

DICTIONARY COORDINATION

Jacqueline Guazzo-Jansen  
Systran Institut, Luxembourg

Dictionary coordination means carefully controlling each entry of each dictionary and comparing the various dictionaries among themselves.

In machine translation, as opposed to human translation, the sacred word is consistency.

Man can be absent-minded and/or creative - which means that an identical source sentence might be translated in different ways.

Machine translation, compared to human translation, is clumsy, heavy, and not always perfectly accurate.

On the other hand, Machine Translation has 2 advantages over man:

- it is faster,
- it is consistent.

Same input, same output:

The same meaning will be taken over and over again for the same word in the same field.

The same sentence will be parsed the same way, synthesized the same way and will produce the same results!

(Of course, this implies that the same versions of the programs and of the dictionaries are used).

A. Problems

This is how it should be! As the situation is now, it is unfortunately not happening that way!

I. Consistency of quality

The French verb 'garder' might be translated by 'keep'.

The French verb 'continuer' might be also translated by 'keep'.

Now, let's imagine that 'keep' (garder) was coded as a regular English verb! While the 'keep' for 'continuer' was coded correctly.

Then we might get 2 sentences in a text:

Il continua à fonctionner sans problème.  
It kept working without problem.

Il garda la température constante.  
It kepted the temperature constant.

- 2 -

Apart from the fact that a form like 'keeped' will be spotted immediately by the revisor, what will strike the end-user is that the computer sometimes makes mistakes and sometimes doesn't. The computer is unreliable and inconsistent.

For people who were told that MT is not perfect, but stable, this is the counter-example, which will be both unexplainable and unforgivable!

Not only is MT far from perfect, it is also unpredictable!

This is worse for the image of MT than not-found-words or a scrambled sentence, once in a while.

Maybe my example is a bit exaggerated.

Let's be more subtle:

We have codes which induce the gerund in English.

("GG" Governs Gerund and "GOG" Governs Object and Gerund)

We can easily imagine one coder coding 'arrêter (de)' as 'stop' + gerund, and another one (or the same one, for that matter) coding 'cesser (de)' as 'stop' without the code for gerund.

Therefore, we might see the sentence:

'The machine stops working when current flow stops to pass through'.

We could easily multiply examples with all the synthesis codes we have in Systran.

The same situation will happen even more often with two systems translating into the same target language.

A customer might get on his desk (or his screen) 'He stopped working' from French and 'He stops to work' from German.

It is irritating.

For two systems sharing the same source language, the problem is the same, just more common because the coding of the analysis is more sophisticated. Let's imagine 'book' being coded as a homograph noun/verb in the English-Italian dictionary and being only a noun in the English-French. 'I book my ticket' will appear as 'Prenoto il mio biglietto' (I reserve my ticket) and 'Je livre mon ticket' (where 'livre' is 'book', not 'deliver'...)

'Stop' should be coded "usually transitive", "governing gerund" and "object plus gerund", with various meanings for different syntactical contexts.

intransitive stop

= to come to a standstill

intransitive stop + gerund

= to cease

transitive stop + gerund

= prevent somebody from doing ...

- 3 -

transitive stop without gerund  
= to fill

Let's suppose now that the English-Italian coder did his work perfectly and coded 'stop' carefully.

In Italian, the sentences

'He stopped the committee meeting'  
and  
'He stopped smoking cigars'

will be translated as

and 'Ha impedito al comitato di incontrarsi'  
'Ha smesso di fumare cigari'.

If the French coder was in a hurry, he maybe forgot to use the GG code ... or the GOG code, or both.

As a consequence, although the programs of English analysis are the same for E-F and E-I, in French, the 2 sentences might come out:

1) GOG is there ... but not GG:  
  
Il empêcha le comité de se rencontrer.  
Il boucha les cigares fumeurs.

or

2) GG is there but not GOG:  
  
Il cessa de rencontrer le comité.  
Il cessa de fumer des cigares.

or

3) Neither GG nor GOG:  
  
Il boucha la réunion du comité.  
Il boucha les cigares fumeurs.

No end-user would ever believe this is only due to a code missing - he would rather think that the Italian synthesis is far more developed than the French and if he encounters later in the text: 'he prevented his mother from coming' coming out 'Il empêcha sa mère de venir' (a perfectly constructed French sentence), he will come to the conclusion that Systran is unpredictable.

Changes in the analysis might be caused by missing codes, wrong codes, superfluous codes...

- 4 -

The complexity of SYSTRAN coding added to the multiplication of different personalities among the coding staff, the depth of their knowledge of each of the two languages - source and target - they are coding, the circumstances of the coding (under pressure or in a hurry, one tends to produce lower quality work) makes the probability of coding 'stop' in E-F, E-G, E-I, E-A, E-S etc... in precisely the same (exact) way, lower and lower and lower.

Embarrassing, if we remember that usually customers translate from one source language into two or more target languages, rather than using language pairs in both senses. (A typical example is a maintenance manual to be made available for various countries).

They will therefore notice almost certainly the divergencies in sometimes short, easy sentences!

This is due to our bilingual, language pair-oriented dictionaries.

This philosophy is wrong - it obliges different coders to go through their coding manuals to find the right codes for the same words, to fill out loads of coding sheets which will be inputted by different keypunchers etc...

Each of them might make a mistake!

All of them, but the first one, are doing a redundant job.

Waste of time, waste of energy and waste of money.

We cannot undo what was done, but we can foresee some better policy for the future!

A first step has been what we call 'the marriage of dictionaries'.

The English-German system was developed years after the English-French.

It seemed essential to have the E-G dictionaries profit from all the work done for the E-F, on the English side. Therefore, the huge E-F dictionaries were copied into the small E-G, ready for a German meaning to be coded.

The same was done between E-A and the now big E-G etc...

A 'marriage' compares a slave to a master.

If an entry of the slave is not yet in the master, it is added without the target part.

If an entry of the slave is already in the master, the two entries are carefully compared:

If identical, the master's entry is kept.

If any difference is noticed, the computer will keep the master's entry but give a message about the difference.

A linguist should go through these messages and correct master and/or slave dictionaries.

This is easily done, if the master is quite small - a pilot dictionary of a 1000 words - so that the entries already in are reduced to a minimum.

- 5 -

It becomes critical to analyse the 'error' messages, if the master is almost as big as the slave, because the entries in common will be many and for the computer a C-line with 'CON,CT' is not equal to a C-line with 'CT,CON'! You can imagine what a task it is to get the real differences!!

And what about two homographs which are both in, in both dictionaries but with inverted DC codes.

00BOOK noun      00BOOK verb  
01BOOK verb      01BOOK noun

Each line of these entries will diverge, although in this case it doesn't really matter, if you put the noun or the verb first.

Marriages are helpful when developing a new system, but they are not a miracle solution.

## II. Consistency of quantity

If we take back the case of our end-user who wants to translate his manuals into 5 different languages, we have to face another essential problem: Dictionaries sharing the same Source language should include the same terminology.

If customer 'X', a well-known computer manufacturer, decides to translate his manuals with Systran, the specific terminology he needs should be in (or added to) all the language pair dictionaries he will use.

If ENG-FRE contains 95% of the required terminology but ENG-ITL only 50%, and ENG-GER, hardly 10% ... the average end-user will not congratulate us for the ENG-FRE adequacy, but will complain about the amount of terminological investment to be made to the ENG-GER.

It is therefore essential to keep all these dictionaries at the same level. This takes us again to the marriage of dictionaries, which will bring young (or less complete) dictionaries to the level of an old bigger reference master.

This will not help for words which would be in the dictionaries, at the source level, but would not have specific meanings for a given field, because a marriage compares source information, not meanings.

On the other hand, as said before, the marriage is a cumbersome tool for everyday life, ie. for keeping two dictionaries identical.

Presently, in Europe we don't have the means to code in parallel one English word or expression with all the meanings in the various target languages. I think WTC took some steps in that direction in the framework of Universal Systran and maybe Systran Japan has solutions or systems to cope with this problem.

### III. Consistency of versions used

Finally, the last part of coordination should be within one language pair.

If six users require development in Eng-Fre, by coding or asking to code their not-found-words, in six different service centers - all six will take part in building a more complete lexical data base.

I don't need to convince you that the development done for (or by) No 1, should automatically profit No 2 through 6 and vice-versa.

### B. Solutions

To our first 2 problems, there exists a solution: Systran dictionaries should not be bilingual anymore but re-conceived as a data base. Each entry would be coded by a native speaker, who would bother about its translation or ... the source word it is the meaning of.

Each entry is therefore entered once and for all with all its codes, source and target.

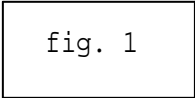


fig. 1

This would be done for all the existing entries in all languages available. If one language is used only at the source level or at the target level, the other field will remain empty for later completion.

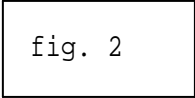


fig. 2

The bilingual coding would be reduced to source word, POS, target word and the E-line, if needed.

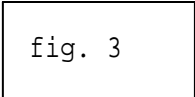


fig. 3

- 7 -

A first update would compare both source word and meaning to a table of the data base where all entries would be listed with their address.

fig. 4

We would deal with different types of 'Not-found-words':

- 1) The word is not in the data base: it could be a source word or a meaning.
- 2) The word is not in the bilingual dictionary used.

BOOK 10  
KITAEB

'BOOK' would look up the English data base for BOOK, part of speech noun, KITAEB would look for KITAEB in Arabic.

fig. 5

&amp;

Same approach for different fields (or TG Topical Glossaries). The German MUTTER would go to the address of the English word MOTHER as general meaning and to the address of the English word 'NUT' in mechanics.

The bilingual dictionaries would only contain addresses.

fig. 6

This approach would solve the problem of consistency both at source and target levels.

However, the problem of using the same bilingual dictionaries is not solved by the data base. The data base would contain monolingual master words, but the bilingual glossaries referring to the data base are dynamic entities, which can be built in different places, by different people using different terminological material.

Systran centers are many in the world and far from identical.

We have (1) Development Centers  
(2) Service Centers  
(3) Customer's Installations

While (3) are usually oriented towards a specific field (eg. DORNIER, KFK, Aerospatiale), (2) and (1) have by definition a 'universal vocation'.

On the other hand, (2) and (3) are rarely involved in their own development - although there are exceptions - while (1) develop both algorithms and dictionaries on the basis of the end-users' feedback.

Finally, (2) and (3) have no contacts between each other, while (1) should have - 'should have' but this doesn't mean that it does actually happen.

Development Centers certainly are adapted for centralization and coordination - their staff are experienced in all levels of coding and programming; they sell (or rent) systems to service centers and/or end-users.

Most of the customers have loaded versions of programs and dictionaries. It is out of the question for them to run updates and they delegate this task to the development center they have connections with.

There are two cases:

- 1 - They have their own coders who draw up update files which are sent for update.
- 2 - They ask for terminology to be added in their field. The development center executes the coding and update.

In the first case, the development center must control the coding of the customer before updating the dictionary.

In all cases, the development center is in charge of updating the dictionaries with the entries requested by the customers and giving to each customer a new loaded version containing that terminology.

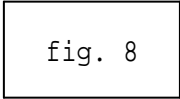
Controlling takes time - but this aspect of the problem would disappear completely with the data base, since the Systran coding would be carried out by qualified staff and the customer would produce only very simple personal bilingual glossaries, without codes, except for the part-of-speech.

Dealing with one customer is easy because it is a one-to-one relationship.

fig. 7
--------



Dealing with two customers in different fields is still all right:



If C1 airplane manufacturer gets, in the new version of the dictionaries the nuclear terminology of C2, and vice-versa, it is probably useless, but certainly not harmful.

On the other hand, the general words (previously not-found) coded for C1 will be available for C2 and reverse. Not only is this useful, but the contrary would be absurd.

Can you imagine, for instance, 'Under-evaluate' being coded for C1 and going back to C1's version only, because C2 didn't ask for it?

It becomes really complicated for DC when customers are numerous - and it happens for Service Centers which can easily have 10 or more end-users - and more customers share a common field, but not necessarily the same terminology!

Let's imagine two airplane manufacturers, one French and one German using both the Fre-Ger and Ger-Fre systems!

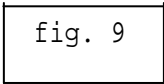
Then both ask for terminology in their field ... and propose different meanings for the same source words.

Customer-specific terminology and Customer's terminological preferences are realities to be taken into account.

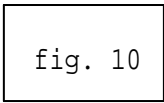
What type of coordination can there be for this case?

First, during a common update, conflicts will occur immediately, because the same term (or the same expression) will be entered twice (or more).

Second, if that problem is overcome by human intervention (running two updates for instance) - Customer 1 will lose his own terminology sooner or later!



1. Customers towards Center



2. Center towards customers

- 10 -

A first help would be to dedicate TGs to Customers rather than to fields, but we are unfortunately limited to  $26 + 9 = 35$  different customers - and some of our customers insist on having different TGs for different applications (eg. DORMER).

Another solution would be to separate customer's dictionaries ... Just the contrary of centralization!

Each customer would have his own bilingual dictionary, including his specific terminology. He could even build it up himself, since the obstacle of Systran coding would have disappeared, and send to the development center the not-found-word lists only, which have to be added to the monolingual data base and introduced to the bilingual ones with a general meaning.

This question is still open.

The other problems we still have to face at the present time are threefold:

#### 1 - Controlling vs censoring

While it is absolutely necessary to check that coding is done properly when the customer does it himself, the coordination center should avoid discussing or refusing meanings (except for spelling errors), because 'Le client est Roi' and should be master of his own terminology.

On the other hand, if two coordination centers take care of different language pairs - a sensible subdivision since one coordination center of all Systran dictionaries is not feasible, each center should respect the coding philosophy of the other.

For instance, if 'A' uses TG's in his approach to customers, as well as in his resolution of stylistic differences by programs, 'B' should not simply suppress the TG meaning, because in his dictionaries TG's are neglected! It could have dreadful consequences at all levels for 'A' and his end-users.

'A' of course, should not impose TG's on an update to be run for 'B'...

#### 2 - An ideal 'turn-over' between updates and re-distribution of new loaded versions should be found by each Development Center.

Running updates too often prevents terminology requests coming in from users who are slower or more distant.

If one waits too long before sending new versions, very dynamic users will grow impatient.

Each center will have its own rhythm according to:

- the number of end-users
- the patience of the end-users
- whether the updates are paid for or not and how much they cost
- whether the terminology is free or not.

- 3 - If the source language of an update is known to be common to more language pairs (like English), these update files should be systematically converted into other bilingual update files, ready for synthesis coding, and that coding done as soon as possible...

However, who would do it?

Who would pay for it?

'A', who sent the file for completion?

'B', who actually does it?

The best answer would be 'reciprocity', but the amount of work for one language pair might be drastically superior to another one and it would become unfair for some partners.

FIGURE 1 one entry in the Data Base

ADDRESS		
WORD	POS	HM connection and cross reference
S	<u>Analysis information</u>	
	Line B POS, MPQ, G, N, CAP, HMYIN, etc...	
	Line C codes...	
	Line D codes...	
T	<u>Synthesis information</u>	
	Line F QMDFQ SEW, MCF, G, etc...	
	F 1, 2, 3...	
	F OS codes...	
	TS codes...	

FIGURE 2

ARB	KITAEB	1	BLANK
S	BLANK		
T	1 123 00200		Singular
	1 123 00100		Plural

ENG	BOOK	1	ADDRESS of IM word
S	B 1010,N,S, BKTB-003		
	C CT,CON		
	D INFORM		
T	1004		

FIGURE 3 entry in a bilingual Master  
 \*\*\*\*\* dictionary

SBC - TOT	
Alimony , 10	
TG = 0	0 Pension 1 Alimentaire F QSWD2
ROCK , 10	
TG = 0	0 LIVRE
Depend , 00	
TG = 0 +	0 DEPENDRE E line : ON=(Beam)
etc...	

FIGURE 4 Data Base

<u>ENGLISH</u>	
00001 ABACUS	C0C41 MERGE
...	...
000B2 ABORT	F2000 RESEARCH
...	...
00483 BOOK	FC532 ZOO
...	...
A8150 KEEP	
<u>FRENCH</u>	
00001 ABRE	A1818 TEMER
...	...
000B5 CHIVRE	CF316 VENIR
...	...
00412 LIVRE	F4444 ZERO
...	...
0A117 PIÉTON	
<u>ARABIC</u>	
00001 ALIF	0C712
...	
00475 KITHIB	
...	
<u>GERMAN</u>	
00001 ABEND	0A477 PROBE
...	...
000A1 NUCB	148BB WAGEN
...	...
00813 LAUFEN	F1000 ZAHLEN
<u>SPANISH</u>	
00001 ABAJO	00C48 LIBRO
...	...
000C5 DISOLUTO	09976 ROJA
...	...
	F7432 ZORRO
<u>ITALIAN</u>	
00001 ACIDO	01133 PIETRA
...	...
000B8 CONTINUARE	14014 TRIESTE
...	...
00A22 LIBRO	F3809 ZUCCHERO
<u>PORTUGUESE</u> ...	
<u>RUSSIAN</u> ...	
<u>NEXT LANGUAGE</u> ...	

FIGURE 5 interrelations between bilingual dictionaries and Data Base

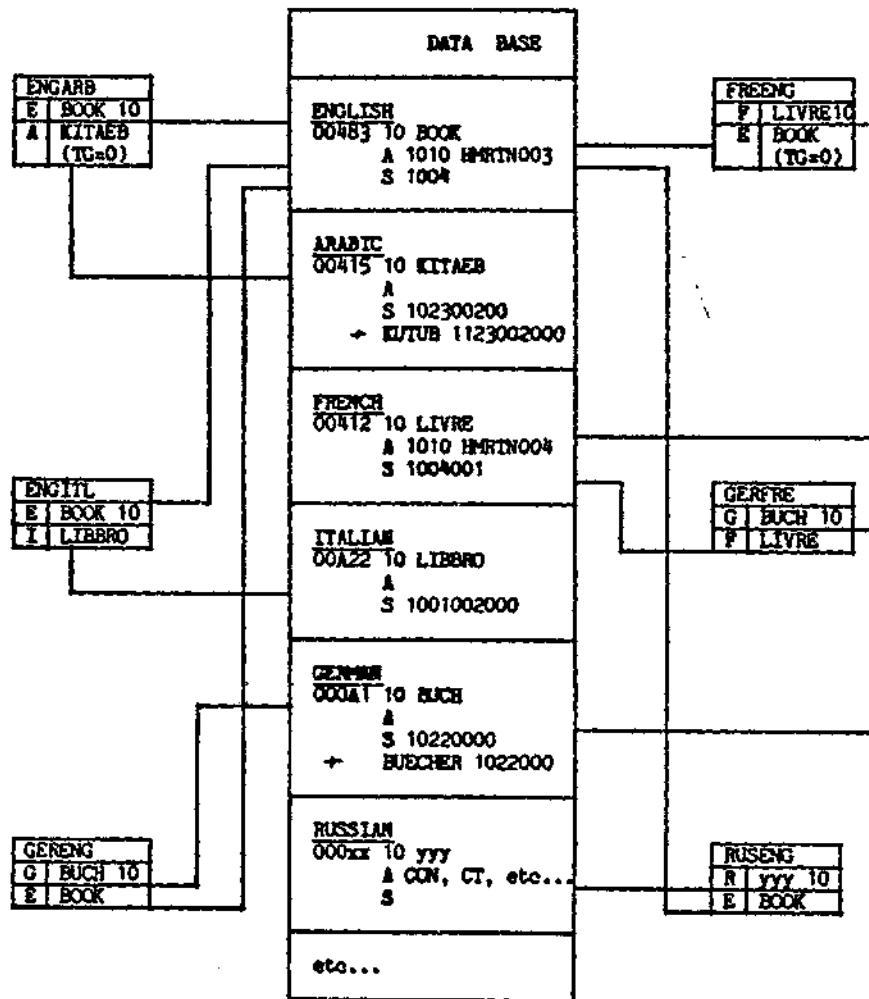




FIGURE 6 entry in a bilingual translation dictionary (after merge)

SRC - TGT	
WORD, POS	
ADDRESS in DB (LANG=SRC Analysis info)	
TG = 0	MEANING 1
ADDRESS in DB (LANG=TGT Synthesis info)	
if Code WD1, WD2 etc...	
	MEANING 1 bis
ADDRESS in DB (LANG=TGT Synthesis info)	
TG = X	MEANING 2
ADDRESS in DB (LANG=TGT Synthesis info)	
if Code WD1, WD2 etc...	
	MEANING 2 bis
ADDRESS in DB (LANG=TGT Synthesis info)	
TG = Y	MEANING 3
ADDRESS in DB ...	
TG = Z	MEANING 4
ADDRESS in DB ...	

FIGURE 7

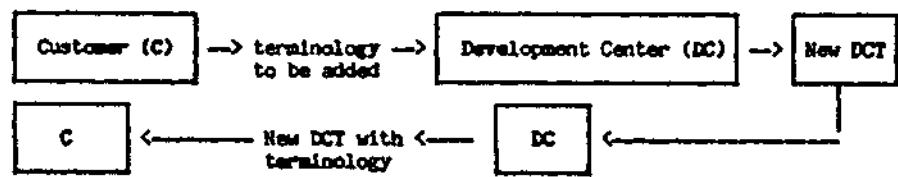


FIGURE 8

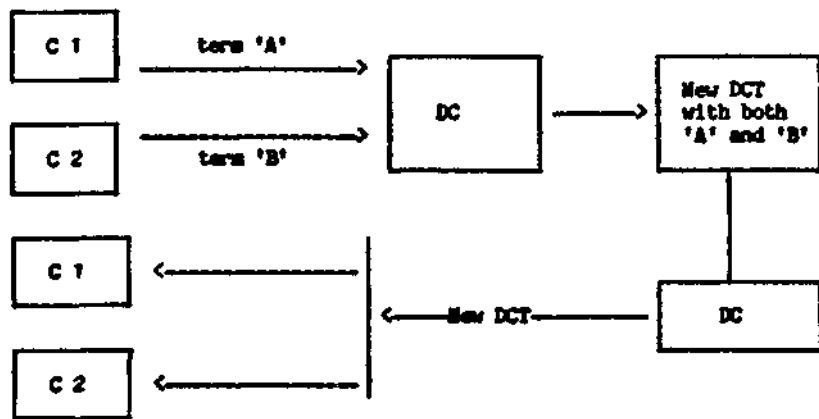


FIGURE 9 Customers towards Center

