Interoperability between DGT-OmegaT and SDL Trados Studio

Thomas Cordonnier, Elio Fedele and Fons De Vuyst – Informatics Unit
Maria José Bellino Machado and Hilário Leal Fontes - Portuguese Language Department
Directorate-General for Translation - European Commission

1. Introduction

In previous articles published in *a folha*[i], we have presented some adaptations, improvements and new features which have been developed by DGT in order to use OmegaT within DGT's workflow.

In this article we focus on the interoperability between DGT-OmegaT (DGT-OT) and SDL Trados Studio (Studio) from a translator's point of view. For more detailed information, see the DGT-OmegaT website[ii].

As Studio is the mainstream CAT tool in DGT, it is naturally important that translators can use DGT-OT if they so wish/need, but nevertheless are able to exchange projects with colleagues who work with Studio. **So the main objective is to make it possible for a user to translate a project in DGT-OT and have it revised by a colleague in Studio or vice-versa, in both cases without losing any data - even concerning segmentation and formatting - using the SDLXLIFF source file(s) created by Studio.**

Modifying a proprietary tool like Studio in order to enable interaction with other tools seems difficult: the only possibility is to do what its API proposes, and this is voluntarily limited by SDL Trados. Furthermore, as only a minority of translators are using OmegaT, Studio users might refuse such changes. So, definitively, if changes have to be done it cannot be in another location than in DGT-OmegaT.

Fortunately, OmegaT is easy to modify and it already has basic support for the intermediate files named SDLXLIFF used by Studio. The work described in this article was to improve this support and add new features specific to SDLXLIFF files.

In particular, DGT-OmegaT users also do not want to be deprived of some of DGT-OT features available when translating source documents directly from native (Office) format files, i.e. without using SDLXLIFF as an intermediate. In other terms, most users should not even see the difference between a project based on SDLXLIFF and one based on the native format.

More precisely, the objectives are to be able to:

1 - Have the same segmentation in Studio and DGT-OT projects.

2 - In DGT-OT, easily generate the target files not only in bilingual SDLXLIFF format, but also in their native (Office or other) format, using the functions: Create (current) translated document(s).
Results should be identical as if the user worked only with Studio.

3 - In DGT-OT, view the source files in their native (Office or other) format using the feature View source file.

4 - In DGT-OT, see the comments from SDLXLIFF target files translated in Studio.

5 - In Studio, see the track changes in target files revised in DGT-OT.

6 - **In Studio, have the segment status (untranslated/translated) correctly set by DGT-OT in files which were translated or revised in DGT-OT.**

7 - Ideally create the SDLXLIFF file (or the Studio project), directly or via the DGT's Studio Wizard (CAT Client), in any case without having to open and use Trados Studio's graphical user interface (GUI).

This article presents the improvements which have been made or are being tested for that purpose. In DGT, these scripts/applications are, by default, integrated in DGT-OT and in its Wizard, but in this article we will not present the new DGT-OT Wizard as it is still being developed.

For the free open source version, the scripts/applications are available – both with versions 3.1 and 3.2 of DGT-OT – in case free-lance translators must provide the translation(s) in SDLXLIFF format (probably from SDLXLIFFf file(s) provided by the client).

2. *CAT tools and file formats*

Any CAT tool or document editor will use the term "filter" to speak about the piece of code which makes the interaction between itself and foreign file formats. But each tool has its own way of doing it, meaning that it is almost never possible to take filters from one tool and use them in another tool. So, it is not surprising that the approach concerning the source file format is different in OmegaT and in Studio.

While Studio always converts all the source documents into an intermediate format named SDLXLIFF, OmegaT will keep data read from the file in the memory and will save only changes made by the user in its "project memory", which is in TMX format.

This is so for historical reasons. First, because the XLIFF format was not fully specified when the OmegaT project started. And second, because the default OmegaT mode is to make the association between source and target texts so, by default, the same segment in the source, if it appears twice in the project, will have the same translation in both cases, unless the translator explicitly defines an "alternative" translation.

Paradoxically, the fact that Studio uses an intermediate format is a chance for us: for OmegaT, this is nothing more than one new format to be added in the list of filters. **On the contrary, using OmegaT's project memory in Studio would have been more difficult because it only contains segments modified by the user.**

That is why OmegaT already has some support for SDLXLIFF: the filters for XLIFF format also work with SDLXLIFF, but with limitations.

3. *Existing filters for XLIFF format*

OmegaT can use XLIFF (and SDLXLIFF) source files via specific filters. There are 2 filters available:

- The native OmegaT filter: **XLIFF filter**
- The **Okapi XLIFF files filter**

They can be selected in the Project - Properties - File Filters menu. Both have advantages and drawbacks. The problems are:

- **Monolingual/bilingual format:** The native OT filter has been designed for XLIFF files produced by Tikal, the batch tool provided by Okapi Framework. These files always have target identical to source. For that reason, and also because this filter is based on monolingual-only filters architecture, it totally ignores the contents of <source> or <seg-source> in the file, and considers the contents of <target> as the source of the segments!

  Therefore, when the translated document is generated the target file is monolingual, i.e., the translated segments will be displayed in the target language and the non-translated segments in the source language, but without the possibility of reusing it as a source file (for revision purposes) as they are monolingual files.

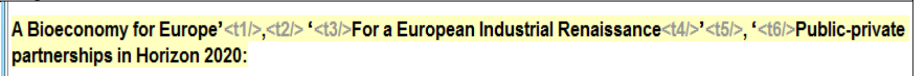
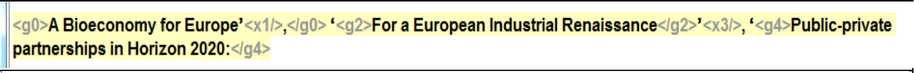
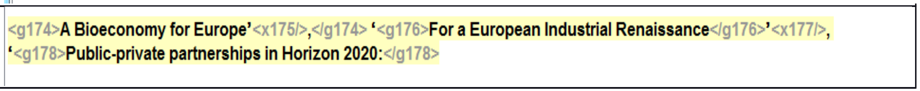
  The Okapi filter on the other hand, generates a bilingual file and can be reused as source file.

- **Tags:** In SDLXLIFF files, tag numbering is sequential from the first segment to the last segment of the project, while it is numbered by paragraph in OmegaT. Usually, this has no consequence for Studio users.

  In the native OT filter, the tags are numbered by paragraph, which is good as auto-propagation of repeated segments with tags is done… but the file format is monolingual, and when you start a project, SDLXLIFF file contains nothing in the <target> markup!

  In the Okapi filter, the file format is bilingual … but the tags are numbered as in the SDLXLIFF file - sequentially from the beginning to the end of the project. As a consequence, auto-propagation of repeated segments with tags is not possible in OmegaT.

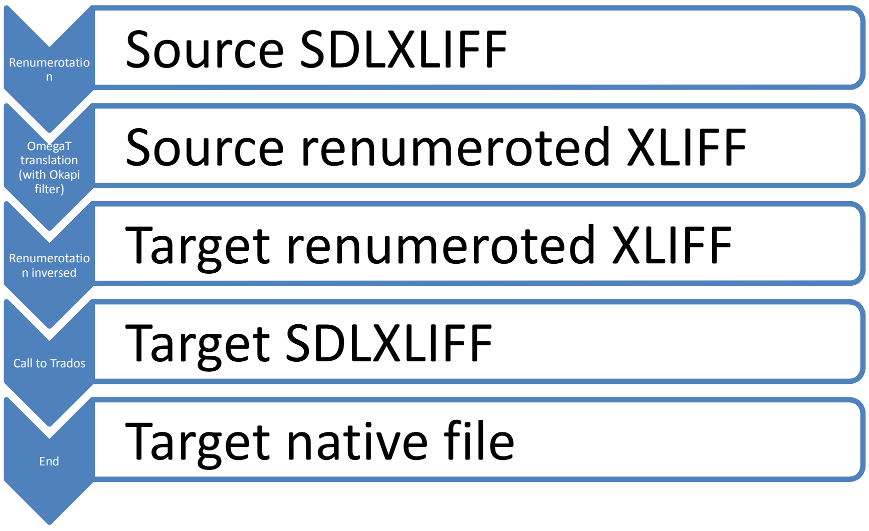
See below how a segment with formatting is displayed depending on the source document being a DOCX or an SDLXLIFF file and, for the latter, the difference when using the OmegaT or the Okapi XLIFF filters.

Format / filter	Segments
DOCX	 A Bioeconomy for Europe’<t1/>,<t2/> ‘<t3/>For a European Industrial Renaissance<t4/>’<t5/>, ‘<t6/>Public-private partnerships in Horizon 2020:
Native OmegaT filter: **XLIFF filter**	 <g0>A Bioeconomy for Europe’<x1/>,</g0> ‘<g2>For a European Industrial Renaissance</g2>’<x3/>, ‘<g4>Public-private partnerships in Horizon 2020:</g4>
Okapi XLIFF files filter	 <g174>A Bioeconomy for Europe’<x175/>,</g174> ‘<g176>For a European Industrial Renaissance</g176>’<x177/>, ‘<g178>Public-private partnerships in Horizon 2020:</g178>

Screenshot 1: Tag format

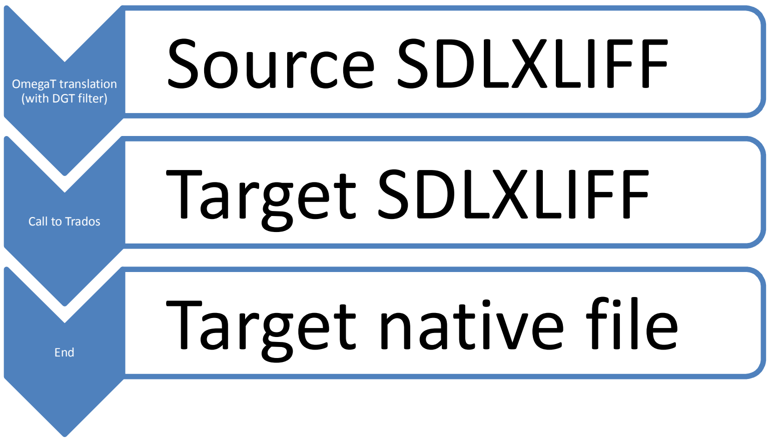
4. *Option chosen: develop a new SDLXLIFF filter for OmegaT and some associated scripts*

Considering the impossibility of modifying the OmegaT native XLIFF filter to make it bilingual and the difficulty of modifying the Okapi filter as it is part of a bigger architecture and therefore any changes could have impacts on other parts of the tool, a first solution was to create a small script called “**renumerotation**”, which “converts” the SDLXLIFF file into an XLIFF file, where tag numbers are reset to zero at each new paragraph, and another script which does the contrary in the translated document, once finished:



This solution works and is included in the public version of DGT-OmegaT 3.1 in case you have a good reason not to continue using the Okapi filter. If it can help, a small script called "call-renumerotation.groovy" is included in DGT-OT scripts folder, so that you do not need to call it manually. But it has never been integrated in the Wizard, as it was initially planned. Furthermore, there are a few SDLXLIFF files which had a strange behaviour in OmegaT with the Okapi filter … and we could not find the bug.

So, finally we chose another option, which takes more development work but which gives us more control of what OmegaT can do: to create a new SDLXLIFF filter from scratch, as well as some other scripts, thereby making it as compatible with Studio as possible. Now for the user (but not for developer!) the schema is simpler:



Both solutions are now integrated in the core of DGT-OT (version 3.1 and 3.2) and are also available as plugins, for those who prefer to use the standard OmegaT. However, DGT-OT adds some specific features, such as the possibility to open the native file or to exchange notes/comments between OmegaT and Studio, which are not in the plugins for the standard OmegaT, as they would not be compatible with it.

Furthermore, to have some possibilities such as creating source SDLXLIFF files or the final native file (via View target file/Create (current) target document(s)), DGT-OT does not totally replace Studio. It simply enables Studio to be called without opening its GUI. To do that, you must have Studio installed and you must add to it a small command-line tool, which can be downloaded in the DGT-OmegaT website[iii].

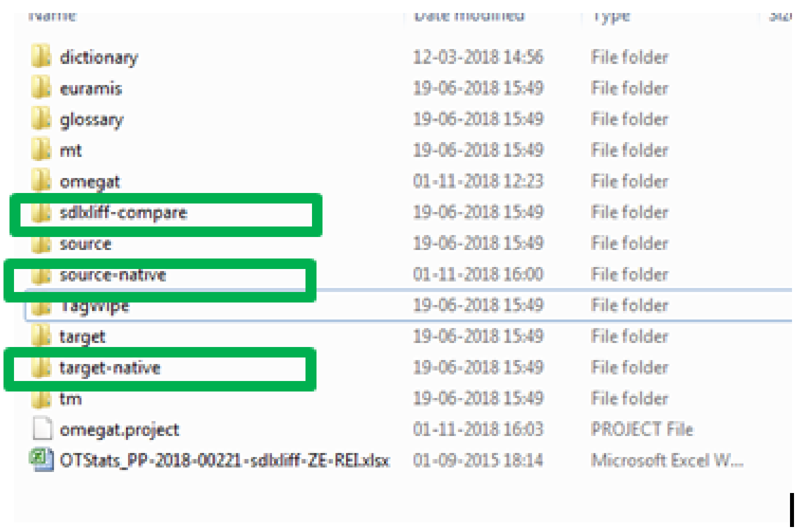
In case you don't have Studio installed, an alternative is to make use of DGT-OT "cross-compilation": this will create an OmegaT project based on the original native file and try to produce the native target file based on the memory of the current project.

This is far from being perfect (there may be problems with segments with tags and segmentation rules may differ from the SDLXLIFF file and the native file read by OmegaT's filter) but it works in any Java platform, even without Studio.

5. *Translating/revising with SDLXLIFF source files - What is different from a translator's perspective*

So, for translators, what are the practical differences of working with SDLXLIFF source files as compared to native format source files, using the new version of DGT-OT?

In the sections below, you will see that several project subfolders are created to manage DGT-OT projects with SDLXLIF source files.



Name	Date modified	Type	Siz
dictionary	12-03-2018 14:56	File folder	
euramis	19-06-2018 15:49	File folder	
glossary	19-06-2018 15:49	File folder	
mt	19-06-2018 15:49	File folder	
omegat	01-11-2018 12:23	File folder	
sdlxliff-compare	19-06-2018 15:49	File folder	
source	19-06-2018 15:49	File folder	
source-native	01-11-2018 16:00	File folder	
Tagwipe	19-06-2018 15:49	File folder	
target	19-06-2018 15:49	File folder	
target-native	19-06-2018 15:49	File folder	
tm	19-06-2018 15:49	File folder	
omegat.project	01-11-2018 16:03	PROJECT File	
OTStats_PP-2018-00221-sdlxliff-ZE-REL.xlsx	01-09-2015 18:14	Microsoft Excel W...	

Screenshot 2: Project subfolders created for SDLXLIFF source projects

5.1. What to install / check

If you download the latest version of DGT-OT (stable 3.1 or development 3.2) most of these features are already included. The only exception is the command line, which must be downloaded separately simply because it must be installed in the Studio's directory.

In case you prefer to use the standard OmegaT, part of these features are compatible with versions 3.6 (but with Java 8) or 4.1 (which is already for Java 8):

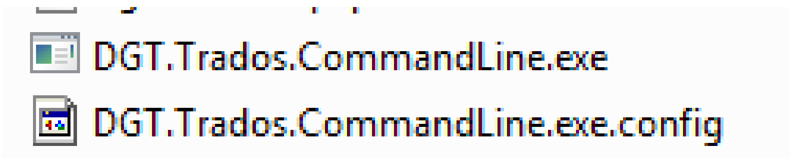
- **The Renumerotation** scripts should work correctly with OmegaT 3.6 or 4. Note that they are written in Perl, so Windows users may have to previously install an interpreter

- The **SDLXLIFF filter** is also available as a plugin compatible with OmegaT 3.6+java 8 and OmegaT 4.1. But the plugin version does not add DGT-specific features such as pseudo-tags or support for notes.

For full explanations and detailed technical information, see the DGT-OmegaT website[iv].

5.1.1. Add-in in Studio

For external use, the following files (available in the DGT-OmegaT website[v]) should be installed in Studio's directory (usually *C:\Program Files (x86)\SDL\SDL Trados Studio\[your version of Studio]):*

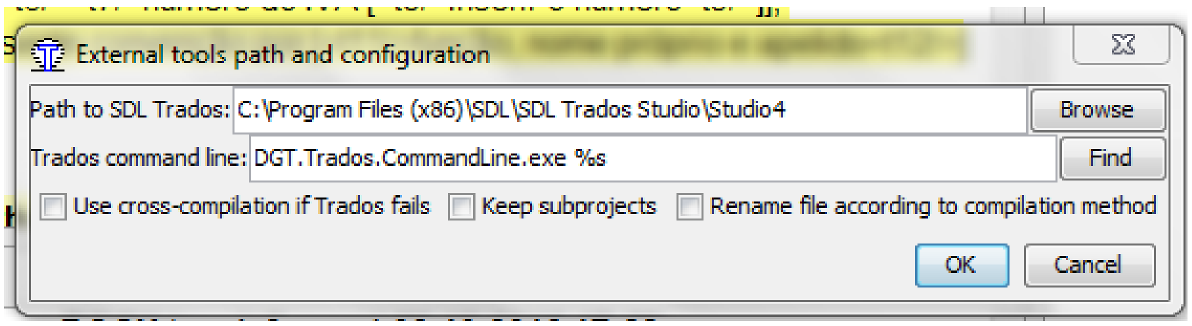


DGT.Trados.CommandLine.exe

DGT.Trados.CommandLine.exe.config

Screenshot 3: External tool for Studio

Once installed, in DGT-OT, in the **Options – External tools** menu, the path and command line must be configured. Even if most Studio versions require the file to be in a dedicated directory, you must configure it (see section 5.2 for details) because otherwise OmegaT doesn't know which version of Studio you want to use. **In the future, this window may be used to call totally different tools such as Okapi Tikal, which also supports (SDL)XLIFF files.**



External tools path and configuration

Path to SDL Trados: C:\Program Files (x86)\SDL\SDL Trados Studio\Studio4

Trados command line: DGT.Trados.CommandLine.exe %s

Browse
Find
Use cross-compilation if Trados fails
Keep subprojects
Rename file according to compilation method
OK
Cancel

Screenshot 4: External tool for Studio

5.1.2. New SDLXLIFF filter for OmegaT

In DGT-OmegaT, the new filter is already integrated in the DGT-OmegaT-core and binaries.

It will be displayed in the File Filters menu (see Screenshot 5). Just make sure that the other XLIFF file filters are inactivated.

For users of standard OmegaT, the filter is also available as a plugin, but the plugin offers only features available in standard OmegaT for other formats.

5.1.3. Scripts: Calculate SDLXLIFF differences, Call Renumerotation, call-trados-commandLine

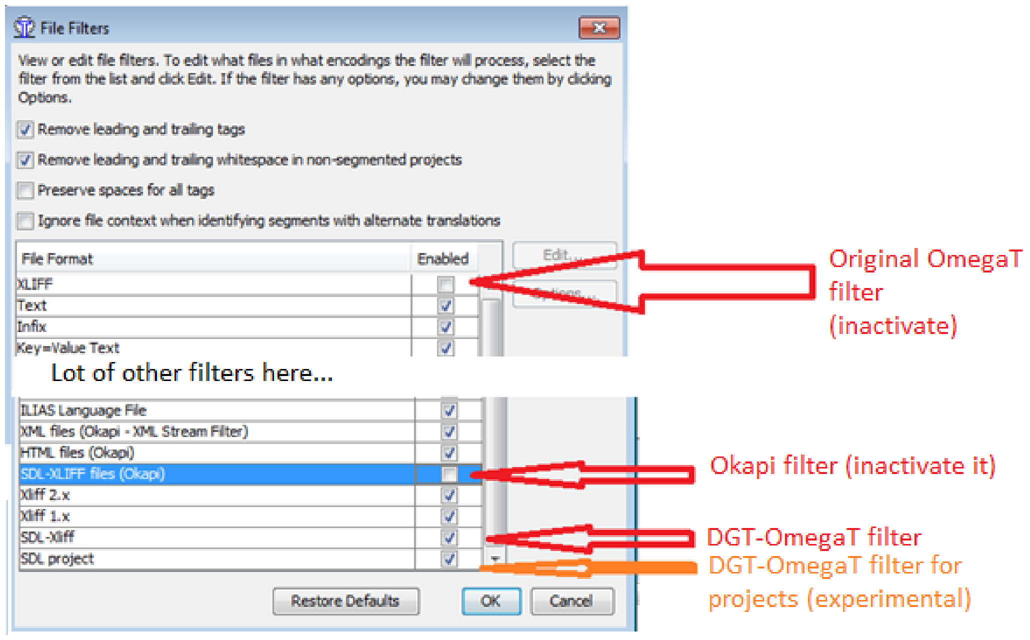
These scripts are integrated in the DGT-OT core and binaries of version 3.1 and 3.2 available in the DGT-OmegaT website. **So, the translator just has to run them from the Tools - Script menu.**

These scripts work in batch mode and will perform the action for all the documents in the project.

These scripts should be compatible with the standard OmegaT 3.6 or 4.1, but we do not guarantee that we will check regularly that they work, in case their API changes.

5.2. Checking the working environment

Before starting working with an SDLXLIFF project, you must check if the active filter for the project is the **SDL-Xliff** filter (the filter developed by DGT for SDLXIFF source files). **Unless, of course, you prefer to use renumerotation, or simply prefer the Okapi filter and don't care about sequential segmentation (which has no consequence on the final result).**


File Filters

View or edit file filters. To edit what files in what encodings the filter will process, select the filter from the list and click Edit. If the filter has any options, you may change them by clicking Options.

- [✓] Remove leading and trailing tags
- [✓] Remove leading and trailing whitespace in non-segmented projects
- [ ] Preserve spaces for all tags
- [ ] Ignore file context when identifying segments with alternate translations

File Format	Enabled
XLIFF	
Text	✓
Infix	✓
Key=Value Text	✓
ILIAS Language File	✓
XML files (Okapi - XML Stream Filter)	✓
HTML files (Okapi)	✓
SDL-XLIFF files (Okapi)	
Xliff 2.x	✓
Xliff 1.x	✓
SDL-Xliff	✓
SDL project	✓

Edit...
Options...
Lot of other filters here...
Restore Defaults
OK
Cancel
Original OmegaT filter (inactivate)
Okapi filter (inactivate it)
DGT-OmegaT filter
DGT-OmegaT filter for projects (experimental)

Screenshot 5: Filters

In order to use the interaction with Studio, in DGT-OT, in the Options – External tools menu, the path and Trados command line must be correctly configured (see Screenshot 4):

- **Use cross-compilation if Trados fails**: Here you can decide, whenever you want, if cross-compilation is to be used or not when the previous and "best" method (which calls Trados Studio) does not work for some reason.

- **Keep subprojects**: Here you can choose to have the temporary Studio project (created either by cross-compilation or by compiling an SDLXLIFF file which did not have a project associated) kept in the disk after compilation or if you want it to be deleted immediately.

- **Rename file according to compilation method**: This will change the target file name, adding the suffix *-existingProject, -dummyProject or –crossCompiled*, depending on which method finally was successful to create it. We cannot add this information anywhere else: if added to the document itself, we would have to do it for lots of file formats, with the risk of sometimes having unexpected behaviour when the documents are opened in native editor.

5.3. Creating a new project with untranslated document(s)

For DGT translators, the possibility of creating a project with SDLXLIFF source files is being integrated in the DGT-OT Wizard.

For external translators, either the client provided SDLXLIFF source file(s) to be translated or not. In the latter case if requested to deliver the translations also in SDLXLIFF format, the translator must first create the project in Studio … which implies that s/he must have Studio installed.

One problem with SDLXLIFF files just generated by Studio is that when you use the default Studio template, these files are not segmented by sentence: they contain one <trans-unit> per paragraph, which is correct, but <seg-source> is not filled in. As a consequence, OmegaT will either show one segment per paragraph, or use **its own segmentation**, which may differ from the segmentation used by Studio!

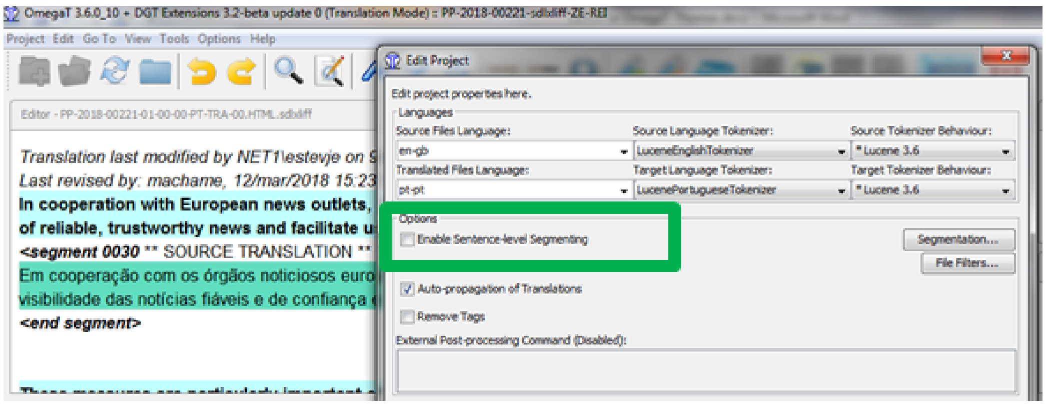
To avoid that, you must execute the "*Analyze files*" task on Studio, which implies that you must have not only the SDLXLIFF file, but also the SDLTM files which contain the segmentation rules (else, Studio will use its own default rules, which may differ from yours!).

For external translators working on DGT projects, the only solution is that DGT executes the task before sending the SDLXLIFF to the translator, so that s/he can use DGT's rules rather than his/her own rules or the rules of his/her company!

The Trados Command line, already discussed in the previous section, also has the capacity to call any Studio task without opening Studio GUI (but you must have Trados installed on your computer).

In DGT-OT there is the small script "call-Trados-commandLine.groovy" which will do nothing more than prepare the correct parameters for the current SDLXLIFF file and call the command line to analyse the file. In the future, this script will be replaced by a call inside the Wizard.

Warning: once Studio applies its own segmentation, in OmegaT you must deactivate - if not already done - sentence-level segmenting in the current project:



OmegaT 3.6.0_10 + DGT Extensions 3.2-beta update 0 (Translation Mode) :: PP-2018-00221-sdlxliff-ZE-REI

Editor - PP-2018-00221-01-00-00-PT-TRA-00.HTML.sdlxliff

Translation last modified by NET1\estevje on
Last revised by: machame, 12/mar/2018 15:23
In cooperation with European news outlets,
of reliable, trustworthy news and facilitate u
<segment 0030 ** SOURCE TRANSLATION **
Em cooperação com os órgãos noticiosos euro
visibilidade das notícias fiáveis e de confiança
<end segment>

Edit Project

Edit project properties here.

Languages

Source Files Language: en-gb
Source Language Tokenizer: LuceneEnglishTokenizer
Source Tokenizer Behaviour: * Lucene 3.6
Translated Files Language: pt-pt
Target Language Tokenizer: LucenePortugueseTokenizer
Target Tokenizer Behaviour: * Lucene 3.6

Options

☐ Enable Sentence-level Segmenting
☑ Auto-propagation of Translations
☐ Remove Tags

External Post-processing Command (Disabled):

Screenshot 6: Sentence-level segmenting deactivated

If you don't, OmegaT will correctly read segments from Studio, but then it will apply OmegaT's rules *in addition to* Studio's rules! This will have no consequence for the resulting SDLXLIFF file, but you may see in OmegaT more segments than really needed…

5.4. Creating a project with SDLXLIFF target files for revision purposes

The process is the same, but the source file selected is the SDLXLIFF translated file.

In DGT, the project is created via the DGT-OT Wizard and the **sdlxliff** box must be ticked so that OmegaT's segmentation by segment is deactivated when opening the project (don't worry about Studio's *Analyze* task in this case: if the file is translated, this has already been done during translation).

5.4.1. Viewing in DGT-OT comments added by the translator in Studio

Segments with comments added by the translator in Studio are displayed in DGT-OT in the Comments pane.

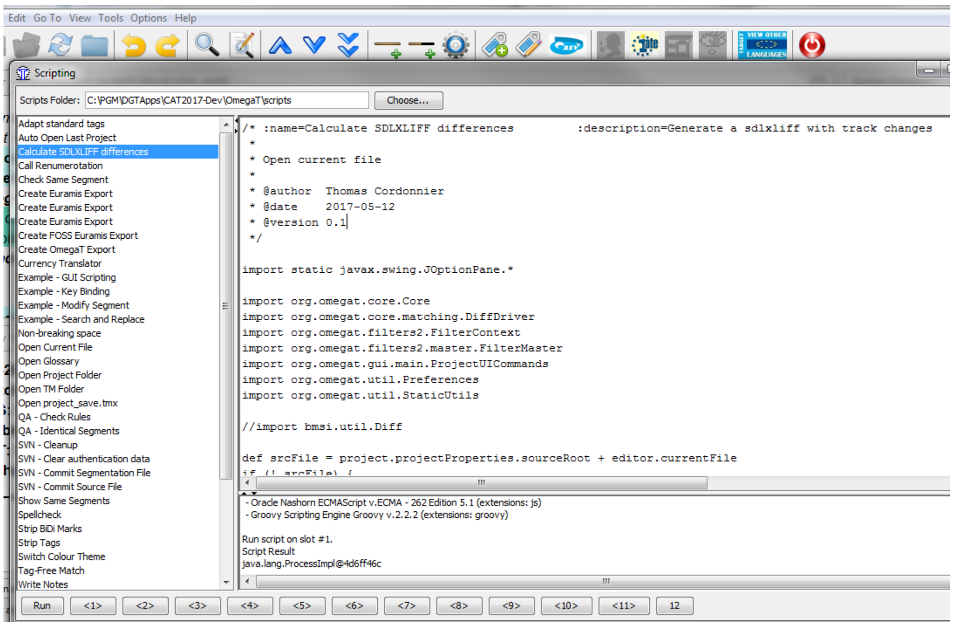
On the other hand, notes added by the reviser in DGT-OT will not be displayed in Studio when the revised project is sent back to the translator. To send the notes to the translator, the reviser can generate a list of all the notes using the script Write notes and send him/her that file.

5.4.2. Generating the SDLXIFF revised target file with track changes

As in DGT the translator generally has the last word, it is important to be able to send back to the translator the revised project downloaded via the Studio CAT Client with track changes.

For that purpose, a script has been developed – Calculate SDLXLIFF differences – to produce a SDLXLIFF target file which compares the SDLXLIFF source file with the SDLXLIFF target file and allows to display the changes made by the reviser when that file is transferred to the Studio project, so that the translator can check – and accept or refuse – the changes made by the reviser.

To generate the SDLXLIFF target file with track changes, in the Tools menu, the translator runs the *Calculate SDLXLIFF differences* script.



Scripts Folder: C:\PGM\DGTApps\CAT2017-Dev\OmegaT\scripts

Adapt standard tags
Auto Open Last Project
Calculate SDLXLIFF differences
Call Renumerotation
Check Same Segment
Create Euramis Export
Create Euramis Export
Create Euramis Export
Create FOSS Euramis Export
Create OmegaT Export
Currency Translator
Example - GUI Scripting
Example - Key Binding
Example - Modify Segment
Example - Search and Replace
Non-breaking space
Open Current File
Open Glossary
Open Project Folder
Open TM Folder
Open project_save.tmx
QA - Check Rules
QA - Identical Segments
SVN - Cleanup
SVN - Clear authentication data
SVN - Commit Segmentation File
SVN - Commit Source File
Show Same Segments
Spellcheck
Strip BiDi Marks
Strip Tags
Switch Colour Theme
Tag-Free Match
Write Notes

```
/* :name=Calculate SDLXLIFF differences        :description=Generate a sdlxliff with track changes
 *
 * Open current file
 *
 * @author  Thomas Cordonnier
 * @date    2017-05-12
 * @version 0.1
 */

import static javax.swing.JOptionPane.*

import org.omegat.core.Core
import org.omegat.core.matching.DiffDriver
import org.omegat.filters2.FilterContext
import org.omegat.filters2.master.FilterMaster
import org.omegat.gui.main.ProjectUICommands
import org.omegat.util.Preferences
import org.omegat.util.StaticUtils

//import bmsi.util.Diff

def srcFile = project.projectProperties.sourceRoot + editor.currentFile
```

```
 - Oracle Nashorn ECMAScript v.ECMA - 262 Edition 5.1 (extensions: js)
 - Groovy Scripting Engine Groovy v.2.2.2 (extensions: groovy)

Run script on slot #1.
Script Result
java.lang.ProcessImpl@4d6ff46c
```

Screenshot 7: Script to generate SDLXLIFF revised target file with track changes

That file is stored in the project *\sdlxliff-compare* subfolder (see Screenshot 2) and must be copied manually to the target folder of the Studio project.

5.5. Segment status in DGT-OT and in Studio

OmegaT has only 2 segment statuses: untranslated and translated. In DGT-OT a new status is generated for revision purposes by adding that information to the project memory.

Studio has 7 segment statuses: not translated, draft, translated, translation rejected, translation approved, sign-off rejected and signed-off.

In DGT-OT, when the translated or revised SDLXLIFF file is generated (Create (current) translated document(s) or View target file), the segment status will be set as «translated» if it is not empty. This works only if you use the DGT SDLXLIFF filter or Okapi filter with renumerotation.

5.6. Generating (and viewing) the source and target documents in native format

The target documents are generated as usual in the project *\target* subfolder with the Create (current) translated document(s). When using SDLXLIFF source documents, the target files are therefore in SDLXLIFF format.

However, with DGT-OT (not with standard OmegaT, even with the plugin!), the native (Office or other) target file will also be generated in the *\target-native* subfolder which is created when SDLXLIFF translated documents are generated.

Both the source and the target files can be viewed as usual in DGT-OT using the feature View source file and View target file if there is a Studio project created, but also if there is no Studio project created … as long as the translator has Studio installed in his/her computer.

If there is a Studio project, the SDLXLIFF target file and the native target file are generated in the Studio target folder and copied to the DGT-OT \target and \target-native project subfolders.

If there is no Studio project previously created (as the translator received the SDLXLIFF source file), DGT-OT creates a basic Studio file which generates the SDLXLIFF target file and the native target file in the DGT-OT \target and \target-native project subfolders.

If Studio is not installed, DGT-OT will not find it and will use a cross-compilation technique which will even so generate the native target file. However, there may be segments which are shown untranslated in the native target document, either because the segment has tags or because the segmentation is different.

5.7. Opening the document in Studio

Furthermore, the feature Open in Trados Studio available in the Project menu allows to quickly open the document in Studio and, if the translator so wishes, to continue working in Studio as the target file in the DGT-OT project is automatically copied to the target folder in the Studio project when the target document(s) is(are) generated.

However if, after modifying the translation in Studio, the translator wants to go (back) to DGT-OT to continue working on that document, then the SDLXLIFF target file must be generated in Studio and used/copied to the **\source** folder in DGT-OT project.

Take into consideration that in Studio, the target SDLXLIFF file has priority over the project memory, while in DGT-OT the project memory (the project_save.tmx file in the project *omegat* subfolder) has priority. So, be careful to use a (new) DGT-OT project without a project memory, as segments in the project memory will override segments differently translated in the SDLXLIFF source file changed in Studio, in the case of a DGT-OT project which had been previously used.

6. Wishes!

This is still work in progress as the interaction with Studio is not easy, but we find it useful for our work. We hope it may also be helpful to users outside DGT … as was the case with TagWipe.

i *Free and open-source software – a translator's good friend*: http://ec.europa.eu/translation/portuguese/magazine/documents/folha45_foss_en.pdf; *Free open source software – **CAT Tools: OmegaT in DGT, its Wizard, Tagwipe and TeamBase (Part 1 and 2):*** **http://ec.europa.eu/translation/portuguese/magazine/documents/folha54_ot_en.pdf; http://ec.europa.eu/translation/portuguese/magazine/documents/folha55_ot_en.pdf;** ***OmegaT in DGT, its Wizard, TagWipe and TeamBase: Documentation:*** **http://ec.europa.eu/translation/portuguese/magazine/documents/folha56_otd_en.pdf**

ii **http://185.13.37.79/**

iii **Link to be added**

iv **Link to the specific DGT-OT section or to your document if it is in pdf format.**

v Link to be added