Encoding Genetic Processes II

Johannes Kepper
Beethovens Werkstatt, Paderborn, Germany
kepper@edirom.de

Susanne Cox
Beethovens Werkstatt, Bonn, Germany
cox@beethovens-werkstatt.de

Abstract

Traditional music philology aims at establishing an edited text, which is supposed to stage a clearly identified and well-reasoned version of a musical work. Such a text will always depend on sources used for its preparation and decisions taken by the editor(s). However, the intention is to deliver a *product* – a static text, which resembles a specific combination of the transmitted sources of the work in question. In Genetic Editing, the focus lies elsewhere: Instead of justifying a specific *product* version, the intention is to trace the creative *processes* involved in the composition of that work. Obviously, those *processes* are only accessible through transmitted documents as well, but those documents do not need to contain full texts, nor are they only relevant when the composition has already matured enough to more or less reflect the final work.

The *Beethovens Werkstatt* project is one of the first endeavors to explore the applicability of Genetic Editing to music. Several years ago, a presentation at MEC 2015 in Florence introduced the first findings of the project and illustrated the then novel approaches of encoding genetic processes in MEI [2]. The discussions of the conceptual model proposed there eventually led to the introduction of several new elements into MEI. Since then, not only MEI has evolved, but also the project. The paper at hand reflects on data model considerations for the project's current module.

Recapitulation

One of the most important aspects of the original data model of *Beethovens Werkstatt* was a clear distinction between the encoding of a work's text and of the documents it is transmitted in. Conceptually, this reflects core aspects of the FRBR data model.[1] This was implemented using a single MEI file that encoded the work's musical *text*. All process-related information was covered there by using MEI's `<add>`, `<del>`, and `<genState>` elements.[2] Everything related to the *scripture* was encoded differently: Each penstroke on paper was digitally traced and stored as an SVG `<path>` element.[3] That element bears no semantic meaning beyond that it's describing an arbitrary shape, and so the visual appearance of the document was encoded as interpretation-free as possible. The connection to the musical text was then established using the `@facs` attribute on MEI `<note>` elements, etc., which referenced the corresponding shapes. That way, every note (as any other component of the music notation) explicitly stated by which shape(s) it was manifested in any given source. A user of the resulting edition could now easily access our findings and challenge the material with different interpretations (see Figure 1).

This general approach worked out pretty well for the first module of *Beethovens Werkstatt*, dealing with compositional revisions in manuscript material. However, the work on Beethoven's 8th Symphony[4] already surfaced some limitations which are now relevant again for the current work.[5] Nevertheless, due to the clear separation of document and text, and the good experience utilizing SVG, the model was considered a good starting point for the work in the current third module of *Beethovens Werkstatt.*

1 https://www.ifla.org/publications/functional-requirements-for-bibliographic-records (accessed January 12, 2022).

2 See the current documentation of these elements at https://music-encoding.org/guidelines/v4/elements.html (accessed January 12, 2022).

3 https://www.w3.org/Graphics/SVG/ (accessed January 12, 2022).

4 See https://beethovens-werkstatt.de/modul-1/ (accessed January 12, 2022).

5 The main difference between the 8th Symphony (Op. 93) and the other examples addressed in this module (excerpts from Op. 111, Op. 59/3, Op. 75/2, and WoO 32) was that here the project had to trace Beethoven's revisions across multiple documents. This brought up deficiencies with regard to establishing the order of writing acts when those acts included simple copying. Then, no MEI 'processual' element like `<add>` or `<del>` could be inserted into the encoding that could identify the genetic state which this copying act would belong to: Mere copying triggered no changes to the work's text. Eventually, this led to the omission of a transcription for this example from Op. 93.

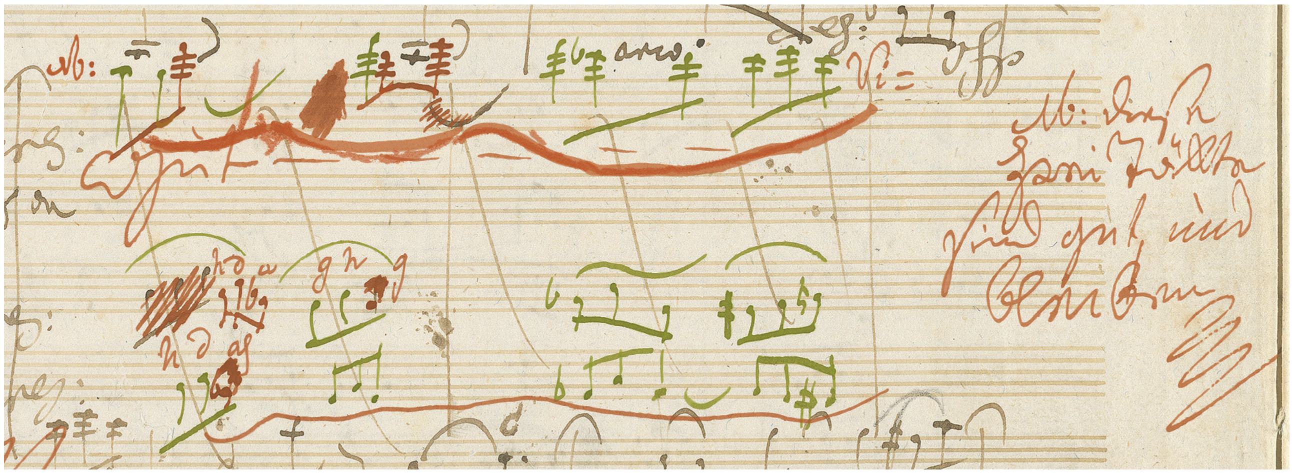

Figure 1: Highlighted musical symbols and text in a digital copy of Ludwig van Beethoven: String Quartet, Op. 59/3, Autograph, p. 18 (D-BNba, BH 62).

Beethoven's Revision Lists

The focus of our current module is on corrections and revisions that Beethoven generally could not carry out himself directly, but instead asked publishers to incorporate in their prints. Here, Beethoven not only corrected engraving mistakes but also produced substantially new variants. He did so by writing down instructions in *revision documents*. Those revision documents can take a number of different forms (see also Figure 2):

- Tabular revision lists consist of listings of the changes requested by Beethoven.[6]
- Quite frequently, Beethoven transmitted the changes he desired in letters to the publishers.[7]
- Finally, there are revision instructions that he entered directly into a music document, such as an engraver's copy, a proof, or a regular edition.[8]

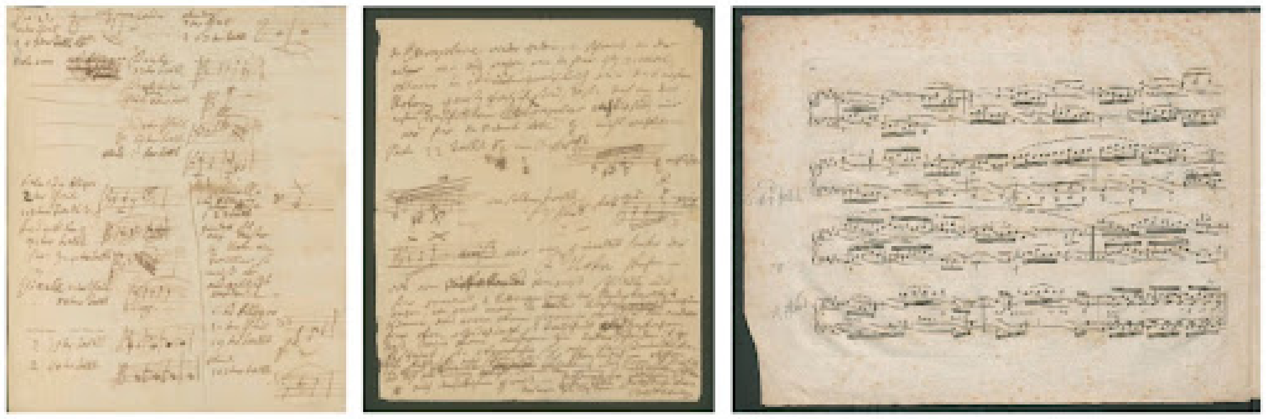

Figure 2: Different forms of revision documents. Left: tabular revision list (D-BNba, Sammlung Wegeler, W 13, p. 2); center: revisions in a letter (D-BNba, Sammlung H. C. Bodmer, HCB Br 214, p. 2); right: original edition of the Piano Sonata, Op. 109, with Beethoven's revisions (D-B, Mus.ms.autogr. Beethoven, L. v. 39,1, p. 16).

Beethoven created revision documents at various stages of the compositional process – some were a response to sample prints he had access to, some were written several months after the publication, and some were written before the work was even printed. The publishers reacted in different ways to Beethoven's instruc-

6 See for example Beethoven's list with revisions for the String Quintet in C minor, Op. 104 (D-BNba, Sammlung Wegeler, W 13).

7 See Beethoven's letter of 3 June 1823 to Maurice Schlesinger, the publisher of his C minor Piano Sonata, Op. 111, in which he reported printing errors in the Paris original edition (D-BNba, Sammlung H. C. Bodmer, HCB Br 214).

8 For the Piano Sonata, Op. 109, a copy of the original edition has survived, in which Beethoven entered several revisions (D-B, Mus. ms.autogr. Beethoven, L. v. 39,1).

tions. Ideally, everything has been changed as requested.[9] Sometimes, however, the instructions were considered only partially, as some changes would have been too complicated to justify the effort.[10] For the same reason, the requested changes were sometimes published as errata lists instead of incorporating them into the prints.[11] In all cases, the revision documents provide interesting insights into Beethoven's compositional processes, as they served as proxies to describe changes that couldn't be carried out directly.

Piano Concerto, Op. 73

A relatively simple case is the revision document that Beethoven created for the Piano Concerto in E-flat major, Op. 73. Breitkopf & Härtel published the German original edition in February 1811. Beethoven proofread the edition and created (among other things) a revision list for the piano voice with 25 requested changes. Most of them are corrections of errors, but some also clarify dynamics. In May 1811, a revised edition was published, in which all issues brought up by Beethoven had been incorporated into the original plates. Copies of both the original[12] and revised[13] editions have survived.

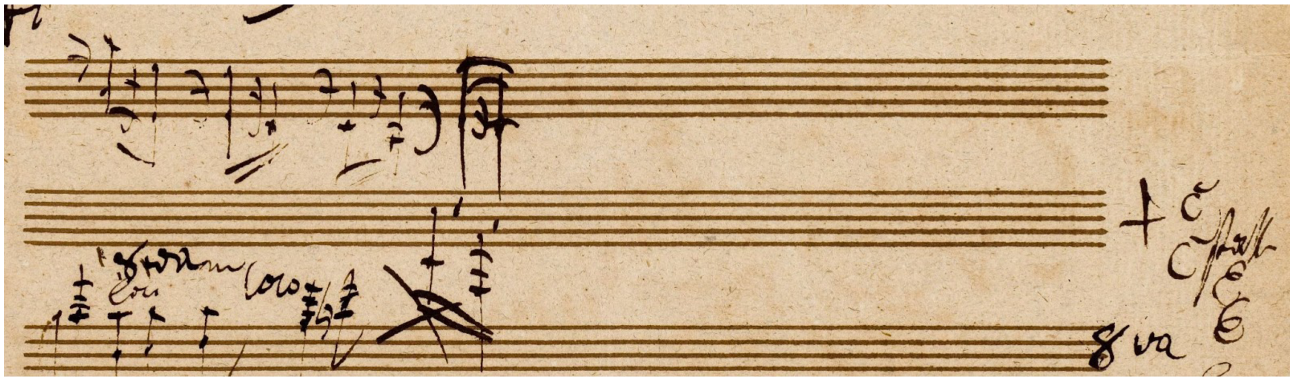

Figure 3: Fifth issue on the revision list for the Piano Concerto, Op. 73 (US-NYj, 31 B393cp no. 5 errata, p. 1).

The fifth issue on Beethoven's revision list for Op. 73 (Figure 3) illustrates the typical components of Beethoven's *monita*.[14] First, Beethoven often identifies the region where the change is to be made with a marginal note, e.g., "2tes großes tutti" [second great tutti]. The very segment to be changed is then given only by the music excerpt, which provides the context for the actual correction. On the right margin, Beethoven explains this correction ("Änderungsimperativ", imperative of change). The "C/C statt E/E" written there indicates that a chord has incorrect pitches. Within the music segment given on the left, Beethoven adds an "X" at the fourth beat to clearly state which chord is wrong. Only when considering all these components does it become possible to understand Beethoven's instruction. The results can be seen when comparing the second-to-last chord in the left hand of the piano in the original and revised edition (Figure 4):

9 One example is the Piano Concerto in E-flat major, Op. 73, for which a revised edition was published (see below).

10 Beethoven's Irish folksong settings, WoO 152, were published in March 1814 by George Thomson in Edinburgh. In September 1814, Beethoven sent a revision list to Thomson (D-B, Mus.ms.autogr. Beethoven, L. v. 29 V, fol. 173–175). In a revised edition published in late 1814 or early 1815, only a part of these revisions were implemented.

11 Beethoven's revision list for the String Quartet in E-flat major, Op. 127, was published in the journal *Cäcilia* (Intelligenzblatt no. 24, April 1827) by Gottfried Weber. The revisions were not incorporated in the printed editions.

12 D-BNba, C 73/9.

13 A-Wn, SH.Beethoven.323.

14 https://beethovens-werkstatt.de/glossary/monitum/ (accessed January 12, 2022).

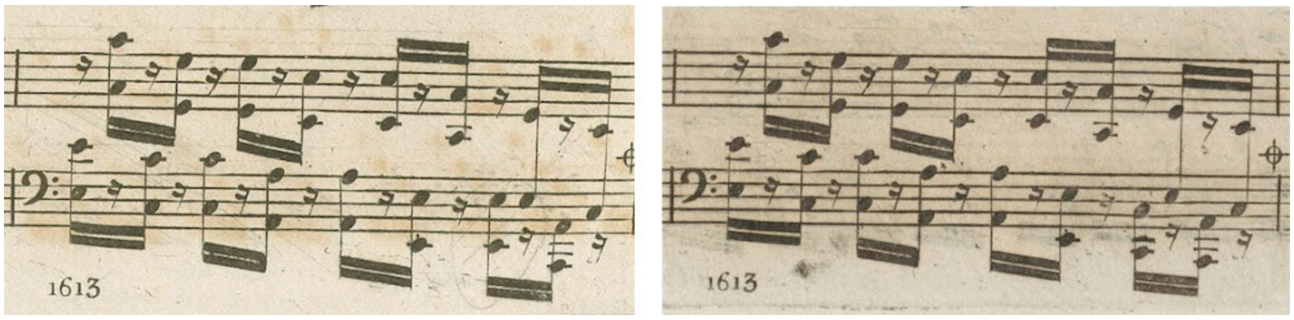

Figure 4: Op. 73, 1st movement, piano part, m. 369: edition *ante revisionem* (left) and *post revisionem* (right) in comparison (D-BNba, C 73/9, p. 15, and A-Wn, SH.Beethoven.323, p. 15).

Encoding Revision Lists

The case of Op. 73 shows relevant similarities to the work done in the first module of the project. In order to trace all metatexts[15] such as the marginal notes and clarifications, it was clear that, at least for the (usually manuscript) revision documents, their scripture needed to be encoded using SVG. An encoding of the affected shapes in all relevant documents (including prints) would better illustrate the context and results of a requested change, and make it easier for users to follow and challenge the edition. Accordingly, the project decided to use SVGs as in the first module.

However, there are also some significant differences to the first module. One of the most notable ones is the number of revision layers for the segments[16] of the text considered: Whereas in the first module, where we sometimes faced more than a dozen revisions as stacked writing layers,[17] the revision lists usually describe a single change, with occasional exceptions of two such revisions for the same position (see below). As all changes requested in any single revision list conceptually come into effect simultaneously, establishing the genetic order of those changes is easy.

Another important difference is the mere presence of multiple documents. This leads to interesting shifts in the relation between *document* and *text*. When writing down the context for a change into the revision list, it was sufficient for Beethoven to clearly identify the segment, without having to copy all the musical content of the section. This, and the occasional mistakes he made when he actually did copy content, sometimes resulted in ambiguity, or at least differences between what those documents read and what the text should actually be like. Likewise, not all changes Beethoven requested were implemented properly by the publishers, leading to a difference between the intended text ("Zieltext") and the achieved text ("erzielter Text"). While the first module had a very clear 1 to 1 relation between *document* and *text*, where the differences between both were sufficiently covered by the use of SVG, we now see a more complex relationship that necessitates an equally elaborate encoding model.

While it seemed still appropriate to cover the scripture of documents by using SVG shapes, it became clear that this isn't sufficient to distinguish document and text properly in situations with more than one witness involved. Our first reaction to this insight was to explore the option to use individual MEI files for each document, plus another separate file for the work text. That way, we could more easily transcribe different texts for each document – initial print *ante revisionem*, revision document, and print *post revisionem*. In addition, the revision instructions that Beethoven included into letters as verbal instructions called for TEI-based markup, which required some kind of structural separation anyway.

15 https://beethovens-werkstatt.de/glossary/metatext/ (accessed January 12, 2022).

16 While the proposed encoding model is designed to allow full encodings of the works in question, *Beethovens Werkstatt* actually only provides the music encoding for those segments that are subject to Beethoven's revision instructions, while everything else is addressed on the level of measure positions in the facsimiles only. This already gives a 'skeleton encoding' of measure elements which allows to encode all clefs, meters, and key changes in the places where they actually occur in the document, instead of compiling the then current parameters into a fictive score definition at the beginning of each revision instruction. That way, the encodings are selective more or less by chance, but enriching them with additional content later will not require any changes to the material encoded now – conceptually, the encodings are complete already.

17 In the first module, those passages were referred to as "Textnarben" (textual scars). See https://beethovens-werkstatt.de/glossary/textnarbe/ (accessed January 12, 2022).

However, using multiple files for an edition comes at the price of having to coordinate them through dedicated markup of some kind. One option considered by the project was to implement something very similar to *Freischütz Digital*'s core model.[18] There, a central MEI file (the 'core') holds the encoding of the musical text, but no information about its visual appearance in any of the sources. These sources are captured by individual encodings of their own, which spare the main parameters of the text and link to the corresponding elements in the core file instead. For example, in that model, a note's pitch is to be encoded in the core, but the stem direction is addressed at the source level. This results in a plethora of links between these files, which are both hard to generate and maintain. However, there are very few alternatives to reduce those efforts. One such approach could be to follow the *Enhancing Music Notation Addressability* (*EMA*) concept.[19] EMA is designed to reference music snippets independent of the data format they are encoded with, by specifying one or more ranges of measures, staves within those measures, and finally beats. The resulting EMA statement is expressed as a URI, and as such can be easily used in conjunction with MEI. With this concept, it would be possible to create links not on the level of individual notes, but for full text segments that are subject to a revision instruction. As current EMA lacks precision to selectively address only some of the content in a linked region, like dynamic markings only, it seems reasonable to mimic the concept with MEI markup that allows greater precision.

```
<!-- antePrint.xml file -->
<measure n="21">
  …
  <annot xml:id="anteSection" tstamp="1" tstamp2="2m+5" staff="19 20"/>
</measure>

<!-- postPrint.xml file -->
<measure n="21">
  …
  <annot xml:id="postSection" tstamp="1" tstamp2="2m+5" staff="19 20">
    <relation rel="isRevisionOf" target="antePrint.xml#anteSection"/>
  </annot>
</measure>
```

Listing 1: Schematic encoding of two file regions identified and linked by MEI <annot> elements.

Listing 1 shows how MEI `<annot>` elements are used to identify specific regions within two files through timestamps,[20] which are then linked using a `<relation>` element and the `isRevisionOf` relation taken from FRBR. An additional `@plist` (participant list) attribute on `<annot>` could be used to specifically identify some content in those regions. This model requires much less effort to connect the different files involved, and therefore seemed like a reasonable approach to follow.

Diabelli Variations, Op. 120

In order to challenge this provisional data model and to verify its applicability, the project had a closer look at other, significantly more complex examples. One such example is a revision document for Beethoven's *Diabelli Variations*, Op. 120, which is included in the "Engelmann sketchbook"[21] that Beethoven used in early 1823. This document significantly differs from the list transmitted for Op. 73 described above. It was not written after but while the work was printed. It holds revisions for a revised copy and the autograph – so it is targeting manuscripts, not a print. Both of those manuscripts were temporarily out of Beethoven's reach at different times,[22]

18 See [1] and https://freischuetz-digital.de/en/datamodel.html (accessed January 12, 2022).

19 https://github.com/music-addressability/ema/blob/master/docs/api.md (accessed January 12, 2022).

20 See https://music-encoding.org/guidelines/v4/content/introduction.html#timestamps (accessed January 12, 2022).

21 D-BNba, Sammlung H. C. Bodmer, HCB Mh 60, p. 16–18.

22 The autograph served for some time as an engraver's manuscript for the publisher Anton Diabelli, and the revised copy was to be the engraver's copy for a planned London edition. Therefore, the revised copy was handed in for transmission to London in the first days of May 1823, but was not actually sent until early July 1823.

so he needed to coordinate the ongoing revision of the work between those documents: In contrast to the Piano Concerto, the *monita* here do not only concern corrections of errors, but also introduce new compositional variants. Implementing the changes later was much easier: Beethoven could do it himself, e.g., by overwriting. At the same time, this type of revision makes the original text much less accessible than is the case for Op. 73, where it is available as a separate printed document.

Another important aspect is that the revision list was written into the “Engelmann sketchbook” – Beethoven’s private working document of the time that he maintained for his own use. Accordingly, the revision instructions are just reminders for himself, and they could be written down in a very rudimentary form as long as they were functional as an *aide-mémoire*. The localisation is less precise, less context is given, less explanations are provided of what exactly is to be changed, and Beethoven’s handwriting is more sketchy.

A major complication is that at some point Beethoven revised the revision list itself, as can be seen from the red ink he used for it. This means that the source and target documents flip their functions – for some revisions, Beethoven changed his mind about the ‘correct’ solution several times. As a result, the identification as source, target, or revision document is not possible at the document level, but needs to be determined for each individual revision instruction. It also means that any single document may hold more than a single state of the work’s text, which results in the need for *process* markup like `<add>` and `<del>` to be used as well in the MEI files for each document. This requires coordinating such process markup across multiple files.

A good example for such complications can be found in measure 16 of the 26th variation of Op. 120 (Figure 5). Here, the source document is the revised copy, and the target document, in which the change is to be inserted, is Beethoven’s autograph.

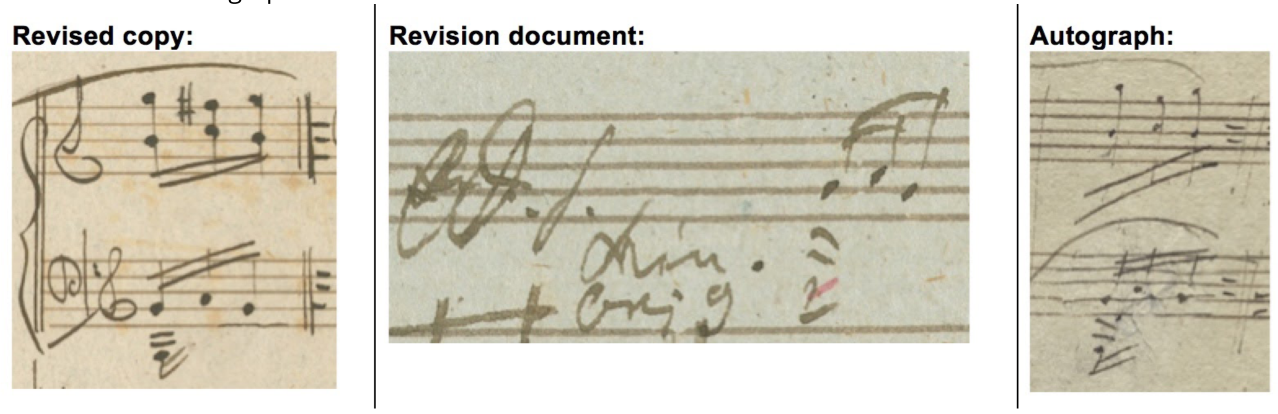
Revised copy:
Revision document:
Autograph:

Figure 5: *Diabelli Variations*, Variation 26, m. 16, in the revised copy, the revision document, and the autograph (D-BNba, Sammlung H. C. Bodmer, HCB Mh 55, fol. 26v; D-BNba, Sammlung H. C. Bodmer, HCB Mh 60, p. 17; D-BNba, NE 294, p. 52).

The revised copy originally read a dotted eighth note *g* in the lower system. This initial reading is no longer present in the manuscript shown in Figure 5 (left), as Beethoven implemented the change in place by erasure. Beethoven’s first change was to remove the augmentation dot. Accordingly, he stated “kein . bej g” [no dot at g] in the revision document (see Figure 6, right).

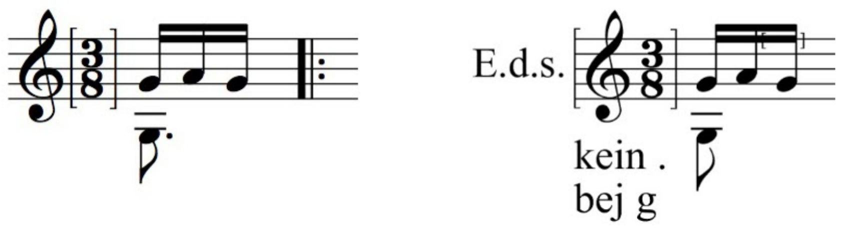

Figure 6: Left: Initial reading of the revised copy of *Diabelli Variations,* Variation 26, m. 16, left hand. Right: Transcription of the revision document in its first version.

Beethoven planned to make this revision in the autograph, to which he had no access at the time of writing because it served as the engraver's manuscript for the edition published by Anton Diabelli. Apparently, Beethoven decided differently when he got the autograph back. He did not only delete the augmentation dot, but also changed the duration of the eighth note to a sixteenth. The second revision was done in the reverse direction: This time, the autograph was the source document, and Beethoven had no access to the revised copy, so again, he kept a note about the intended change in his sketchbook (Figure 7).

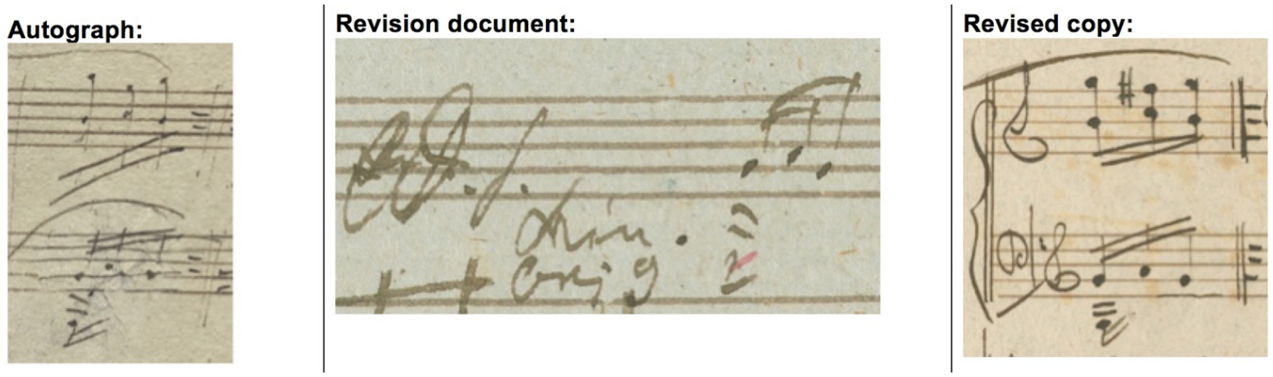

Figure 7: *Diabelli Variations*, Variation 26, m. 16, in the autograph, the revision document, and the revised copy (D-BNba, NE 294, p. 52; D-BNba, Sammlung H. C. Bodmer, HCB Mh 60, p. 17; D-BNba, Sammlung H. C. Bodmer, HCB Mh 55, fol. 26v).

Beethoven used red ink to update the revision instruction by adding a second flag to the note in question (Figure 7, center). Later, he copied that flag to the revised copy, so that all documents ultimately read the same final version of the text.

Coming back to the encoding, the challenge of having to include the textual development not only in the encoding of the text but also in the document files becomes very apparent in this example: All four files provided in the original data model would have about the same content. The text file would embrace encodings of the original version (A), the first revision (B), and the second revision (C). The autograph would hold at least versions A and C, the revised copy would provide versions A, B, and C, and the revision document would hold versions B and C. Both the autograph and the revised copy serve as source *and* as target of separate revision processes, and so their respective encodings need to do the same. The markup between those files could be coordinated using `<genState>`, which also clarifies the chronological order. But still, this results in a very widespread duplication of content across multiple files. At first sight, this seems to contradict the intended separation of document and text. Accordingly, we continued our considerations of the different alternatives and tried to explore if we couldn't simplify the model again by returning to the conceived clarity of the first module's data model, which was based on a single MEI file, used in conjunction with SVG only.

Going Back to a Single File?

Instead of utilizing a separate file for each document, plus one file for the text (which can easily end up in almost all other files as seen above), it seemed reasonable to return to the basic principles of the first module's data model. When multiple files end up having the same content, why not restrict oneself to just one file? Such a step would fully avoid duplication of both markup and marked up content. This single file would then focus on encoding the work's text. Multiple `<facsimile>` elements used to capture the page setup of all documents could be easily integrated into this file – a single `<measure>` element could be linked to `<zone>` elements within the `<surface>` encodings of different documents. Retrieving the location (or absence) of each measure within a given document would be equally possible (see Listing 2).

```
<manifestation xml:id="antePrint"/>
<manifestation xml:id="postPrint"/>

<facsimile decls="#antePrint">
  <surface label="page of print ante correcturam">
    <zone xml:id="anteMeasure1Rect" ulx="…" uly="…" lrx="…" lry="…"/>
    <zone xml:id="anteMeasure2Rect" ulx="…" uly="…" lrx="…" lry="…"/>
  </surface>
</facsimile>
<facsimile decls="#postPrint">
  <surface label="page of print post correcturam">
    <zone xml:id="postMeasure1Rect" ulx="…" uly="…" lrx="…" lry="…"/>
  </surface>
</facsimile>

<!-- measure available in both documents -->
<measure facs="#anteMeasure1Rect #postMeasure1Rect">…</measure>
<!-- measure available in ante doc only -->
<measure facs="#anteMeasure2Rect">…</measure>
```

Listing 2: Referencing measure positions in multiple documents. Genetic processes like additions or deletions, which would explain the different measures, have been omitted for clarity of the example. Individual notes may link to their SVG shapes in respective documents following the same principle.

Determining the right level of 'diplomatic accuracy' for the encodings seemed more challenging. When the encoding aims to capture the textual development across multiple documents, most of the content will be found in more than one source. Just like measures, individual notes (and other components) may link to their respective SVG shapes in multiple documents. *Explicit* metatexts found in individual documents, like marginal notes, revision instructions, and so on, are encoded using MEI's `<metaMark>` element. This element offers the `@source` attribute, so it is easy to identify in which document any given explicit metatext can be found.[23]

But even when leaving those specifics aside, during our considerations of using only a single file again revealed problems with differences in the text provided by the documents: Occasionally, Beethoven unintentionally introduced 'variants' or made copying errors when copying the context into his revision lists. For him, it might not have been important to faithfully copy every slur etc., but for a reader, it is sometimes hard to separate those pragmatic omissions from intentional change requests. Accordingly, those differences need to be made explicit in an encoding. They can be captured using a combination of `<orig>` (original, i.e., correct reading) and `<sic>` (markup for apparent errors), nested inside a `<choice>` element.[24] As with any other editorial markup, those elements may use a combination of `@source` (the document in which this reading can be found), `@resp` (identifies the agent responsible for this content), and `@state` (helps to clarify the chronology of the composition).

But even such an elaborate data model may not always allow to create satisfactory transcriptions. The third complaint from Beethoven's list for Op. 73 illustrates yet another significant challenge. Whereas the original text for the piano is written on both systems of the grand staff, Beethoven provides only the right hand as context for his complaint and puts it on a single staff, with clefs added to avoid at least some ledger lines (see Figure 8). While the notes stay the same, they are written in quite different ways. With a single file approach, it is not possible to reproduce these individual layouts – unless `<app>` and `<rdg>` elements are used to trace such alternatives. This, however, would result in quite complex files, contradicting the intended ease of use with only a single file. The example also shows that an earlier assumption should be reconsidered: Beethoven's handwriting is not easy to decipher here, and it requires some time to figure out how the two sources relate. If

23 *Implicit* metatexts, such as an uncommon spacing of individual signs on a page, cannot be encoded easily in this model, though, as they often rely on positional attributes. Covering them would result in excessive use of `<app>` and `<rdg>` elements for rather marginal information, and was thus considered dispensable. Moreover, high-res facsimiles are always available as a fall-back.

24 Like TEI, MEI has no explicit element to encode the absence of a feature (available in different witnesses or elsewhere) – the `<gap>` element available in both formats is somewhat related, but has a slightly different meaning. It is usually connoted as an editorial omission, not an authorial one.

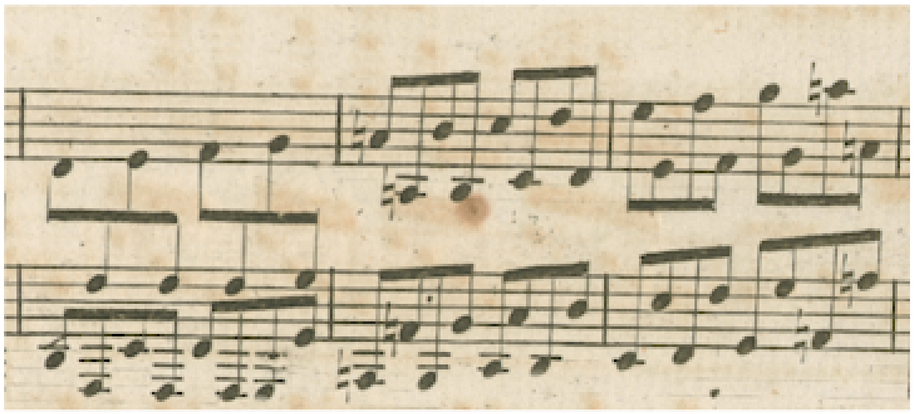
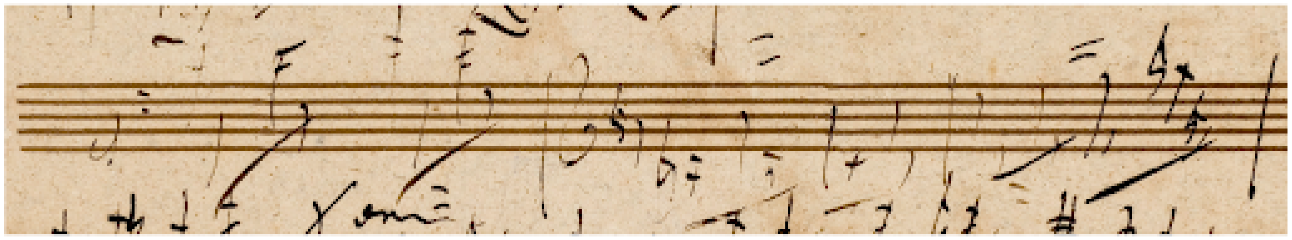
Figure 8: Top: Measures 212–214 of the first movement of the Piano Concerto, Op. 73, in the edition *ante revisionem* (D-BNba, C 73/9, p. 10). Bottom: Third issue on the revision list concerning this section (US-NYj, 31 B393cp no. 5 errata, p. 1). Beethoven asks for extenders (the 'equal' signs above the system) for a *crescendo* that started in the preceding measure.

the transcription reflects only the content of the notes, but not necessarily their layout as found in the sources, it provides only limited support. In situations like this, it would be extremely helpful for users of the edition who are not deeply familiar with Beethoven's handwriting to have faithful transcriptions for each document. It would be equally important to have one-to-one relationships between the notes across documents. A graphical user interface to these encodings could highlight the corresponding notes between multiple documents and their respective transcriptions. In the first module of the project, this type of 'in-place assistance' was considered very helpful to better understand the genetic processes taking place in manuscript documents. Even though the examples at hand for the third module usually don't offer the complexity of multiple writing layers stacked in one place, it seems sensible to prepare for such more challenging examples in the upcoming modules of the project. Accordingly, it seems counterproductive to pursue the idea of using a single file only – the limitations of that approach seem too severe to ignore, and the potential benefits turn out to be smaller than expected.

Data Model Conclusions

After all these considerations, it becomes evident that any solution needs to be a compromise. There are three independent requirements for the envisioned encoding model, and they will not combine easily.

First, the data model should support a significant level of diplomatic accuracy. While incorrect stem directions might be tolerable, a wrong distribution of notes across staves is certainly not. Examples from the current project module, but certainly also from future modules will be incomprehensible otherwise. Resorting to a common encoding in one file is not only a step back compared to prior achievements of the project, it would also complicate sufficiently accessible editions that faithfully transcribe the available documents as *products*. Any solution based on a single file would require the use of <app> and <rdg> elements to a level where this solution is no less complex than an approach based on multiple files.

Second, from a genetic perspective, it is crucial to fully trace the textual development: How do notes in multiple documents relate to each other? If everything were encoded in a single file, *process markup* like <add>

and `<del>` would be sufficient to clarify these relations. Any approach based on multiple files requires explicit linking, which not only takes significant work to generate, but is also challenging to maintain.[25]

The third requirement is something very generic: A data model should be as intuitive and easy to implement and process as possible. The less markup is used, the less error-prone the resulting edition is – making it easier for editors to generate the required markup and to proofread their files afterwards. In addition, it will be easier for developers to write the code for the edition's user interface. So, a 'simple' data model has a value in its own right, as the lower implementation effort is likely to allow more content to be covered.

Obviously, it is not possible to match all three requirements at the same time: A data model may allow diplomatic faithfulness in a relatively simple form, but that will not cover the relations between multiple documents properly. Another approach would be capable of covering these genetic connections in an elegant and simple form using a single file, but that will not allow diplomatic accuracy at the same time. Finally, the combination of diplomatic accuracy for multiple documents and the coverage of the overarching genetic development of the work through some form of linking will inevitably result in a fairly complex data model – an example of a complex problem requiring a complex solution.

As our considerations have shown that the project must not not forego requirements one and two, the only option is to minimize the effort of utilizing a complex data model that covers both of them. The most promising starting point for that was discussed in the context of Op. 120. Here, one MEI file was intended to trace the textual development of the work. In addition to that, one file per document would provide an (editorially enriched) transcription of the textual development accessible in that document. While this approach introduces a lot of redundancy especially for easy-to-read printed music documents, it supports a helpful distinction between a more diplomatic transcription of what is actually written (the 'achieved' text) and a more normalized version of what was originally requested (the 'intended' text). As both perspectives often do not align properly, it seems legitimate to allow for both of them independently. The main challenge was then to create the necessary connections between the text and document encodings. As mentioned above, entering thousands of UUID-based references manually seemed dangerously error-prone. Here, the project sidestepped the problem by utilizing the available SVG shapes.[26] This process simplified data entry for those links significantly. The resulting annotations (see Listing 3) provide great flexibility in linking content at any level of granularity. While this approach does not simplify processing the data model, it simplifies data entry considerably and makes the adoption of this data model actually feasible. With individual files for each document, it allows diplomatic precision in the encoding where necessary. The duplication of content that comes with this approach is actually necessary to allow a proper distinction between (achieved and intended) *text* and the *documents*. The very scripture of each document is still encoded using SVG shapes. The genetic processes, though being traced to some degree in the document encodings as well, are properly treated in a separate file, which can establish order of and relations between the documents.

The data model of the project's first module is mostly compatible with this on a conceptual level, so that existing data may eventually be converted to the current model. At the same time, the model seems sufficiently robust to handle research questions and materials that are scheduled for upcoming modules of the project, including an edition of a sketchbook in relation to the works contained. Those modules will serve as a benchmark to evaluate if *Beethovens Werkstatt* is actually approaching a consistent model for encoding genetic editions with MEI.

25 Especially when using 'non-speaking' identifiers like UUIDs (see https://en.wikipedia.org/wiki/Universally_unique_identifier; accessed January 12, 2022), it is barely possible to manually control such links. An additional challenge is that links may need to operate on different levels of granularity: While right now probably all links may operate on the level of individual notes, rests, and so on, for sketch material larger musical 'ideas' or 'phrases' may need to be linked. It is evident that such linkage is based on human interpretation and may neither be auto-generated nor auto-validated.

26 SVG shapes are available and linked to their corresponding MEI elements for all document encodings using a very simple web application called "Genetic Sandbox", which was developed in the project's first module. Here, a single page of a document is shown as facsimile, with all SVG shapes laid on top. The user may now click on individual shapes and highlight them in different colors. A click will bring up the ID of that element, which can then be inserted into the corresponding `@facs` attributes in the MEI file. While mistakes are still possible, this mutual reference has proven to help reduce their number significantly, resulting in very reliable connections between encodings and shapes. This approach can be used to support the entering of links between document and text encoding as well. Although the SVG shapes of notes, etc., are relevant for the document files only, they can be temporarily associated to their corresponding elements in the text file using the same Genetic Sandbox.

```
<annot xml:id="aab2668bf-91fb-4880-a7fd-6fd146c66bl6"
  plist="#xe542fc95-cb05-4731-90d0-146cd0b06246" type="note">
  <relation rel="hasEmbodiment" label="D-BNba_C73-9"
    target="#x047398ae-d689-4541-88bf-13988ebed121"/>
  <relation rel="hasEmbodiment" label="A-Wn_SH.Beethoven.323"
    target="#x2a8429eb-bla7-400d-9543-18808cf3620f"/>
  <relation rel="hasEmbodiment" label="US-NYj_31_B393cp_no.5_errata"
    target="#x8cf86ccc-6ee3-4f2a-9847-fd2c35d0d473"/>
</annot>
```

Listing 3: Semantic reference between elements from the text file and their corresponding elements in the document encodings. Here, the note with ID `xe542fc95-cb05-4731-90d0-146cd0b06246` is linked to the elements by `relation/@target`. FRBR terminology (`hasEmbodiment`) is used to describe the relationship: The documents serve as manifestations for the abstract text file. Here the attributes `annot/@type` and `relation/@label` serve as temporary aids to proofread the automatic generation of these links. Since they don't contribute in relevant ways, they will be removed for final publication. All links are relative within a file only as XInclude is used to combine the contents of all files into one logical MEI document.

Acknowledgments

This paper is also based on work of all other project members of the *Beethovens Werkstatt* team: Bernhard R. Appel, Kristin Herold, Ran Mo, Andreas Münzmay, Elisa Novara, Salome Obert, Agnes Seipelt, Richard Sänger, Joachim Veit, and Jan-Peter Voigt. We would also like to thank Jens Dufner, Perry Roland, and Mark Saccomano for their invaluable input.

Works Cited

[1] Kepper, Johannes. "Wie? Was? Entsetzen! Lessons Learned from the Freischütz Digital Project" in *Music Encoding Conference Proceedings* (MEC 2016), 95–105.

[2] Kepper, Johannes, and Richard Sänger. "Encoding Genetical Processes" in *Music Encoding Conference Proceedings* (MEC 2015), 37–44.