# User Interface Aspects of a Translation Typing System

Philippe Langlais, Guy Lapalme, and Sébastien Sauvé

RALI-DIRO, Université de Montréal
C.P. 6128, Succ Centre-Ville
Montréal, Québec, Canada H3C 3J7
{felipe,lapalme,sauvese}@iro.umontreal.ca

**Abstract.** This paper describes the user interface design and evaluation of TransType, a system that watches over the user as he or she types a translation and repeatedly suggests completions for the text already entered. We show that this innovative approach to a translation tool, both unobtrusive and very useful, can be very productive for the translators.

## 1 Introduction

TransType is a project set up to explore an appealing solution to the problem of using *Interactive Machine Translation* (IMT) as a tool for professional or other highly-skilled translators. IMT first appeared as part of Kay's MIND system [6], where the user's role was to help the computer analyze the source text by answering questions about word sense, ellipsis, phrasal attachments, etc. Most later work on IMT, such as Brown [2], has followed in this vein, concentrating on improving the question/answer process by having less questions, more friendly ones, etc. Despite progress in these endeavors, systems of this sort are generally unsuitable as tools for skilled translators because the user serves only as an advisor, with the MT components keeping the overall control over the translation process.

TransType originated from the conviction that a better approach to IMT for competent translators would be to shift the focus of interaction from the *meaning* of the source text to the *form* of the target text. This would relieve the translator of the burden of having to provide explicit analyses of the source text and allow him to translate naturally, assisted by the machine whenever possible. In this approach, a translation emerges from a series of alternating contributions by human and machine. In all cases, the translator remains directly in control of the process: the machine must work within the constraints implicit in the user's contributions, and he or she is free to accept, modify, or completely ignore its proposals.

The core of TransType is a completion engine which comprises two main parts: an *evaluator* which assigns probabilistic scores to completion hypotheses and a *generator* which uses the evaluation function to select the best candidate for completion. The evaluator is a function $p(t|t', s)$ which assigns to each target-text unit $t$ an estimate of its probability given a source text $s$ and the tokens

$t'$ which precede $t$ in the current translation of $s$. We use a linear combination of separate predictions from a language model $p(t|t')$ and a translation model $p(t|s)$. Our linear combination model is fully described in [8] but can be seen as follows:

$$p(t|t',s) = \underbrace{p(t|t')\ \lambda(\Theta(t',s))}_{\text{language}} + \underbrace{p(t|s)\ [1-\lambda(\Theta(t',s))]}_{\text{translation}}$$

where $\lambda(\Theta(t',s)) \in [0,1]$ are context-dependent interpolation coefficients. $\Theta(t',s)$ stands for any function which maps $t'$,$s$ into a set of equivalence classes. Intuitively, $\lambda(\Theta(t',s))$ should be high when $s$ is more informative than $t'$ and low otherwise. For example, the translation model could have a higher weight at the start of sentence but the contribution of the language model can become more important in the middle or the end of the sentence.

The language model is an interpolated trigram [5] trained on the Hansard corpus (about 50 million words), with 75% of the corpus used for relative-frequency parameter estimates, and 25% used to reestimate interpolation coefficients.

Our translation model is a slight modification of an IBM model 2 [1] in which we account for invariant entities such as English forms that almost invariably translate into French either verbatim or after having undergone a predictable transformation e.g. numbers or dates. These forms are very frequent in the Hansard corpus.

TransType is a specialized text editor with a non intrusive embedded Machine translation engine as one of its components. In this project we had to address the following problems: how to interact with the user and how to find appropriate multi-word units for suggestions that can be computed in real time. The former has been described by Langlais [9] but this article focuses on the latter.

## 2 The TransType model

### 2.1 User Viewpoint

Our interactive translation system is illustrated in figure 1 for an English to French translation. It works as follows: a translator selects a sentence and begins typing its translation. After each character typed by the translator, the system displays a proposed completion, which may either be accepted using a special key or rejected by continuing to type. This interface is simple and its performance may be measured by the proportion of characters or keystrokes saved while typing a translation. Throughout this process, TransType must continually adapt its suggestions to the translator's input. This differs from the usual machine translation set-ups where it is the machine that produces the first draft which then has to be corrected by the translator.

TransType mode of interaction requires a synchronization between the user interface module and the translation engine in order to maintain a coherent state:

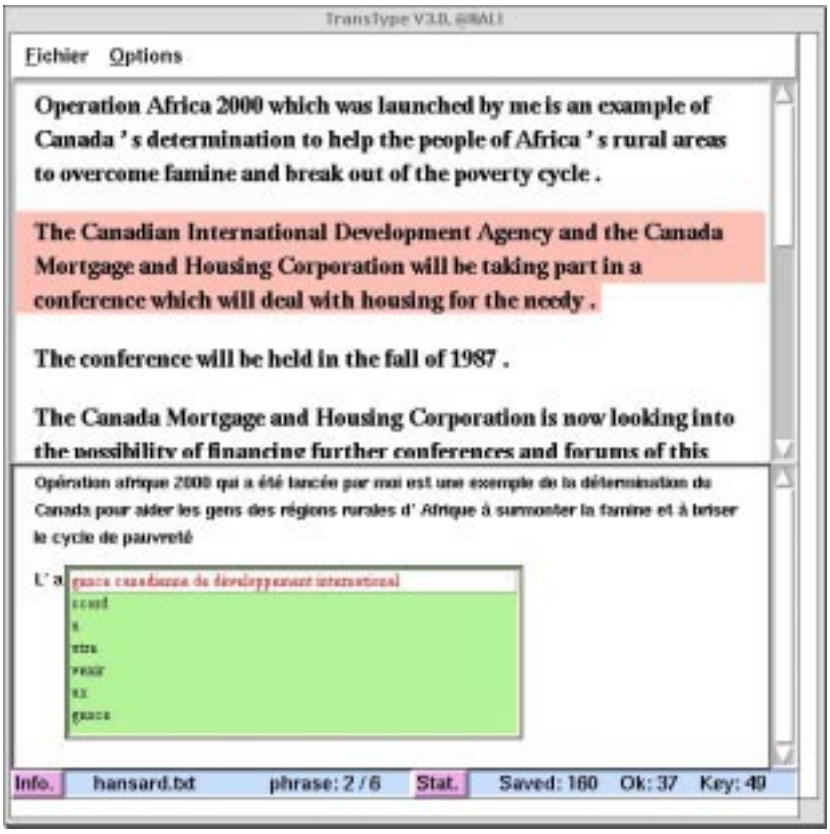


**Fig. 1.** Example of an interaction in TransType with the source text in the top half of the screen. The target text is typed in the bottom half with suggestions given by the menu at the insertion point. Unlike the version used in the evaluation, the current prototype which offers unit completion is illustrated here.

the translation engine must be aware of the sentence the translator is working on and continuously keep track of the part of the sentence that precedes the cursor. The synchronization must always be kept even in the case of cursor movements with the mouse or in the case of cut and paste operations.

## 3 Development of the user interface elements

A major part of the TransType project went into the design of a real-time translation engine fast enough to respond after each action of the user. This work was first described by Foster [4] and was implemented with a rudimentary line-oriented interface. The user was presented suggestions one at a time and control keys were used to cycle through them and to select one. This prototype

showed the feasibility of the underlying translation engine but was not really "usable" by translators.

We then defined the following objectives for a better user-interface:

- hide the inner workings of the translation engine
- provide an adequate display for the user showing both the source text and appropriate suggestions
- embed the engine in a more convenient and intuitive text editor similar to the usual working environment of a translator.

We developed a first version of the editor in order to find the best way to display the text and the suggestions: we tried to display the text and its translation side by side but it seems that a synchronized display of the original text and its translation one over the other is better; we also tried displaying suggestions in a separate window but we finally chose the set-up shown in Figure 1 where the seven best suggestions are shown as a floating menu positioned at the cursor. In the first version, editing was limited at going from left to right. The only way to correct what had been typed was by hitting the backspace key. This reflected the left to right working of the translation engine. But we quickly saw that this was too rigid (users could not even move the cursor with the arrow keys) and that the results would not be meaningful. Even though, our goal was only to prove the feasibility of our translation engine, we found that these interface limitations would hide the usefulness of TransType to the translators.

So we decided to invest more time in a translator friendlier interface that allows a free movement of the cursor either with the mouse or arrow keys. We also allowed all usual editing such as cut and paste of arbitrary selections of text. This implied a synchronization mechanism between the user interface and the translation engine of TransType in order to follow these cursor movements and to update in real-time the context of the engine. We also added easier means of dealing with suggestions which can either be cycled through using `PageUp` or `PageDown` keys; the current element of the menu always appear at the same level of the text to ease reading and can be accepted using either the `Tab` or the `Return` key. A user can also click directly any suggestion of the menu using the mouse.

User preferences can tailor some aspects of the interface dealing with:

- relevance of suggestions: mixing coefficients of the language and translation models, minimal score for a suggestion to be given;
- number of suggestions displayed, prefix length before a suggestion is made (currently 0) and the minimum number of letters that a suggestion must have before being shown.

We have not done a systematic comparison of all these parameters but we chose a set of what seemed to be the most adequate settings for the purpose of our evaluation if this tool really supported a more productive way of producing translations.

This interface was implemented using a text widget in Tcl/Tk linked with our translation engine written in C++. The text widget is limited to the edition of plain character files and thus is not a full featured text editor such as Microsoft Word which allows for formatting of characters using bold and italics, for paragraph indenting and centering and for creating figures and tables.

As we wanted to only test the speed of typing translations of isolated sentences, we did not need a full text processor but one that we could customize. We instrumented the interface to keep track in a file of all user actions. This file was then analyzed off-line to deduce measurements about the behavior of the user.

## 4 User-interface Evaluation

We first defined a theoretical evaluation of TransType on a word completion task, which assumes that a translator carefully observes each completion proposed by the system and accepts it as soon as it is correct. Under these optimistic conditions, we have shown that TransType allows for the production of a translation typing less than a third of its characters, see Langlais et al [11] for more details.

The goal of the evaluation was two-fold: first to see if this behavior of a hypothetical user is similar to the one of a human translator while composing a translation; second to gauge if TransType could help in other ways such as giving ideas for translations for terms for which there is some hesitation. As the suggestions of TransType are correctly spelled, their selection insures that there are less misspellings; this is particularly useful for completed proper nouns or numbers which must always be carefully transcribed and are often error prone.

### 4.1 User Protocol

We asked ten translators with various work years of experience and areas of expertise, to try TransType in a controlled setting. We took for granted that the translations were correct because we wanted to evaluate our system and not the translators themselves. All translators were given the same sentences to translate; these sentences were chosen arbitrarily from our corpus.

The protocol consisted of three steps:

1. **6 minutes without TransType** to reassure the translators that our text editor was quite conventional for typing texts: the usual keys for deletion, motion, cutting and pasting are present. There is no provision for formatting though. This step measures the "natural" typing speed of each translator, i.e. their speed of thinking and typing a translation in our text editor but without TransType activated.
2. **25 minutes with TransType** in which the user types a translation while being able to select suggestions given by the system. At about the middle of the experiment, we stopped and gave the translator some advice on trying

an alternate way of using TransType in order to make a better use of the facilities of the system. We soon realized that this intervention was more of an annoyance than a help but we kept it in order to have comparable results.

3. **6 minutes with longer suggestions** that were inspired by the work of Langé [7], we wanted to check if some longer suggestions that we called *briskels* (bricks and skeletons) could be useful. Briskels were presented to the user as soon as a user selected a sentence. The briskels were determined by hand for the sentences of our experiment but Langlais [10] has shown that is possible to automatically compute units longer than one word.

**Table 1.** Number of characters typed, automatically by accepting the suggestions of TransType, erased, the number of acceptations and the number of characters finally present in the text produced at Step 2 of our protocol. The last column shows the proportion of characters manually typed over the number of characters in the final text. The last line indicates the mean.

| | typed | auto | erased | accept. | final | % typed |
|---|---|---|---|---|---|---|
| 1 | 223 | 748 | 33 | 117 | 938 | 40% |
| 2 | 578 | 1469 | 118 | 238 | 1929 | 48% |
| 3 | 281 | 746 | 64 | 129 | 963 | 49% |
| 4 | 887 | 985 | 124 | 152 | 1748 | 67% |
| 5 | 817 | 1446 | 143 | 228 | 2120 | 56% |
| 6 | 189 | 505 | 92 | 82 | 602 | 60% |
| 7 | 669 | 885 | 85 | 151 | 1469 | 62% |
| 8 | 588 | 820 | 201 | 119 | 1207 | 75% |
| 9 | 222 | 962 | 93 | 166 | 1091 | 44% |
| 10 | 405 | 1156 | 155 | 198 | 1406 | 54% |
| | 486 | 972 | 111 | 158 | 1347 | 55% |

## 5 Results

### 5.1 Comparison with the theoretical evaluation

As we have discussed in section 3, the theoretical gain in the number of keys saved using TransType is about 0.45 if a user does not change his mind once something has been typed, does not move the cursor with the mouse and does not erase whole words or parts of the text.

Table 1 shows the number of characters that were typed during step 2 of the protocol. We observe that on average a translation can be obtained by typing only about a third for the characters. This figure roughly agrees with our theoretical user performance which had been used in developing our translation engine.

The number of suggestions that were accepted was quite high which shows the usefulness of TransType.

### 5.2 Productivity

We define productivity as the ratio of the number of characters in the final text over the time it took to produce the text. Interviews with the translators had shown that almost all of them thought that TransType had improved their productivity. Unfortunately Table 2 on the left does not corroborate this favorable impression because on the average raw productivity went down by 35%!

**Table 2.** Table on the left gives the raw productivity of the translators at each step of the protocol; Table on the right gives the corrected productivity rate not taking into account inactivity periods. The last line indicates the mean for all translators.

| | Step 1 | Step 2 | Gain | Step 3 | Gain |
|---|---|---|---|---|---|
| 1 | 67.2 | 54.9 | -18 % | 83.7 | 25 % |
| 2 | 143.9 | 85.0 | -41 % | 102.4 | -29 % |
| 3 | 79.3 | 60.0 | -24 % | 89.3 | 13 % |
| 4 | 87.7 | 86.5 | -1 % | 98.5 | 12 % |
| 5 | 131.9 | 92.6 | -30 % | 90.4 | -32 % |
| 6 | 70.0 | 34.9 | -50 % | 38.2 | -45 % |
| 7 | 141.7 | 84.3 | -40 % | 131.1 | -7 % |
| 8 | 116.8 | 45.9 | -61 % | 79.3 | -32 % |
| 9 | 77.1 | 46.4 | -40 % | 63.7 | -17 % |
| 10 | 101.6 | 58.5 | -42 % | 69.4 | -32 % |
| | 101.7 | 64.9 | -35 % | 84.6 | -14 % |

| | Step 1 | Step 2 | Gain | Step 3 | Gain |
|---|---|---|---|---|---|
| 1 | 123.4 | 128.4 | 4.1 & | 194.0 | 57% |
| 2 | 155.6 | 112.8 | -28 % | 137.2 | -12% |
| 3 | 118.0 | 107.7 | -8.7 % | 147.2 | 25% |
| 4 | 138.6 | 189.4 | 37 % | 221.8 | 60% |
| 5 | 148.1 | 127.5 | -14 % | 145.1 | -2% |
| 6 | 115.5 | 105.4 | -8.7 % | 84.0 | -27% |
| 7 | 156.9 | 127 | -19 % | 193.6 | 23% |
| 8 | 156.0 | 126.4 | -19 % | 185.9 | 19% |
| 9 | 138.5 | 188.9 | 36 % | 163.5 | 18% |
| 10 | 131.3 | 97.1 | -26 % | 153.5 | 17% |
| | 138.2 | 131.1 | -5 % | 162,6 | 17% |

This can be attributed to the learning process involved in using a new tool: some users did not hit the right keys to accept the suggestion, stopped for some periods or were stunned by some suggestions given by TransType. Interestingly enough, translator 4 who managed to get the most out of TransType used mainly the mouse to enter the translation. This means that the right suggestions almost always appeared in the menu. Some translators would have liked to temporarily deactivate TransType for reformulating some sentences that seemed to have "gone on the wrong track". We did not want to burden our voluntary translators for more than an hour although some of them would have liked to bring TransType home to use it regularly.

In order to partially take into account some of these factors we removed all inactivity periods of more than five seconds from the productivity computation. After these corrections, we see (Table 2 on right) that three translators managed to improve their productivity in step 2. But in step 3 where larger suggestions (briskels) were proposed, then 7 translators managed to improve their productivity. This prompted us to develop further work in computing longer suggestions than only one word [10].

### 5.3 Saved Effort

Another useful measure is the effort saved by TransType in producing a translation. Effort is defined as the number of actions (key press and mouse clicks) done in a unit of time. An ideal tool would increase productivity while redirecting the effort by inserting more characters with the least number of actions.

Figure 2 shows the relation between the effort and productivity at each step of our protocol. The diagonal corresponds to a ratio of one action for each character and would be observed by a translator who would type correctly all the text on the first try. This line roughly corresponds to the underlying assumption made in the theoretical evaluation.

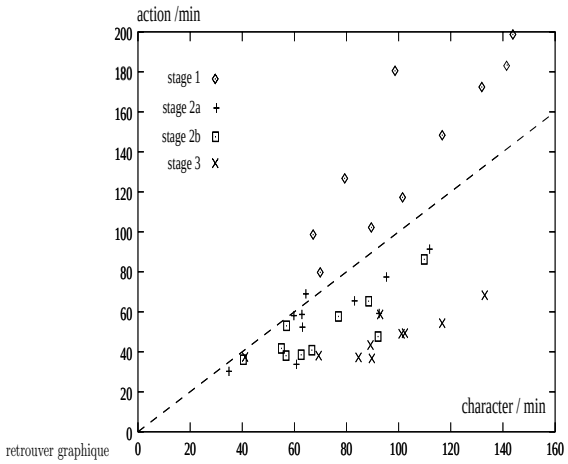


**Fig. 2.** Productivity versus effort of each subject over each stage of the protocol. The x-axis indicates the productivity, that is: the number of characters produced by unit of time (here a minute). The y-axis (the effort) indicates the number of keystrokes (or mouse clicks) produced on average each minute.

We see that actions of Step 1 of the protocol are over the diagonal and that the points of steps 2 and 3 are under the diagonal which means that each action produced more than one character.

We define efficiency as the ratio of productivity over effort. For example, an efficiency of 0.6 means that a user only produces 60 characters for 100 actions. Table 5.3 shows that the efficiency for all translators increases with each step of use of TransType.

### 5.4 Qualitative evaluation

All our testers (except one) were enthusiastic about this concept of translation typing tool even though our prototype was far from being perfect. They liked

**Table 3.** Average productivity, effort and efficiency of all subjects for each stage of the protocol.

| stage | productivity | effort | efficiency |
|---|---|---|---|
| 1 | 102.1 | 139.1 | 0.7 |
| 2 | 72.4 | 56.4 | 1.3 |
| 3 | 91.1 | 47.0 | 1.9 |

the idea that they could work at their own pace either accepting or ignoring TransType suggestions, contrarily to other translating tools that are always there even when they are not needed. The translators appreciated the fact that they did not have to check for the correct spellings of suggestions. Most of them were confident that with time they would become more proficient at making a better use of TransType.

The translators had more mixed feelings about the influence of TransType on the literary quality of their translations: some were under the impression that TransType induced a literal mode of translation. But they also noticed that it could have a positive effect because TransType allowed them to easily get the "long formulation" of a translation in cases where they would probably have typed an abbreviated form.

Translators also liked the idea of "false briskels" because they are long suggestions. But as it takes more effort to read them, it is often not easy to think about them at the right moment. This reinforces the idea that longer suggestions that would pop up at the appropriate moment would be very useful. We plan on evaluating this aspect later. More details about this evaluation are given by Sauvé [12].

## 6 Related work

It is hard to compare TransType with other systems because it is unique thanks to the statistical translation engine that drives it.

Although the style of text prediction proposed in TransType is novel, there are numerous precedents for text prediction in a unilingual setting. Many programs such as *GNU Emacs* and *tcsh* offer built-in word or command completion features, and word-completion add-ons are also available for standard word processing environments. For example the "small floating yellow windows" that Microsoft Word pops up when a prefix of a unique known word in a special table is recognized. In this case, the strings to be suggested were determined either when Word was compiled or they were painstakingly added by the user. Word only suggests one possibility while TransType determines many suggestions at run-time depending on the contexts of both the target and the source texts.

Dynamic completions also occur in the field of *alternative and augmentative communication* (AAC), which deals with communication aids for the disabled such as the *Reactive Keyboard* [3]. The system then tries to guess what the

user wants to type next. In this case, the suggestions or choices only depend on what has already been typed. In TransType, it is possible to vary the relative contributions of both the language and translation models; so in principle, we could set it up so that only the language model is used but we have not done any experiment with this.

Translation memories such as the one implemented in the Translator's workbench of Trados [13] also address the problem of speeding up the typing of translations. A translation memory is an interface to a database of pairs of sentences and their associated translations. Within a text editor, the translation memory manager first checks if the current sentence can be found in the database of previous translations and if so, it proposes its previous translation that can either be accepted or modified by the translator. This environment can be quite efficient in the case of repetitive texts or for revisions of already translated texts. Although some "fuzzy matches" are allowed for finding close enough sentences (for example sentences can vary by dates or numbers) this approach is not as flexible as the dynamic suggestions of TransType. Another drawback is the fact that once a user operates in the context of a translation memory, it is often awkward to stop it from proposing new sentences even if they are not relevant or to go around them. TransType on the other side is a silent helper whose suggestions can be quietly ignored when the translator already knows what is to be typed.

## 7 Conclusion

Although some drawbacks have been identified, this user evaluation was very useful to show the interest of the innovative TransType concept in a real setting. It is thus possible to develop computer aided translation tools that can help to improve the efficiency of translators who are more and more in demand in the new global economy.

## Acknowledgements

TransType is a project funded by the Natural Sciences and Engineering Research Council of Canada. We thank the Machina Sapiens company for its support in this project. We are greatly indebted to Elliott Macklovitch and Pierre Isabelle for the fruitful orientations they gave to this work.

## References

1. Peter F. Brown, Stephen A. Della Pietra, Vincent Della J. Pietra, and Robert L. Mercer. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312, June 1993.
2. Ralf D. Brown and Sergei Nirenburg. Human-computer interaction for semantic disambiguation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 42–47, Helsinki, Finland, August 1990.

3. John J. Darragh and Ian H. Witten. *The Reactive Keyboard*. Cambridge University Press, 1992.
4. George Foster, Pierre Isabelle, and Pierre Plamondon. Target-text Mediated Interactive Machine Translation. *Machine Translation*, 12:175–194, 1997.
5. Frederick Jelinek. Self-organized language modeling for speech recognition. In A. Waibel and K. Lee, editors, *Readings in Speech Recognition*, pages 450–506. Morgan Kaufmann, San Mateo, California, 1990.
6. Martin Kay. The MIND system. In R. Rustin, editor, *Natural Language Processing*, pages 155–188. Algorithmics Press, New York, 1973.
7. Jean-Marc Langé, Éric Gaussier, and Béatrice Daille. Bricks and Skeletons: Some Ideas for the Near Future of MAHT. *Machine Translation*, 12:175–194, 1997.
8. Philippe Langlais and George Foster. Using context-dependent interpolation to combine statistical language and translation models for interactive machine translation. In *Computer-Assisted Information Retrieval*, Paris, April 2000.
9. Philippe Langlais, George Foster, and Guy Lapalme. Unit completion for a computer-aided translation typing system. *to appear in Machine Translation*, page 25 p., Dec 2000.
10. Philippe Langlais, George Foster, and Guy Lapalme. Unit completion for a computer-aided translation typing system. In *Applied Natural Language Processing 2000*, page 10 pages, Seattle, Washington, May 2000.
11. Philippe Langlais, Sébastien Sauvé, George Foster, Elliott Macklovitch, and Guy Lapalme. Evaluation of transtype, a computer-aided translation typing system: A comparison of a theoretical- and a user- oriented evaluation procedures. In *Conference on Language Resources and Evaluation (LREC)*, page 8 pages, Athens, Greece, June 2000.
12. Sébastien Sauvé. L'évaluation d'un logiciel de traduction interactive: l'usager tire-t-il profit de transtype ? Internal report, Université de Montréal, avril 2000.
13. Trados Translator's Workbench 2, product description. URL: www.trados.com/workbench/index.html, 1998.