The Link is the Data

On Realisations, Representations, and the Link Betwixt

Werner Kriechbaum   Gerhard Stenzel
IBM Deutschland Entwicklung GmbH
Schönaicher Strasze 220
D-71032 Böblingen, Germany
{kriechba,stenzel}@de.ibm.com

ABSTRACT

The paper shows how links can be used to imprint structural and supplementary information from the representation of multimedia data on the realisation and thus enable a structure-based navigation. The authors suggest that similar linking mechanisms can and should be used to connect descriptive metadata with the essence.

1. INTRODUCTION

Most if not all multimedia data exist or can be expressed in two equivalent forms, either as a symbolic representation, or as a realisation. This duality is most obvious for audio material but holds for audio-visual material or “non-traditional multimedia” data, like e.g. buildings or proteins, as well. Examples of representations are the score of a piece of music, the storyboard of a movie, and the layout of a building – corresponding to the realisations of an audio recording, the filmed movie, and the building either built in reality or virtual reality. In many cases, the symbolic representation captures all but the “emotional” aspects of a realisation. The loss of this aspect is often compensated by the fact that the symbolic representation either contains additional structural information or makes it comparatively easy to derive this information. Therefore the representation is a powerful tool to ease the navigation and the interpretation of the corresponding realisation, given the relation of elements from the representation to elements of the realisation can be established and encoded.

2. REPRESENTATION AND REALISATION

Representation and realisation are different manifestations of the same content and can – at least in principle – be generated from the respective other form (Fig. 1): A representation can be rendered to create a realisation and a realisation transcribed to produce a representation. Many attempts to make multimedia data more accessible for indexing, search, or navigation try to derive a structure from audio or video streams (see e.g. [16]). Despite tremendous advances in rendering as well as in transcription technologies (for recent examples in the visual domain see [8],[10],[13],[17]) the automatic extraction of a symbolic representation from multimedia data is still much harder than the generation of a faithful realisation. And even if an accurate transcription can be achieved much of the underlying structure of the data is lost: State-of-the-art speech recognition systems still rely on their user to explicitly supply punctuation marks. But in cases where a symbolic representation is available, the structure can be extracted from the representation and imprinted on the realisation.

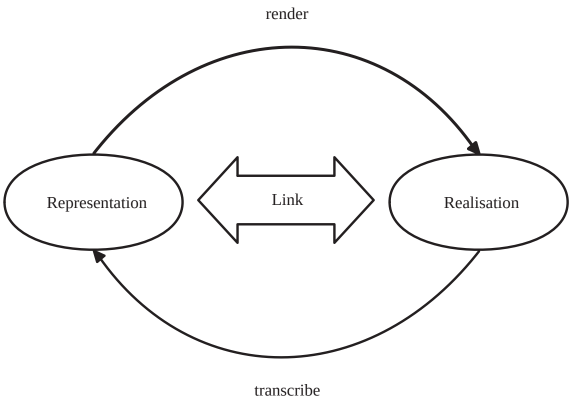

Figure 1: The relation between representation and realisation

Consider for example the audio or video recording of the staging of a theatre play. The text of a play like Shakespeare’s Hamlet has a rich structure built from acts, scenes, and dialogues (Fig. 2) [18] that can be made explicit by markup.

```
<ACT><TITLE>ACT I</TITLE>
<SCENE><TITLE>SCENE I.  Elsinore. A platform before
the castle.</TITLE>
<STAGEDIR>FRANCISCO at his post. Enter to him
BERNARDO</STAGEDIR>
<SPEECH>
```

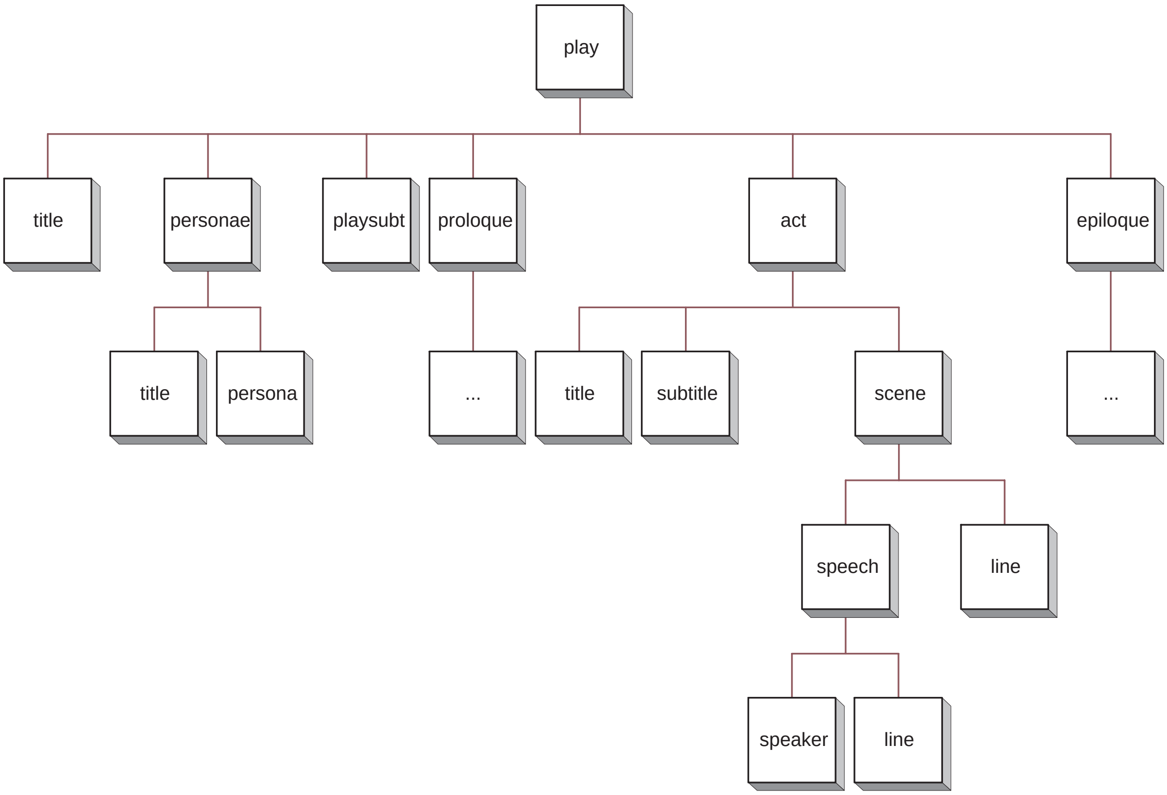

Figure 2: The simplified TEI document type definition for a play

```
<SPEAKER>BERNARDO</SPEAKER>
<LINE>Who’s there?</LINE>
</SPEECH>
<SPEECH>
<SPEAKER>FRANCISCO</SPEAKER>
<LINE>Nay, answer me: stand, and unfold yourself.
</LINE>
</SPEECH>
<SPEECH>
<SPEAKER>BERNARDO</SPEAKER>
<LINE>Long live the king!</LINE>
</SPEECH>
<SPEECH>
<SPEAKER>FRANCISCO</SPEAKER>
<LINE>Bernardo?</LINE>
</SPEECH>
<SPEECH>
<SPEAKER>BERNARDO</SPEAKER>
<LINE>He.</LINE>
</SPEECH>
<SPEECH>
<SPEAKER>FRANCISCO</SPEAKER>
<LINE>You come most carefully upon your hour.
</LINE>
</SPEECH>
...
<STAGEDIR>Enter HORATIO and MARCELLUS</STAGEDIR>
...
<STAGEDIR>Exeunt</STAGEDIR>
</SCENE>
```

```
<SCENE><TITLE>SCENE II.  A room of state in the
castle.</TITLE>
<STAGEDIR>Enter KING CLAUDIUS, QUEEN GERTRUDE,
HAMLET, POLONIUS, LAERTES, VOLTIMAND, CORNELIUS,
Lords, and Attendants</STAGEDIR>
<SPEECH>
<SPEAKER>KING CLAUDIUS</SPEAKER>
<LINE>Though yet of Hamlet our dear brother’s death
</LINE>
```

Tags like ACT or SCENE allow structure based navigation (“Go forward one scene”) and structure aware queries (“Which speaker referred to Hamlet for the first time and in which scene?”). Imprinted on the audio recording the same navigation is possible for the realisation – quite a difference from the customary navigation based on track numbers and timecodes. In addition the text of a play contains a wealth of stage directions that provide additional information. The original stage directions of a play can never be derived from the realisation, even if some sophisticated analysis might be able to interfere them, it is not possible to ascertain that the performance honours the original stage direction without analysing the representation. A similar argument applies to the structuring of music. It is possible to structure an audio recording according to derived dynamical patterns, but without access to the score it is not possible to structure the realisation according to the dynamical patterns intended by the composer. Like structure stage directions can be used for navigation (“Go to where Horatio and Marcellus enter stage”) and query but in addition some of the stage directions can provide helpful hints for automatic transcription tools: For a person tracker it is rather useful to know who is – or should be – on stage and who not.

Often a representation has more than one structure: A printed book has a surface structure (pages) as well as a deep structure built from parts, chapters, subchapters, footnotes and cross-references. Finding and addressing specific structures (pages, chapters, etc.) in its audio book realisation is either time consuming, complex, or impossible. Whereas this is only inconvenient for the casual user of an audio book, the capability to search, navigate, and cite conforming to the printed edition is essential for blind students when working with audio text books. Linking representation and realisation solves many of these problems: The structure information can be used to segment the audio signal in meaningful units like sentence, paragraph or chapter, and standard text-mining technologies can be used to locate sentences of interest. Even, if the transcript is unstructured in the sense that it does not contain formal markup, punctuation can serve as means to partition the document and thus create a very basic kind of structure.

3. COMPLEX STRUCTURES

In most cases the document structure of a play or a book seems to be rather simple: There is a surface structure that refers to the physical appearance of the printed representation (page and line numbers) and a structure that describes the content as such (acts, scenes, speeches or chapters, subchapters, paragraphs). More complex structural relationships exist but these are more apparent in music and will therefore be discussed in the context of classical European music. Music has a complex temporal organisation and a rich semantic structure that defines a natural segmentation for the audio stream. Like in images, in music many features are characteristic for segments, vary from segment to segment, and become meaningless when averaged over all segments. For most, if not all music, the structure is not arbitrary but conforms to one of a small set of possible pattern [11]. The sonata form, or better sonata forms [15] since it is a set of closely related patterns, is one of the most important patterns of Western music in the Classical Era and a highly simplified sketch of the structure of this form is depicted in Fig. 3. The sonata form is built from at least three pieces: An optional Introduction, an Exposition, a middle part (called "Durchführung" or Development), a Repeat and an optional Coda. The Exposition is not an atomic structural element but built from the succession of a First Theme, a Bridge, a Second Theme, an optional Third Theme, and an Epilogue. The structure of the sonata form is governed by two relationships between segments: A segment may be part of another segment (signified by rhomboid arrows) and the segments are ordered in time (signified by "follows" arrows). These relations are similar to those found in a play (an act is made of scenes and the acts or scenes are ordered in time) and are captured by trees with ordered siblings like the ones that can be defined with SGML or XML document type definitions. But the relationship most interesting when analysing music is a transformation relationship: In the sonata form and in most other musical forms some segments are derived from other segments by applying a transformation. For example, in the first movement of Haydn's Symphony No. 82, the second theme is the first theme transposed to a different key. Common transformations are

- transposition of a theme from Tonic to Dominant
- expansion of a Motif
- fragmentation of a Theme
- inversion and reflection
- doubling or halving of the tempo
- changes in intensity

Many of these transformations do not operate on time intervals of the complete score but act on segments of individual voices of a piece thus complicating the correspondence between representation and realisation. In addition, as soon as more than one monophonic instrument plays, a single segment in the realisation corresponds to many segments in the representation. In essence the transformations establish relations between segments in the representation and/or segments in the realisation like inversion(A,B), i.e. segment B is an inversion of segment A. These additional relations between segments transform the structure tree of a piece of music into a structure web. Such a web can be modelled in two different ways: Either in an object-oriented flavour by making segment the central data type encapsulating all its links (cf. Fig 3) or by taking a document structure approach where the parent-child relationship is singled out to build a structure tree and the transformational links are added to the tree nodes as attributes. In both approaches the links between the segments are the dominant features to describe the representation.

The approach to architecture or document structure illustrated in Fig. 3 is prescriptive: As in 19th century music theory, it is assumed that there is an ideal architecture for a sonata form, and that a piece of music not conforming to these rules is in error. In our century this concept has come under considerable criticism (e.g. [15], [9]) and a descriptive approach has been advocated. In a descriptive approach a set of features induces a document structure that may or may not correspond to last century's idealistic conception. As a consequence, each musical work may have a plenitude of valid structures describing different aspects of its analysis or interpretation. Without the means to link these different structures to each other – and to the realisation – such a multi-faceted description is rather useless. Like with different transformational relations within one description there are two basic alternatives to structure this set of document structures: One may single out one document structure as a master and try to relate all other document structures to this master structure. Such an architecture is rather well behaved from a computational point of view; each new structure needs only to be synchronised with the master structure. The disadvantage is that comparisons between child-structures are only possible by comparing each child with the master and further processing of the results of these comparisons. As an alternative one may create a web (or net) of different structures, relating each individual structure to all others. Here all structures are peers and direct comparison between them is straightforward but when an additional structure is added it has to be synchronised

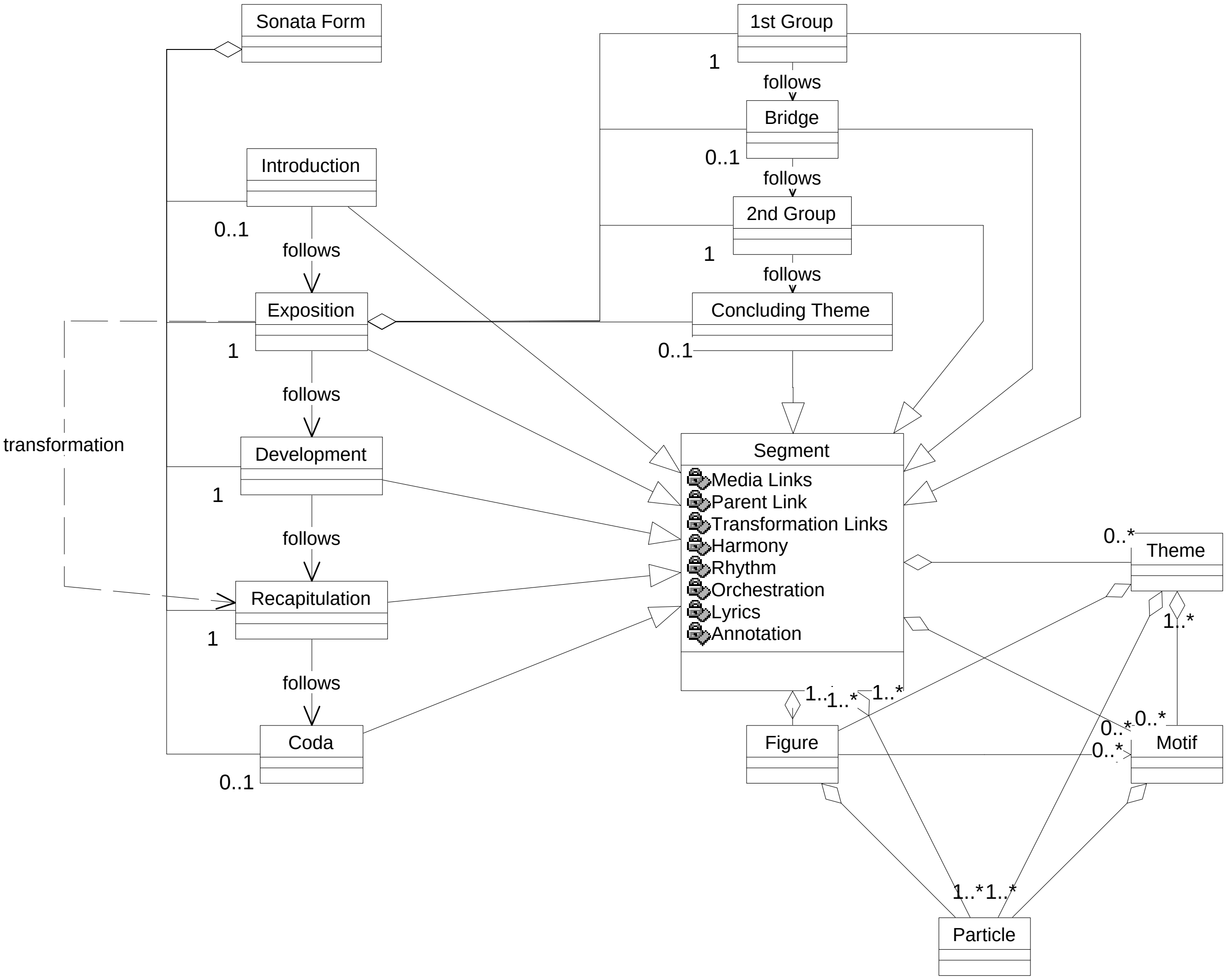

Figure 3: The prescriptive structure of a sonata form. Building blocks are links and segments. Segments have features like rhythm and harmony, serve as linkends, and can be typed with semantic labels like Motif or Particle.

with all the other structures. For more complex documents like a classical symphony, the size of the data needed to synchronise the different structures may easily surpass the size of the structure descriptions themselves.

4. TRADITIONAL "METADATA"

Besides relations within and between structural elements of a representation or a realisation, traditional metadata like author information can be interpreted as a relation (e.g. isAuthorOf(person, opus)) and therefore as a link. For minimal metadata sets like e.g. Dublin Core [3] where elements like "Creator" or "Contributor" are usually quite small and embedded in the essence as tags such an approach might be considered unnecessary complicated. But it offers at least one important advantage: It separates the description from the essence and an archive can provide *one* curated set of e.g. author data that can be linked to all pieces produced by this author. Trivial as this may seem it nicely solves the problem of spelling variants or misspelled entries generating phantom persons, a major problem in all existing catalogues. For more complex metadata sets like MIDAS [6] a separation of description and essence becomes mandatory. Whereas in Dublin Core the creator information is just a character string, the MIDAS description scheme for an artist (Fig. 4) is a tree with a considerable size. Even a "simple" descriptor like the name of an artist has eleven elements most of which can be instantiated multiple times. A system that can make use of this complexity can find "Some like it hot" as an answer to a query for all the movies with Norma Jean Mortenson acting. But the size of a fully instantiated MIDAS description schema is definitely beyond what one would want to embed into a low bandwidth stream for annotation. Links into the tree together with a tree transformation language like XSL [1] can be used to filter specific elements from the artist description and provide the means to adapt the scope of the knowledge base to the desired usage without changing the knowledge base.

5. LINKS

Up to now the term "link" has been used as a concept but the actual linking mechanism that connects the different representations and realisations has not been specified. A

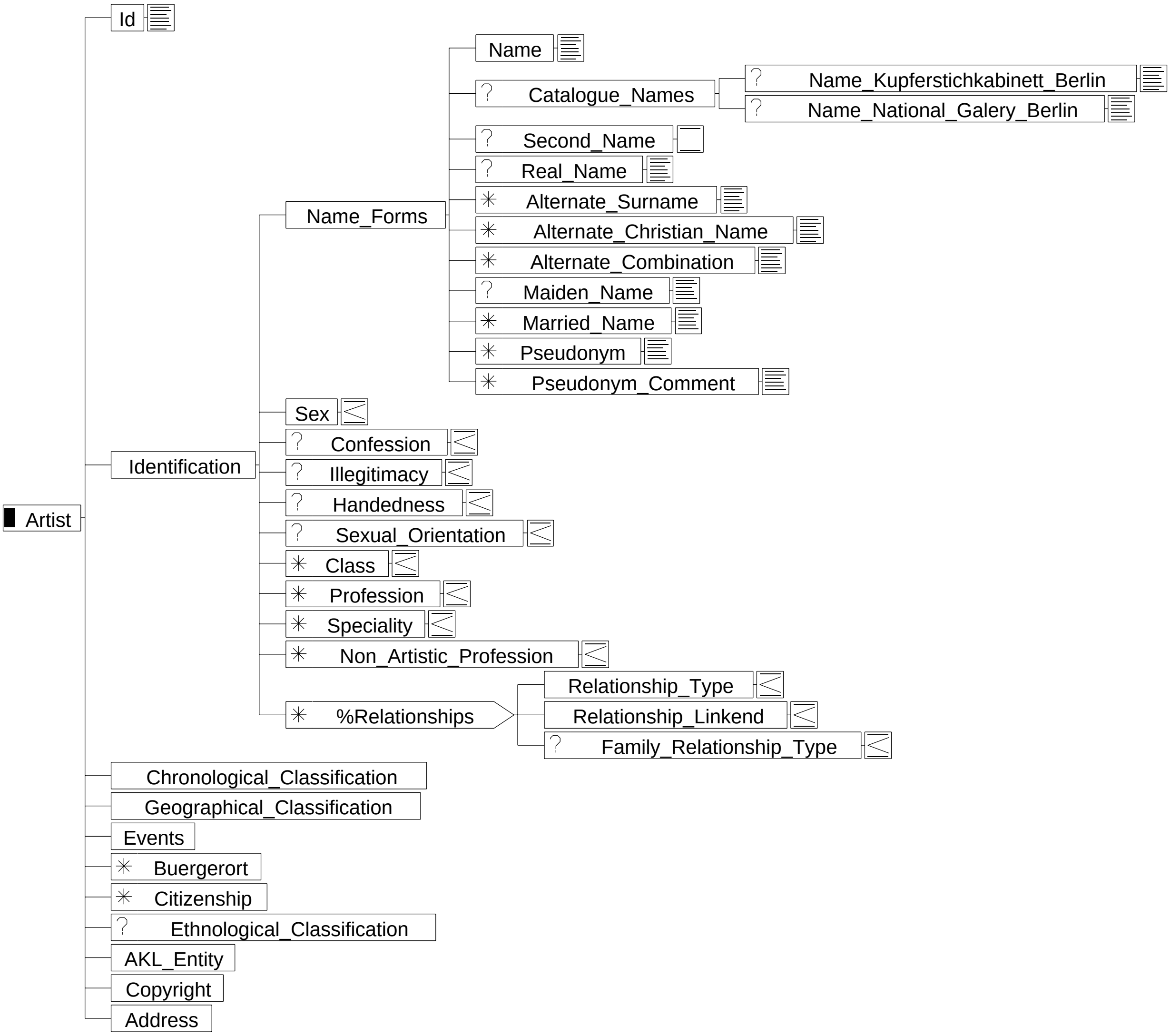

Figure 4: An person according to MIDAS

versatile linking mechanism has to fulfil a diverse set of requirements, especially it should support:

- arbitrary formats
  Representations may be unstructured (plain text) or structured using a variety of markup languages ranging from standardised (e.g. SGML) over de facto standards (e.g. LaTeX) to proprietary formats (e.g. Word). Realisations are usually stored as bit streams, nonlinear compression formats with varying bitrates are quite common (e.g. MP3), and new encoding schemes are emerging at a steady rate (e.g. Ogg Vorbis [12]),

- read only media
  More and more content is distributed on read only media like CD or DVD. Even if the data can be transferred to a writeable medium, many bitstreams do not support the insertion of additional data at arbitrary locations and a link must be able to locate its linkend without modifying the data. In addition data stores for metadata like an author database usually do not grant write access to their users and therefore have to be accessed without changing the content of the data store.

- many-to-many links
  Often different versions of the same realisation are available: In the case of images or music many archives store the same realisation in multiple versions with different quality (compressed for free preview and linear for fee). A data collection that is focused on performing history is likely to link equivalent segments in different realisations. And a descriptive approach to musical form has to deal with more than one representation for the same realisation.

- bidirectional link traversal
  This is a direct consequence of many-to-many links. If all the linkends are peers there must be a way to traverse from each linkend to all the other ones.

- points and intervals
  Since the links are used to map structural elements like the scene of a play on a bitstream there must be means to address intervals in the realisation.

An immediate consequence of the requirement to address read only media is that the link information has to be self contained such that it can be stored in a separate document. The requirement to support arbitrary formats has as a consequence that the linking mechanism can not rely on its targets to supply some unique identification to specify the anchor point of the link. This problem can be solved by isolating the link from the linkend with a locator layer that deals with media-specific addressing issues and presents a unique identification for the linkends to the link proper. This indirection approach has been pioneered by the HyTime ([2],[7]) independent link (Fig. 5) and the mechanics of linking will be discussed using the HyTime ilink as an example. Some aspects of the ilink functionality have influenced the design of XLink, XPath, and XPointer and can be expressed in XML but its ability to refer to segments of non-SGML data is still unique. A HyTime independent link does not need to reside at any of its endpoints, the ilink tag has an attribute linkends which may specify any number of identifiers each of which points to a locator.

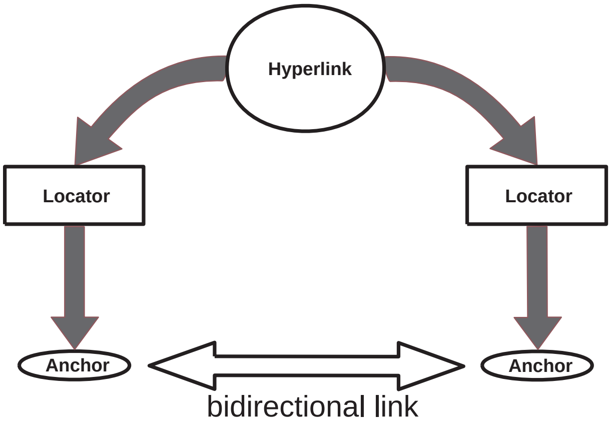

Figure 5: Elements of an independent hyperlink

```
<myLink linkends="repA1 repA2 repA3 relA1 relA2">
```

One of the most useful locators is a treelocator, HyTime's ancestor of XML's extended pointer concept, which allows the navigation of untagged trees:

```
<treeloc id="repA1" locsrc=sgmllink>1 2 3 2
</treeloc>
```

Each node in the tree is identified by a list of integer values. The list of integers describes how to get from the root of tree to the specific node. The root node is assigned the number '1' and each successive number describes the position of the node among the children of the parent node by counting the children from left to right starting with '1' for the first child. Thus when applied to the excerpt from Hamlet used above, the tree locator 1 2 3 2 selects the first spoken words "Who's there":

1 specifies the root node ACT

1 **2** specifies the SCENE, node 1 1 is the TITLE of the ACT

1 2 **3** specifies the SPEECH, node 1 2 1 is the TITLE, node 1 2 2 STAGEDIR

1 2 3 **2** specifies the LINE, node 1 2 3 1 is the SPEAKER

Since they do not rely on the presence of tags, tree locators provide a powerful addressing mechanism that covers a wide range of special cases like e.g. lattice structures or matrices. Whereas HyTime does not predefine the locators needed to refer to segments in the realisation it provides mechanisms to define coordinate systems and measurement units which can be used to define "custom-made" locators (for details see [7, 2]) for points and intervals in arbitrary media.

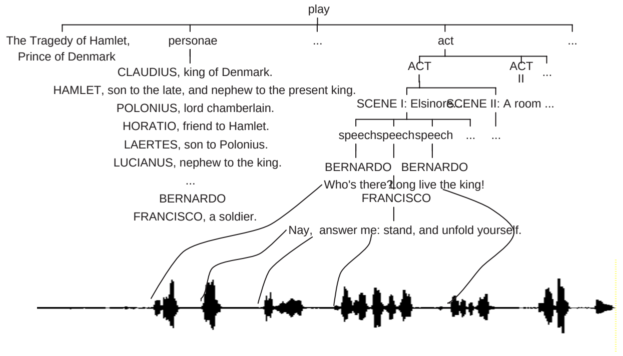
play
The Tragedy of Hamlet, Prince of Denmark
personae
...
act
...
CLAUDIUS, king of Denmark.
HAMLET, son to the late, and nephew to the present king.
POLONIUS, lord chamberlain.
HORATIO, friend to Hamlet.
LAERTES, son to Polonius.
LUCIANUS, nephew to the king.
...
BERNARDO
FRANCISCO, a soldier.
ACT
ACT II
...
SCENE I: Elsinore
SCENE II: A room ...
speech speech speech ... ...
BERNARDO
BERNARDO
Who's there? Long live the king!
FRANCISCO
Nay, answer me: stand, and unfold yourself.

Figure 6: Linking Hamlet

Inverse traversal from anchor to link is possible by searching for the locators that cover the current position and selecting all the links referring to these locators.

6. GENERATING THE LINKS

In order to link a performance of Hamlet recorded on CD with the tagged text of the play a separate link document has to be generated that connects recorded utterances with the text and imprints the structure information on the audio stream (Fig. 6).

To simplify the description of the link generation (Fig. 7) it is assumed that a linear recording of the performance (e.g. a WAVE file) and an SGML or XML tagged text of the play are available. The principle processing steps are illustrated in Fig. 7: Structure and plain text are extracted from the tagged representation and separated. The plain text is decorated with time tags that specify start time and end time of each word in the plain text. This timestamped representation is merged with the extracted structure and formatted conforming to HyTime syntax.

Generating the plain text from the tagged Hamlet representation requires some pre-processing: All tags that markup unspoken content (ACT, SCENE, etc.) and all implied markup like punctuation are filtered from the representation and from the remaining tags only the untagged content is used:

```
Who’s there Nay answer me stand and unfold
yourself Long live the king Bernardo He You
come most carefully upon your hour
```

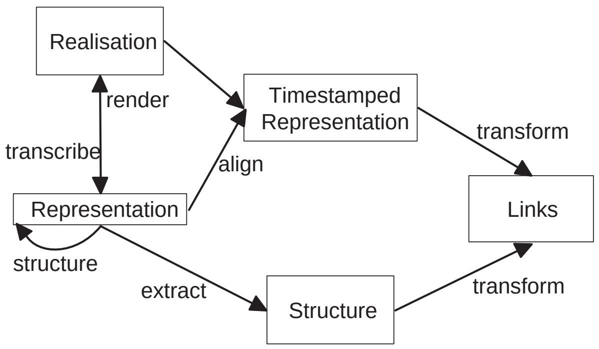

Figure 7: Generating the links

The structure is represented by a sequence of tree locators for the spoken words indexed with the words:

```
1 2 3 2 1  Who’s
1 2 3 2 2  there
1 2 4 2 1  Nay
1 2 4 2 2  answer
1 2 4 2 3  me
1 2 4 2 4  stand
1 2 4 2 5  and
1 2 4 2 6  unfold
1 2 4 2 7  yourself
1 2 5 2 1  Long
1 2 5 2 2  live
1 2 5 2 3  the
1 2 5 2 4  king
1 2 6 2 1  Bernardo
1 2 7 2 1  He
1 2 8 2 1  You
1 2 8 2 2  come
1 2 8 2 3  most
1 2 8 2 4  carefully
1 2 8 2 5  upon
1 2 8 2 6  your
1 2 8 2 7  hour
```

The audio recording is fed through a speech recognition engine that produces a transcript and tags each recognised word with its start and end time. As can be seen from Fig. 8 [19] there is an n:m relation between the plain text (called reference text in the figure) and the speech recognition transcript. Plain text and transcript are aligned with a dynamic programming algorithm [4] and after alignment the word times from the recognised text are transferred to the corresponding words in the plain text. Most of the speech recognition errors do not influence the timing values: An isolated word recognition error (alignment quality 1 in Fig. 8) still gives the correct timing. An error were instead of a single word in the representation a group of words is recognised (alignment quality 2) does not pose a problem, the correct word is marked with the start time of the first word and the end time of last word in the recognised group. Only errors where a group of words in the representation is transcribed as a group of incorrect words (alignment quality 3) yield unrecoverable timestamp errors. In this case the missing time values can either be generated by interpolation or the timing for the afflicted word boundaries can be marked as unknown. After aligning and resolving the speech recognition errors we can tag the words from the plain text with start and end times:

```
Who’s      210  211
there      211  215
Nay        220  221
```

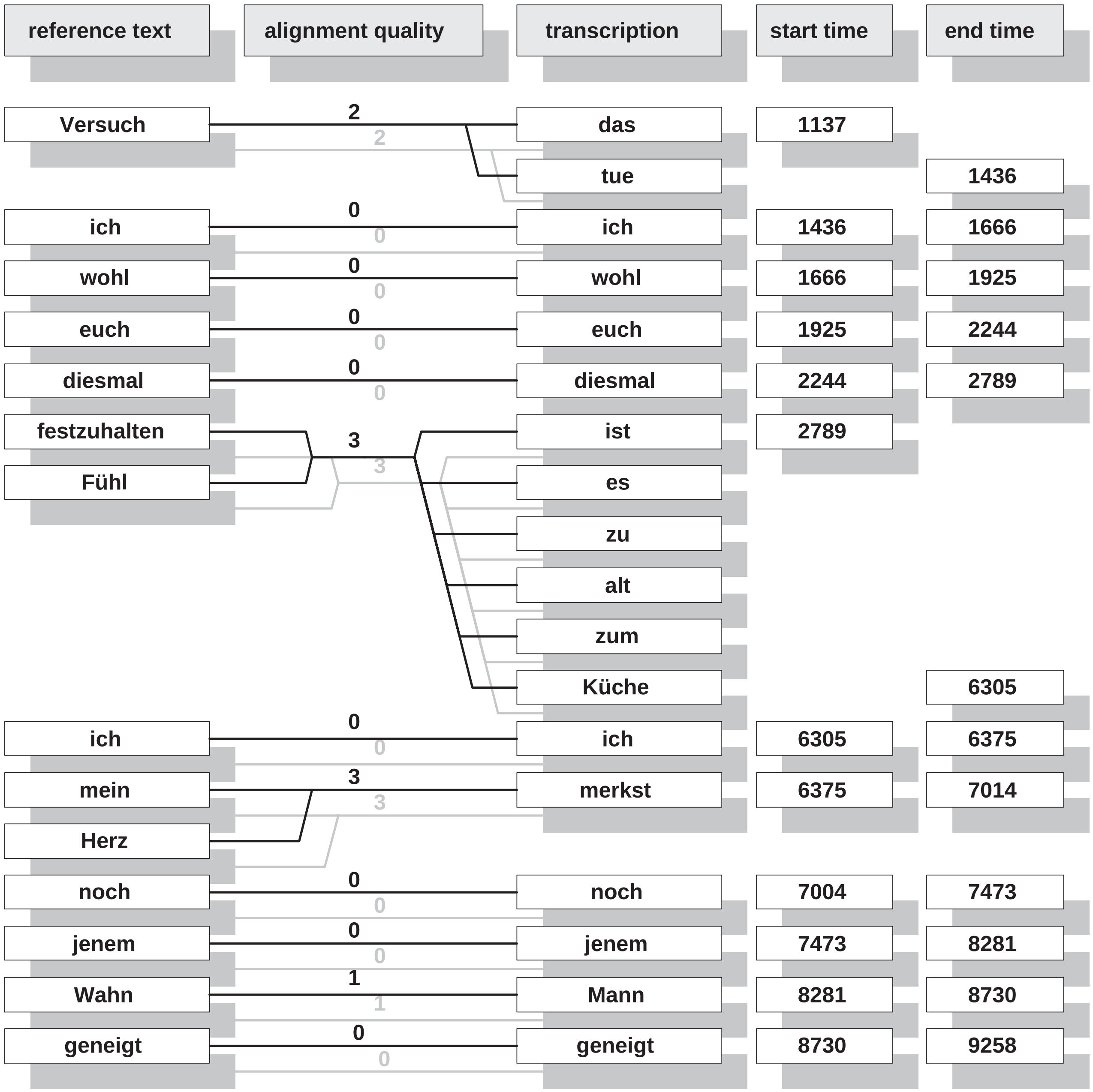

Figure 8: Aligning plain representation and the transcript

```
answer     221  226
me         227  228
stand      232  234
and        235  236
unfold     237  242
yourself   242  250
Long       260  262
live       262  263
the        264  266
king       266  267
Bernardo   275  276
He         280  281
You        290  292
come       292  294
most       294  296
carefully  297  301
upon       301  302
your       303  304
hour       304  305
```

This time tagged plain representation is merged with the word tagged structure

```
1 2 3 2 1  Who's      210  211
1 2 3 2 2  there      211  215
1 2 4 2 1  Nay        220  221
1 2 4 2 2  answer     221  226
1 2 4 2 3  me         227  228
1 2 4 2 4  stand      232  234
1 2 4 2 5  and        235  236
1 2 4 2 6  unfold     237  242
1 2 4 2 7  yourself   242  250
1 2 5 2 1  Long       260  262
1 2 5 2 2  live       262  263
1 2 5 2 3  the        264  266
1 2 5 2 4  king       266  267
1 2 6 2 1  Bernardo   275  276
1 2 7 2 1  He         280  281
1 2 8 2 1  You        290  292
1 2 8 2 2  come       292  294
1 2 8 2 3  most       294  296
1 2 8 2 4  carefully  297  301
1 2 8 2 5  upon       301  302
1 2 8 2 6  your       303  304
1 2 8 2 7  hour       304  305
```

and after eliminating the words the result is a raw link file connecting the representation with the realisation

```
1 2 3 2 1  210  211
1 2 3 2 2  211  215
1 2 4 2 1  220  221
1 2 4 2 2  221  226
1 2 4 2 3  227  228
1 2 4 2 4  232  234
1 2 4 2 5  235  236
1 2 4 2 6  237  242
1 2 4 2 7  242  250
1 2 5 2 1  260  262
1 2 5 2 2  262  263
1 2 5 2 3  264  266
1 2 5 2 4  266  267
1 2 6 2 1  275  276
1 2 7 2 1  280  281
1 2 8 2 1  290  292
1 2 8 2 2  292  294
1 2 8 2 3  294  296
1 2 8 2 4  297  301
1 2 8 2 5  301  302
1 2 8 2 6  303  304
1 2 8 2 7  304  305
```

that needs only reformatting to become a valid HyTime hub document.

7. CONCLUSION

Independent hyperlinks are a versatile and powerful mechanism to link between and among representations and realisations and to establish relations between data. Since the linkends can be arbitrary objects, nothing precludes their interpretation as object and interpretant and thus the link becomes a sign (or the sign a link): "A Sign, or Representamen, is a First which stands in such a genuine triadic relation to a Second, called its Object, as to be capable of determining a Third, called its Interpretant, to assume the same triadic realation to its Object in which it stands itself to the same Object."[14], quoted from [5]. Adopting this interpretation allows to rephrase the title of the paper to something more fitting for a semiotics conference: "The Sign is the Data".

8. REFERENCES

[1] J. Clark. XSL transformations (XSLT) 1.0. Recommendation, World Wide Web Consortium, Nov 1999.

[2] S. J. DeRose and D. D. Durand. *Making Hypermedia Work – A User's Guide to HyTime*. Kluwer, 1994.

[3] Dublin Core Metadata Initiative. Dublin core metadata element set (DCMES) Version 1.1. Recommendation, Dublin Core Metadata Initiative, 1999.

[4] D. Gusfield. *Algorithms on Strings, Trees and Sequences*. CUP, 1997.

[5] C. R. Hausmann. *Charles S. Peirce's Evolutionary Philosophy*. CUP, 1993.

[6] L. Heusinger. *Marburger Informations-, Dokumentations- und Administrations-System (MIDAS)*. K. G. Saur, second edition, 1992.

[7] ISO/IEC JTC1/SC18/WG8. Hypermedia/time-based structuring language. Technical report, 1997.

[8] L. Jin and Z. Wen. Adorning VRML worlds with environmental aspcts. *IEEE Computer Graphics and Applications*, 21(1):6–9, 2001.

[9] C. Kühn. Form. *MGG Sachteil*, 3:607–643, 1995.

[10] F. Long, D. Feng, H. Peng, and W.-C. Siu. Extracting semantic video objects. *IEEE Computer Graphics and Applications*, 21(1):48–55, 2001.

[11] S. Mauser, editor. *Handbuch der musikalischen Gattungen*. Laaber, 1993–.

[12] http://xiph.org/ogg/vorbis/index.html. Technical report.

[13] G. Papaioannou, E.-A. Karabassi, and T. Theoharis. Virtual archaeologist: Assembling the past. *IEEE Computer Graphics and Applications*, 21(2):53–59, 2001.

[14] C. S. Peirce. 1902 manuscript.

[15] C. Rosen. *Sonata Forms*. Norton, 1988.

[16] A. Sheth and W. Klas, editors. *Multimedia Data Management*. McGraw Hill, 1998.

[17] I. Shlyakhter, M. Rozenoer, J. Dorsey, and S. Teller. Reconstructing 3D tree models from instrumented photographs. *IEEE Computer Graphics and Applications*, 21(3):53–61, 2001.

[18] C. Sperber-McQueen and L. Burnard, editors. *Guidelines for Electronic Text Encoding and Interchange*. 1999.

[19] J. Stöffler. Optimizing speech recognition resources for text-audio matching, 1998.