The Diff-concept of a Digital Music Edition

The paper introduces the workflow of comparing two digital editions (based on different sources of the same musical work) and displaying the results in a user interface. This workflow is being developed in the context of the project N. A., and is the critical edition of the musical works by N. A.. Both editions — ‘ref’ and ‘edt’ — are encoded in MEI . Willing to show the users the differences between the ‘ref’ and the ‘edt’, our team faced some challenges on technical and philological aspects of this process, which will be described in this paper.

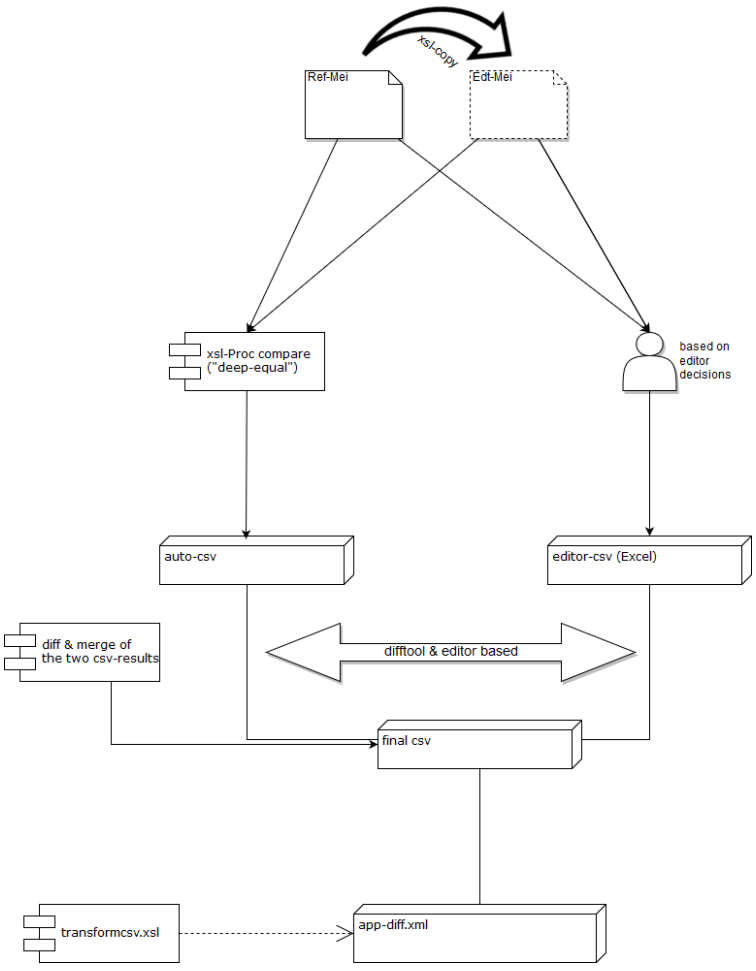
Encoding

The encoding should have some restrictions. The both editions have to be encoded as independent files and don’t contain any reference information between thems like pointers, etc. The reasons for this constraint are the project premise *encoding is the edition* and that the ‘ref’ as a digital re-mastering of a printed edition is considered to be as ‘sealed’ edition. However, a solution which respects the constraints but allows to make the files corresponding, was achieved by the following approach. After the encoding of the ‘ref' was finished, a base file for the ‘edt’ was created by pre-processing ‘ref' by an XSLT-stylesheet (prepare_edition.xsl). After the transformation, all ‘ref’-specific information was deleted, but the 'element ID’s base' was preserved, only a suffix with edition type was added, for instance: @xml:id=”slur_123” in ‘ref’ corresponds to @xml:id=”slur_123-B1” in ‘edt’; this state of the files allows automatic comparison and the suffix helps the editors to distinguish between the editions while working on both files.

Locating differences

The workflow of the comparison and displaying the differences can be splitted into the following steps:

- while editing, the editors entry the element IDs which differ or are absent/present in the editions into a list (diff-list.csv)
- a diff_auto-edt_merge.xsl creates another list (notInEditors.csv) which contains proposals for the differences which the editor might forgot to entry
- these proposals are processed by the editor manually and are either copied into the ‘main section’ or into the ‘ignore section’ of the diff-list
- create_app-diff.xsl creates an app-diff.xml from the ‘main section’ of the diff-list.csv and enhances it with some additional information needed for rendering
- based on the information provided in the app-diff.xml, respective measures are displayed in the user interface and the differences are highlighted



Graphic 1 Workflow of creating the app-diff.xml

Automatic comparison and its limitations

As mentioned above, a computer program (diff_auto-edt_merge.xsl) for the automatic comparison was developed. The concept of this comparison is based on the corresponding IDs; it expects therefore that all elements in one edition might have or not have their respective elements in the another edition:

- if they do: defined attributes of these elements will be compared, e. g. @startid and @endid of a <slur/>
- if they do not: an element is present in one edition, but not in the another

Graphic 2 illustrates this approach: The slur in the 'ref' is longer than in 'edt' (i. e. the value of the @startid was changed) and they are considered to be different. The entry in the diff-list will look like this: "slur_123, slur_123-B1"

Graphic 2: slurs with different length

However, facing philological needs on a higher semantical level, this concept has some limitations. For instance, the difference between the slurs on the graphic 3 could be considered in several ways:

1) the first short slur in 'ref' (picture to the left) is a reference slur to the slur in the 'edt' (picture to the right) and it is shorter, the second short slur in the 'ref' is not present in the 'edt'
2) the second short slur in the 'ref' is a reference slur to the slur in the 'edt and it is shorter, the first short slur in the 'ref' is not present in the 'edt'
3) both slurs in the 'ref' were merged into a longer slur or vice versa, the slur in the 'edt' was splitted into two shorter slurs

Graphic 3: merged/splitted slurs

Letting a computer program to decide which variant should be chosen might lead to over-complexity and not to be quite satisfactorily. So the editors should perform these decisions. However, this has an impact on the encoding and entries to the diff-list: the slur in 'edt' should get a new ID (i. e. not to be a corresponding slur) and the entry in the diff-list should be as following: "slur_123 slur_456,slur_789-B1" (Note that first two IDs are divided by white space only, which groups them). Additionally, in order not letting the program to treat all three slurs like those which are not present in another edition, there should be three entries to the ignore-part of the diff-list: "slur_123, 0", "slur_456, 0", "0, slur_789-B1" which will exclude these cases from the results of the automatic comparison.

Conclusion.

The diff-concept combines editorial decisions with the support of computer programs.
It is work-in-progress and is constantly being updated in order to make this process more efficient as well as philologically more precise.