

TUMATXA USER MANUAL

Note: I apologise for my English because it is not as good as I would like it to be. However, I will try to do my best writing this user's manual.

Tumatxa - ZTMX provides Zope with an easy and useful TMX files management system (try <http://www.lisa.org/tmx> to know what a TMX file is).

Notes on installation

Just unpack the product in the Products directory and restart Zope. Then import the ZTMX.zexp file into your Zope instance (the .zexp file must be placed in your Zope 'import' directory). It adds a ZTMX Corpus named 'ZTMX' and titled 'Tumatxa' to your computer. It contains some objects that are necessary for Tumatxa to work, such as a Catalog, a MessageCatalog, a Localizer object and some forms (search forms, local menu...) and images. You could modify them as you wish.

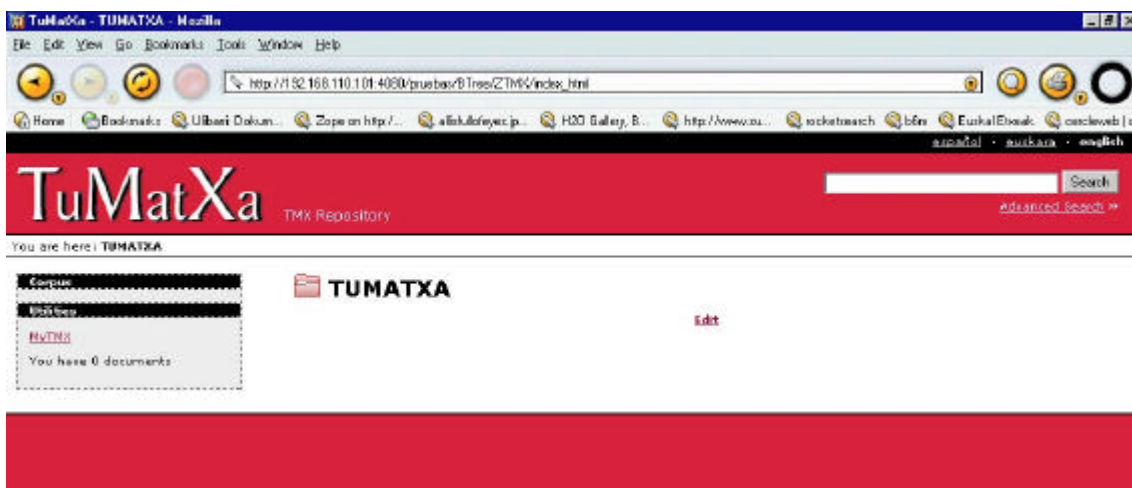
If you are going to import UCS-2 encoded files or other files whose encodings cannot be recognized by Zope, (like Trados exported files) you must have Linux installed, since Tumatxa uses the 'iconv' Linux command.

Programs that your computer needs to have installed:

- ⇒ Zope 2.6 or later
- ⇒ Python 2.1 or later
- ⇒ BTreeFolder2 product - (developed with 0.5.0 version)
 - <http://hathaway.freezope.org/Software/BTreeFolder2>
- ⇒ TextIndexNG2 product - (developed with 2.0.1 version)
 - <http://www.zope.org/Members/ajung/TextIndexNG>
- ⇒ Localizer product - (developed with 1.0.1 version)
 - <http://www.j-david.net/software/localizer/>

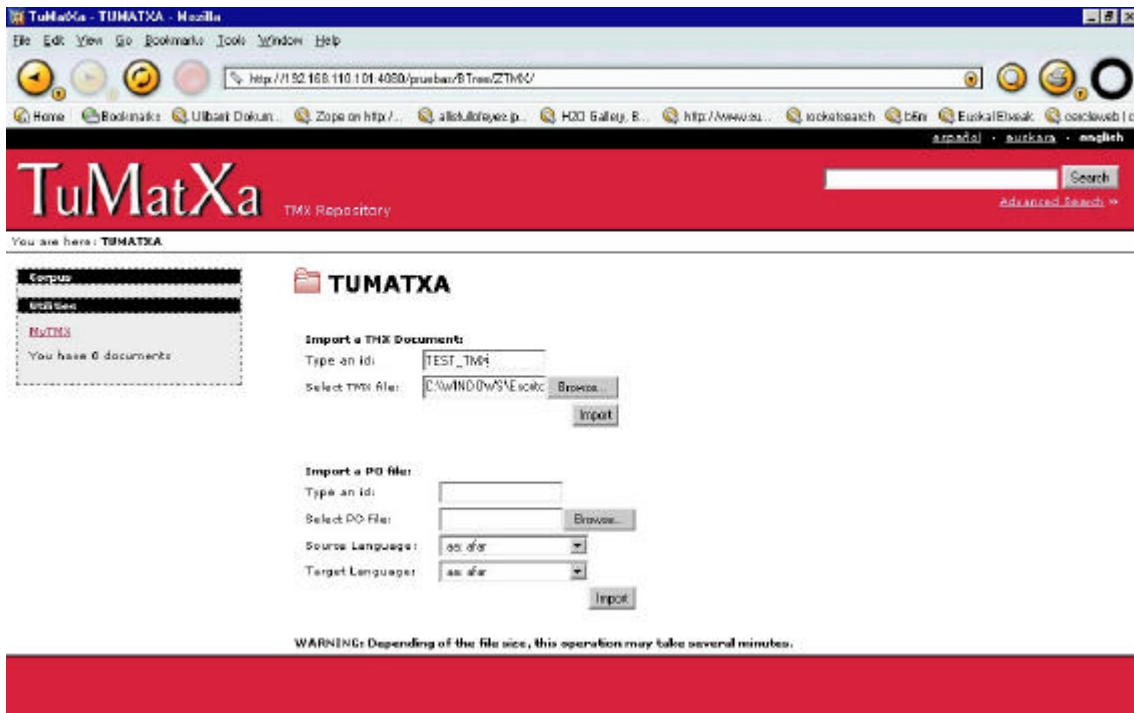
Starting with Tumatxa: edit a ZTMX Corpus

The first thing you should see when you install the ZTMX product is the main page of the ZTMX Corpus. The initial screen is empty at the moment. However, it will show contents as you go on.

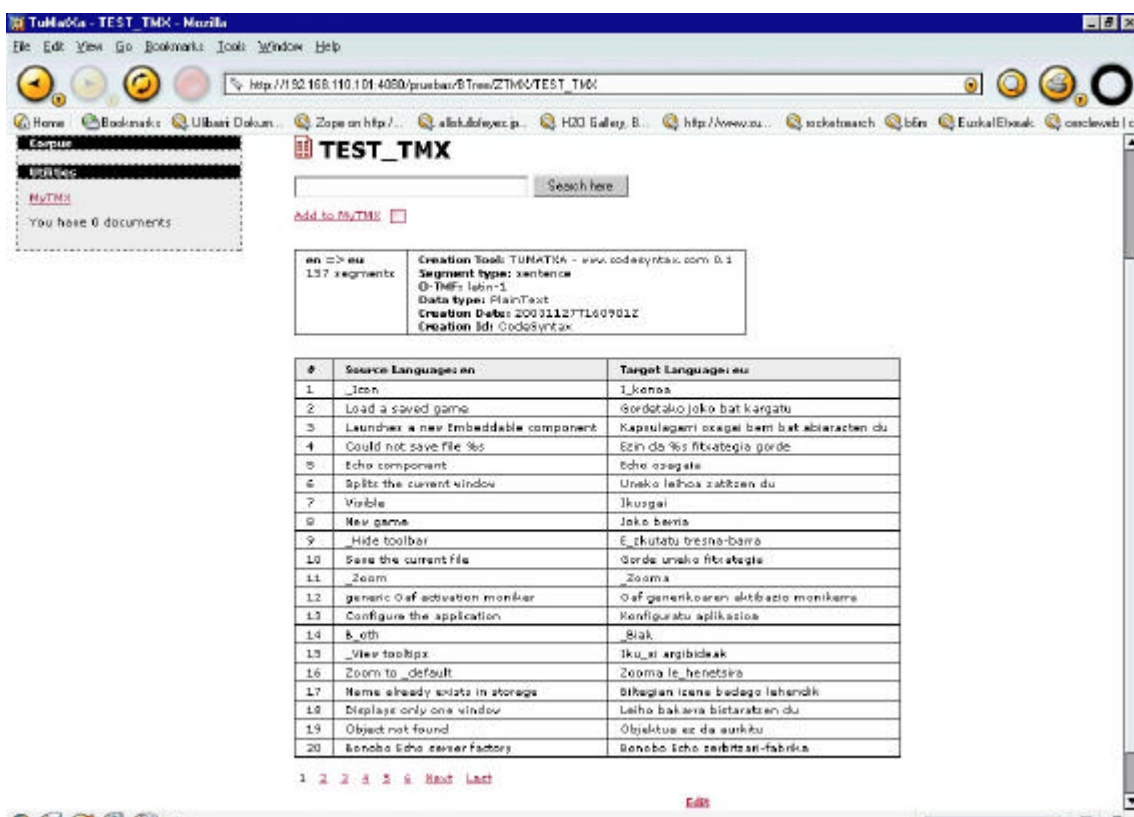


You can see a link called 'Edit' in the middle of the screen. If you click on it, you will be able to edit the current item, namely a ZTMX Corpus or a ZTMX Document. You can see a text input on the right top of the screen. This allows you to make a search in the entire repository. There is also a link called 'Advanced Search' that allows you to refine your search parameters. It will be explained in more detail later.

If you click on 'Edit' you can either modify the current ZTMX Corpus or ZTMX Document in two ways. On the one hand, you could add new Corpus or Documents, or rename, cut, delete... the existing ones as if it were a Zope folder (for a ZTMX Corpus). On the other hand, you could add new segments or edit the document header (for a ZTMX Document). Let's add a new ZTMX Document. Here is the next screen:



Here you can add a TMX file or a Po file. If you import a Po file, you must specify the source and target languages, since they are two parameters that don't appear in a Po file header. Select your file and click on Import. If you do not specify an id, Tumatxa will generate a numeric one for you.



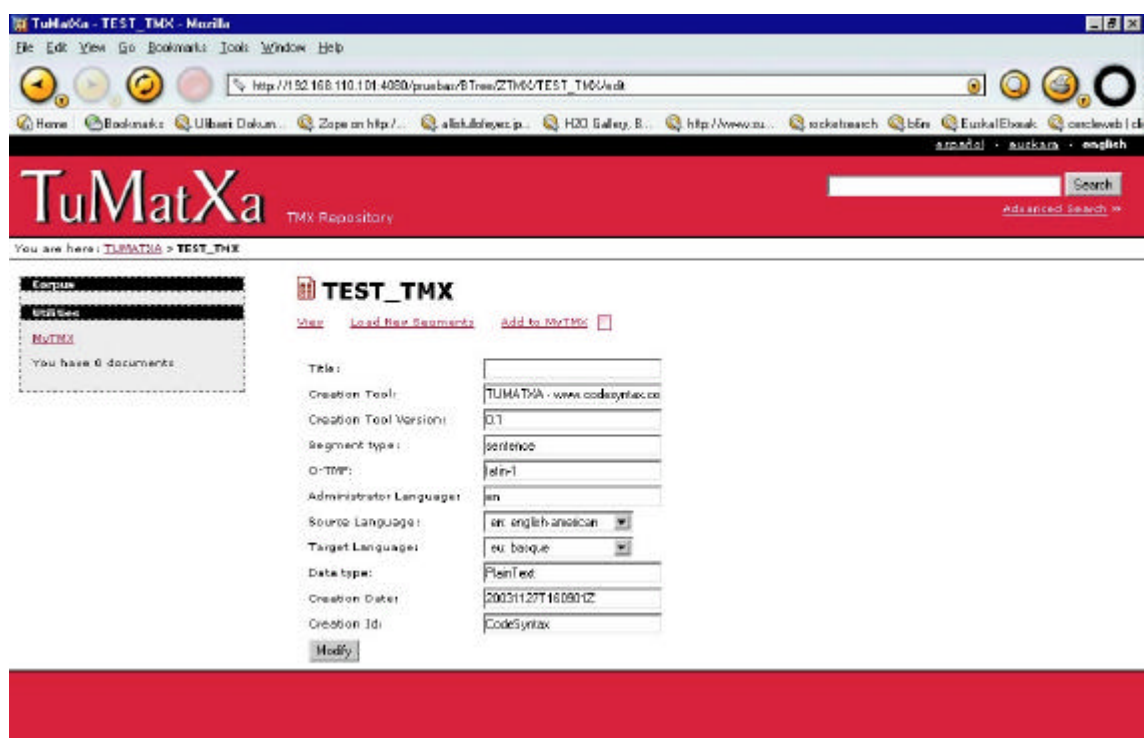
Here you can see the header parameters, such as the creation tool, the segment type... On the other hand, you can also see the source and target languages and the size of the document (the number of segments it has). You also can see segments and surf between pages watching the segments. You can see the 'Edit' link below, which will let you modify the header parameters of the document and add new segments (you could do this in different cases: in case you would like to update the current document, since it has grown since the last time you translated it; in case you would like to append new segments that have relation with this document; in any other case you would like).

You can repeat this process as many times as you wish in order to have your own TMX repository. You could organize them in the Corpus adding new ZTMX Corpus inside Corpora.

ZTMX Document

We have seen how the index view of a ZTMX Document is like in the previous chapter. Here you can surf between the different pages which contain segments and see the document header. Here are the different features we have in a ZTMX Document.

A ZTMX Document and a TMX file are the same. It is an object that stores the header parameters as properties and stores the different segments from the TMX file as ZTMX Segment objects. If you click on the 'Edit' link of a ZTMX Document, you will have the following screen:

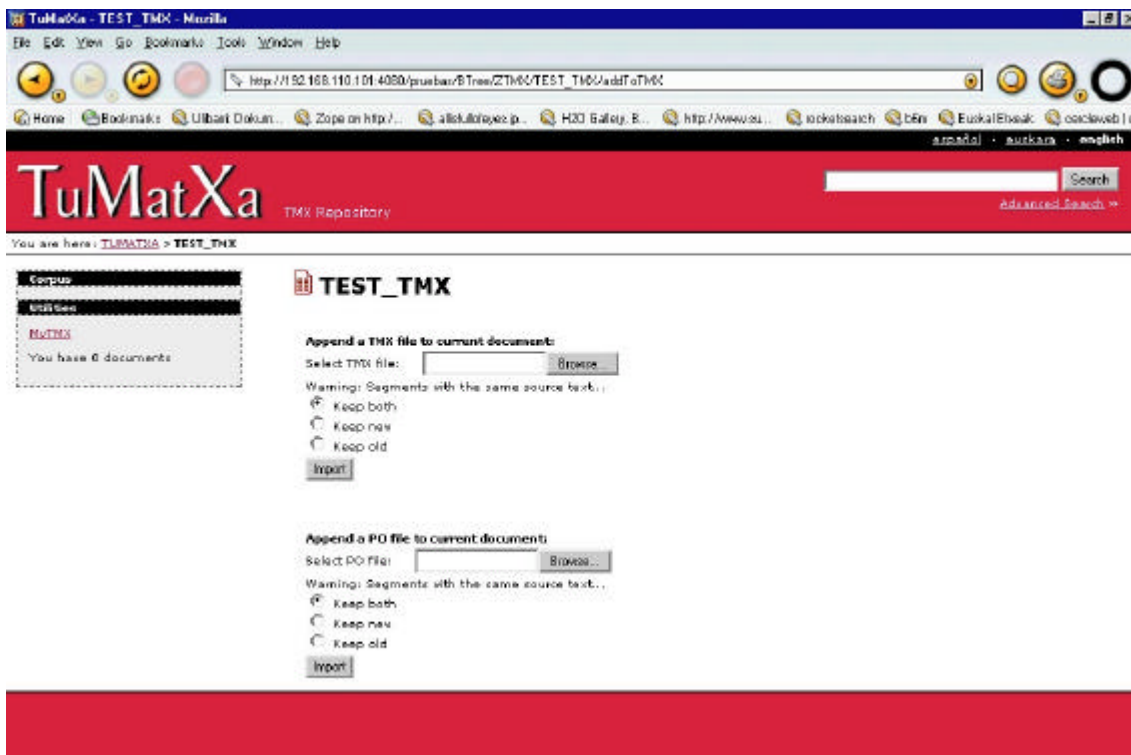


Here you can edit the header parameters of the documents. There are two properties that you cannot find in a TMX file: the Title and the Target language. These two properties will not appear when you export the document, so you should not worry. They are used to have a better information about the ZTMX Document.

You can see some links under the id (or the header) of the document. The 'View' link allows you to go back to the index view of the document. The 'Load New Segments' link lets you add new segments to the current document. On the other hand, the 'Add to MyTMX' link allows you to stick this document down in order to have it exported later. If the document has been stuck down already, it would appear 'Remove from MyTMX'.

If you make any change on the properties, you must click on "Modify" to make changes become operative.

When you want to load TMX document (file) with new segments, you must click on the appropriate link. After that, the following will appear:



Here you can import new segments either from a TMX file or from a Po file (as when you imported the document). You must specify which file it is and what to do with the segments whose source language text will be repeated. You can keep both segments (Keep both), you can overwrite old segments with the new ones (Keep new) or you can keep the old segments (Keep old). It is obvious that if a segment is repeated (same source and target text), it will not be loaded to the document.

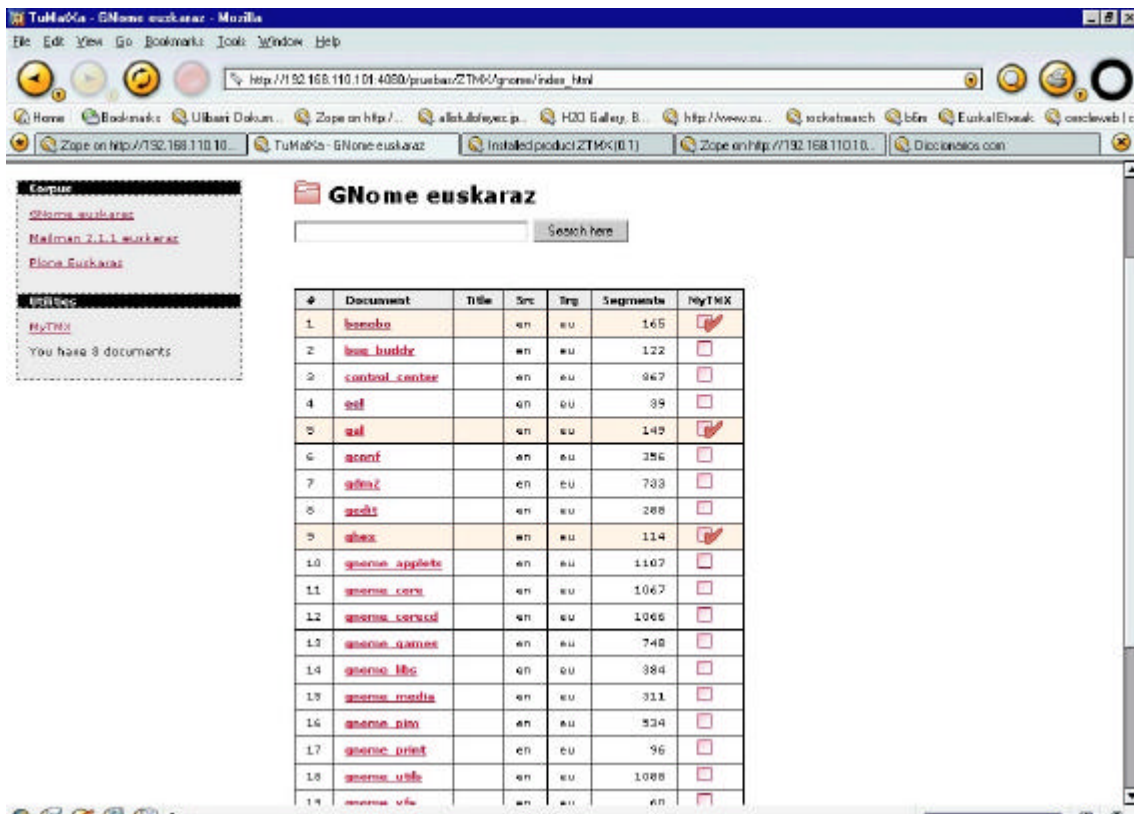
ZTMX Segment

It is not possible to add a single segment from the Tumatxa interface (that must be done from the Zope interface). However, it is possible to modify a single segment. In order to do so, you must go to the document index view. Here the segments of the document appear following a numeric order. When you click on the 'Edit' link, you log on Zope. Well, when you go back to the index view, you will see that you can click on the number of the segments. That link will take you to an edit view of the segment. document ou can modify any segment of the document here.

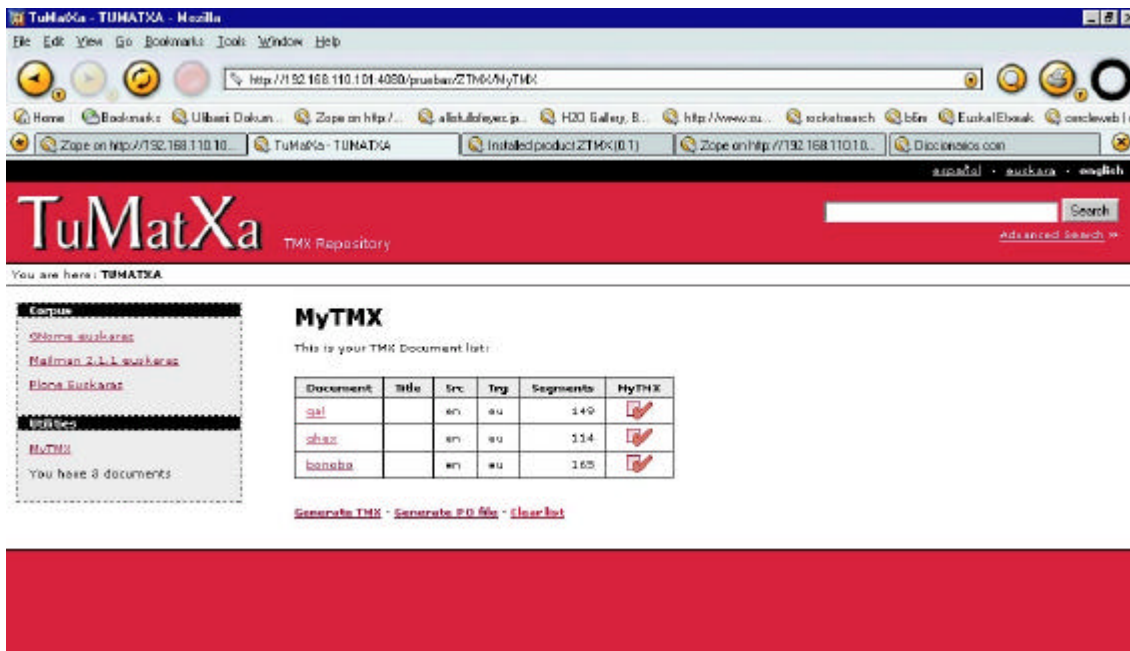
MyTMX

MyTMX works like a shopping cart. You can add documents to this cart in order to download them like a TMX file or a Po file. You can stick documents down from any document view (as we have seen before), or from a Corpus index view by clicking on the MyTMX box. If the document has not been stuck down yet, you can make it come unstuck.

On the left side of the screen you can see how many documents we have stuck down to our MyTMX. It goes without saying that you cannot stuck down documents whose source and target languages are different from the others.



Next, if you click on the MyTMX link on the Utilities block (right side of the screen), you will see the following:



Here appears the list of documents you have stuck down to MyTMX. You can see some links as well: 'Generate TMX', 'Generate Po file' and 'Clear list'. As you may suppose, 'Generate TMX' takes us to a screen that allows us to generate a TMX file. 'Generate Po' lets us generate a Po file. And 'Clear list' clears the document list we have in MyTMX.

If you want to generate a TMX file, click on 'Generate TMX', and you will see the following:

The screenshot shows the TuMatXa web interface. At the top, there is a navigation menu with links for 'Corpus', 'MyTMX', and 'You have 3 documents'. Below this, the 'My TMX header' section is displayed, featuring a table of configuration parameters:

Creation Tool:	TUMATXA - www.codesyntax.com
Creation Tool Version:	0.1
Segment type:	sentence
O-TMFX:	Latin-L
Administrator Language:	en-english-american
Source Language:	en
Target Language:	eu
Data type:	PlainText
Creation Date:	20091208T164645Z
Creation Id:	CodeSyntax

Below the table, there is a 'Segment order' dropdown menu set to 'en - eu' and an 'Export' button.

Here you can see the header parameters that your TMX file will have (except target language, obviously). You can select the Administrator Language to adjust to your requirements. You can also choose the order of the segments. This is an interesting feature, since you can invert the order of the segments of any TMX file (and have the translations as source segments). When you select this, click on Export and you will have your TMX file.

If you want to generate a Po file, you should follow the same process (by clicking on 'Generate Po file', obviously). You will see the Po file header and you can choose the order of the segments.

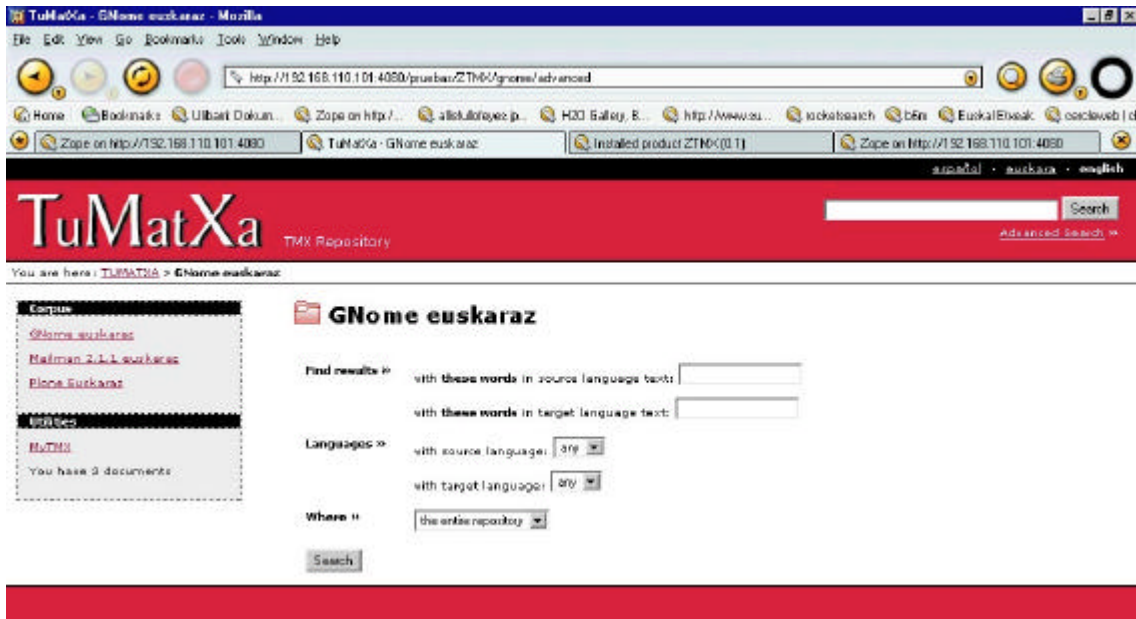
Search

If you would like to search for something, you have three options: you can make a search in the entire repository, in the current Corpus/Document or make an advanced search.

If you would like to search for something in the entire repository, you can use the input box you can find on the right top of your screen. You can type the words, the phrase or whatever you would like to search for there. Tumatxa will return a list of segments where those search terms can be found on their source language text.

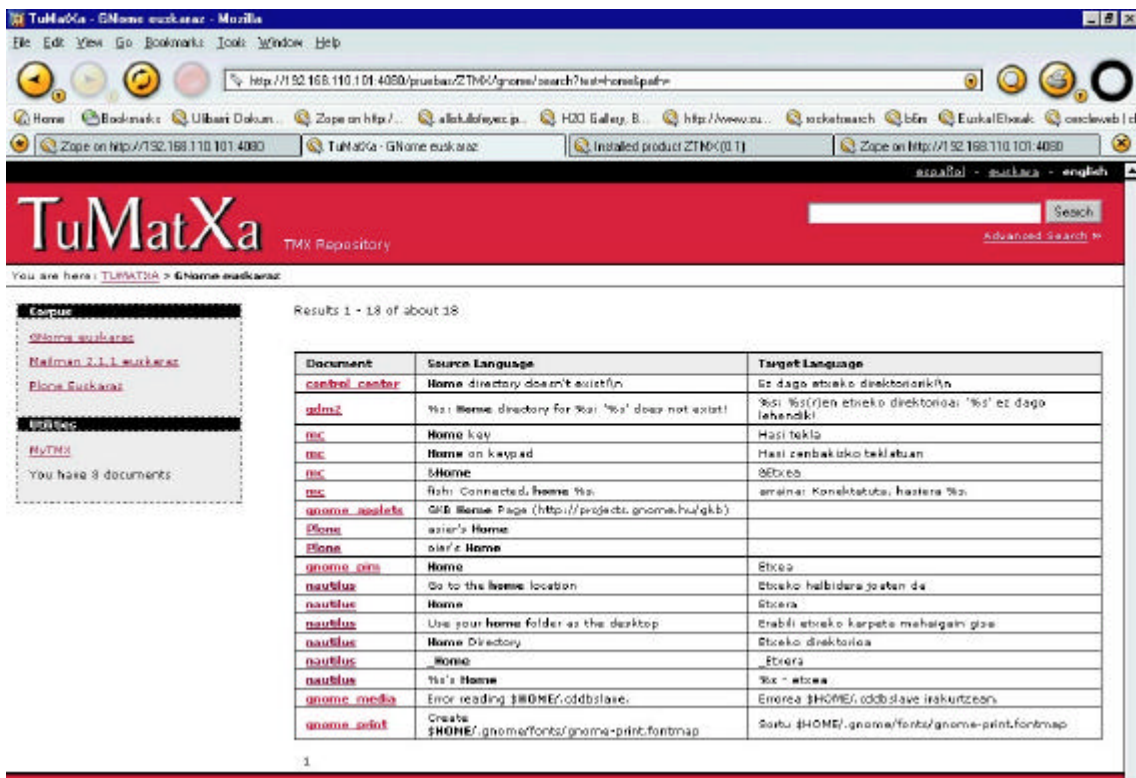
The procedure you should follow in order to search for something in the current Corpus or Document is the same, although you must type your search term on the input box that appears under the header of the Corpus or the Document. You will search on the segments it contains or the segments contained in any subcorpus that it contains.

If you want to do an advanced search, you must click on the 'Advanced Search' link on the right top of the screen. Then you will see a site like this:



You will be able to refine your search there. You can search for terms in the source text and in the target text. You can only search in segments whose source or target language is one of the selected languages or any. You can search in the entire repository, or in the current Corpus/Document.

When you click the 'Search' button, you will see something like this:



In the results page you will see the segments that contain your search term, their translation and a link to the document that contains them so that you can go there and have it added to your TMX. The results page is the same for all the searches you make and it shows 20 segments per page. You can surf between pages with the links below the table.

Quick reference

I have included a list of the methods you can find in the ZTMX product.

ZTMX Corpus

```
def manage_addZTMXCorpus( dispatcher, id, title="", date="", author="",REQUEST=None):
    """ Adds a new ZTMX Folder object with id *id*. """

def addCorpus(self, id, title="", REQUEST=None, RESPONSE=None):
    """ Adds a new TMX Corpus. It checks the id. """

def delCorpus(self, id, REQUEST=None, RESPONSE=None):
    """ Deletes a TMX Corpus """

def autoDetectEncoding(self, file, REQUEST=None, RESPONSE=None):
    """ buffer -> encoding_name. The buffer should be at least 4 bytes long. Returns None if encoding cannot
    be detected. Note that encoding_name might not have an installed decoder (e.g. EBCDIC) """

def getHeader(self, file, REQUEST=None, RESPONSE=None):
    """ Gets the header of a TMX file """

def doTmp(self,file, REQUEST=None,RESPONSE=None):
    """ Makes a copy of the file in the var directory, and creates a second temp file for use later if necessary. """

def getLangs(self, file, REQUEST=None, RESPONSE=None):
    """ Gets source and target language from a two languages TMX file """

def createDoc(self, id, file, REQUEST=None):
    """ Previous step for making a TMX Document """

def makeDoc(self, id, file, REQUEST=None, RESPONSE=None):
    """ Makes a TMX Document """

def convert(self, REQUEST=None, RESPONSE=None):
    """ Converts an UCS-2 doc (Trados, Wordfast...) to an UTF-8 doc """

def delTMX(self, url, rq="", REQUEST=None, RESPONSE=None):
    """ Deletes an enqueued TMX from the REQUEST.SESSION variable. The rq parameter is for the case
    we call this method from the index of a Corpus """

def clearTMX(self, REQUEST):
    """ Clears the TMX list from the REQUEST.SESSION variable """

def getIds(self, REQUEST):
    """ Returns the Ids from the documents enqueued in the REQUEST.SESSION """

def getValue(self, id, REQUEST):
    """ Gets the value of an id from the REQUEST.SESSION TMX list variable """

def parentsURL(self, origin, REQUEST):
    """ Returns the URL list of the containers of the Corpus """

def getCurrentPath(self, origin, REQUEST):
    """Returns the current path """

def parentId(self, url, REQUEST):
    """ Returns the container ID """

def parentURL(self, origin, REQUEST):
    """ Returns the container URL """
```

```

def export(self, st, al, sl, tl, dt, cd, ci, order, REQUEST, RESPONSE=None):
    """ Makes a TMX from the TMX list we have in REQUEST.SESSION """

def exportPo(self, order, REQUEST, RESPONSE=None):
    """ Exports our TMX list to a PO file """

def dateFmt(self):
    """ Returns the current date in a format to be included in the TMX header """

def datePo(self):
    """ Returns the current date in a format to be included in the PO header """

def isEnqueued(self, url, REQUEST=None):
    """ Returns 1 if the document is enqueued to the TMX list. """

def createList(self, REQUEST=None):
    """ Creates the TMX list in the REQUEST.SESSION variable """

def po2tmx(self, id, file, srclang, trglang, REQUEST=None):
    """ Imports a po file. """

def getPoSegments(self, file):
    """ Gets the segments from a po file. """

def search_pages(self, total_item, start_item=1, item_per_page = 10, page_number = 10):
    """ It returns the number of pages of a search. """

def normalizeLang(self, lang):
    """ We normalize the languages as said in the ISO 639 document. """

def text_searched(self, text = "", searched = "", KOP=50, KOP_OSOA=400, JUN='...'):
    """ Allow us to return the substring searched as strong """

```

ZTMX Document

```

def manage_addZTMXDocument( dispatcher, id, title="", creationtool="", creationtoolversion="",
segtype=",o_tmf=", adminlang="", srclang="", trglang="", datatype="", creationdate="",
creationid="",REQUEST=None):
    """ Adds a new ZTMX Document object with id *id*. """

def addSegments(self, file, REQUEST=None, RESPONSE=None):
    """ Adds segments to the current TMX Document """

def addOneSegment(self, srctxt, trgtxt, REQUEST=None, RESPONSE=None):
    """ Adds only one segment to the current TMX Document """

def parseTMX(self, file, src, trg, REQUEST=None, RESPONSE=None):
    """ Parses the TMX file and adds the segments """

def getSegmentList(self, file, src, trg, REQUEST=None, RESPONSE=None):
    """ It returns the list of the segments from the TMX file """

def chgSymb(self, text):
    """ We filter the rare symbols that may appear when we did the conversion from UCS-2 to UTF-8 """

def addTMX(self, id, url, REQUEST, rq=""):
    """ Appends one TMX to the REQUEST.SESSION TMX list. The rq parameter is for the case we call
this method from the index of a Corpus (in that case, the value must be 1) """

def addSegmentsToTMX(self, file, act, REQUEST=None, RESPONSE=None):
    """ Imports the segments from one TMX document to another """

```

```
def addPoToTMX(self, file, act, slang, tlang, REQUEST=None, RESPONSE=None):  
    """ Adds segments to an existing TMX """
```

```
def delTMX(self, url, rq="", REQUEST=None, RESPONSE=None):  
    """ Deletes an enqueued TMX from the REQUEST.SESSION variable. The rq parameter is for the case  
    we call this method from the index of a Document """
```

ZTMX Segment

```
def manage_addZTMXSegment(self, id, title="", idc="", srctext="", trgtext="", REQUEST=None):  
    """ Adds a new ZTMX Segment """
```

Author and license

Copyright 2003 Roberto Quero - CodeSyntax <<http://www.codesyntax.com> - info@codesyntax.com> I am sure that the code could be written better, so if you would like, do it, please!

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your choice) any later version. This program is distributed hoping that it will be useful, but WITHOUT ANY WARRANTY, without even the implied warranty of MERCHANTABILITY or SUITABILITY FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.