ACHIM BLATT

EURAMIS Alignment and Translation Memory Technology

1 Introduction

Translation memory (TM) is a software package for the storage and retrieval of sentences[1] and their translations. Translation pairs are either created during the translation process, or by a sentence alignment program.

The Translation Service (SdT) of the European Commission has bought licences for Translator's Workbench (TWB), which is used interactively on the PC: as soon as a translation for a sentence has been typed in, both the source and the target sentence are stored in the TM, from which they can be retrieved if an identical or a similar sentence (a so-called "fuzzy match") occurs later on in the same document or in a subsequent document for which the same TM is used.

EURAMIS is being developed specifically for the Commission: among other things, it provides an e-mail-based central TM; its design facilitates data sharing and the combination of different products. EURAMIS results can be used both with TWB (via

[1] In general terms, a sentence in the TM context is the same as in general linguistics. For technical reasons, it is however assumed that sentences do not go beyond carriage returns, table cell boundaries and the like; ie a table cell would be regarded as a sentence, even if it consists of a single character.

import) and directly with stand-alone word processing (see " EURAMIS: Added Value by Integration" , p 59).

The following article will not so much give a general description of TM and alignment technology, but rather concentrate on the specific features which have been implemented in the framework of the EURAMIS project.

In order to use TM in such a large translation organisation as the SdT in an optimal way, it is necessary to provide fully sharable central translation memories in addition to the existing TM technology which tends to work on isolated individual TMs of a rather limited size.

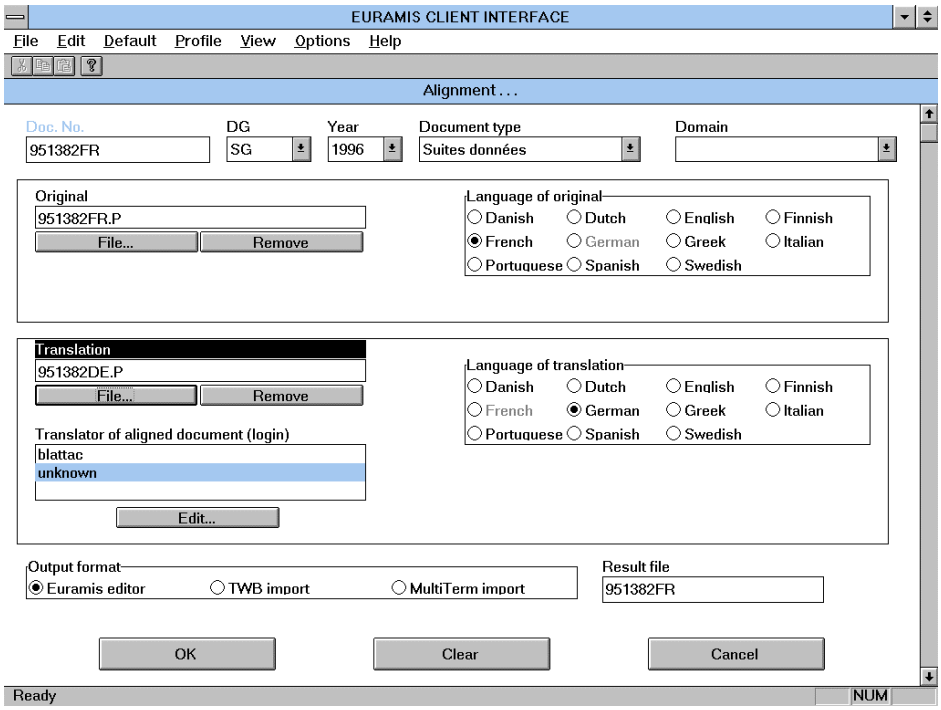
2 Alignment[2]

When work on the EURAMIS project started in the beginning of 1996, the commercially available alignment programs did not offer an acceptable quality. To give an example, one application which was examined more closely had apparently been developed for user manuals and technical documentation produced in an environment where strict formatting rules (eg use of tabulators) can be imposed on technical writers. In such an environment, it is not imperative to offer customisation of sentence segmentation rules and similar functions. On the other hand, these are essential in the SdT as it cannot and does not wish to impose too many rules on its clients. In addition, existing alignment editors did not offer enough functions to handle alignments efficiently. It was therefore decided to develop applications with more functions and more customisation facilities.

2.1 *Alignment request*

Like most of the EURAMIS applications, alignment is offered via an e-mail based client-server environment, ie users launch their requests with the EURAMIS client interface, and later on receive the corresponding result by e-mail:

[2] Throughout this paper, the term " alignment" will be used for what should more correctly be called " sentence alignment" .



In their request, users already add information about the requesting service (" DG"), the year in which the document was produced (it may in fact be aligned only years later), the document type (contract, speech, letter etc) and the domain (agriculture, budget, consumer policy etc). This information will then be available on every sentence pair stored in any EURAMIS or TWB TM.

Apart from the EURAMIS format (which is used by the EURAMIS alignment editor and TMs) and TWB import format, it is also possible to request MULTITERM[3] import format; this has proven useful for a number of reasons:

[3] MULTITERM, the local terminology application used by the SdT, is completely integrated with TWB (see also " EURAMIS: Added Value by Integration" p 59).

- It is possible to align glossaries for traditional use in MULTITERM;

- if MULTITERM is loaded with TWB, smaller text units such as headings (eg 'Character Data and Mark-up') can be reused in larger contexts (eg 'XML consists of intermingled character data and mark-up');

- it appears that sometimes, sentences for which no result is found with TWB, are found in MULTITERM.

2.2 *Core program*

The EURAMIS core alignment algorithm uses the statistical model of sentence lengths which was developed by GALE and CHURCH (1991) and has been used as a basis for many alignment applications. The model of sentence lengths is based on the number of characters in a sentence: it works on the principle that source sentences and their translations tend to have similar lengths. For any possible translation pair, a probabilistic score is allocated which is based on the difference in length of the sentences in question[4]. This score is then used in a dynamic programming framework to find the most likely alignments.

Unlike the original algorithm, EURAMIS alignment is run in two phases: paragraphs are aligned first; on the basis of this, sentences are treated. Since original texts and their translations frequently differ in the number of paragraphs and sentences, both alignment levels support 2-to-1 and 1-to-2 correspondences (provided they fit into the statistical model). Originally, 1-to-many and many-

[4] The general difference in length of the languages involved is also taken into account, eg French texts appear to be about 5 per cent longer than their English equivalents (German: about 10 per cent).

to-1 correspondences had been supported as well, but it turned out that this generally leads to poor results.

Unlike other batch programs of this type, EURAMIS alignment always tries to align all sentences involved. This means that alignment fails, if establishing 2-to-1 correspondences is not enough to cover all the sentences involved (eg because one document contains an annex, the other does not). In such cases, a kind of brute force alignment is carried out (containing a clear warning to the user), which blindly establishes 1-to-1 correspondences from the beginning of each document until the sentences of one document are used up. It is left up to users whether they want to correct such alignments with the alignment editor (see below), or whether they want to adapt the documents to be aligned and submit them again.

In 1996, an early version of the EURAMIS alignment program was compared with a number of commercial products and it turned that it was already at least as good in terms of accuracy (the ratio between correct and overall alignments), and better in terms of coverage (the ratio between aligned and overall sentences). Since then, a number of improvements of the program and more customisation have led to even better results so that for certain text types, perfect alignment without human intervention is not too far.

2.3 *Extensions*

It has been shown that the statistical algorithm suggested by CHURCH and GALE (1991) is more powerful than other approaches both in terms of accuracy and coverage (see BLANK 1995 and 1997). Since no linguistic information is needed, this is even truer if a large variety of documents and languages have to be treated, as is the case in the SdT.

It is therefore astonishing that people with little or no knowledge of the languages involved (eg secretaries in support units who work on a large number of languages) are able to correct alignment results quite reliably. The reason for this is that they apply a kind of pattern recognition scheme in both source and target sentences.

It therefore appeared promising to come back to an older approach (the so-called "linguistic" or "lexical" approach) which tries to link sentences by word correspondences (see KAY and RÖSCHEISEN 1988).

EURAMIS has been extended in such a way that a "light" lexical approach is used on top of the statistical approach. The idea is to define a kind of grid on the basis of "anchors" and to limit statistical alignment to the contents of parallel grids. In order to keep performance high, only such anchors are used for which no heavy dictionary lookup is needed: numbers, all kinds of sentence enumeration, acronyms[5], but also formatting information such as styles. A first version of this combined approach has been implemented recently. In order to make this extension fully operational, some fine tuning is however still needed.

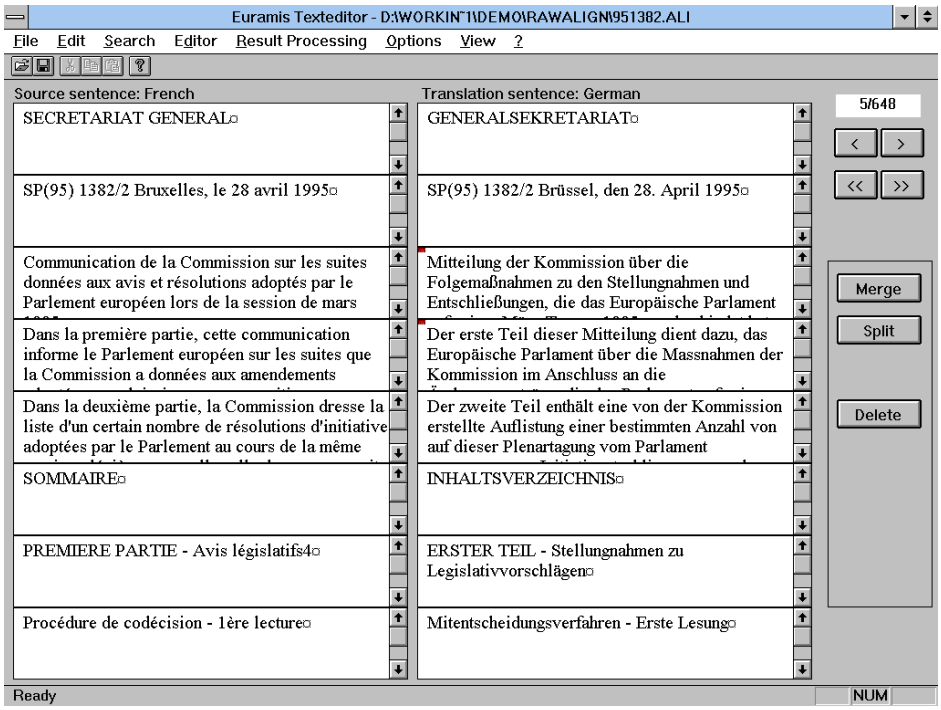
[5] The term "acronym" is defined in a very superficial way here, ie any sequence of upper case characters.

There is also a first version of multilingual alignment, which is nothing more than a small application which takes as input a number of bilingual alignments and merges them together into one file. The advantage of this is that it appears more economic to correct one file with, say, three languages than two bilingual alignments.

2.4 *Alignment editor*

The EURAMIS alignment editor is probably the application which is most appreciated by users, because it is simple to use and at the same time offers all the functions users might want when they correct alignment output.

The general idea is that users look at alignment results screen after screen. As long as results are correct, users continue using " page down" as they run over the pages. The figure below shows what the interface looks like:

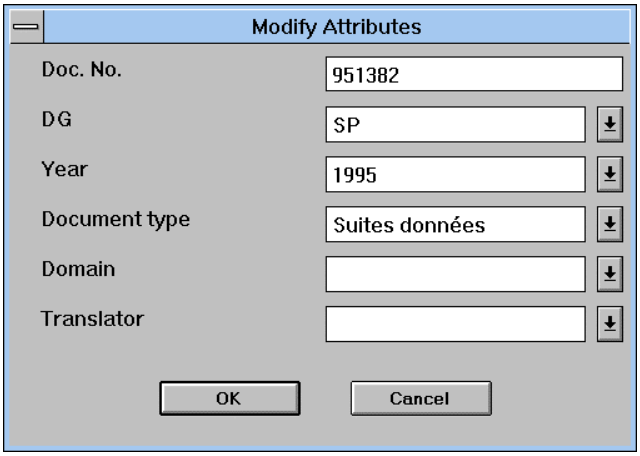


The screen can be customised by changing the number of rows, the font and the font size used: settings will depend on sentence length and user preferences. The gentle reader might notice that vertical scroll bars are always supplied, even if the text fits perfectly in the cell; the reason for this is that users find it less straining if the structure of the screen is not changed with every " page down" command.

Sentence cells can be deleted, merged or split (splitting is not restricted to undoing merge operations). In order to keep operations simple, it is assumed that in the translations produced in the SdT, the sequence of source and target sentences is parallel. In order to deal with the rare cases of cross-over translations, the " cut" , " copy" and " paste" functions have to be used.

The main difference with most commercial products is that sentences are not " locked", ie that all normal text editor functions are available (it appears that at least some functions are foreseen in Trados' Winalign software package, but not yet available in the current version): if users encounter errors in the source or target text, they can correct them immediately. In addition, the standard " find" and " search and replace" functions are available for the whole text.

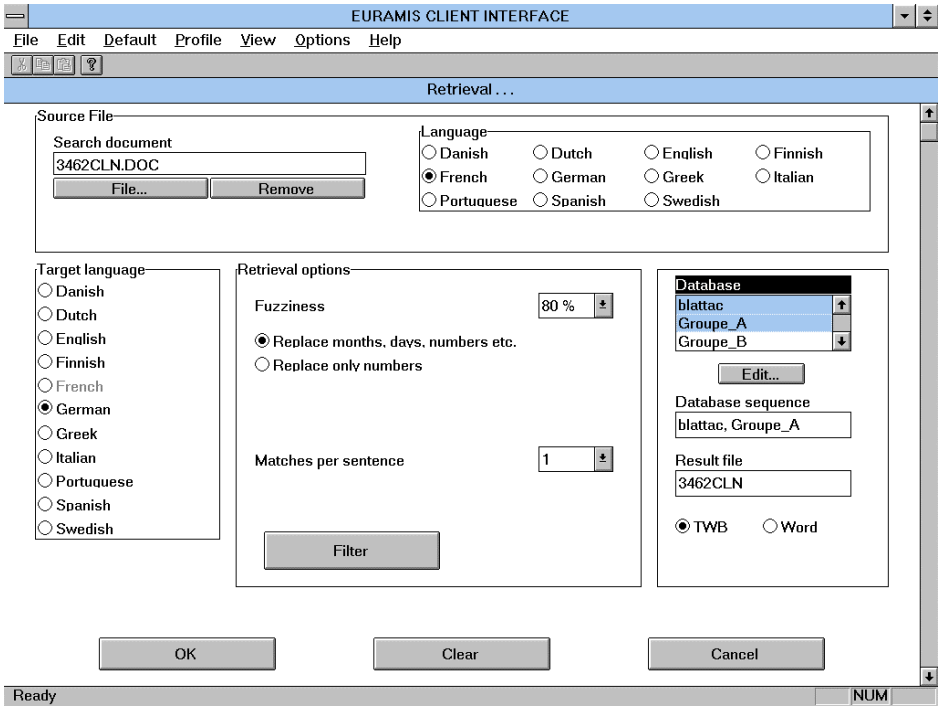
The EURAMIS alignment editor internally works with EURAMIS pivot format[6], but users can both load and save files in TWB import format; TWB attributes which are not defined in EURAMIS can be converted interactively. It is also possible to change or add attribute values to a given document:



[6] For a description of the pivot format, see " EURAMIS: Added Value by Integration" p 59.

3 TM retrieval request

The EURAMIS TM technology is embedded in a client-server environment, which uses the EURAMIS client interface as a communication means for batch retrievals. The main screen of TM retrieval looks as follows:



In order to have a valid request, a number of details have to be indicated (it should be mentioned that all settings except the name of the source document and the result file can be saved for subsequent sessions):

- the name of the document for which the retrieval is requested;

- the source and the target language;

- at least one TM in which a search should be carried out;

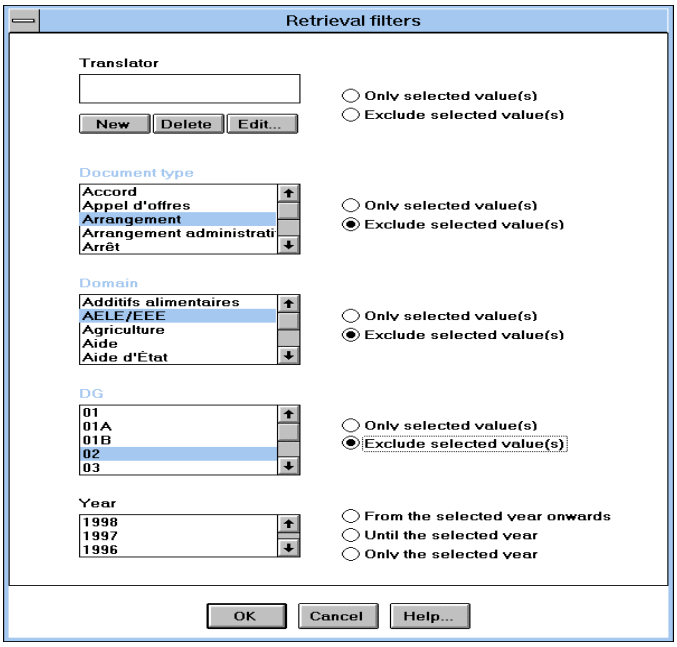
- the output format.

It is possible to change the threshold from which the user is prepared to accept fuzzy matches, to disable the automatic replacement of months, weekdays etc (see section 4 on automatic replacements), and to select the maximum number of matches per sentence (automatically restricted to 1 for output in Word format).

By default, the list of databases available contains the user's personal TM and all group TMs (see section 5 on TM levels). The " edit" button underneath the database list allows the user to add items, eg the personal TMs of colleagues.

One important feature of EURAMIS retrieval is that the user can define a number of databases in one retrieval; it is for instance possible to use one's own personal TM, one's group TM, and a specific TM such as the legislative TM: all TMs indicated are inspected, the matches from each TM are compared and the best match(es) - as many as requested by the user - are returned.

In order for the user to distinguish between the results of different requests, a result file name can be indicated. In addition, the user can indicate filters to fine-tune or restrict results:



Apart from the year field, where a different logic imposes itself, all filters can be used positively or negatively, ie one can either stipulate that for a translation to be acceptable, it must carry a certain attribute (" document type" etc) with a selected value or values, or that it may not have such a value. It is even possible to exclude systematically the results stemming from a certain colleague by saving the corresponding settings as a default.

4 Automatic replacements

Automatic replacements are carried out mainly in order to spare the user trivial changes which would otherwise be necessary in order to adapt a fuzzy match so that it can be reused in a different context. Numerals are the most obvious example of such a treatment; a user probably does not care too

much about the difference between the following two sentences, and he probably does not even want to be bothered with the changes necessary in order to produce a correct translation:

(1) Directive 77/93/EEC is hereby amended as indicated in the Annex to this Directive.

(2) Directive 83/107/EEC is hereby amended as indicated in the Annex to this Directive.

TWB offers the possibility to replace Arabic numerals automatically. If, for example, the string " reference: 4-0111/96" is already in the translation memory, " reference: 4-0999/96" will be found as a 100 per cent match and " 111" would be replaced by " 999" . This process can be regarded as a reliable background operation where the user is not irritated by any specific hint (the original sentence can still be inspected in the " TM window").

In addition to this treatment of Arabic numerals, EURAMIS makes it possible to define sets of entries for which automatic replacements are also carried out, eg names of months, weekdays, Roman numerals etc. There are two different types of sets:

- Open sets which can in principle be defined by means of regular expressions, eg cardinals (regular expression: [0-9]+) and Roman numbers;

- closed sets which are defined by means of database entries in the different languages, eg months (January, janvier, Januar etc).

Automatic replacement is carried out only if a number of conditions are met (in the examples below, open sets are in italics, closed sets are put in bold):

Condition or action	Example
the fuzzy part of the search sentence must contain elements of a replacement class	Le règlement (CE) n° *3295/94* du Conseil du *22* **décembre** *1994* fixant...
the fuzzy part of the database sentence must contain the same number of elements of the same replacement classes in the same places	Le règlement (CE) n° *1626/94* du Conseil du *27* **juin** *1994* fixant...
the database translation must contain corresponding parts, ie elements which are identical (for open sets) or equivalent (for closed sets)	Council Regulation (EC) No *1626/94* of *27* **June** *1994* laying down...
only under these conditions, a replacement is carried out on the database translation	Council Regulation (EC) No *3295/94* of *22* **December** *1994* laying down...

The degree of fuzziness is calculated only after replacements have been carried out, ie as if the search sentence did not contain the differences in question; this means for the example above that there is a 100 per cent match as opposed to 87 per cent, if only numbers were replaced. It should be pointed out that this replacement mechanism works even if fuzzy parts remain after the process; in that case, a match would be raised from, say, 66 per cent to 79 per cent.

In order to make all automatic replacements available in TWB[7], without incorrect restrictions on

[7] For output in Word format, a separate colour is used for replaced items.

fuzziness, the artificial " retrieval" (3) - (4) is created[8]:

(3) Le règlement (CE) n° *3295/94* du Conseil du *22* **décembre** *1994* fixant...

(4) Council Regulation (EC) No *3295/94* of *22* **December** *1994* laying down...

The difference between this and a real occurrence is made visible:

- Automatic replacements are highlighted by means of colours in the translation unit to be imported to TWB (via RTF tags in the import file);
- the attribute " ChangeUser" receives the value " replacement" ; a user who accepts the translation created becomes ChangeUser himself so that this feature can be used as a filter to the central EURAMIS TMs: translation units which still carry the value " replacement" are not saved, as they do not constitute real translations.

5 TM levels

According to the general EURAMIS philosophy (see also LEICK's paper, p 52), there are different TM levels (individual, group, all) which allow validation at the transition from a lower to a higher level (" harvesting"). As far as translation memories are concerned, validation concerns above all correctness of alignment, coherent and correct information on domain, text type etc, whereas the translation quality of an individual sentence will be checked

[8] It would not be necessary to treat Arabic numerals during EURAMIS retrieval, but this generalisation makes it possible to have only one treatment, independently of whether TWB or word processing output is created.

only in rare cases. The following describes the different TM levels in more detail.

Users who wish so can have their own personal TM. This makes sense mainly in the context of integration of different products, eg TM and machine translation. In order to avoid confusion, a unique name for this personal TM is used (the user's login which is available as a local environment variable). Although the individual user is the only person who can write into the TM in question, there is a common consensus that everybody can consult everybody else's TM. This approach is based on the principle that the institution rather the individual (who is after all paid for his work) " owns" the data produced.

Above this level, there are currently unconsolidated group TMs, where practically every member of a group (independently of his proficiency concerning TM technology) has the right to add data. Although this approach implies a high risk (" garbage in - garbage out"), it had to be accepted temporarily for practical reasons. The development of TM management tools (see section 6) has not yet been finished so that harvesting is not yet fully operational. If under such conditions, a strict separation between individual and validated group TMs were maintained, users would have to list a large number of individual TMs for their queries; this problem is aggravated by the fact that users have opted for rather large groups. In the future, these unconsolidated group memories will be replaced by individual memories which are linked logically so that a user can consult a virtual group memory which consists of all the members of a group (not only his own).

Once the unconsolidated group TMs are treated only as logical concepts, consolidated group memories will be established which contain the validated data of a group. Each group TM has a TM manager who harvests

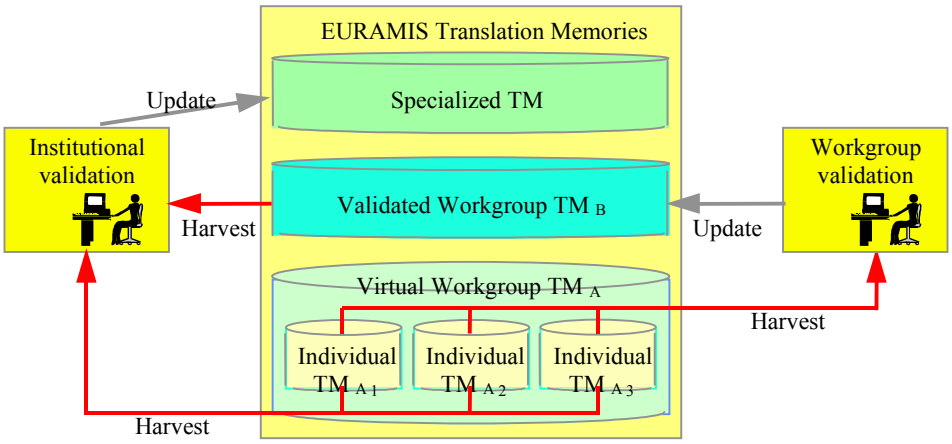
data from individual TMs, coordinates the content of the TM, sees that attributes are used coherently in that group, possibly corrects attribute values for certain texts, grants write access etc.

Above group TMs, there are two types of specialised TMs:

- TMs of general interest: these are used as reference databases which accommodate sample translations, eg, there is one TM with the model sentences for call for tenders, another one which contains the most important legislative texts and court rulings etc;

- TMs with material which is produced across groups: since the translation of the monthly *Bulletin* and of the yearly *General Report* implies large volumes and short deadlines, work is distributed over several work groups who for those purposes use specific TMs.

The following figure shows the interaction between the different levels:

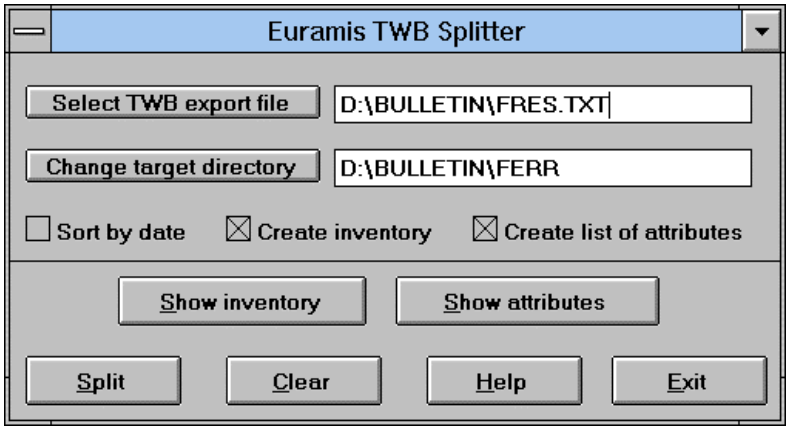


6 Management tools

It is evident that active management of the different TM levels is possible only if user-friendly and efficient management tools are provided. A nucleus of such tools is currently available only for TMs at PC level. An application which provides the same (and more) services for central TMs is currently being implemented.

6.1 *Local TM management*

When users started to upload their TWB translation memories to central EURAMIS TMs, it was felt that the functions to filter out unwanted entries were not powerful enough: filters have to be defined during export, but frequently, users do not have an overview over all the attributes and values used. In addition, it was felt that it would be very useful to separate a TM into the documents (translations or alignments) on which it is based. In order to fill these gaps, a small EURAMIS client application has been developed:



This application takes as input a TWB export file and splits it into as many files as there are different sets of attribute-value pairs[9]: since users regularly put the document reference (both during translation and for alignments), this is generally the most distinguishing feature. The order of the original TWB export file is kept (which in turn tends to keep the original document order), making it possible to create pseudo-alignments. The resulting files can be corrected with the EURAMIS alignment editor. As the alignment editor makes it possible to map unkown attributes to EURAMIS attributes, and to change (or insert) values for a given document, differences in usage can be ironed out. If it is felt that a certain file should not be part of a TM, it just can be deleted before uploading (or reimport into a new local TM). In order to facilitate standardisation, a list of all the attributes and their values pertaining to the TWB TM in question can be requested:

```
<Att L=Doc. Ref.> 95/1382

                  95/2868

                  96/0905

<Att L=Doc. Type> <none>

                  Suites données

<Att L=Source> Euramis TM

<CrU> mustema
```

[9] Since date can be regarded as insignificant in most cases, it can be left out.

In the example above, the attribute " Doc. Type" has been left empty at least once in the different attribute value sets. A second list gives the details for each of the files created. This list can be used for fast identification of the file(s) to be corrected:

SUI-FRE1.wal:

<CrU>mustema

<Att L=Source>Euramis TM

<Att L=Doc. Ref.>95/1382

<Att L=Doc. Type>Suites données

SUI-FRE2.wal:

<CrU>mustema

<Att L=Source>Euramis TM

<Att L=Doc. Ref.>95/2868

<Att L=Doc. Type>Suites données

SUI-FRE3.wal:

<CrU>mustema

<Att L=Source>Euramis TM

<Att L=Doc. Ref.>96/0905

<Att L=Doc. Type><none>

In this example, it would suffice to search for " <none>" , possibly look at the file(s) concerned in more detail, and add the missing information.

6.2 *Central TM management*

Central EURAMIS TM management is currently being implemented. Since data will be migrated to a relational database management system (RDBMS), management tools will offer much more flexibility than with the database which is currently used.

TM management will be very much centred around the same notion of document as shown in the previous section. TM management will therefore be based on the same kind of inventories and attribute lists as described in the previous section.

There will be two main differences to the treatment which is already implemented locally:

- Inventories will be integrated in a user-friendly interface which enables TM managers to manipulate data efficiently;

- since EURAMIS TMs typically consist of multilingual entries (see section 7), two views will be necessary: an upper level view, where the existence of a language version of a document is just listed and language-specific information (eg translator) is not taken into account, ie there is one " multilingual document" across languages; and a lower level view, where the opposite is the case: there is a breakdown of the different attributes used for each language version.

Management of central TMs will basically consist of two aspects:

Firstly, there will be manipulations where the document constitutes the largest entity to be dealt with. These activities will be based on TM extractions in alignment format; these extractions

will be launched from the TM management tool. The alignment editor will be used for correcting wrong alignments, deleting wrong sentences etc.

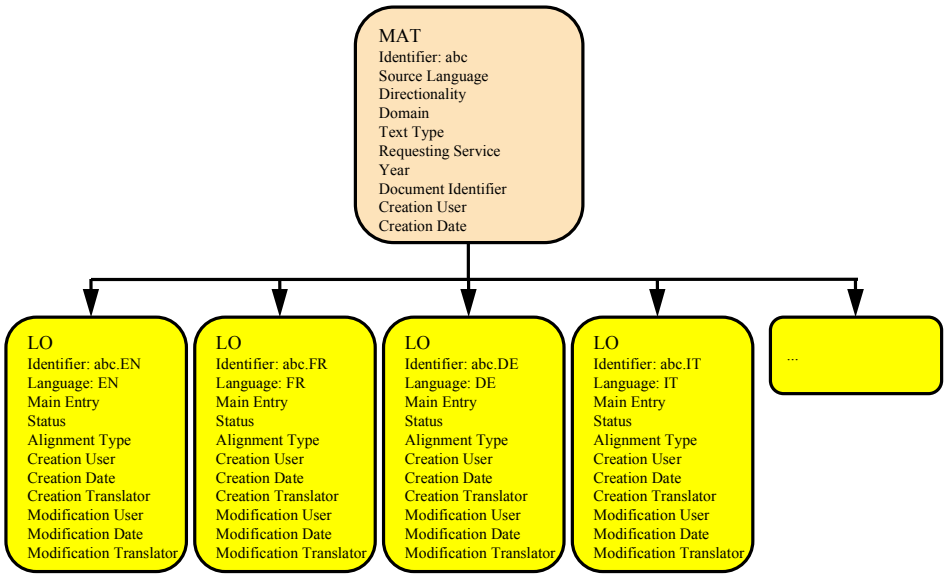
Secondly, there are activities where the document constitutes the smallest entity to be dealt with. This type of data manipulation embraces harmonisation of attributes, making parts of the translation memory available to larger groups (" harvesting") and similar activities. The management tool will provide facilities to request inventories, use inventories for value management and harvesting, and carry out the resulting modifications of the database(s) involved.

7 Structure of TM entries

All entries of the EURAMIS Linguistic Resources Database (LRD) have the same general design: there is a language level where the information pertaining to the different languages (eg translator, alignment type) is stored in parallel sections; above these " linguistic objects" (LOs), there is a second level with information common to all languages (eg domain, text type, requesting service, document name[10]); upper level entries are called " multilingually aligned texts" or MATs).

As a general rule, users cannot define attributes freely, but have to stick to commonly accepted features. As a result of this, pre-defined attributes can be used as powerful filters to refine retrieval efficiently (see section 3 on TM retrieval requests):

[10] Document names are generally the same across languages; information on language is kept separately.



The MAT and the LOs entries are linked by a system of unique identifiers where the identifier of the MAT entry is suffixed with a language identifier for each linguistic object[11].

8 Language direction

Commercial products currently tend to support only one search language per TM whereas EURAMIS offers the possibility to search with changing source languages; this means that it is not necessary to create

---

[11] This scheme offers a number of performance advantages in that access via indexes can be reduced considerably:

- A whole multilingual entry set can be retrieved from the database by suffixing the MAT identifier with an asterisk (for the figure above: " abc*");
- in the same way, all LOs pertaining to a MAT can be retrieved by suffixing the MAT identifier with question marks (" abc.??");
- translations of an LO can be retrieved suffixing the MAT identifier with the corresponding language suffixes.

different TMs just because different source languages are involved.

As to indexing of main entries[12] (which is a condition for subsequent retrieval), the following practical solution has been chosen:

- The source language is always indexed;
- English, French and German are indexed even if they are target languages.

The reason for this is that SdT has to produce translations mainly from the working languages (English, French and, to a lesser extent, German) to all the eleven official languages. It is necessary to handle translation direction in a more flexible way than may be sufficient normally: it happens frequently that the target language of a document becomes source language in a subsequent version, eg because at a certain stage, a dossier is passed on to a service where a different working language prevails.

A compromise had to be found between this need for flexibility and the reliability of entries which might suffer if the translation direction is reversed (or an indirect relationship is deduced between two translations) without the user being aware of it: a difference was introduced between adirectional entries where language direction does not matter and directional ones where it does. The current implementation defines this difference as a property of individual entries (at MAT level); this proved to be too cumbersome for the user, because for each storage to a central TM (be it an alignment or a TWB export file), users would have had to indicate the correct information. On the other hand, it turned out

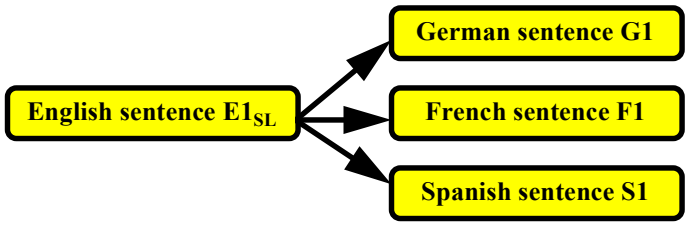
[12] For more details on the indexing mechanisms used, see subsequent section.

that directionality (or its absence) can be seen as a property of an entire TM: this approach is currently being implemented.

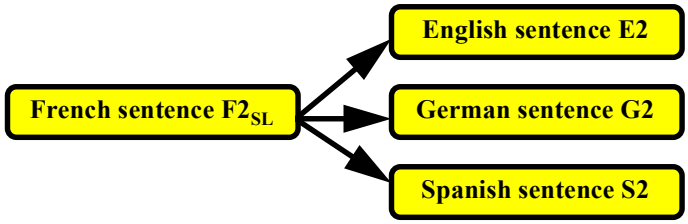
Typically, adirectional TMs are TMs of general interest (as described in section 4); they are usually created on the basis of alignments where the source language is of little importance up to the point that it is even unknown. LOs are organised completely in parallel (information on the original source language is kept at MAT level as additional information to the user)[13]:

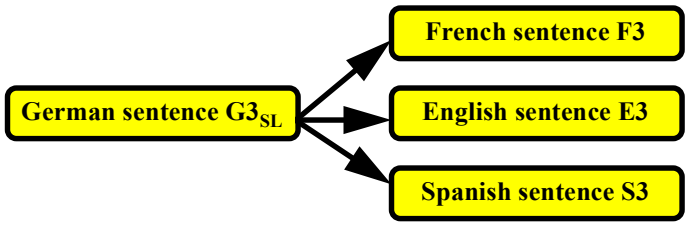
Directional TMs emerge from day-to-day operation, ie these are TMs which are maintained by individual users or workgroups. TWB assumes directionality with the restriction that there may be only one source language per TM:

[13] The figures shown here only represent a logical view: the only differences in the database are the values for directionality and source language.



As already mentioned, EURAMIS makes it possible to put entries with different source languages into the same TM:





Since most of the translation memories are directional, this will be the default. A simple list will contain the names of the adirectional TMs.

Since the value for source language is kept for information purposes, a change in the definition will always be possible. It will also be possible to extract subsets from one TM to another, and to thereby change the type.

For directional TMs, the following retrieval options will be added (note that any language direction will be taken from adirectional TMs):

- " this language pair in this direction" : this is the most conservative approach and it coincides with what one gets from TWB; it will therefore be the default; applied to the examples above, a retrieval from English to French would yield E1-F1 only;

- " this language pair in either direction" : this option is particularly interesting when the source language changes from one version to the next; a retrieval from English to French would yield E1-F1, but also E2-F2;

- " these languages in any entry" : this option is interesting if for the language pair needed, there are only few entries available; a retrieval from English to French would yield E1-F1 and E2-F2, but also E3-F3.

For output in TWB, the attribute " ChangeUser" will have the values " reverse translation" respectively " indirect translation" . The name of the translator of the original language direction will disappear in such cases.

9 Indexing and retrieval mechanisms

Generally speaking, the efficiency of access to records in a database is increased by using indexes. Such indexes are built according to well-defined ordering criteria (eg numerical value): for each ordered value, they contain the reference(s) to the relevant record(s) in the database.

Language storage and retrieval mechanisms such as terminological or documentary databases usually order linguistic data by splitting it into words and putting these words into an index (since very frequent words are not significant for the search, they are usually left out). Queries can be carried out on the basis of exact strings[14] (eg " feasibility study" , " feasibility studies" etc), or using wild cards (eg " feasibility stud*").

TMs differ from other linguistic databases in that they have to provide fuzzy matches, ie similar strings which might differ from the search string at any position, the only condition being that they are still similar enough. Theoretically, conventional access mechanisms could be used if wild cards were generated in a large number of permutations, but this would unacceptably slow down the search.

This section describes the fuzzy technology underlying the EURAMIS TMs in more detail. This part can safely be skipped by those readers who are less interested in the technical aspects of TM technology.

[14] Multi-word queries are interpreted as the intersection of all references attached to the index words listed in the query.

9.1 *Fuzzy index*

One basic requirement for EURAMIS TMs is that it must be possible to find similar sentences in very large databases. In order to achieve this, for each sentence which is put in the database, a numerical value is calculated (the " fuzzy key") and stored in a specific index file, together with a unique identifier of the database sentence; accordingly, for each sentence searched in the database, a fuzzy key is calculated and searched in the fuzzy index: the fuzzy keys with the lowest distance to the search key are most likely to provide similar sentences (equal sentences have identical fuzzy keys, ie the distance between search and database sentence is zero).

The index file which incorporates the fuzzy key of each database record can be seen as a kind of map, where searches are carried out within a certain radius from a given point (the radius being the maximal fuzziness allowed).

The fuzzy key approach is based on trigram[15] technology, which has proven to be very powerful in linguistic applications. In order to be able to represent very long sentences in relatively little space, a number of generalisations are made before the inventory of trigrams of a sentence is stored in the fuzzy key: upper case is converted to lower case and non-alphabetic characters are removed. The resulting normalised sentence (5b) is segmented into overlapping trigrams, ie all possible sequences of three characters (5c):

(5a) Le règlement (CE) n° 1626/94 du Conseil du 27 juin 1994 fixant...

(5b) lereglementcenduconseildujuinfixant...

[15] A trigram is a string of three characters.

(5c)
ler,ere,reg,egl,gle,lem,eme,men,ent,ntc,tce,cen,end,ndu,duc,uco,...

Now, for each trigram, a number is calculated, whereby the first character's position in the alphabet is multiplied by 1681 (ie 41*41), the second character's position is multiplied by 41, and the third character's position is taken as it is. The sum of these values is divided by 47; the remainder of the division is used as the numerical value of the trigram, eg " ler" yields the value 44:

(6) (12*1681 + 5*41 + 18) / 47 = 433 remainder 44

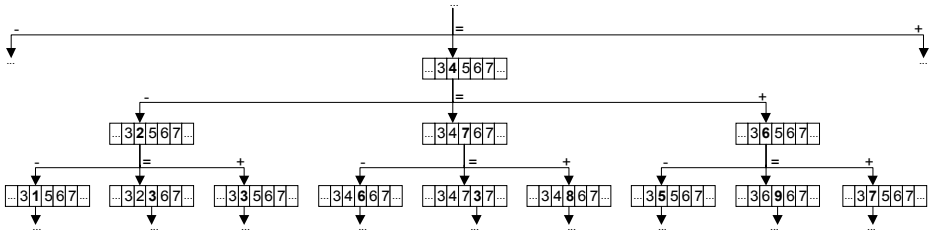
Since there is a division by 47, the resulting remainder ranges from 0 to 46. Hence the fuzzy key consists of an array of 47 bytes where each byte is used to count the number of occurrences of one trigram value: for sentence (5a), byte number 45 would therefore have the value 1 after the first trigram has been dealt with. This approach guarantees that sentences of practically any length can be represented by a numerical key which is 47 bytes long.

What is needed as a divisor is a prime number which is bigger than the multiplier, ie any prime number from 43 onwards would do. The bigger the divisor, the lower the chance of getting similar keys for very different sentences - at the cost of requiring more space in the index file. 47 has been established as the optimal compromise between space requirements and accuracy. The reason why a multiplier of 41 is chosen in the above calculation is that the alphabet used consists of 40 characters.

The need to have a small character set is the reason why normalisation takes place.

A fuzzy key is inserted in the index file only once, ie if different database sentences yield the same fuzzy key (be they equal or different strings), the references to these sentences are stored under the same fuzzy key. If a key with multiple references appears to be relevant in a given search, all the sentences subsumed by it must be inspected further on the basis of a string comparison algorithm (see below).

The index file is organised as a linked list of fuzzy nodes where at each node it is possible for one of the byte positions to branch to smaller, equal, or greater values (this schema is called a ternary tree)[16]:



This structure offers the following advantages:

- Ternary trees tend to grow in width rather than in depth; since the depth of a tree determines the average number of nodes to be inspected before the best node is reached, access times grow only

[16] Branching to an equal value leads to shifting the focus by one position to the right; branching to different values leads to keeping the same focus.

marginally with each order of magnitude by which the database grows; this property makes large databases without performance losses possible;

- a search algorithm which is specially tuned for such trees makes it possible to find the fuzzy keys with the lowest distance very fast.

The distance between the search key and the key for the database sentence can be defined as the sum of the distances (ie absolute differences) of the individual slots:

search key	2	2	1	0	1	2	6	2	...
database key	0	3	1	0	0	0	5	2	...
distance	2	1	0	0	1	2	1	0	...

A number of optimisations have been implemented in order to make this approach more powerful without losing too much of its function:

Firstly, for a sentence to yield a trigram (which is then represented in the fuzzy key), it must contain at least 3 alphabetical characters; this leads to a situation where a lot of " sentences" would have to be checked which are referenced under an empty fuzzy key, ie simple numbers (eg from table cells), but also strings such as

(7) A4-0195/95

(8) <T4>11.2.2.1.

(9) C (3)

A search with a sentence yielding an empty fuzzy key would necessitate inspecting all the database records referenced under the empty key in the fuzzy file. This would take quite some time, although such " sentences" are generally not very useful; it has therefore been decided to exclude them from both storage and retrieval.

Secondly, the fuzzy key is prefixed with a byte containing the number of words of the sentence in question; the distance for this byte is given a higher weight (factor of five). Although this does not change the search results, the search in the tree is guided much faster to the area with the best fuzzy key.

The fuzzy algorithm as described above is an efficient way to find similar sentences in a large database, since all similar sentences have fuzzy keys with a small distance. However, not all fuzzy keys with a small distance are similar. Moreover, distance between fuzzy keys is no absolute measure for similarity, ie a more distant fuzzy key could still refer to a more similar sentence. There are mainly two reasons for that:

- Normalisation which is needed to keep the alphabet used as small as possible, wipes out all differences concerning capitalisation, numbers, punctuation marks etc;

- since all trigrams are represented by a number between 0 and 46, it is unavoidable that different trigrams can be counted in the same slot so that no difference for that trigram can be detected in the fuzzy key.

9.2 *String comparison*

Since the fuzzy algorithm yields nothing more than a stack of references to potentially similar sentences, these sentences have to be inspected by a string comparison algorithm[17]. Since quite a number of string comparisons may have to be carried out, it is essential to use a very efficient algorithm. The approach taken in EURAMIS is based on the Levenshtein distance (LEVENSHTEIN 1965), which is defined as the minimal number of deletions, insertions and substitutions needed to transform one word into another one. A number of algorithms have been developed to calculate this distance (for an overview see MICHAEL 1994).

The best method appears to dynamically calculate a matrix which at any position contains the minimum number of manipulations needed in order to transform the partial word up to the column in question to the partial word up to the row in question, eg in order to transform "loves" to "ha", 5 manipulations are needed (2 substitutions and 3 deletions) - cf the value of the last column, second row in the following example. The Levenshtein distance for the entire strings can therefore be taken from the downmost cell in the rightmost column, in this example, the distance between "loves" and "hates" is three (3 substitutions are needed to transform one word to the other):

	l	o	v	e	s
h	1	2	3	4	5

[17] Filters criteria (see section 2) are checked before the string itself is validated.

a	2	2	3	4	5
t	3	3	3	4	5
e	4	4	4	3	4
s	5	5	5	4	3

Although the algorithm had been developed for words, it can be applied to sentences as well. Since similar sentences tend to have large parts in common, the algorithm can be optimised by building the matrix only from the first different character to the last. In order to identify this maximal difference, the sentences have to be compared from their beginnings to the right and from their ends to the left:

	l	o	v
h	1	2	3
a	2	2	3
t	3	3	3

Even for relatively short real life sentences, the optimisation is quite significant:

(1) Directive 77/93/EEC is hereby amended as indicated in the Annex to this Directive.

(2) Directive 83/107/EEC is hereby amended as indicated in the Annex to this Directive.

The original matrix would be 82 by 83 (ie 6 806 cells), whereas the reduced matrix is 5 by 6 (ie 30 cells) which is less than half a per cent. It should mentioned that different " penalties" can be given for the different manipulations. The following penalties are currently used in EURAMIS:

- Substitutions between upper and lower case of the same character: 1;
- other substitutions within the same basic character (é<->e, ä<->a etc): 2;
- all other manipulations: 4.

One could argue that the deletions needed to transform the database sentence to the search sentence should get a lower weight than insertions (the effort needed to adapt the target sentence will in most cases also be lower).

9.3 *Finding 2-to-1 correspondences*

Many TMs cannot deal in an automatic way with cases where two sentences in the source language are translated with one target sentence: during retrieval, only matches pertaining to simple sentences are offered in the first place. In order to get a retrieval for more than one sentence, users have to expand the search to two sentences manually - if they suspect that there might be something useful. The reason for this is that 2-to-1 correspondences have no specific property in the database by which they could be identified easily. Expanding searches automatically would therefore make it necessary to always look up sentence pairs as well - a measure which would deteriorate performance in an

unacceptable way, given the fact that 2-to-1 correspondences are not that frequent.

In EURAMIS, 2-to-1 correspondences are dealt with by a simple efficient mechanism. The database record contains the two sentences AB in question together with a separator. All the rest is done in the fuzzy index. Fuzzy keys are calculated for A and for AB. The node for A contains both a reference to the database record and the fuzzy key for AB.

If for a retrieval of a sentence A', the fuzzy key A is encountered, A' is expanded by one sentence to the right, and the corresponding fuzzy key is compared with the key for AB. If the fuzzy distance is below the threshold used, the sentence pair is inspected further by means of string comparison.

10 Future work

More integration will be provided with full-text databases such as CELEX[18] (see also paper " EURAMIS: Added Value by Integration" , p 59): the general idea is to extract reference documents for a given search document from a full text database, align them and create *ad hoc* TMs. The search document is then used for a query in such an *ad hoc* TM; alternatively, users can ask for the entire TM created. Since there is no further need for such a TM on the server side any more, it can be deleted.

Such a strategy presupposes a near-to-perfect quality of alignment. As far as CELEX documents are concerned, this is already around the corner: firstly, CELEX contains legal documents where care is taken that paragraphs and even sentences correspond

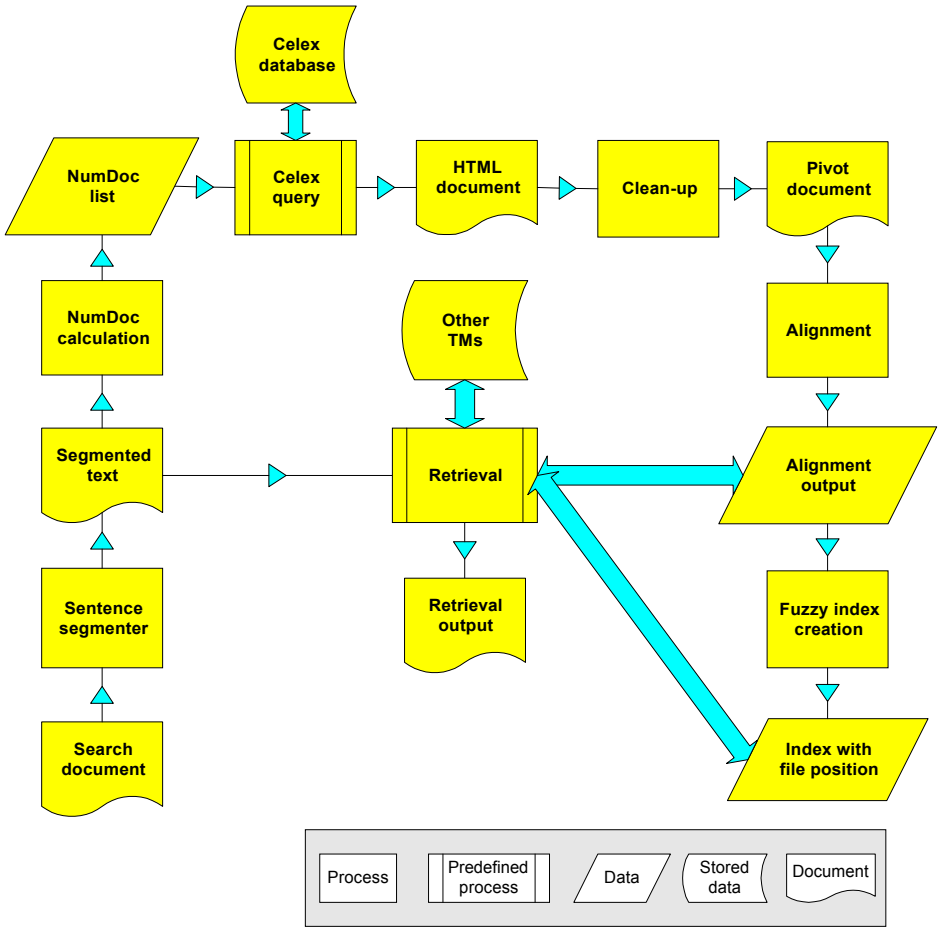
[18] CELEX is the full text database of the Office for Official Publications of the European Communities and covers a wide range of legal documents.

to each other in the different languages (so that they can be referred to without checking the different language versions); secondly, since these documents are aligned frequently, a lot of experience has already been gained as to where languages (or documents) usually differ and how these differences can be ironed out.

It is evident that a relational database is not a good place where *ad hoc* TMs should be stored: firstly, importing to a relational database (and deleting the data afterwards) takes too much time; secondly, since the data comes from foreign sources, the secondary information which can under normal circumstances be used for filtering (document type, domain etc), will not be available anyway.

In such a scenario, it is more efficient to create a fuzzy index directly on the alignment output: the index file then contains the fuzzy keys for the sentences in the source language, with references to the positions of these sentences in the alignment file(s). The following flow chart shows how CELEX could be used as a virtual translation memory[19]:

[19] NumDoc is the name of the unique document identifier used in CELEX.



ACHIM BLATT

Translation Service

European Commission

Luxembourg

Bibliography

BLANK I (1995) " Sentence Alignment: Methods and Implementations" 81/99 *Traitement automatique des langues* 36, 1/2

BLANK I (1997) *Computerlinguistische Analyse mehrsprachiger Ausdrücke* Diss München

GALE W A / CHURCH K W (1991) " A Program for Aligning Sentences in Bilingual Corpora" 177/184 *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics* Berkeley, California

KAY M / RÖSCHEISEN M (1988) *Text-Translation Alignment. Technical Report* Xerox Palo Alto Research Centre, Palo Alto

KAY M / RÖSCHEISEN M (1993) " Text-Translation Alignment" 121/142 *Computational Linguistics* 19-3

LEVENSHTEIN V I (1965) " Binary Codes Capable of Correcting Deletions, Insertions and Reversals" 707/709 *Soviet Physics Doklady* 10

MICHAEL J (1994) " Joker im Spiel. Erweiterung der Levenshtein-Funktion auf Wildcards" 230/239 *c't* 3