

# Notes on a Live Demonstration

Terence Lewis  
Translator & Developer  
terence.lewis@language-engineer.co.uk

## Introduction

These notes have been written to accompany a “live” demonstration of the Dutch-English translation software, known as “Trasy”, independently developed by the author with the support and encouragement of Siemens Nederland. The demonstration will focus on the aspects of the software that integrate Machine Translation and Translation Memory by using a common XML data source that can be accessed by both MT and TM tools. The thrust of the author’s recent work has been to implement a more organic integration of the two types of tools in the workflow he has designed to translate documentation for Siemens Nederland.

## The MT Software

The translation environment presented in the demonstration comprises a Dutch-English “Translation Box”, a Dictionary Manager and a Language Resources Manager. The Translation Box serves as a point of entry to the various features of the actual machine translation software. The function of the Dictionary Manager is self-evident. The Language Resources Manager is a module the author has refined over the past 18 months. It is totally language independent and can, in theory, be used to manage and mine resources for any language pair. This module includes a “Pretranslate” function that can be applied to translate between any language pair for which a translation memory has been provided; it’s like having a “mini translation memory” application within the MT application. The Language Resources Manager also offers tools for building Multiword Expression resources, a Concordance feature and a tool for “harvesting” bilingual content for incorporation in the Multiword Expression Repository.

## Background

The translation software has been built entirely from freely available Java packages and components. Its modular structure has greatly facilitated the continuous development and refinement of the components of the translation package, which combines simple corpus-based translation techniques (similar to those deployed in Example Based Machine Translation) with a conventional rule-based approach to machine translation. The presentation will show how the complete separation of user interface and the underlying language engineering allows the translation program to be run in a variety of set-ups, from a “Translation Box”, from the command line, on a server, and from within any other application that will allow a Java Virtual Machine to run inside it. This 100% Java application runs on virtually any platform (Windows XP, Solaris, Linux...) for which there is a Java Virtual Machine.

## The back office set-up

The software has a “Translation Box” interface that will be familiar to users of online translation services, and it’s possible to run an entire project from this GUI. In the document flow designed for Siemens Nederland the MT application is usually deployed via one or several servers. The current back office set-up involves a mail gateway running on a Linux platform, several servers on Solaris 10 and Linux platforms, and clients running on the Windows XP platform.

## Latest developments

One of the basic aims in developing this software has been, not only to keep the language data used by the program separate from the translation engine, but to keep them in a totally “open” format so that they can be maintained even from a simple text editor and – more importantly – so that they can be utilised by other language tools. Use of non-proprietary formats and platform independence have in fact been the two key planks in the design and further development of this MT application. This has also been a major consideration in the author’s recent development of a prototype Malay-English translator, which will hopefully be taken from prototype to something more useful by using crowdsourcing to review bilingual resources acquired from public sources. Easy access to the data will be essential here.

Over the past five years, as a result of using the Dutch-English translation software to handle major projects, we have seen significant growth in the volume of bilingual data which has been stored in two parallel repositories: a Multiword Expression Repository consulted by the MT application and a Trados Translation Memory database containing over a million translation units relating to industrial subjects. At the beginning of this year it was decided to place all the language resources in a single container which could be exploited by both MT and TM programs. After trying out a number of structures to provide such a container, we decided to adopt a simplified form of XLIFF.

## What is XLIFF?

A fair definition of XLIFF is found at <http://developers.sun.com/dev.../xliff.html>.

“XLIFF is an XML-based format that enables translators to concentrate on the text to be translated. Likewise, since it's a standard, manipulating XLIFF files makes localization engineering easier: once you have converters written for your source file formats, you can simply write new tools to deal with XLIFF and not worry about the original file format. It also supports a full localization process by providing tags and attributes for review comments, the translation status of individual strings, and metrics such as word counts of the source sentences.”

The XLIFF format grew out of a collaboration between a number of companies, including Sun Microsystems, but was soon brought under the management of an Oasis-Open.Org XLIFFcommittee. According to this Committee, the XLIFF format aims to:

“Separate localizable text from formatting.

Enable multiple tools to work on source strings and add to the data about the string.

Store information that is helpful in supporting a localization process.”

## XLIFF provides two-way access

The XLIFF format provides an easy-to-manage structure for the repositories of multiword expressions (phrases and short sentences) mined by the MT program prior to the rule-based translation stage.

Recognising that the vast amount of data held in our legacy translation memories was unavailable to the machine translation application yet could be very useful, we decided to store the approved translation memory for every completed new project in XLIFF and gradually convert our legacy memories into this format. This is being done by first exporting these memories to TMX and then transforming them into the XLIFF format.

The nice thing about XLIFF is that it really does provide an excellent, non-proprietary structure for holding bilingual data gleaned from the work of human translators, since it allows the use of elements to hold alternative translations and elements containing human comment. The MT program can now access and exploit any translation memory data stored in the XLIFF format (but unlike the human translator cannot yet choose between alternatives!).

The latest version of our software also incorporates a “Guest Memory” feature that allows a user to select an “outside” translation memory in TMX format that can be searched for matches via the Pretranslate feature. Matches found during the Pretranslate can be transferred into the source document prior to machine translation, or can be kept in a separate document for subsequent comparison with the MT results (there may be cases where the MT result is superior to the TM result). By the same token, most Translation Memory applications that can import a TMX file can perform a search in Trasy's own data repositories for 100% matches and fuzzy matches. So, data are shared in both directions. Whilst this bidirectionality is possibly of little relevance to the individual translator working with both the MT program and a TM program, it facilitates data sharing among a large group of translators.

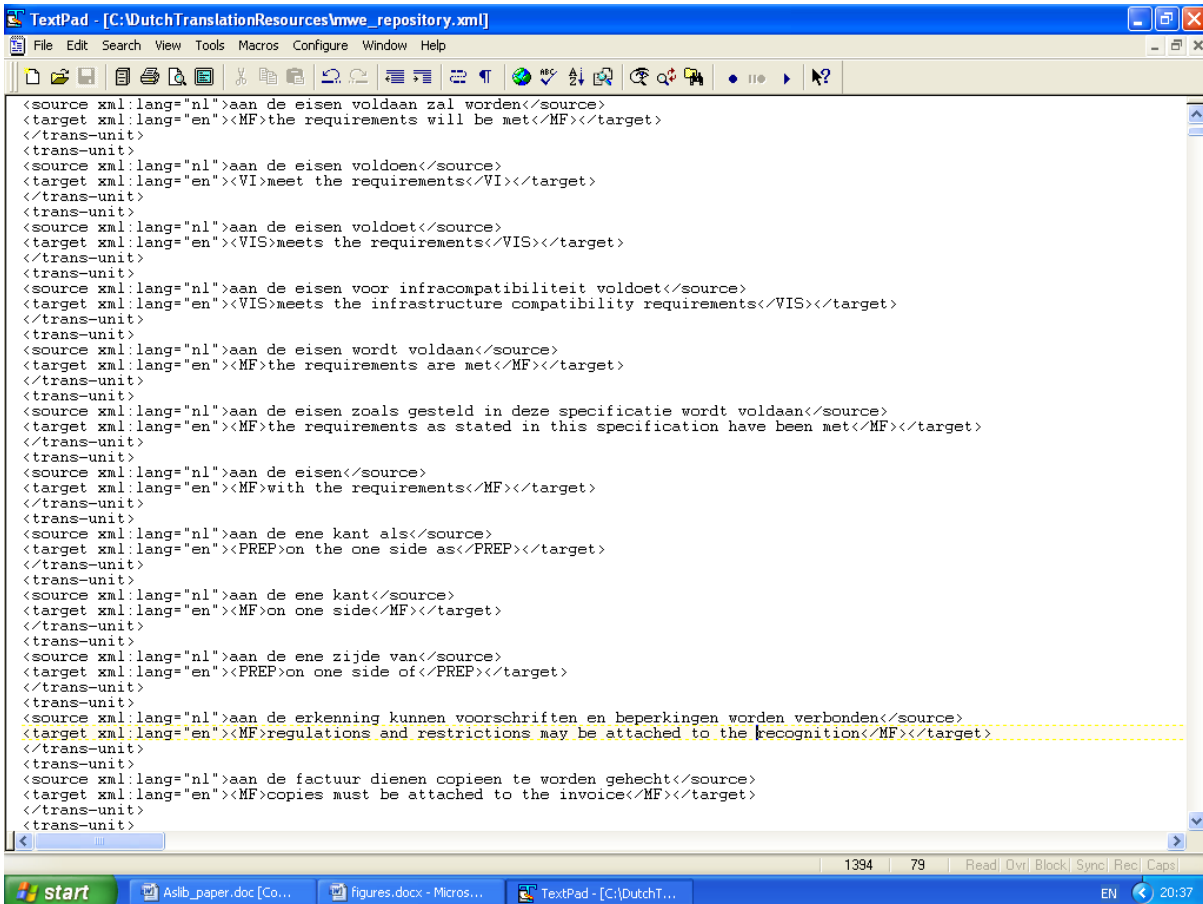
## The importance of ngrams

Like developers and practitioners in the field of Statistical Machine Translation, the author sets great store by the role of bilingual corpora in the automatic translation process. The approach demonstrated differs, however, from purely statistical machine translation by the fact that the data exploited by the translation software are grammatically tagged. The XLIFF file structure now employed in the Trasy system, incorporates a part-of-speech tag allowing the part of speech to be a child of the target element that can even include attributes to hold syntactic and semantic information. Figure 1 shows elements from Trasy's data repository. Using pattern matching and other simple linguistically based data exploitation techniques, the translation engine can match these ngrams to phrases in the source sentence and so produce a more “natural-sounding” translation. This “Near Human Quality” of some of the output has certainly been one of the main factors in customer acceptance of our automated translations at Siemens Nederland.

But, as we've stated above, the process works the other way round too. To give an example: for use with an application such as Memoq® or Translators Workbench®, the MT program's XLIFF repository can be converted into a TMX file, which can in turn be imported into a commercial translation memory program. The translator can then consider the matches mined from the resources of the MT program as translation proposals in his translation memory environment. Memoq® is one of the translation memory environments that actually allows a user to specify several translation memories. In practice, the percentage of matches found in the Translation Memory will determine whether a job is tackled as a Translation Memory or a Machine Translation project. At the end, however, the approved edited data will be available for both applications. The approved translation memory can be used by the MT program; a repository prepared for

an MT run can be imported into a translation memory.

Of course, the beauty of the XLIFF structure is that it can accommodate both the document to be translated and the resources used to translate it. Figure 2 shows a simple example of an XLIFF file translated by the software. We can see that there's no difference between the structure of this document and the structure of the file containing the language resources (Figure 1). Where these resources provide total coverage of the document, so they can be used to generate a usable translation for every sentence in the document, that document is nothing less than a subset of the language resources, and the target elements of the resources become the target elements in the XLIFF document to be translated. However, 100% matches between a large repository of resources and randomly chosen text are rarer than many practitioners would wish.



```
<source xml:lang="nl">aan de eisen voldaan zal worden</source>
<target xml:lang="en"><MF>the requirements will be met</MF></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de eisen voldoen</source>
<target xml:lang="en"><VI>meet the requirements</VI></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de eisen voldoet</source>
<target xml:lang="en"><VIS>meets the requirements</VIS></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de eisen voor infracompatibiliteit voldoet</source>
<target xml:lang="en"><VIS>meets the infrastructure compatibility requirements</VIS></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de eisen wordt voldaan</source>
<target xml:lang="en"><MF>the requirements are met</MF></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de eisen zoals gesteld in deze specificatie wordt voldaan</source>
<target xml:lang="en"><MF>the requirements as stated in this specification have been met</MF></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de eisen</source>
<target xml:lang="en"><MF>with the requirements</MF></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de ene kant als</source>
<target xml:lang="en"><PREP>on the one side as</PREP></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de ene kant</source>
<target xml:lang="en"><MF>on one side</MF></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de ene zijde van</source>
<target xml:lang="en"><PREP>on one side of</PREP></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de erkenning kunnen voorschriften en beperkingen worden verbonden</source>
<target xml:lang="en"><MF>regulations and restrictions may be attached to the recognition</MF></target>
</trans-unit>
<trans-unit>
<source xml:lang="nl">aan de factuur dienen copieën te worden gehecht</source>
<target xml:lang="en"><MF>copies must be attached to the invoice</MF></target>
</trans-unit>
<trans-unit>
```

Figure 1 - Ngrams with translations held in the XLIFF structure in the data repository

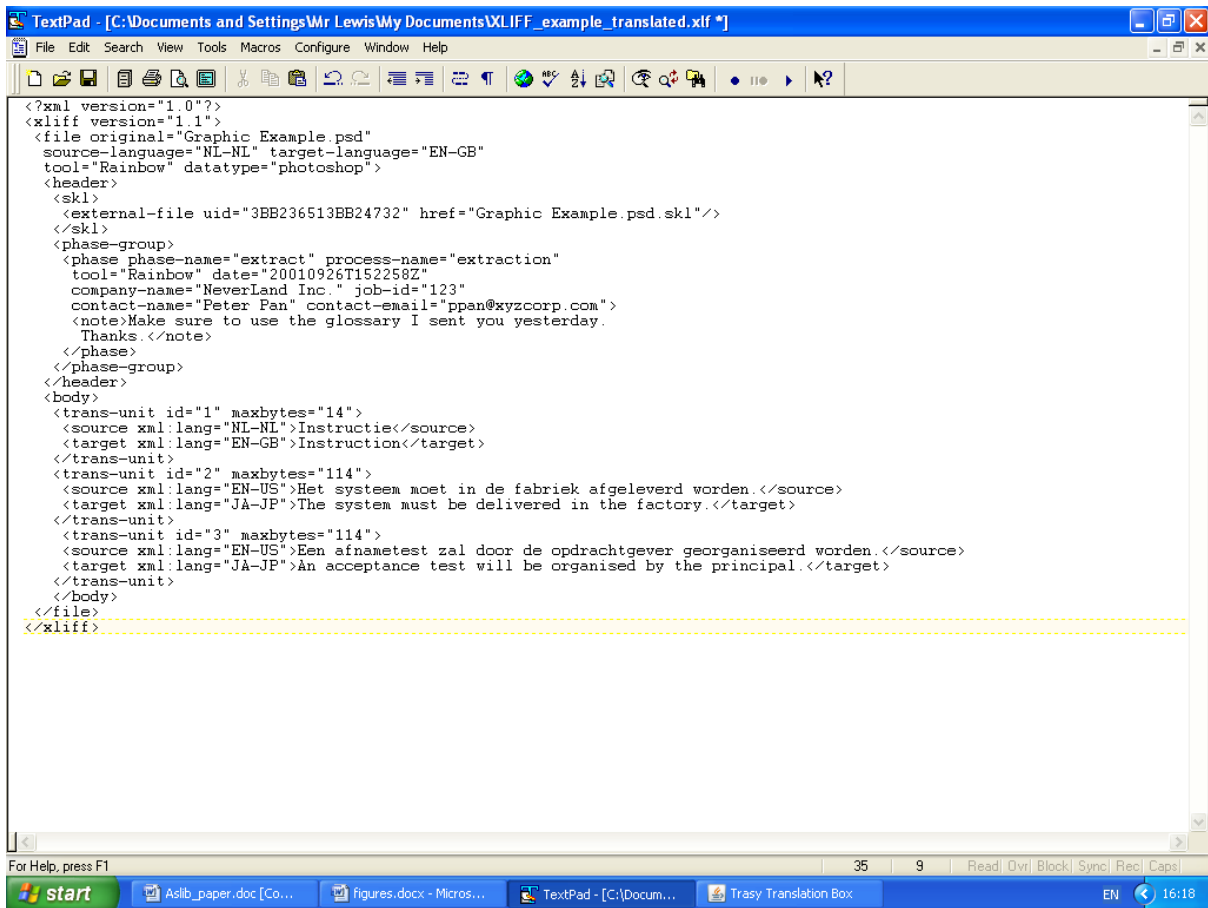


Figure 2 - Example of an XLIFF file translated by the MT Software

Matches are more likely to be found between bigrams, trigrams and four-element or five-element ngrams in a source text and stored phrases of these lengths. The repository used by the software, which is nowadays a very large XLIFF file, therefore contains not only complete sentences (with their translations) but the ngrams into which these sentences are fragmented. These ngrams typically take the form of noun phrases (e.g. “decision taken by the board”) and verbal phrases (e.g. “is considered to reflect best practice in the industry”). The translation program tries to exploit this data by means of a variety of techniques before any parsing in the conventional sense is attempted. In its regularly visited domains, the program will frequently assemble the translations of whole paragraphs from these fragments. Figure 3 shows the output from two of the tools used to incorporate ngrams (phrases and short sentences) with their translations in the Multiword Expression Repository. The Ngram Frequencies list contains the most frequently occurring combinations of words in the document to be translated. The list on the right hand side of the screen shown in Figure 3 contains all the ngrams in the document in alphabetical order. Both these lists offer a means of feeding the MT program with common collocations and phrases that need an idiomatic rather than word-for-word translation.

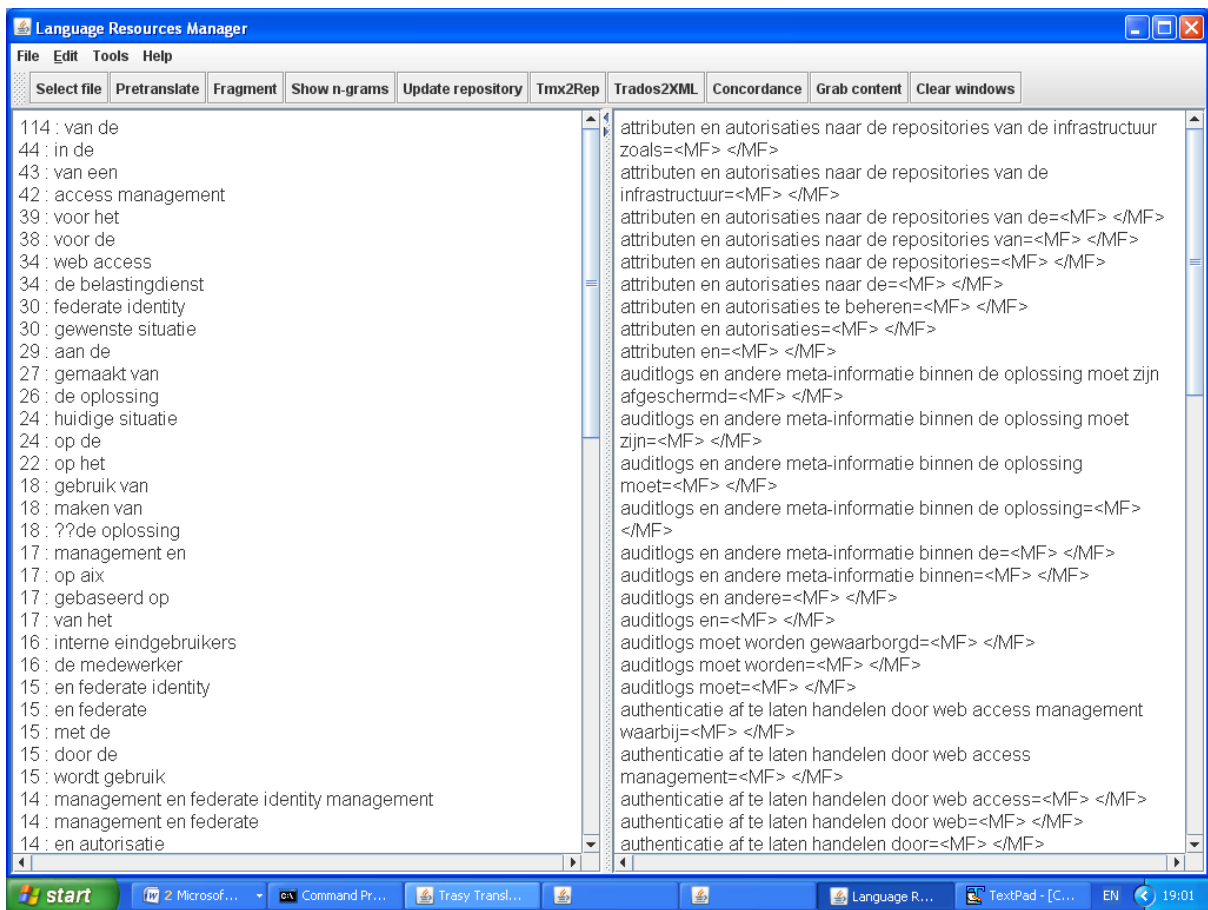


Figure 3 - Ngrams generated by the Fragmentation Tool.

### Practical use of the software

The demonstration for which these notes are written will present the workflow currently used to generate English translations for the Corporate Translations Office of Siemens Nederland. This process combines the Trasy Dutch-English MT Program and the SDL Trados Translator's Workbench®. We will assume that the source document is contained in a MS-Word file. If the translation office receives a PDF, step 1 will be preceded by the steps needed to convert the PDF into a processable Word file. In the set-up we have designed for Siemens Nederland, files are received and returned in MS-Word, but the translation software is, in fact, capable of handling files in a variety of formats such as \*.odt, \*.ods, \*.odp, \*.htm, \*.xml, \*.xlf, \*.tmx, \*.txt and \*.docx files. The routine workflow looks like this:

1. Save the Word file as a text file.
2. Run Analysis in Translator's Workbench on the text file.
3. Export the "Unknown segments" into a TMX file (-> DutchMemory\_Unknown.tmx).
4. Open the Dutch-English Translation program
5. Open the Dictionary environment (single word dictionary)
6. Check and import any word lists provided by customer
7. Check the document for unknown words (i.e. words not in the core dictionary)
8. Provide translations of unknown words and update core dictionary
9. Open the Language Resources Manager
10. Using the Fragmentation Tool, generate table of ngram frequencies and add most frequent ngrams with translations to the Multiword Repository
11. Open file containing all the ngrams in the document, search for phrases of interest and add them with their translations to the Multiword Repository
12. Use the MT engine to translate the TMX file (-> DutchMemory\_Unknown\_translated.tmx)
13. If necessary (i.e. grave errors), post-edit the TMX file (in text editor or in TMX enabled TM application).
14. Import TMX file into Translator's Workbench (or other TM application)
15. Generate final Word document from translation memory application
16. (If required) check the Word document for lay-out and completeness
17. Convert finally approved TMX file to XLIFF format and add to Trasy's Multiword Repository

This 17-step workflow is probably the most efficient way of handling large projects, especially as all the “Unknown segments” in all the files can be dumped into a single TMX file for translation, editing and import into the Translation Memory application.

### Interactive use

The MT application can also be combined with a Translation Memory application in an interactive set-up using Cut & Paste or Drag & Drop techniques. Figure 4 shows the Trasy MT program being used in conjunction with MemoQ® but it could be used in a similar way with TagEditor®. This is likely to be the approach adopted by a professional translator who is treating the MT software simply as the proposer of drafts which he/she will review and improve.

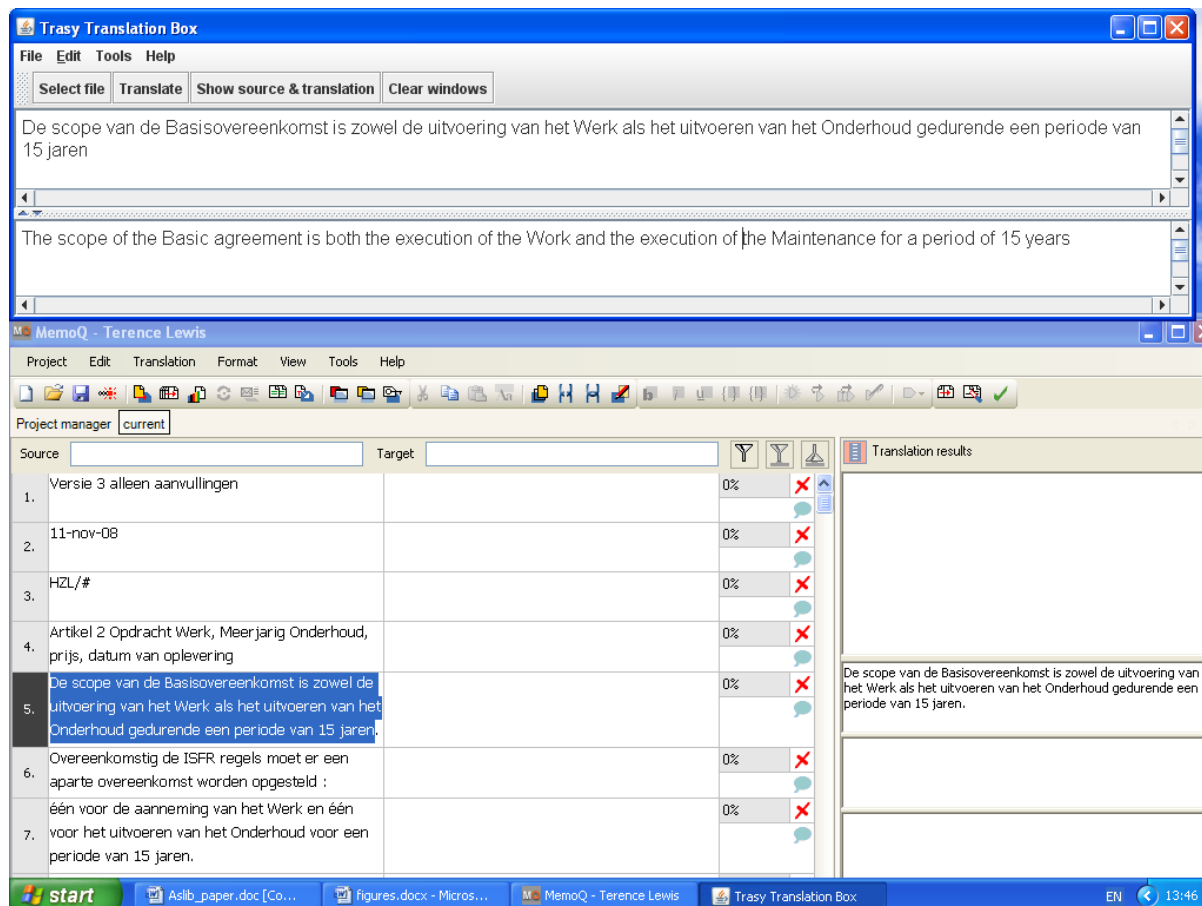


Figure 4 - Trasy MT software used in conjunction with MemoQ®

### Use as a stand-alone translator

The previous paragraphs describe the use of the MT software in conjunction with a Translation Memory application. Figure 5 shows all the file formats the software can translate directly. Of course, the use of the MT software without using the Translation Memory software doesn't mean that the TM resources are excluded from the process. Since the move to TMX & XLIFF, all new translated documents are stored (broken down into translation units) in both the Translation Memory as such and in the MT program's Multiword Expression Repository. So, really, there's no such thing as a stand-alone Trasy translator!

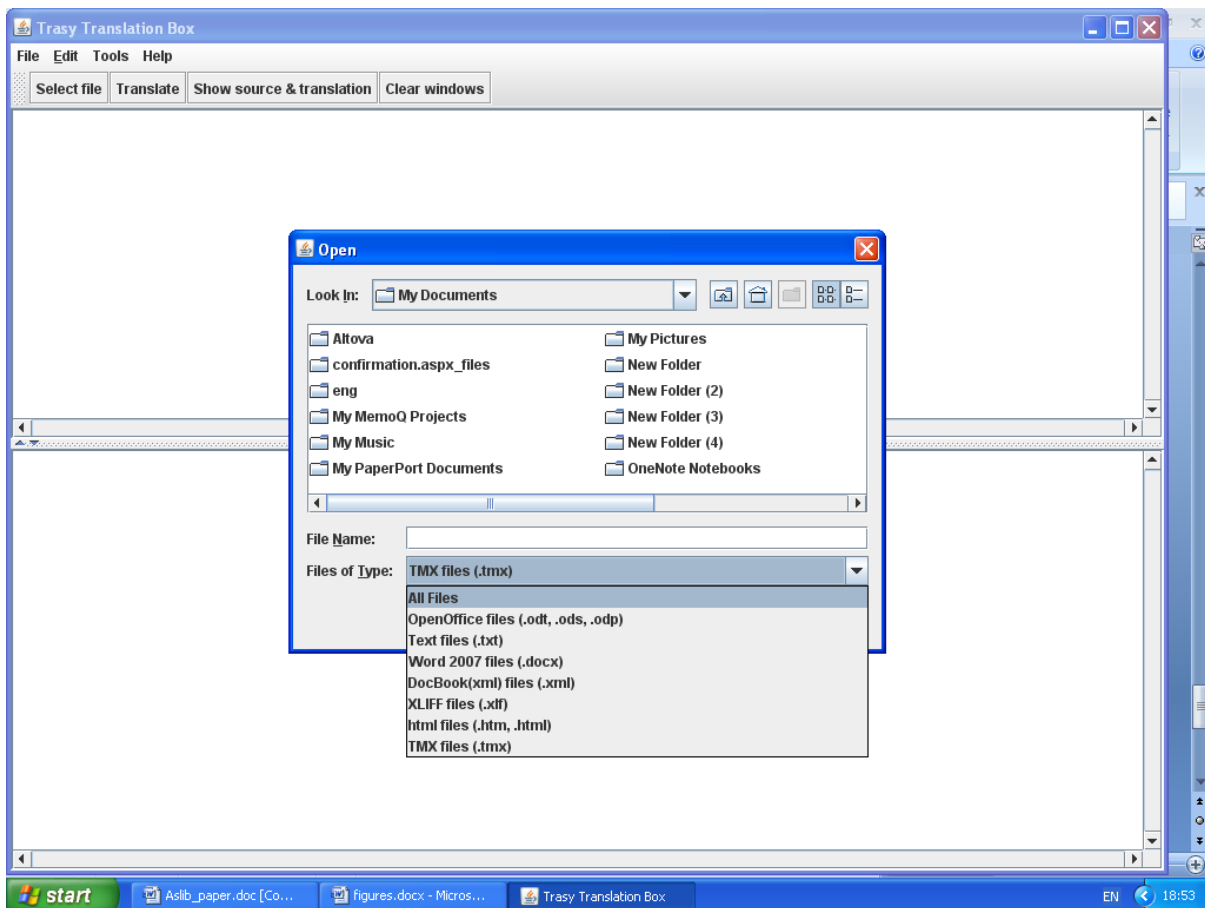


Figure 5 – “Translation Box” with list of files that can be translated directly

## Conclusion

These “notes” have set out to provide some background to a live demonstration of the “Trasy” Machine Translation Software. While the “Translation Box” feature is a handy way of generating a quick translation of a short Dutch text, the emphasis has been on showing how this machine translation software is used within a large-scale document flow to produce automated translations of technical texts. In this context, the value of open, non-proprietary formats for data exchange has been underscored. TMX and XLIFF will be the key formats for storing language data in all future developments of the Trasy Dutch-English Translation Software.