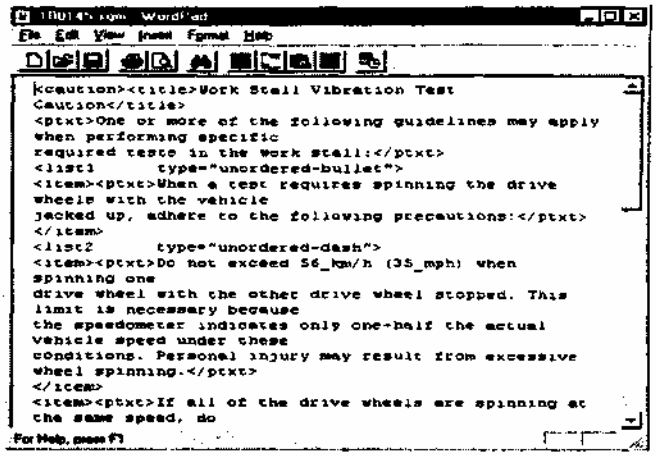
To set the scenario for the description of this dream system, I will describe the production process we have developed for what we call *extreme volume MT projects.* These projects involve thousands of files containing (together) at least one million words and have a very tight deadline (such as 90 days).

2. Project Description

The greatest challenge in the process of translating such high volume projects lies in the management of the work, in streamlining the processes in a way that bottlenecks are largely avoided. A typical project for the purpose of this presentation is a service manual of about 4.000 pages, containing approximately 1.5 million words. The service manual does not exist in printed form, even in the source language, as it is designed to be distributed only on CD.

Files are received divided in various topic areas, such as maintenance, repair, information, service, etc. Other than that, there is no clue as to where a certain file fits into the publication, as even the file names are purely numeric, too. It has even happened that files from more than one publication were combined in the same release. For example, the service literature for various different transmissions was grouped into one release of files.

The different topic areas may contain anywhere from 50 to 3,000 files, ranging in size from 1 Kbyte to l00 Kbytes, and they are in SGML format.



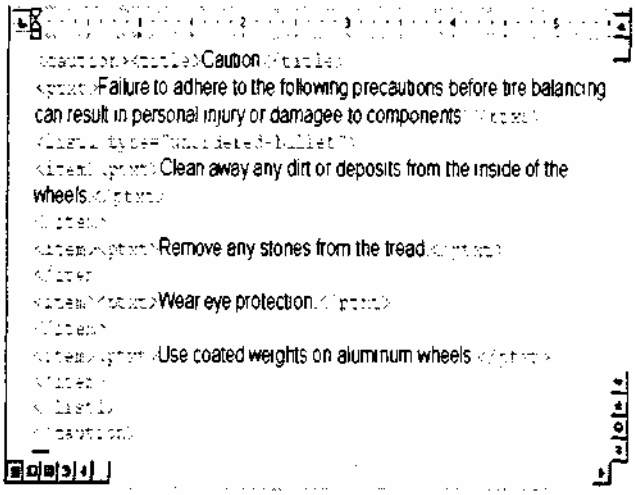
Example of SGML file in .txt format

This format poses a problem for most MT programs that prefer to process files in either text or rtf formats. Systran is an exception here, as it allows for one-step filtering of SGML codes before and after translation.

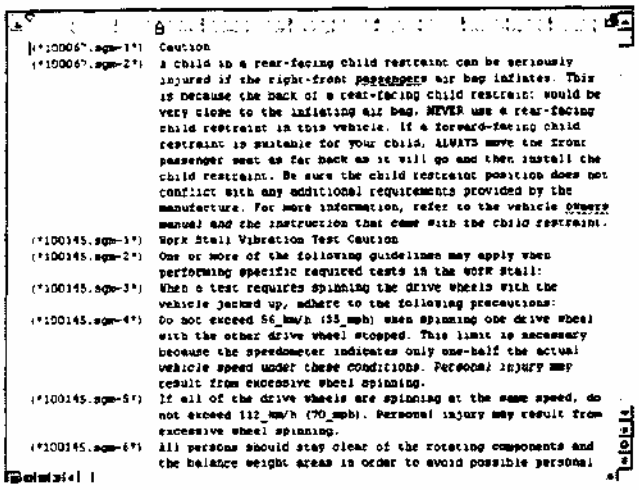
2.1. Original Approach

In order to process the files through other MT programs, a filter is needed to hide the SGML tags first. The tags must be made invisible by either replacing them with untranslatable place holders or by making them invisible to the MT/TM program by other means, such as special filters.

The following two figures depict two such filters in use, one distributed by Trados the other a proprietary L&H filter.



Example of Trados .SGML to .rtf filter



Example of HIS Parser (placeholders)

Next, it is necessary to determine the terminology unrecognized by the MT program. For this, it is necessary to make a first pass through a MT program that yields a list of words not found in its terminology lists. The lack of a good and reliable batch feature renders most MT programs ineffective, as they can only translate a handful of files in a given batch. This requires manpower to process the small batches, reducing the processing time to normal work hours. It therefore takes between 3 and 5 days to process 8,500 files, using a 350 MHz Pentium II computer with 64 MB of RAM.

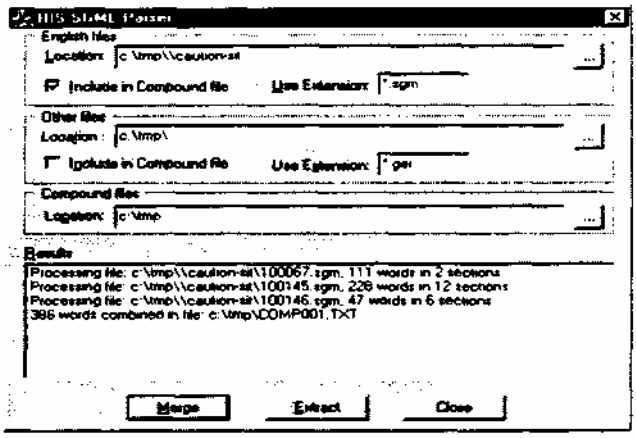
Unfortunately, and in keeping with the single-file approach to MT, each file processed through MT programs results in one single file containing the words that were not found in the electronic dictionaries. It is, therefore, necessary to combine all these data files into one and eliminate duplicates so as to gain a quick and concise overview of the terminology to be added.

For good measure, this word list resulting from the first pass will also contain a good amount of misspellings and other orthographic defects in the source texts. The correction of these mistakes by means of batch search and replace programs is very useful and enhances the output quality of the MT program, but it is, at the same time, the only type of pre-editing the source text time will permit.

Once the mistakes are corrected and the unknown words are programmed into the CSD, the files are ready for translation by MT and subsequent post-editing. Again, much time is lost here due to the lack of an efficient batch process.

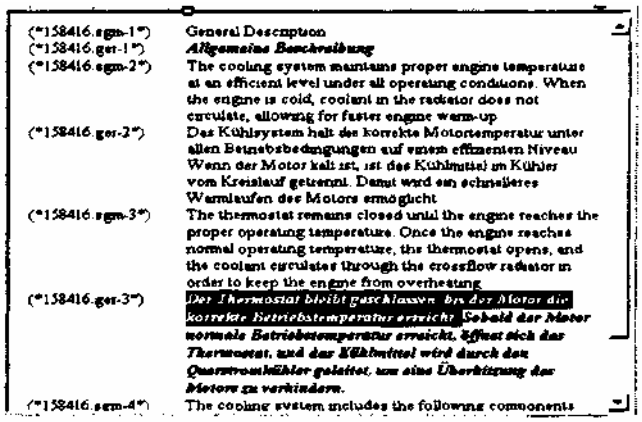
In practice, it is not very useful to try and post-edit each of the thousands of text files individually. Rather, it is highly inefficient, given that it is not possible to perform search and replace functions on more than the currently open file or to use other efficiency tools.

For this reason, we developed a parser/concatenation tool that allows us, in one pass, to replace the SGML tags with place holders and to concatenate a number of small files into one large file which can then be saved in plain text format. In general, we have set the size for concatenated files at about 20,000 words, which reduces the amount of files to be processed from 8500 to about 90.



HIS Parser, after replacing SGML tags and concatenating three files

With this parser, we can also combine each source segment produced by removing the SGML tags with the same segment in raw MT output. This allows the post-editor to have source and target in one file for easier reference:



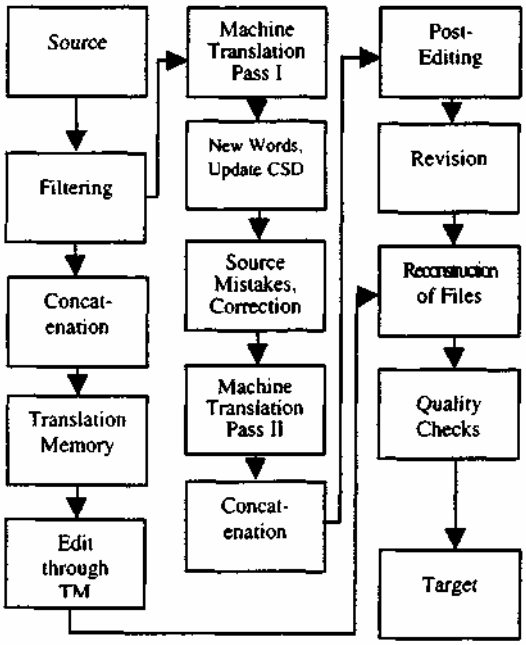
After the post-editing and revision of these files, the aforementioned parser can extract the translated segments into the source files, replacing the source segments with their edited translations. All that is now left is the performance of several quality control measures to ensure that all source files were indeed translated, that no source language remains in the files and that all files are where they are supposed to be.

The time line, therefore, for processing files before and after MT will be approximately like this:

Activity	Days
File quantification and logging	1
First pass through MT	4
Creation and evaluation of NFW lists	2
Programming the CSD	1
Second pass through MT	4
File concatenation and combination	1
Post-editing 90 files @ 20,000 words/ 6 translators, and revising	75
Merging translation with source files	1
Performing various operations on files	2
Checking file quality	2
Preparation for delivery, and delivery	1
Total:	**94**

From this table it is obvious that, despite the tight schedules, the goal of 90 days is barely within reach, even under ideal conditions (i.e. none of the post-editors or revisers gets sick or needs time off, none of the freelancers drops out for another, more lucrative project, etc.).

It is also quite clear that the simplistic production model shown above does not bear any resemblance to reality, as the process should be depicted more like this[3]:



How, then, can we improve the process so that not only the deadline can be met but also can be met with a somewhat comfortable margin for unforeseen circumstances?

A reduction of the time allotted for post-editing and revision is not practical, as this would invite more errors and lesser-quality revision of the texts. To add more translators to the team would endanger consistency. We have noticed that control over consistency becomes exponentially more difficult as more translators and/or revisers are added to a team.

The answer, then, must lie in the pre- and post-production times. We need to further shorten the time it takes to get the source text and raw translation to the post-editors.

The most logical approach would seem to be the use of Translation Memory as a tool to shorten translation time. This is true, to a certain extent. Most TM programs, however, require files to be in a specific format, such as .txt or .rtf, which requires yet another step in the process if these formats differ from the one required by the MT engine.

Also, TM introduces some extra steps which slow down the production in the extreme volume range. In order to produce completely translated files, the source text must be run against the TM first, yielding the 100% matches and exportable files of non-matches. These unmatched segments now must be processed through MT, post-edited and revised, then aligned and read into TM again. The next pass of the same source files should yield, at least in theory, a 100% edited and publishable file.

The argument against this approach is the difficulty of post-editing these segments that are even more devoid of context than the complete files themselves. You may recall that, in our example, the naming processes and sequence of the source files leave no clue as to the context of their contents. Yet within themselves, they possess at least a semblance of context. This context, however is stripped away by the segmenting of the source file into 100% matched and unmatched parts of the files.

As a work-around, we have decided to process files in a different way: One copy of the complete source release is immediately concatenated into larger, but fewer files, and these files are then analyzed by the TM program. This process is performed at the same time as the first run of the complete source release through MT and adds therefore no additional time to the overall process. The files with a relatively high number of matches in the 85% to 100% range are kept for processing through TM in interactive mode, while the files with lower match rates are machine translated and post-edited in the traditional sense. We were able to do this after amassing a translation memory of about 4 million translated words in approximately 150,000 aligned segments.

The effect of this approach on the timeline is measurable, even if it is not entirely satisfactory:

Activity - Machine Translation	Activity - Translation Memory	Days
File quantification and logging		1
First pass through MT	Concatenation and first pass through TM	4
Creation and evaluation of NFW lists	Preparation of files editable in TM, project memories, etc.	2
Programming the CSD		1
Second pass through MT		4
File concatenation and combination		1
Post-editing 75 files @ 20,000 words/5 translators Revising (62 days)	Editing/translating within TM program of 15 files @ 20,000 words/1 translator (60 days)	62
Merging translation with source files		1
Performing various operations on files		2
Checking file quality		2
Preparation for delivery, delivery		1
Total:		**84**

This approach has proved itself to be somewhat more efficient, yet it still does not provide us with enough of a safety margin to ensure on-time delivery. And, furthermore, it has actually increased personnel demand, as the two processes, TM and MT, must be run concurrently. Moreover, it is inefficient insofar as many 100% matches are not properly utilized, because they may appear in files with a high number of near- or non-matches.

An even smoother approach is needed, one in which the demand on processing time and operator time can be cut even further so a large enough time cushion is created for the biggest variable in the process, the post-editing/editing.

3. The Dream Machine

The system I envision, therefore, consists of an all-in-one black-box approach containing all the functions and capabilities necessary for projects of the size and complexity such as I have described.[4]

I envision an application that will allow, as a first step, the filtering of source files into the format preferred by the MT engine. The user should see a dialog box in which I could choose the appropriate filter, from SGML, PageMaker, FrameMaker and so on.

Next, the program would ask the user to define a range of files to be processed, from one to as many as can be handled within the limitations of RAM and storage capacity on the individual workstation. An option to concatenate files in cases where there is an abundance of small files is necessary, as is the possibility of choosing the size of the concatenated files.[5]

The ideal program has, of course, translation memory integrated within the machine translation program. Any text will have the option of being passed through translation memory, and only the near- and non-matches below a certain fuzzy level will be machine translated.

After defining glossaries, etc., a pre-run should be conducted by the application, yielding a single file of new words, as well as statistics of TM matches, word counts, and an opportunity to stop and make changes to and correct the source files. In the next step, after an analysis of the new words lists, the source files will be amended and the CSD updated. Then the second run through MT can proceed.

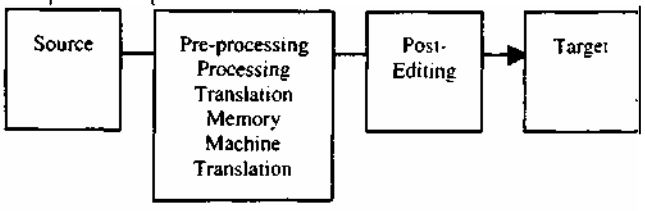
A choice of output files, monolingual raw translation or combined source/MT output should be the result of the second pass through MT.

Finally, a reintegration of the post-edited files is needed, for automatic alignment integration into the TM database, as well as for restructuring and re-filtering the files according to the source file formats. And that will give us the finished files, ready to be sent to the client after a final quality assurance step.

The time savings will be rewarding:

Activity - Machine Translation	Days
File quantification and logging	1
First pass through MT	2
Evaluation of NFW list	1
Programming the CSD, Corrections in Source	1
Second pass through MT	2
File concatenation and combination	1
Post-editing 75 files @ 20,000 words/5 translators; Revising (62 days)	62
Merging translation with source files	1
Performing various operations on files	1
Checking file quality	1
Preparation for delivery, delivery	1
Total:	74

This schedule makes the 90 day goal attainable. The other benefit is that most non-translator functions can be performed by one person, yielding efficiencies in terms of manpower. And it will happen in a transparent process, a black-box approach with only a few buttons to press, reducing, at least outwardly, the process again to a simple concept.



One might argue that what I am describing here is actually equivalent to a "eierlegende Wollmilchsau", as they say in German, an egg-laying, milk-producing pig in a sheep's coat. Of course, combining all these functions, most of which, by the way, are already available in separate products and programs[6], will not be easy. But then, in keeping with the Summit's theme of MT in the Great Translation Era and finding ourselves at the doorstep of a new Millenium, in time, it will happen, I am sure.

---

[1] It is important to point out that the basis for this paper dates back a few years in which the process described was developed. At the beginning, the business to which reference is made, was Hartmann International Services, Inc, a small translation company in upstate New York. HIS was subsequently bought by a German translation company , Heitmann International GmbH (1997) which, in turn, was acquired by Lernout & Hauspie in 1998.

[2] Earlier descriptions of high-volume translation environments have been presented by Lou Cremers. (1997) "Using MT in a Corporate Setting". In Proceedings of MT Summit VI, pp. 240 -241; and by Christine Kamprath. (1997) "Using MT in a Corporate Setting". Presented at MT Summit VI, but not contained in the Proceedings. What distinguished these scenarios from the one described here is that they dealt with in-house, corporate settings, where it was possible to adapt the input to the MT system's demands. In an translation business environment, this is not a viable solution.

[3] It should be noted that these diagrams are still very much simplified and do not pretend to encompass the complete process. For illustration purposes, they should, however, suffice.

[4] A predecessor of such a system, albeit limited in its scope and possibilities and leaving room for many "wishes", is described in Bech, A. (1997) "MT from an Everyday User's Point of View". In Proceedings of MT Summit VI, pp. 98 - 105.

[5] For files to be post-edited off-site which contain the source and the raw translation, a file size of 20,000 source words is appropriate, while sizes up to 35,000 source words are useful in cases where they will be processed through a separate TM system.

[6] See, for example, Schwall, U., and Thurmair, G. (1997) "From Metal to Tl: Systems and Components for Machine Translation Applications". In Proceedings of MT Summit VI, pp. 180 - 190.