Advanced automatic MT post-editing

by Rafael Guzmán

This article was originally published in Multilingual Computing, #95, vol. 19, issue 3, pp. 52-57 (April 2008)

Key words: MT post-editing, machine translation post-editing, regular expressions

“Why is translation difficult for computers? Because it involves problems that resist an algorithmic solution, including common sense reasoning, learning, and combinatory explosive tasks.” – Doug Arnold, “Why Translation is Difficult for Computers”

Although machine translation (MT) will hardly ever be able to use “common sense reasoning”, it is possible to overcome this limitation to a large extent. In a previous article (*Multilingual*, #90 September 2007), I described how regular expressions can be used in text editors to automatically search for and post-edit linguistic errors in the translation outputs generated by MT: spelling mistakes, lack of grammatical agreement, bad syntax, redundancies, style issues, etc. By contrast, this article will show how regular expressions can be used to search for ambiguous text in the source segments and decide if their translation in the MT output is wrong, in which case it will be automatically post-edited.

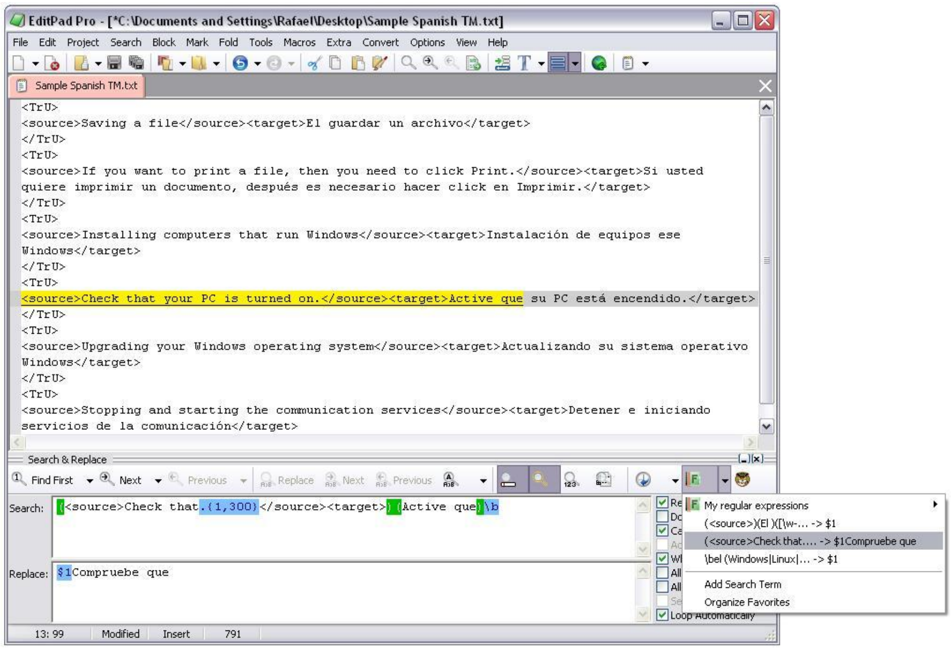
Most of the linguistic error patterns described in this article will look familiar to people involved in manually post-editing Spanish outputs generated by rule-based MT engines: mistranslations of *-ing words,* subordinate clauses, and verbs. The regular expressions used to fix the problems described in this article can be adapted to other languages.

Providing linguistic context to disambiguate mistranslations

Segments containing the slightest ambiguity are almost doomed to be mistranslated by MT engines. By ambiguity I mean words, phrases, and full sentences that are either missing or have more than one meaning and as a consequence, need disambiguation. Fortunately, many recurring ambiguity patterns in source segments will often be mistranslated in the same or similar way by a rule-based MT system. However, in order to automatically post-edit these mistranslations using regular expressions, some *post-editing context* needs to be provided. A practical way of doing this is aligning *both* source and translation segments side by side to allow regular expressions to look at both of them — not just the translation segments, as it was the case in the examples in my previous article — and fix the problem, if any.

There are tools in the market that provide the possibility of creating translation memories (TMs) by aligning source and translation segments in separate translation units: SDL Trados WinAlign, STAR Transit or the TM Generator Alignment Tool available in Language Weaver. By source segments I refer to the segments entered in an MT system for translation. By translation segments I mean the translation output generated by an MT system, such as WorldLingo, Babel Fish or Sytran desktop client.

Figure 1 shows a sample translation memory (txt format) being automatically post-edited using EditPad Pro, a text editor that supports regular expressions.


EditPad Pro - [*C:\Documents and Settings\Rafael\Desktop\Sample Spanish TM.txt]

```
<TrU>
<source>Saving a file</source><target>El guardar un archivo</target>
</TrU>
<TrU>
<source>If you want to print a file, then you need to click Print.</source><target>Si usted
quiere imprimir un documento, después es necesario hacer click en Imprimir.</target>
</TrU>
<TrU>
<source>Installing computers that run Windows</source><target>Instalación de equipos ese
Windows</target>
</TrU>
<TrU>
<source>Check that your PC is turned on.</source><target>Active que su PC está encendido.</target>
</TrU>
<TrU>
<source>Upgrading your Windows operating system</source><target>Actualizando su sistema operativo
Windows</target>
</TrU>
<TrU>
<source>Stopping and starting the communication services</source><target>Detener e iniciando
servicios de la comunicación</target>
```

```
Search: (<source>Check that.{1,300}</source><target>)(Active que)\b
Replace: $1Compruebe que
```

```
My regular expressions
(<source>)(El )([\w-... -> $1
(<source>Check that.... -> $1Compruebe que
\bel (Windows|Linux|... -> $1
Add Search Term
Organize Favorites
```

Figure 1: *EditPad Pro*'s *Search* and *Replace* fields support the use of regular expressions to search for and fix linguistic recurring errors in text files (e.g. Trados TMs in txt format).

Each TU contains both a source and a translation segment, each one of them separated by self-explanatory tags:

<TrU>
<source>Printing a document**</source><target>**Imprimiendo un documento**</target>**
</TrU>

These surrounding tags are key anchor points for letting regular expressions know where each source and its corresponding MT translation segment starts and ends. Otherwise, it would not be possible to distinguish between them.

Post-editing MT outputs automatically using regular expressions

Once both the source and its MT translations are aligned in a TM text format, it is just a matter of opening this text file in a text editor that supports regular expressions and search for typical errors patterns in the translation segments associated with a specific ambiguity problem in the source segment (such as the scenarios described in the next sections of this article) and replace the mistranslations with the correct text.

As it can be imagined, copying and pasting regular expressions manually, one by one, in a text editor to search for different patterns can be very tedious. Interestingly enough, text editors such as EditPad Pro can store regular expressions in a library. The user can then automate the post-editing task by recording one or more macros in this tool to run all the regular expressions sequentially on the TM (Figure 1). This tool functionality is fine when dealing with just a few regular expressions. On the other hand, sometimes it may be necessary to apply a hundred regular expressions or more in order to automate a significant amount of post-editing. When this is the case, it will be more practical to develop some type of in-house script that will sequentially execute as many regular expressions as necessary. This should be a rather straightforward job for an experienced programmer.

Finally, once all the regular expressions have identified and fixed all the mistranslation errors in the translation memory caused by ambiguity, human post-editors will need to check and fix manually the remaining errors that could not be found using regular expressions. At this point, a vast amount of tedious work should have been spared to the human post-editors. Once the TM has been appropriately post-edited, the TM will be ready for future leverage.

The rest of this article will describe a number of typical Spanish mistranslations created by rule-based MT systems and how to fix them using regular expressions. All the regular expressions for all the examples described in this document are Perl-compatible and will assume TUs in the format described in the sample TM of Figure 1.

Scenario #1: Mistranslations of *–ing* words

This section will describe some examples of typical mistranslations of *–ing words* and how they can be fixed using regular expressions.

–ing words are formed by adding the *–ing* inflection to the base of the verb: *open – opening*; *view – viewing; shop – shopping; import - importing.* Their actual meaning and translation in Spanish will always depend on the part of speech (verb, noun, adjective, and adverb) and the function they perform in a sentence (subject, direct object, or subject complement). For instance, the verb *to import* in the sentence "Importing files saves time" has become an *–ing word* that functions like a noun (English gerund) and "Importing files" is functioning as the subject of the sentence. On the other hand, the verb *to import* in the sentence "Windows is importing files" is a present participle preceded by a conjugated form of the verb *to be*, and their combinations form a verb in the present progressive tense.

It should also be clarified that English and Spanish gerunds are not the same thing. Generally speaking, the English gerund is an *-ing word* that functions like a noun. The Spanish gerund (*gerundio*) is a *present participle verb formed by adding -ando or –iendo to the base of the verb (importando; abriendo)*. It always occurs in combination with *ser/estar* in a progressive tense (*está importando; está abriendo*).

Let us begin by reflecting on the ambiguous source segment in the example of Table 1.1 and Table 1.2: *Importing files*. In principle, this may give the impression of being exactly the same sentence in both tables and so does the translation generated by MT: *El importar archivos*. But is this really the case?

The first problem that stands out is the Spanish definite article *El* preceding the infinitive verb: *El importar*. This is a common MT error when translating some *–ing words* and should be corrected by removing *El* in the translation of both examples. But, how can a regular expression know for sure that *importar* is actually an infinitive and not a noun or something else? Here is the first ambiguity problem that needs to be resolved.

Spanish infinitives may end in *-ar* (first conjugation), *-er* (second conjugation), and *–ir* (third conjugation). This is useful information, but not enough to make a solid determination, because these endings can also appear in Spanish nouns like *primer* and *router*. However, if we also know that the first word in the source segment ends in *–ing*, we can be almost certain that *El importar* is a mistranslation of an *–ing word* and, therefore, the article *El* can be safely removed*.*

Table 1.1: Mistranslation of -*ing words* as *El* + *infinitive* in titles.

Raw MT output	<source>Import**ing** files</source><target>**El** import**ar** archivos</target>
Automatically post-edited MT output	<source>Import**ing** files</source><target>**Cómo importar** archivos</target>
Regular expression	**search=**(<source>[\w-]+ing.{1,100}<target>)(El)([\w-]+[aei]r\b)(.{1,150}[^\.\?:!]<) **replace=**$1Cómo $3$4

Table 1.2: Mistranslation of *–ing words* as infinitives in software process descriptions.

Raw MT output	<source>Import**ing** files...</source><target>**El** importar archivos...</target>
Automatically post-edited MT output	<source>Import**ing** files...</source><target>Import**ando** archivos...</target>
Regular expression	**search=**(<source>[\w-]+ing.{1,100}<target>)(El)([\w-]+)(ar\b)(.{0,100}\.\.\.) **replace=**$1\u$3ando$5

The next step is deciding what to do with the Spanish infinitive verb. This will depend on what the *–ing word* in the source text really is and the function it performs. This is also ambiguous because there is something being omitting in both sentences. *Importing files* could be either a noun phrase or a present progressive verb phrase. We can assume that if the *–ing word* occurs in a title, the *–ing word* should normally be translated as a noun (*Importación de archivos*), or paraphrased as *Cómo* (How) followed by the infinitive: *Cómo importar archivos* (How to import files). A way of guessing if the sentence is a title is by checking that the sentence does not end in a period, question or exclamation mark (Table 1.1). On the other hand, if the *–ing word* occurs in a sentence that ends in three dots (Table 1.2), chances are that the *–ing word* forms part of a present continuous verb (*to be* + present participle) that is describing an action or process happening now. In this example, the conjugated verb *to be* preceding the *–ing word* is implicit in the sentence. This type of strings is typical in the graphical user interface (GUI) of software applications to notify the user of a software process currently taking place.

Let us see now how the proposed regular expressions solve all these ambiguity problems and how the mistranslations get post-edited with the correct text.

As it can be seen in the *search* field of Tables 1.1 and 1.2, there are four different groups captured between brackets. In the first group, the *(<source>[\w-]+ing)* expression checks if there is a word ending in *–ing* at the beginning of the source segment. The expression *.{1,100}<target>* means any number of characters between 1 and 100 (i.e. it doesn't matter what follows) until the beginning of the target segment is reached. The second group captures the Spanish article *El* that will need to be replaced by *Cómo,* provided the sentence is a title. The *([\w-]+[aei]r\b)* expression in the third group captures the word that ends in *–ar, -er, -ir* next to *El in the translation segment*. If the full pattern (the three captured groups) is matched, the regular expression will *assume* that the first two words in the target segment are the Spanish article *El* followed by an infinitive. Up to this point, the search strategy is almost identical for both examples.

The fourth group in the searching expression basically checks how each sentence ends. This is a key anchor point that will enable the regular expression to identify the type of sentence and the solution to apply. This last part of the searching regular expression varies in both examples. For instance, in Table 1.1, the last part of the searching regular expression (*.{1,150}[^\.\?:!]<)* places the condition that the translated segment should not end in a period, comma, or question mark in order to be considered a title. Because this condition is matched in Table 1.1, the expression in the *replace* field will fix the mistranslation by simply keeping the first and third groups (*$1* and *$3*) but replacing *El* with *Cómo*. The last part of the searching expression in Table 1.2 checks if the sentence ends with three consecutive dots: *(.{0,100}\.\.\.)*. Because this condition is matched, the regular expression knows that the Spanish infinitive needs to be converted into a Spanish gerund. This is done by the regular expression in the *replace* field: *$1\u$3ando$5*. Everything is kept except the *El* article. which simply gets removed, and the infinitive's ending (*-ar*) is replaced with the appropriate ending (*-ando*) to turn the infinitive into a Spanish gerund (*Importando*). It should also be pointed out that for simplicity's sake, the regular expression in Table 1.2 ignores infinitives ending in *–er, -ir* (second and third conjugation respectively). These would require two separate regular expressions.

As Table 1.3 shows, some English gerunds occurring in titles (i.e. *-ing words* functioning as nouns) may get mistranslated as a Spanish gerunds. Most of the times, ambiguity is behind this problem. The searching pattern of the regular expression is very similar to the previous examples. The main difference is that it searches for a Spanish word ending in *–ando*, instead of *El* + *infinitive*. If found, the regular expression inserts *Cómo* at the beginning of the sentence and converts the Spanish gerund into an infinitive by just dropping *–ando* and adding *–ar to the base of the verb*.

Table 1.3: Mistranslation of *–ing words* as Spanish gerunds in titles

Raw MT output	<source>Upgrad**ing** your Windows operating system</source><target>Actualiz**ando** su sistema operativo Windows</target>
Automatically post-edited MT output	<source>Upgrading you Windows operating system</source><target>**Cómo actualizar** su sistema operativo Windows</target>
Regular expression	**search=**(<source>[A-Z]\w+ing\b.{1,300}<target>)([A-Z][\w-]+)(ando)\b **replace=**$1Cómo \l$2ar

Finally, the example in Table 1.4 shows an example of a gerund mistranslated as a Spanish gerund, instead of a noun or infinitive, when preceded by verbs such as *See*, *Consult* or the preposition *About*.

Table 1.4: Mistranslation of *-ing words* as Spanish gerunds when preceded by *About*, *See*, *Consult*

Raw MT output	<source>**See** "Sav**ing** a file" on page 25</source><target>Ver "Guard**ando** un archivo" en la página 25</target>
Automatically post-edited MT output	<source>See "Saving a file" on page 25</source><target>Ver "Cómo guard**ar** un archivo" en la página 25</target>
Regular expression	**search=**(Ver|Vea|Véase|Consulte|Acerca de|Sobre|Información sobre)(.{0,2})([\w-]+)(ando\b) **replace=**$1$2Cómo \l$3ar

This mistranslation type probably occurs because the MT system interprets the *–ing word* as the way in which the user needs to see or consult whatever follows in the sentence. Because this error is obvious enough, the proposed regular expression in this case does not check the existence of an *–ing word* in the source. Instead, it assumes that if the word following *Ver* (See), *Consultar* (Consult), *Acerca de* (About) ends in *–ando*, there has been a mistranslation of a *–ing word* as a Spanish gerund. The proposed solution is almost identical to the example of Table 1.3.

Scenario #2: Mistranslation of subordinate clauses

Mistranslation of subordinate clauses by MT usually happens because subordinating conjunctions such as *that, then and while* have more than one meaning, or simply because the conjunction is missing in the sentence, resulting in ambiguity.

Table 2.1 shows a mistranslated sentence caused by the absence of *while* to introduce the subordinate clause. From the point of view of the MT system, the source segment means that the computer was not supposed to download an update at all (i.e. the error was downloading the update). A human translator would have easily realized that the intended meaning in Spanish is "there was an error *while* downloading the update". Sometimes, this type of ambiguous pattern can be found in software warning messages: *There was an error* + *-ing word*.

Table 2.1: Mistranslation of subordinate temporal clauses

Raw MT output	<source>There was an error downloading the update.</source><target>Hubo un error **que** descarg**aba** la actualización.</target>
Automatically post-edited MT output	<source>There was an error downloading the update.</source><target>Hubo un error **mientras se** descargaba la actualización.</target>
regex	**search=**(Hubo un error) (que) ([\w-]+aba\b) **replace=**$1 mientras se $3

Because this mistranslation pattern is so obvious, the regular expression does not need to check the existence of the *–ing word* in the source. Instead, it just searches for the pattern *Hubo un error que* in the translation segment. Then *que* gets replaced by *mientras* (while) followed by the pronoun *se*.

Table 2.2 shows a typical mistranslation pattern of relative *that-clause*s consisting of mistranslating *that* + verb (present tense) as *ese* (instead of *que)* + past-participle verb functioning as an adjective (instead of a verb in the present tense).

Table 2.2: Mistranslation of subordinate relative clauses

Raw MT output	<source>Installing computers **that** run Windows</source><target>Instalación de equipos **ese ejecutados** Windows</target>
Automatically post-edited MT output	<source>Installing computers that run Windows</source><target>Instalación de equipos **que ejecutan** Windows</target>
Regular expression	**search=**(<source>.{0,300}that.{1,300}<target>.{0,300})(ese)(.{1,300}[\w-]a)(d[oa]s\b) **replace=**$1que $3n

Spanish regular past participle verbs are formed by simply dropping the infinitive ending (*-ar, -er, -ir*) and adding *-ado* (for *-ar* verbs) or *-ido* (for *-er, -ir* verbs). For instance, the past participle of *ejecutar* (to run) is *ejecutado* (run). Unfortunately, there are also irregular verbs. For example, the past participle form of *abrir* (to open) is *abierto* (not *abrido*). For the sake of clarity, only Spanish past participle verbs functioning as adjectives ending in *–ados, -adas* are considered in this example.

Using all this information, the regular expression in the *search* field (Table 2.2) checks if *that* occurs in the source segment. Then it checks if *ese* occurs in the target segment followed by a past participle verb working as an adjective: *(ese)(.{1,300}[\w-])(ad[oa]s\b).* If this pattern is found, it is safe enough to assume that *that* has been mistranslated and so has the verb following it. The expression in the *replace* field will just replace *ese* with *que* (second group surrounded with parenthesis). Then it will drop the *–do(s) or –da(s)* verb ending and add *–n* to convert the verb into a present tense verb in plural.

Table 2.3 shows another typical scenario of mistranslation of subordinate clauses introduced by *then.* In instruction manuals, *then* usually has two possible meanings: (1) *Next, afterward*. In Spanish, this translates as *luego, después, or a continuación*. (2) *In that case*. This usually translates as *entonces* in Spanish.

In table 2.3, *then* has been mistranslated as *después*, instead of *entonces*, giving the sentence the wrong meaning. Obviously, the solution to this problem can not be simply searching for the word *después* and replace it with *entonces* because it could actually be correct, depending on the context.

Table 2.3: Mistranslation of subordinate conditional clauses

Raw MT output	<source>**If** you want to print a file, **then you need to** click Print.</source><target>Si usted quiere imprimir un documento, **después** es necesario hacer click en Imprimir.</target>
Automatically post-edited MT output	<source>If you want to print a file, then you need to click Print.</source><target>**Si** usted quiere imprimir un documento, **entonces** es necesario hacer click en Imprimir.</target>
Regular expression	**search=**(<source>[Ii]f\b.{1,300}<target>Si.{0,300})(después) (es necesario)\b **replace=**$1entonces $3

One possible way of disambiguating the real meaning of *then* is assuming that when *then* means *in that case* it will often follow the following pattern in the source and translation segments respectively: *If... then... Si... entonces...*. It could even be assumed that, in these cases, *you need to*/*have to*/*must* will often follow the *If* conditional conjunction in the source segment. When this is the case, *then* should be translated as *entonces es necesario*, *in the translation segment, as shown in Table 2.3*.

Scenario #3: Mistranslation of verbs with several meanings

One of the potential down-sides of MT dictionaries in rule-based MT systems is that that there can be only one term for each concept. Obviously, this can be a serious problem if the terminology in the source text is not *controlled*. For instance, the verb *to check* can have two possible translations in the context of GUIs: *comprobar* (to confirm or verify something) or *activar* (to activate or enable an option in a dialog box, etc). Suppose that for some reason it suits better to encode *to check* as *activar*, instead of *comprobar*, in the MT dictionary. In this case, *to check* will be mistranslated in the following sentences:

Check that there is power.
Turn on the switch *to check that* you have power.
You can *check to* see *if* there is power.
What is the file *to check?*

The good news is that most of these scenarios present useful anchor points that can be used by regular expressions to identify the ambiguity problem and fix it. Sometimes, however, this will not be 100% crystal clear. For example, how should *to check* be translated in "Y*ou need to check this option"*?

Table 3: Mistranslation of verbs with several meanings

Raw MT output	<source>**Check that** your computer is turned on.</source><target>**Active que** su equipo está encendido.</target>
Automatically post-edited MT output	<source>Check that your computer is turned on.</source><target>**Compruebe que** su equipo está encendido</target>
Regular expression	**search=**(<source>Check that.{1,300}</source><target>)(Active que)\b **replace=**$1Compruebe que

For example, the regular expression in the *search* field of Table 3 focuses on checking if *Check that* is mistranslated at the beginning of the source segment as *Active que* (instead of *Compruebe que)*. If this is the case, the regular expression just leaves everything as it is, replacing just *Active que* with *Compruebe que*.

Conclusion

Regular expressions provide plenty of scope to automatically post-edit MT mistranslations caused by ambiguities in the source text. This requires two main ingredients: recurring patterns and linguistic context. The latter can be provided by aligning source and the machine translated segments in a TM which must be subsequently post-edited using regular expressions. These need to be carefully tested and fine-tuned before hand, otherwise side-effects may occur.

On a final note, it is often argued that rule-based MT (as opposed to statistical MT) is restricted to the development of MT dictionaries and that this is time-consuming, expensive and can only fix a subset of the MT system's problems. However, one of its big advantages is that the output (good or bad) generated by rule-based MT systems becomes *predictable* to a large extent after several analyses. This is precisely the main ingredient that allows for automatic post-editing using regular expressions.

References

Aranberi, N. 2007. Exploding the Myth (The gerund in machine translation). *In: The LRC XII Conference.* 27/28 September 2007, Dublin. Limerick: The Localisation Research Centre, pp. 211-219

Arnold, D. 2003. Why translation is difficult for computers. *In: H.* Sommers, (ed). *A translator's guide: Computers and Translation,* Philadelphia: John Benjamins B. V., pp. 119-141

Dugast, L., Senellart, J. and Koehn, P. 2007. Statistical Post-Editing on Systran's Rule-Based Translation System. *In: Second Workshop on Statistical Machine Translation*, 23 June 2007, Prague. Madison: Omnipress, pp. 220-223.

Guzmán, R. 2007. Automating MT post-editing using regular expressions, *Multilingual Computing*. 18 (6), pp. 49-52.

Guzmán, R. 2007. Manual MT Post-editing: if it's not broken, don't fix it!, *Translation Journal* [online]. 11 (4), [Accessed 1st October 2007]. Available from World Wide Web: <http://www.accurapid.com/journal/42mt.htm>

Regular-Expressions.info – Regex Tutorial, Examples and Reference – Regexp Patterns. 2007. [Accessed 1st February 2007]. Available from World Wide Web: <www.regularexpressions.info>

Simard, M., Goutte, C., Pierre, I. 2007. Statistical Phrase-based Post-editing. In: *Proceedings of NAACL-HLT-2007 Human Language Technology: the conference of the North American Chapter of the Association for Computational Linguistics*. 22-27 April 2007. New York, pp. 505-515.

About the author

Rafael Guzmán's personal website is at www.rafaelguzman.ie