Convert from Toolbox into LEXUS

A guide to importing data

Original Author: Jacquelijn Ringersma

Updates for version 2.0: Konrad Rybka

Updates for version 3.0: Kasia Wojtylak

Max Planck Institute for Psycholinguistics, Nijmegen, The Netherlands
May 2012

Convert from Toolbox into LEXUS
A guide to importing data

Original Author: Jacquelijn Ringersma

Updates for version 2.0: Konrad Rybka

Updates for version 3.0: Kasia Wojtylak

Published LEXUS Version 3.0

Table of Contents

1. Introduction 5
    1.1. Defining problematic areas 5
    1.2. The structure file 5
    1.3. The Data File 9
2. Problematic areas for import: Structure file 11
    2.1. Defining markers and their hierarchies 11
    2.2. Useful markers and useless markers 13
    2.3. Double defining markers 13
3. Problematic areas for import: Data file 15
    3.1. Data file inconsistent with the hierarchy of the structure file 15
    3.2. Some practical advice 17

Chapter 1. Introduction

Importing data from Toolbox to LEXUS does not have to be a very complicated matter provided that some requirements have been fulfilled. In the following chapters we discuss those features of Toolbox files and LEXUS structures that might often make the transfer more time-consuming than expected. To take the best advantage of the following manual, some knowledge of Toolbox is necessary, in particular of the idea of a marker and markers' hierarchy.

1.1. Defining problematic areas

When talking about Toolbox and LEXUS, we have to concentrate on two different types of files. The first one is your structure file, the second one is your data file. The former one provides your Toolbox project with a structure: it defines the markers you are using and the relations between them, that is their hierarchy. The latter one will include your lexical information per se, that is the particular lexical entries together with all the information you have about them. The two types of files entail different problems.

1.2. The structure file

Your structure file, also referred to as the *.typ* file here, consists of a list of markers and their definitions. These definitions include the information about the relation between the markers. In fact, this is the most important part of this file for the purpose of importing your lexicon into LEXUS. In Figure 1.1, you can see an example of a typical structure file viewed in a text editor.

```
\+mkr ps
\nam Part of speech
\desc Classifies the part of speech. This must reflect the part of speech of t
\lng English
\rngset -n ??? adj adv conj dem idiom inc interj n ni npfx nsfx num onom persn
\+fnt
\Name Times New Roman
\Size 11
\Italic
\rgbColor 0,0,0
\-fnt
\mkrOverThis lx
\mkrFollowingThis pos
\CharStyle
\-mkr

\+mkr rd
\nam Reduplication form(s)
\desc A repository for reduplication forms for later study (these forms should
\lng vernacular
\mkrOverThis ps
\CharStyle
\-mkr

\+mkr re
\nam Reversal (E)
\desc English word(s)/phrase(s); used to reverse the dictionary for an English
\lng English
\mkrOverThis sn
\mkrFollowingThis lt
\CharStyle
\-mkr

\+mkr rf
\nam Reference
\desc Used to note the reference for the following example sentence.
\lng English
\mkrOverThis sn
\mkrFollowingThis xv
\CharStyle
\-mkr
```

Figure 1.1. An example of a Toolbox .typ structure file

As you can see, the structure file is divided into small sections, each defining a separate marker. The markers are listed alphabetically. The first line of each of these sections identifies the marker, for instance the phrase: \+mkr ps identifies the marker ps, that is part of speech. The next line gives the full name of the marker: \nam Part of speech and yet the next one gives its description: \desc Classifies the part of speech...and the language in which the values appear: \lng English. Most of this information will be copied to LEXUS and therefore should be complete and correct. The \rngset line defines the possible values that this marker can have (this applies only to markers that have such closed sets of values): \rngset -n ??? adj adv conj dem... The few lines that are eclosed by: \+fnt and \-fnt contain the information about the font: type, size, color etc.

The most important part of the marker definition for the purpose of importing data into LEXUS is the line \+mkrOverThis which specifies which marker is above the currently discussed one in the hierarchical structure. In this case, the marker that is directly above ps is lx, that is lexeme. That is how we know that the hierarchical order of these two is the following: lx is directly above ps. Similarly, all other markers are defined in the structure file, so that they all form one hierarchy. This hierarchy can be found not only in the structure file, but also in the Toolbox project itself. If you open your Toolbox project, which runs on your structure file, and go to View and from this menu choose Markers and Marker Hierarchy then in the left part of your workspace all the markers for the particular entry you are looking at, will be listed and their hierarchy will be visualized by the 'stair-like'structure (See Figure 1.2):

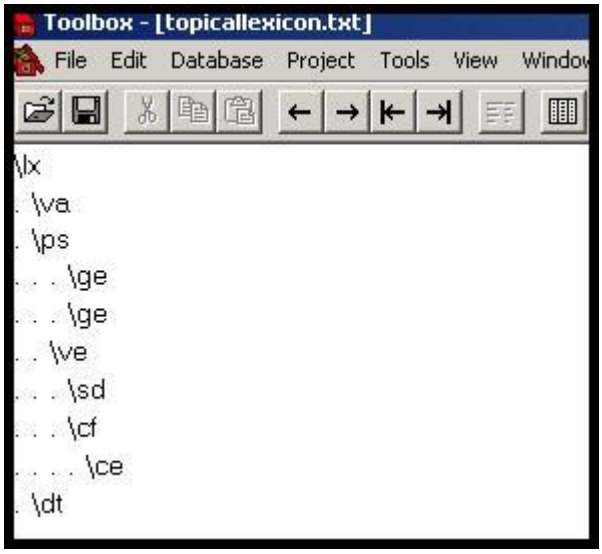


Figure 1.2. Structure file in Toolbox

Here again you can see that ps, part of speech is found under lx, lexeme. We can also notice that it is not only the marker ps, but also va that has been defined under lx. This visualization of the hierarchy however is not completely clear as what we see is not the entire hierarchy that is in the structure file, but only the markers used in the particular entry. Thus, we can see that the marker ge is lower in the hierarchy than ps or va; this is apparent from the number of dots before it, as one dot separates nodes that are on different levels in the structure. But we do not know what is the precise hierarchy. In fact it could be the case that ge is linked under ps or va but not directly (the number of dots shows that there is another level between them) or it could be linked indirectly under something else than ps or va. Similarly, it is not immediately clear which marker is above ve: ps, va or a different marker that did not appear in this particular entry. Toolbox shows here only the markers that have been inserted by you in this particular entry and not the underlying hierarchical structure of the whole lexicon.

Only by looking at Figure 1.2 we cannot decide what the hierarchy really is. One way to check it is to look again into the structure file, find the part that defines the relevant markers and see what is written there in the line: \+mkrOverThis.

Another way to find this information in Toolbox is to right-click on the maker itself in the hierarchy. A window will pop up with all the information about the marker:

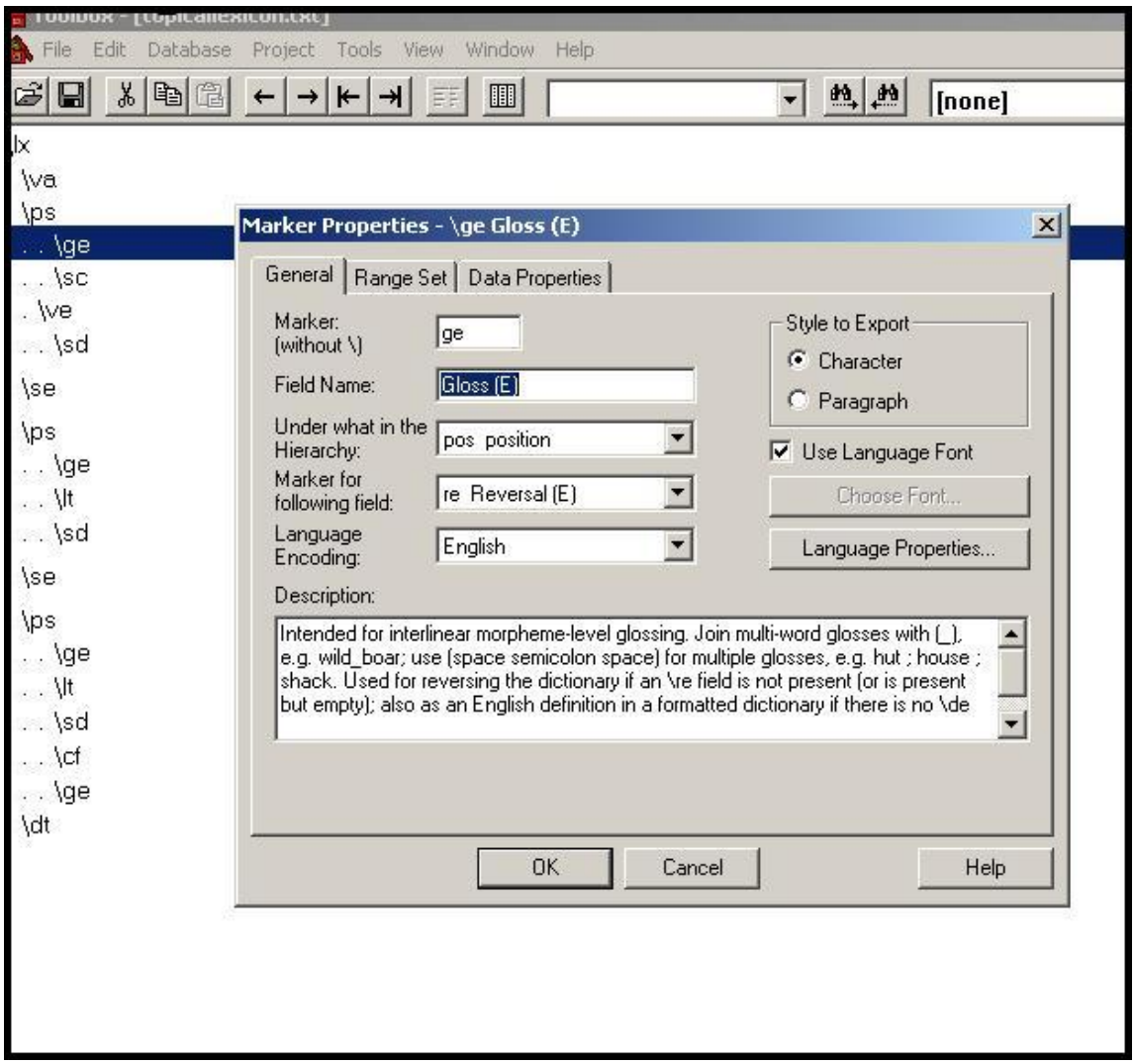


Figure 1.3. Marker information in Toolbox

Here we can see that ge is actually specified under pos, position in the hierarchy. This pop-up window is very important as it contains information about the marker: its field name, its place in the hierarchy, language encoding and definition. All these elements can be modified here. As for changing the place of a marker in the hierarchy, you simply have to click on the roll-out menu next to Under what in the Hierarchy and choose the marker you want from the list:

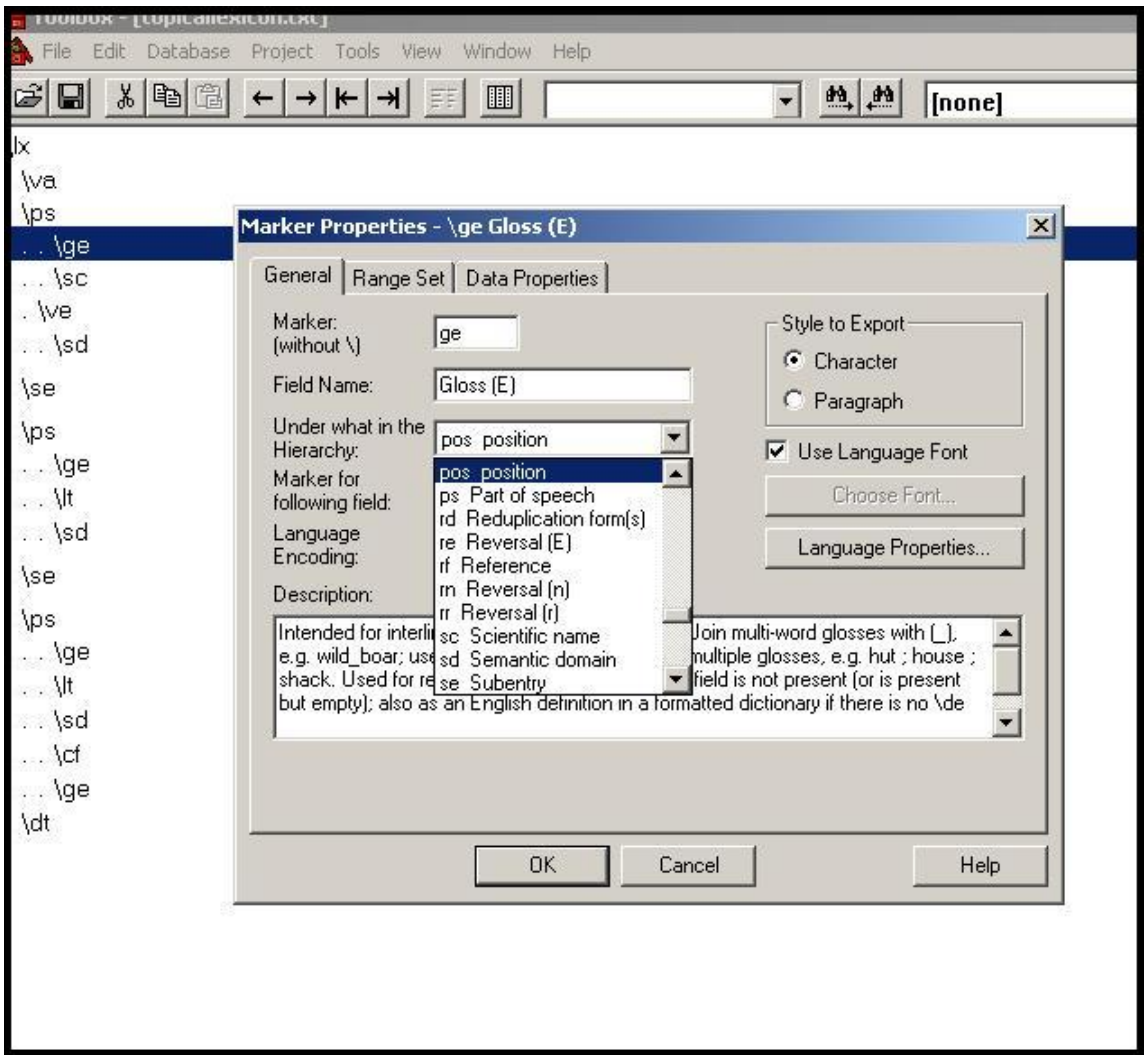


Figure 1.4. Redefining the position of a marker in the structure in Toolbox

This change will be automatically saved in your structure file.

It is also possible to rearrange the marker hierarchy in the structure file itself. You simply have to substitute the marker \mkrOverThis with another one. Compare the two screenshots in Figure 1.5:

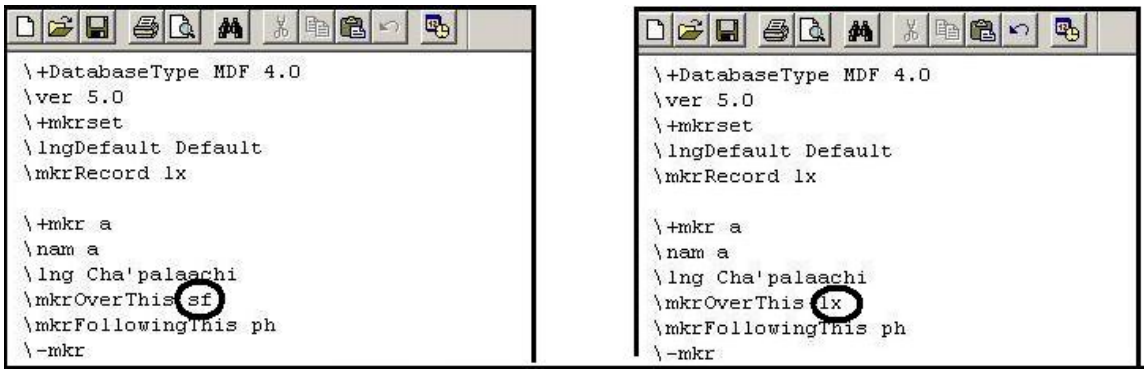


Figure 1.5. Example: Redefining the position of the marker in the .typ structure file. Before: defined under 'sf'. After: defined under 'lx'

It is important to remember to save your changes in the *.typ* structure file every time you do them.

The last important option in Toolbox is the possibility to view a list of all the markers that you are using. To do that, click on Database in the main Toolbox menu and choose Properties from the roll-out menu. A window will pop-up with all the markers, See Figure 1.6:

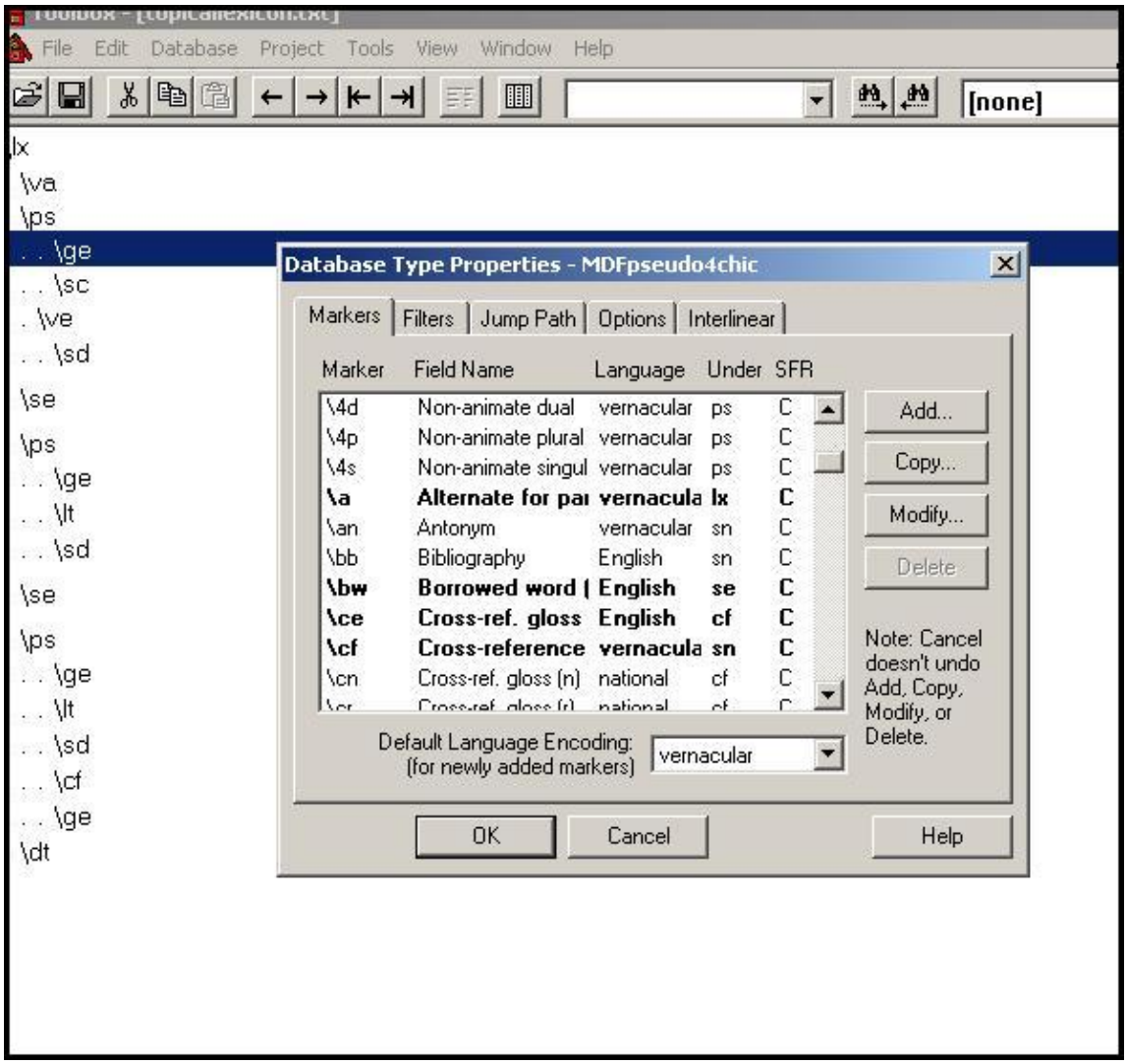


Figure 1.6. List of markers in Toolbox

Here you can see all the markers with the information about their place in the hierarchy. This window also shows which markers are actually in use (that is, which markers have at least one value specified somewhere in the lexicon). It is easy to see which markers do not fulfill this criterion - they are not written in bold font like the rest. These markers can be deleted by clicking on them and choosing the option *Delete* from the right panel.

1.3. The Data File

Your data file usually consists of a number of entries. Each of them in turn consists of a series of markers and their values. A typical entry in a data file will look like the following example Figure 1.7 (remember, however that in your project you might additionally be using different markers as well):

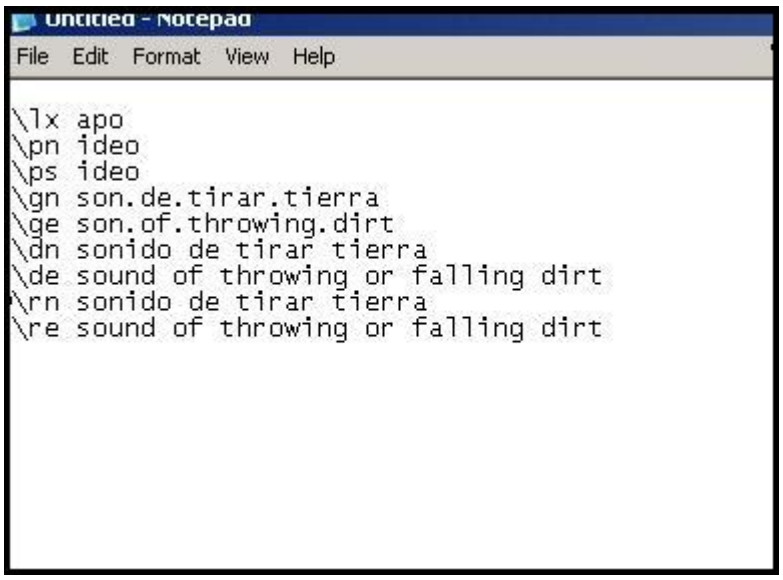


Figure 1.7. An example of an entry in a Data File.

Importantly, this flat text version of your lexicon, does not show the underlying hierarchy, which we have been able to see in the Toolbox project. The data file only records the information about the markers used in a particular entry and their values. LEXUS will make use of this file during the import process. It is, therefore, important that it is consistent with the *.typ* file.

Chapter 2. Problematic areas for import: Structure file

2.1. Defining markers and their hierarchies

As shown in the previous chapter, the structure file is a list of all the markers that you use in your project and the definitions of the relations between them. Before embarking on any serious lexical enterprise, it is worthwhile to think about the markers you will use in your lexicon and the relations between them. For instance, in order to introduce examples into the lexicon, researchers very often use markers such as sfx (sound file example), xv(example vernacular), xe(example English) and xn (example national language). Obviously all these categories belong together. If you have more than one example per entry, you will want it to be clear which sound file goes with which translation etc. In Toolbox you could simply list them one under another, without arranging them hierarchically.

However, when transferring your data to other programs, in this case LEXUS, this information might be lost. Therefore, you need to make the relation between them explicit by ordering them in a hierarchy. As has been demonstrated above, this applies to all markers in your Toolbox project. Toolbox allows you to view the markers together with their hierarchy. For the purpose of importing your data into LEXUS, in some cases, you will need to change it. This is not difficult, however.

For every marker you specify under which other markers should be defined and save it in the *.typ* file. The top marker is usually the lexeme marker (lx) but the position of the rest can be defined according to your needs and the particularities of your lexicon. To come back to our example, you might, for instance, want to define sfx, xe, xn under xv. In that case for each xv that you want to include, you will have a different sets of sfx, xe, xn.

Let us now see what would happen if you imported such a hierarchy of markers into LEXUS. In general, based on your *.typ* file, LEXUS will create a structure for your lexicon that will be used for your data. When LEXUS encounters a structure in the *.typ* file such as given in Figure 2.1, it will take the top node and create a group node out of it. LEXUS always turns markers that have other markers defined under them into group nodes. Then it will put all the markers that were defined under the top nodetogether with the top node into that group.

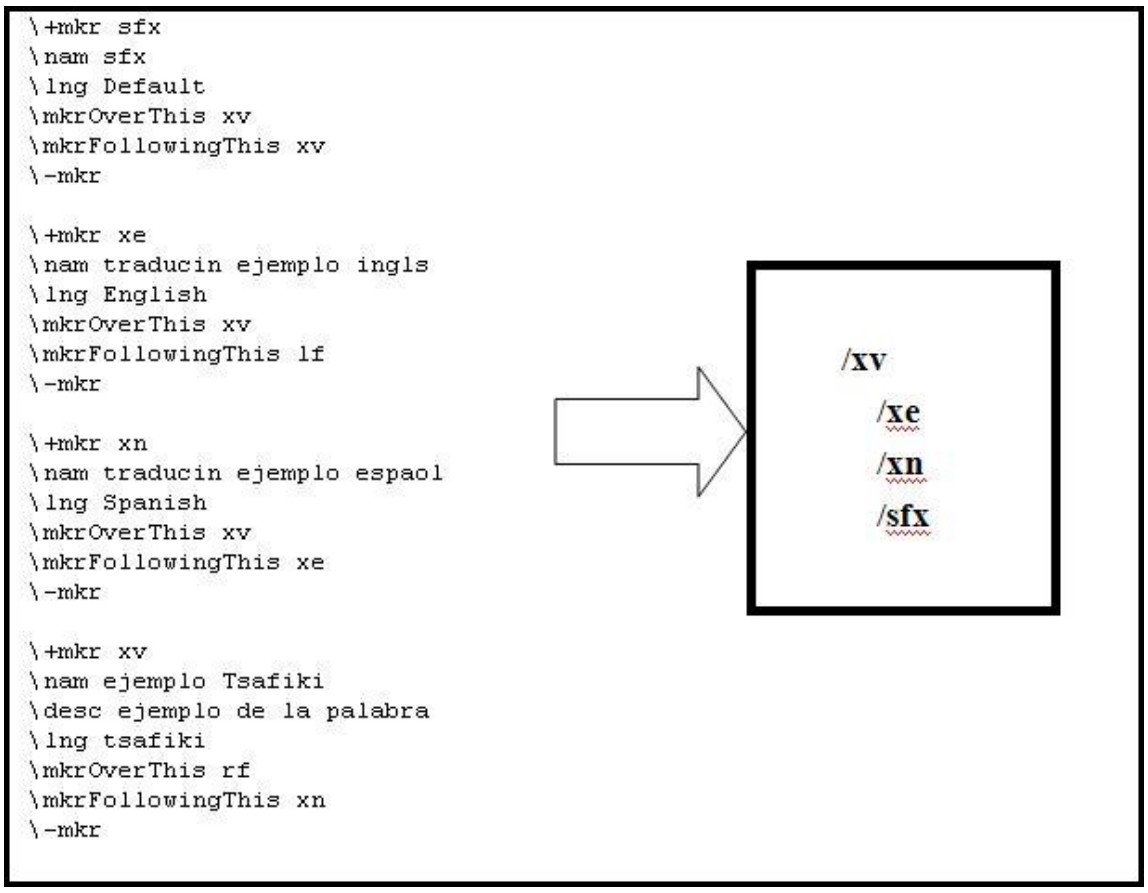


Figure 2.1. Example of a marker hierarchy in Toolbox

The resulting structure in LEXUS will be the following:



Figure 2.2. The structure in LEXUS resulting from the .typ file

Of course the names of the particular nodes and group nodes can be changed later on in LEXUS so that xv group could be called example group and it would include in it all the necessary nodes: the sound file, transcription and the translations. As you can see, in our example, we have already changed the definitions of the markers in Spanish from the *.typ* file (i.e. "ejemplo Tsafiki", "ejemplo Espaol", "ejemplo Ingls") into names in English ("example Tsafiki", "example Spanish", "example English"). As a rule, LEXUS will create a group node out of everything that has at least one other marker defined under it in the *.typ* file.

2.2. Useful markers and useless markers

When using Toolbox for your project, you usually have a number of predefined markers to choose from. The predefined set is called the MDF (Multi-Dictionary Formatter). It is very advisable to use them, so that working with your lexicon is easier for people not that familiar with your project. It is simply a question of uniformity and comparability between different lexica. Naturally, as your project might consists of definitions of markers specific only to your lexicon, it might not always be possible to use the predefined markers for all the elements.

It will more often be the case that you will actually not make use of some markers from the predefined list, rather then add some of your own. In that case, when importing your data into LEXUS, it is advisable to erase those markers that you are not using from the list of markers in Toolbox. Simply, keep in your *.typ* structure file only markers that you actually make use of.

If you decide to include a marker in your project that you define on your own, make sure that it has not only a symbol (e.g. lx), but also a name (e.g. lexeme). It is also highly advisable to give descriptions to your markers, especially if you have introduced some new markers, potentially unknown to others. Giving a name to your markers is not only a practical idea, but it is necessary for the import of the *.typ* file into LEXUS. Although LEXUS will recognize these markers, they will be included in the structure without names. If there are many of such markers, it might be difficult later to find out what the nameless markers are supposed to stand for. If this happens, however, you can always turn to the LEXUS technical group to help you solve this problem.

2.3. Double defining markers

A final step in making sure that the import into LEXUS will cause problems is to be aware of the possibility of double-defining markers. As in Toolbox markers are defined under other markers, it happens often that certain data categories seem to appear consistently in more than one place in the data file. Most often this is the case for part of speech, ps. In the MDF file, it is defined under subentry. However, except for appearing under the subentry marker, it will also be found throughout your lexicon also under lexeme. The structure is actually in each case the same: part of speech appears under subentry which is under lexeme. In those cases, where it is found under lexeme in the data file, the subentry marker (which is in the structure between lexeme and part of speech) is simply not expressed explicitly. For Toolbox, as we already know, this is not a problem.

In this case, however, LEXUS will have to create an empty subentry group between every lexeme and part of speech. Importantly, part of speech in this place is very consistent throughout lexica and it is desirable to have it there. To solve this problem, LEXUS allows to double define markers manually by simply copying the definition of ps in the *.typ* file and changing in the copy of that definition the node under which it is defined from subentry to lexeme. In that way, in the structure file we have two definitions of the marker ps: one under lexeme and another under subentry. This double definition will not be recognized by Toolbox, but LEXUS will be able to cope with it. The only limitation is that there should be no more than three double-defined elements in your *.typ* file, and the general rule is the fewer the better.

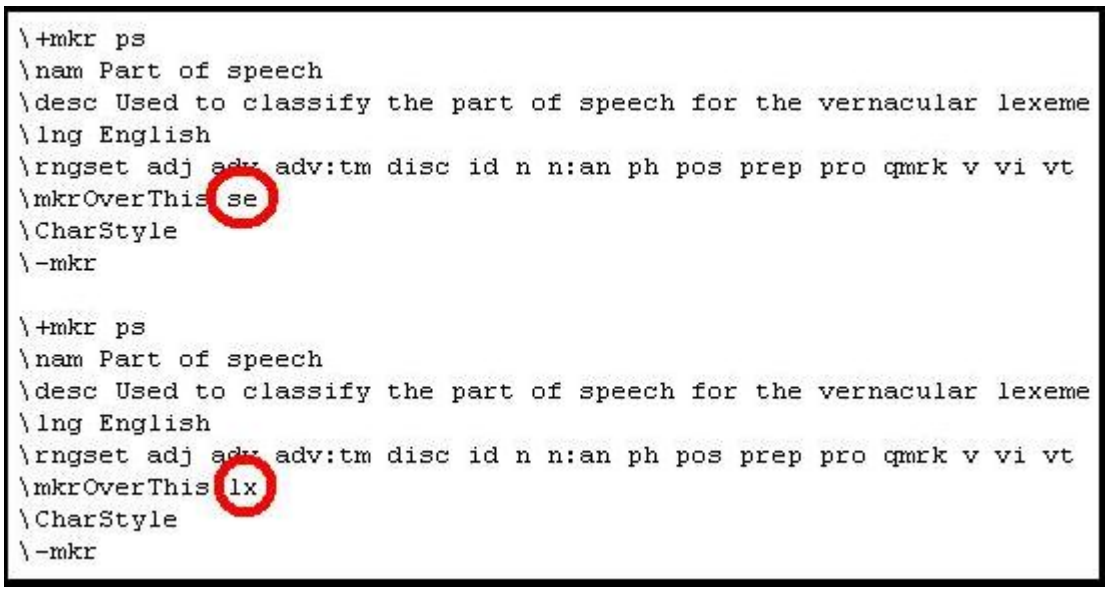


Figure 2.3. Double defined markers in the .typ file.

Often, there are more than three markers that you would like to double-define. The only solution here is to first order them in a hierarchy and to double define only the very top marker of that hierarchy. The markers that are defined under it will automatically appear under the double-defined markers, both under lexeme and subentry for instance.

Chapter 3. Problematic areas for import: Data file

3.1. Data file inconsistent with the hierarchy of the structure file

There are a few types of problems that might arise when importing your data into LEXUS from the *.typ* data file. The size of a typical data file and numbering hundreds of entries makes it very important to think in advance about the potential issues that you might deal with. The solution to all of them is simple: consistency.

In the previous chapters we have stressed the importance of having a clear idea of how you want to organize your markers. Once that has been accomplished, it is crucial to keep that order consistent throughout your data. It is clear that often, as the data is added to the lexicon, such order becomes less systematic. For Toolbox keeping a certain hierarchy is not as important as it is for LEXUS.

When importing the data from Toolbox, problems usually begin when any of your entries contains a string of markers and their values that is against the hierarchy defined in the *.typ* file. Let us come back to the xv example. The black box in Figure 3.1 presents the structure of the *.typ* file with the relevant part of the entry description:

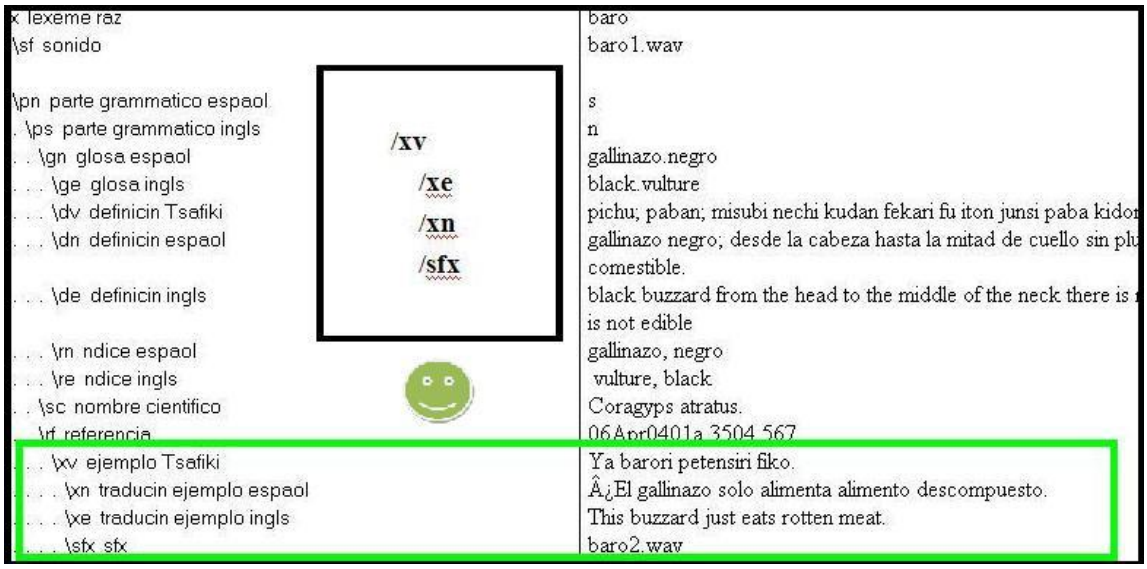


Figure 3.1. Hierarchy of markers and an entry that follows this pattern

The translations xn and xe and the sound file sfx are placed under the appropriate example in Tsafiki - xv. If all the entries follow this order, there will no problems. Notice that the order of xn, xe and sfx does not matter - this is because they are all defined under the xv in the structure, and as long as they follow the xv in the data file, their ordering is of no relevance.

Let us assume, however, that there is an entry in your data file in which the definition markers and their values have a different order. In Figure 3.2 the structure of the *.typ* file is shown in the box together with the relevant part of the entry:

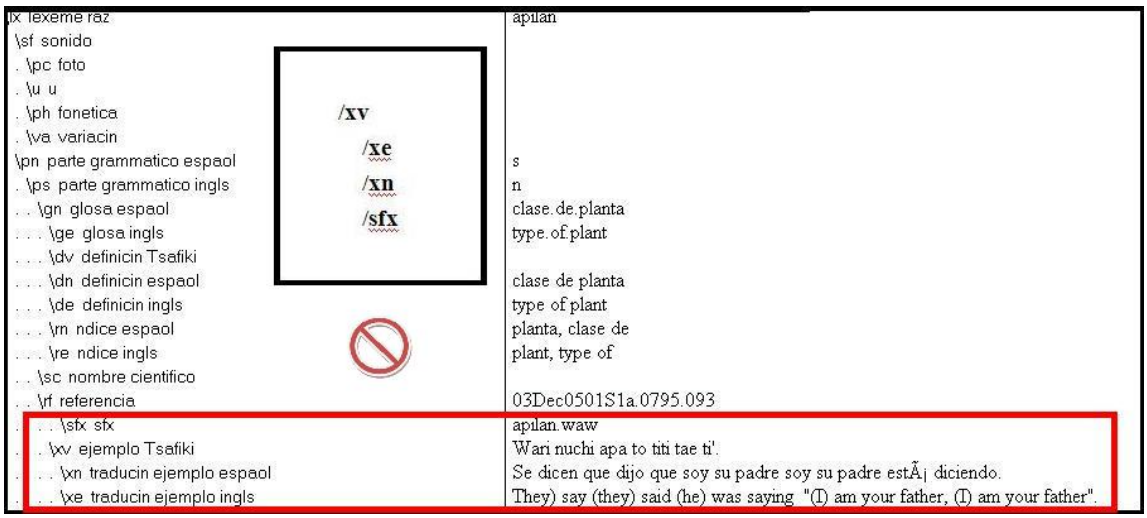


Figure 3.2. Hierarchy of markers and an entry that does not follow the pattern

As LEXUS reads the entries linearly, line after line, and fills the structure that the *.typ* and the data files provide, it will treat such an entry differently. Whenever it encounters a marker that has a certain value, LEXUS checks under which marker this marker was defined in the *.typ* file. Subsequently, it looks back through the part of the entry that has already been created to see whether this higher marker has already appeared in the structure or not. If it has, then the currently analyzed marker will be linked under it.

For the purpose of our example, let us assume that (1) the *.typ* file and the data file follow the same structure, (2) xv in our structure file is linked under rf (reference group), and (3) rf has already appeared in the file and LEXUS has created a node for it. In this situation (see Figure 3.1) LEXUS will behave in the following way.

When encountering xv in the data file, LEXUS will check in the structure file where this definition marker should be linked to – in this case it will be under rf. As rf already exists in the structure of this entry, xv will be linked under rf. Remember, however, that xv has also other nodes linked under it in the structure file. Therefore, first a group node will be created out of it (xv group). It will be linked under rf and xv will be linked under that group node. The next marker that LEXUS will encounter will be xe. Here again LEXUS will check if the marker (that in the hierarchy is above it - xv group) already exists in the structure. Since it does as LEXUS has just created it, the xe marker will be linked under xv. This operation will be repeated until all the relevant markers are linked under the xv group definition marker. As a result, the following structure will be created for that entry in LEXUS. In this example, this is how we want our lexicon to look like:

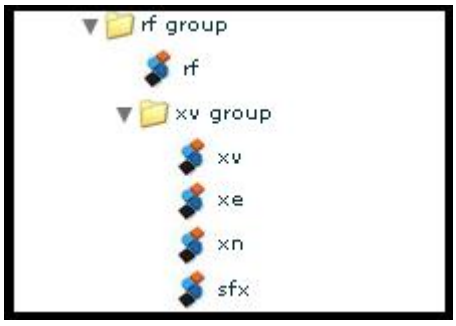


Figure 3.3. LEXUS structure for the entry that follows the structure of the .typ file

However, when the order of the markers in the *.typ* file is as presented in Figure 3.2 above, the outcome will be different. LEXUS will first encounter sfx , not xv. Then, it will check under which node sfx is defined in the structure file. As we already know, it is defined under xv group. As the xv group definition marker has not appeared in this entry yet (remember that xv is placed after sfx), LEXUS will create xv group and link sfx under it. Then, LEXUS will encounter xv and create another xv group with xv, xn and xe linked under it. Eventually, the following structure will be the outcome:

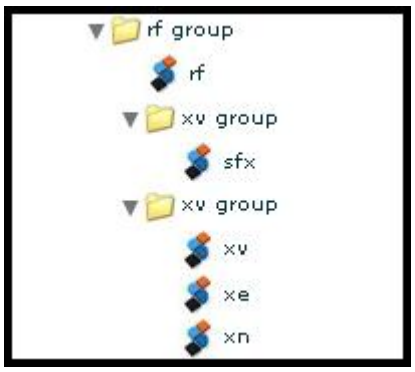


Figure 3.4. LEXUS structure for an entry that does not follow structure of the .typ file

This is problematic, because the information about the sound file (sfx) that goes with its translations (xv, xe, xn) is now lost – it is distributed between two different xv groups: one missing a sound file, the other missing the translation information. Such situations will always happen if a marker that is higher in the hierarchy of the structure file, appears also placed on a lower level.

That is why, when you want to import your data into LEXUS, you have to make sure that such situation does not occur in your data file. As a practical guideline, we suggest, therefore, **to follow the hierarchy of markers from your .typ file in your data file**. This means manually placing a marker right under the marker under which it was defined, and never above it.

3.2. Some practical advice

When importing data from Toolbox into Lexus we have not forget about the following.

(1) Remember that not all markers are used in every entry and it would be a waste of time to list all the unnecessary markers just for the sake of keeping to the hierarchy. LEXUS reads only the markers that are used in a particular entry, and makes a structure out of them according to the *.typ* file.

(2) LEXUS does not see, it does not include. That is to say, it is not enough for a marker to appear in an entry for LEXUS to read it. It also has to have a value to be recognised by LEXUS. Whenever you have a marker that has no value this marker will be simply omitted. In some cases, however, this can work to your advantage, becaause LEXUS will always create a coherent minimal structure out of the lexical entry according to the given *.typ* pattern. Therefore, an empty marker can never lead to the afore-mentioned complications. Do not forget, however, that too many nameless markers can lead to problems addressed in section 2.2 above.

In case you encounter problems, you can contact the LEXUS technical group at: lexus@mpi.nl.

The full online LEXUS manual is available at: http://tla.mpi.nl/tools/tla-tools/lexus