Routledge
Taylor & Francis Group

Journal of New Music Research

ISSN: 0929-8215 (Print) 1744-5027 (Online) Journal homepage: www.tandfonline.com/journals/nnmr20

Data structures for music encoding: tables, trees, and graphs

Joshua Stutter

To cite this article: Joshua Stutter (2024) Data structures for music encoding: tables, trees, and graphs, Journal of New Music Research, 53:3-4, 208-221, DOI: 10.1080/09298215.2024.2412598

To link to this article: https://doi.org/10.1080/09298215.2024.2412598

JOURNAL OF NEW MUSIC RESEARCH
2024, VOL. 53, NOS. 3–4, 208–221
https://doi.org/10.1080/09298215.2024.2412598

Routledge
Taylor & Francis Group

OPEN ACCESS Check for updates

Data structures for music encoding: tables, trees, and graphs

Joshua Stutter

Digital Humanities Institute, University of Sheffield, Sheffield, UK

ABSTRACT

One of the challenges in planning a digital edition is the selection of the encoding system and representation format that the edition will use to encode musical information. This decision affects the form of the edition not only through the descriptive capabilities of the encoding format, but also through the format's adherence to certain data structures such as tables, trees, and graphs. Far from being straightforward containers, each of these structures possess unique qualities that constrain the encoding process into considering the music within the parameters of that data model. Data that does not fit must be cajoled into the chosen model, with varying degrees of success. Recent work has developed arguments for the use of certain formats in specific cases based on features and interoperability, but a thorough review of the suitability and sustainability of the underlying data structures has yet to be conducted.

This paper explores the problem of music representation in data structures from the perspective of musical domain. By encoding the same passages of music in multiple hypothetical formats it demonstrates the musical aspects and, particularly, the relationships that the data structures commonly used in music encoding variously excel at, privilege, or struggle with representing. I argue that encoding projects should consider the constraints that the data structures of a format impose on their encoding; whether there are instances that would be unsatisfactorily modelled, the data biases this creates, and therefore whether it would be advisable to extend the representation format to accommodate another data structure.

ARTICLE HISTORY
Received 29 December 2023
Accepted 30 September 2024

KEYWORDS
Encoding; hierarchy; representation; domain; ontology; structure

1. Introduction

It is impossible to process music by computer without a machine-readable encoding of that music in some form. As an encoding comprises the data set from which computer analyses or digital editions begin, it is vital that the encoding of that music accurately represents the features that will be processed. However, it has long been argued that the way in which something is encoded frames and alters its meaning (McLuhan, 1964). What do we mean when we say that we are encoding music? For any encoding project, musical or otherwise, a data model must be constructed that is not only capable of representing the features and objects of study, but also the relationships and hierarchies that link those objects together. For example, an encoding of an English text must be able to represent, at the most basic level, the order of its characters. Similarly, an image encoding must be capable of representing the two-dimensional layout of the constituent pixels. Crucially, the problems of data modelling concern not just the representation of the data itself, but also the way in which it is organised – i.e. the data structure.

Encoding music is unlike these basic models, however, as what constitutes 'music' in individual cases can be difficult to define both in ontological and epistemological terms (Cross & Tolbert, 2021; Kania, 2023). When we speak of music we are often calling to mind not just its auditory and ephemeral manifestation, but also visual or graphical notation, structural forms, gesture or performance information, and countless other objects and *simulacra* that together create the musical experience. How these individually relate to the broader concept of 'music' is a difficult question. Some argue that it is the score that 'defines a work' (Goodman, 1968, p. 178), others that the work exists solely through its manifold manifestations (Kania, 2008), and still others that such questions are fundamentally 'worthless' (Ridley, 2003, p. 203). Nonetheless, we want to be able to encode music to its fullest, including all of these issues and more, but the ways in which music is transmitted to us are frequently at odds with one another: for example, what may be apparent in performance might not be notated in score or vice versa.

CONTACT Joshua Stutter j.stutter@sheffield.ac.uk Digital Humanities Institute, University of Sheffield, Sheffield S10 2TN, UK

Supplemental data for this article can be accessed online at https://doi.org/10.1080/09298215.2024.2412598.

Music representation rarely reflects this fuzzy musical reality, as it is commonly held that music encodings require the concepts of music theory to be fully defined and rationalised by logical rules (Pazel, 2022, p. 10). This is despite the options for how to represent key aspects such as time and alignment often being contentious (Dannenberg, 1989, p. 74), and there not being any hard and fast rules even for Common Western Music Notation (CWMN), by far the most studied notational system (Byrd, 1984). Against thorny philosophical concerns and music that often contains more exceptions to the rules than conforming examples, music encodings are forced to make difficult representational decisions to satisfy the need for hierarchy and structure governed by their data models and thus to be able to move forward with their goals. As an example of such philosophical problems, before encoding any music, a music representation must say first that there is ontologically something worth representing in common between the artefacts of musical practice such as notation and performance (Matheson & Caplan, 2011), second that we can say epistemologically what we know about the music – such as by stating facts about its operation or by circumscribing its finite set of possibilities (Pace, 2009) – and finally that it is possible to represent key parts of these objects in a computer. In very few cases are answers to these questions as simple as we would like to believe.

In implicitly forming solutions to these issues, a music encoding system creates its own music theory through the design and implementation of its schema, and it is deceptively easy to get lost in questions of what exactly that theory is trying to represent rather than how best to represent it. Controlling for these issues in the realm of music theory is difficult; for example Cook (2002, p. 78) readily admits that for music theories 'you can easily find yourself asking, without any clear sense of what the answer might be: is this theory about acoustic events or perceptions, about notational traces or ideal content?' Despite this, the selection of music encoding tools (and their accompanying theories of music) often appears to be driven more by personal preference than by a rigorous assessment of their underlying qualities, and this trend within independent and divergent research conducted at various centres has in the past led to 'countless barely compatible initiatives' (Pugin, 2015, p. 2). This paper investigates the theories of music that common encoding data models espouse, with a view to highlighting the relevant issues within each.

2. Musical domain

In order to be able to represent multiple and often conflicting aspects of music, music encoding systems commonly rely on the concept of musical domain. An encoding can be said to consist of a number of representations for musical features such as pitch information or rhythm, and certain elements within individual representations can then be tagged as belonging to a musical domain or set of domains. For example, a certain pitch inflection may be present only in performance, or the encoding may include analytical information not originally present in the score. Such division of music into constituent domains has been described as 'the most basic representation of music theory', and has been a goal of theories of music since antiquity (Blasius, 2002, p. 27). These distinctions are key, not only for defining and limiting the scope of the music encoding system, but also for avoiding any confusion or conflation between musical domains, such as between the graphical appearance of notation and an interpretation of its underlying semantic content.

Confusingly, there is a significant terminological overlap between the concept of domain as used in music encoding, and as used in data modelling more generally. The process of data modelling typically divides into three distinct stages or levels: a conceptual model is a high-level and somewhat abstract specification of the main entities and relationships within a model, a logical model is a more precise yet technology-independent description of the required data structures, and a physical model the way in which that model will be created using a specific database or markup language (Jannidis & Flanders, 2019, pp. 82–83). Some of the musical domains cited in this paper are named similarly, and the levels of data modelling should not be confused with the particular divisions of musical domain. At the same time, the term 'domain' has a specific definition in a physical data model, referring to the accepted inputs for a particular attribute or set of attributes. This, too, should not be confused for the looser idea of musical domain which relates more to the conceptual than the physical model. Throughout this paper, any mention of domain is used with regards to musical domain unless specifically noted as either a data modelling domain or a physical data domain.

To create a complete taxonomy of music, the list of domains into which music can be divided must be exhaustive: all music that an encoding system is wishing to represent must fit into at least one of its defined domains. However, there is little agreement as to the number and extent of musical domains, and each assessment begins to formulate its own rudimentary theory of music. Although the division of music into independent domains can be seen at least as far back as the sixth-century philosopher Boethius' *De institutione musica* (Pesce, 2011), in modern study Babbitt (1965)

Table 1. Distribution of domains in selected literature.

	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
Babbitt (1965)	✓	✓		✓				
Maxwell (1981)	✓	✓		✓				
Huron (1992)	✓	✓	✓	✓	✓		✓	✓
Leman (1993)	✓	✓	✓					
Sloan (1993)		✓	✓	✓	✓			
Selfridge-Field (1997)	✓	✓	✓	✓	✓			
Haus and Longari (2005)	✓	✓	✓	✓	✓	✓		
Steyn (2013)			✓	✓	✓	✓		✓
Roland et al. (2014)		✓	✓	✓	✓	✓		
Baratè et al. (2016)	✓	✓	✓	✓	✓	✓		

was the first to make explicit the delineation of musical process into a number of domains, arguing that the conversion between notation and sound is a difficult one at best.[1] Since then, there have been numerous hypotheses for the number and scope of the domains that music should fall into within encoding systems. The following list is not exhaustive either, but serves to demonstrate the great variety between available domains in encoding systems. There are slight differences between terminologies and definitions given by authors documented below but, for the purposes here, I believe these not to be significant. In each case I have taken the earliest available citation, and Table 1 shows how the domains are distributed across literature:

(a) **Audial:** Sound waves, the ephemera of music or audio recordings. Babbitt (1965) calls this 'auditory', Maxwell (1981) 'physical', Huron (1992) simply 'sound', Leman (1993) 'acoustic', and Selfridge-Field (1997) 'phonological'. Haus and Longari (2005) and Baratè et al. (2016) term it 'audio'.
(b) **Performative:** The ways in which performers create music. Babbitt (1965) calls this 'acoustic', Huron (1992), Haus and Longari (2005), and Baratè et al. (2016) 'performance'. Leman (1993) calls it 'subsymbolic'. Sloan (1993), Selfridge-Field (1997), and Roland et al. (2014) term it 'gestural'.
(c) **Logical:** A semantic interpretation of music, typically an abstraction away from its graphical domain. Huron (1992) calls it 'common musical notation', Leman (1993), Sloan (1993), and Roland et al. (2014) term it 'symbolic', Selfridge-Field (1997) 'the semantic context of musical perception and understanding', Haus and Longari (2005) 'music logic', and Baratè et al. (2016) 'logic'. Steyn (2013) terms it 'concept'.
(d) **Graphical:** The notation of music on the written page. Babbitt (1965) calls it 'graphemic', Huron (1992) 'visual notation', Sloan (1993) 'visual'. Haus and Longari (2005) and Baratè et al. (2016) term it 'notational'. Steyn (2013) terms it 'written format'.
(e) **Analytical:** The structural or theoretical analyses of music from another domain. Haus and Longari (2005) and Baratè et al. (2016) term it 'structural', Steyn (2013) terms it 'theory'.
(f) **Bibliographical:** All forms of structured or unstructured metadata. Haus and Longari (2005) and Baratè et al. (2016) call it 'general', Steyn (2013) terms it 'language'.
(g) **Meta-scores:** Huron (1992) adds a meta domain to graphical notation, consisting of notational representations that set out the processes by which a score may be created. He gives as examples 'Xenakis-like tendency masks, tables of conditional probabilities associated with information-theoretic analyses, self-similar or recursive processes, transformational-generative grammars, etc.' (Huron, 1992, p. 15).
(h) **Digital:** Huron (1992, p. 12) adds also the intermediate representation of 'sound synthesis information', being one step removed from the audial domain and including all models of physical sound, such as 'algorithms and note-lists'. Steyn (2013, p. 9) includes a similar meta-domain, that of 'markup expressions', whereby the encoding is a distinct domain within itself.

It can be concluded from this brief literature review that the taxonomy of musical domain is far from decided. Such differences emerge naturally from the varying size of the scope of what exactly constitutes 'music' in each case. However, these decisions are critical to the success of the representation: if there exists a music object within the scope of a use case of the encoding that is not adequately covered by one of its domains, then that object must be omitted, or its attributes theorised to be part of another domain through a transcription or conversion process. For example, Babbitt's (1965) three domains may appear perfectly capable within a limited encoding scope, but they become inadequate if the scope is widened to include, say, bibliographic information. Given the broad trend towards increasing the overall number of

[1] Babbitt cited Michael Kassler as the origin of this idea but did not say where. The terminology appears in Kassler (1965), but not in the same form.

domains so as to accommodate a wider view of music, it must therefore be questioned whether the current lists of musical domain are comprehensive. Either there are other as yet unknown domains of music that will in future have to be added to expand the scope of representation schemata, or our current set of musical domains must be further fragmented into a greater number of ever-shrinking regions.

Furthermore, the ways in which these domains are related must also be considered, as the manner in which they interact with one another can create structures within the encoding. One simple way of organising the domains listed above would be as an unordered set; a group of related items but without explicit mappings between them. This would be simple to implement but may not be very useful: musicologists are much more often interested in the relationships between musical domains rather than the domains themselves, such as the ways in which the graphical domain of music notation is expressed in performance, or the creation of an analysis from the logical domain.

More commonly, domains are organised in ways that make clear the theorised relationships between them according to the encoding's intrinsic music theory. When there are few domains in consideration, they can each be related to one other, as illustrated by Wiggins (2009b) (see Figure 1). However, when more domains are considered this becomes untenable, as there would be an unreasonable number of connections between all possible pairs of domains. Structure is therefore often created to systematise the domains.

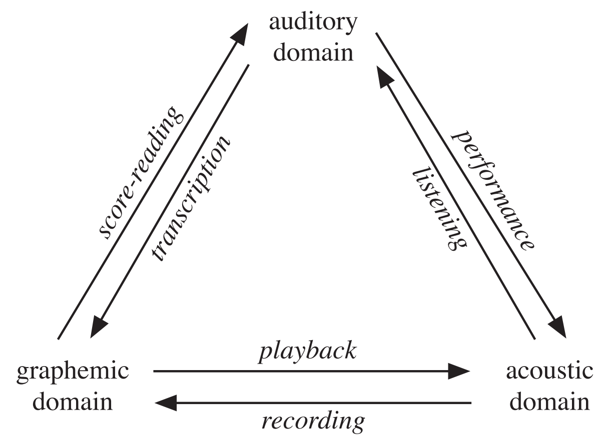

Figure 1. Wiggins' (2009b) representation of the relationships between Babbitt's (1965) three domains.

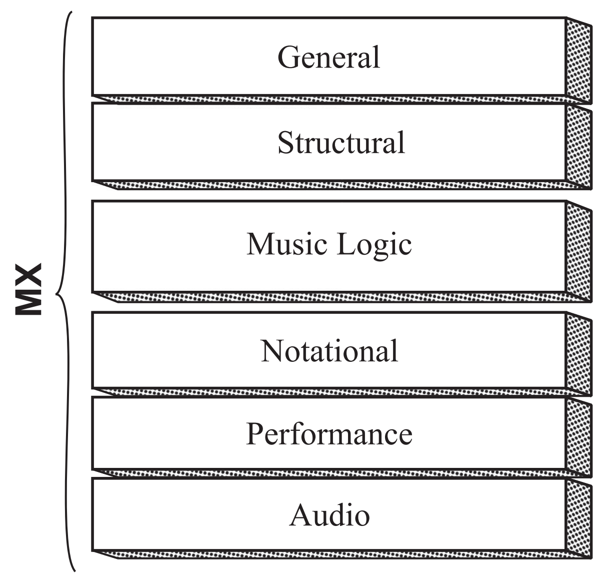

Figure 2. From Haus and Longari (2005).

A simple structure is to organise the domains linearly as a stack, as is demonstrated in Haus and Longari (2005) (Figure 2), where each successive domain is an abstraction of the last: notation is an abstraction away from performance, and musical structure an abstraction of the logical domain. For many cases this structure is sound, but can be quickly found to be far too rigid for many cases. For example, it does not allow us to describe the structure of a performance without first passing through the intermediary layers of notation and music logic. Anyone who has recognised a sonata form from its performance alone knows that these stages are often unnecessary.

This same argument against rigidity can be made in more complex cases, such as the hierarchical domain arrangement described in Lindsay and Kriechbaum (1999) (Figure 3) and the hub-and-spoke model of Huron (1992) (Figure 4), where there may be more diverse connections possible, but remaining possibilities not accounted for. For example, it is not possible in Huron's model for a meta-score to be performed without first being transcribed to CWMN. This is contrary to the intention of many graphic scores that exist precisely due to the limitations of CWMN (Chew & Rastall, 2014). These models, in linking pairs of domains together, also do not account for the distinct possibility of complex interrelationships arising, where three or more domains may need to be linked together at once.

By dividing music into a discrete set of categories with limited relationships between each, these limiting sets of domains and domain relationships create basic, but at times conflicting, theories of music. Even in the most advanced cases where there are numerous theorised domains, the list is likely not exhaustive, nor are the possibilities for the relationships between domains, except in the special case where it is possible to link all domains to all others. It is also possible to have multiple instances of the same type of domain, such as two editions of the same work. Encoding schemata should therefore aim for flexibility in the domains that are supported, as well as

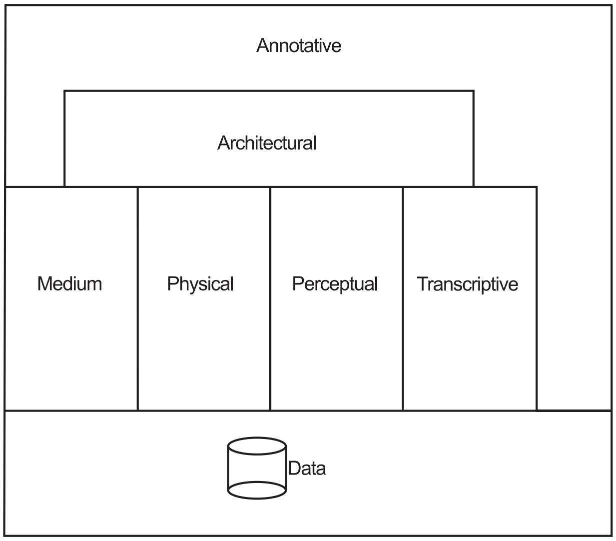

Figure 3. From Lindsay and Kriechbaum (1999).

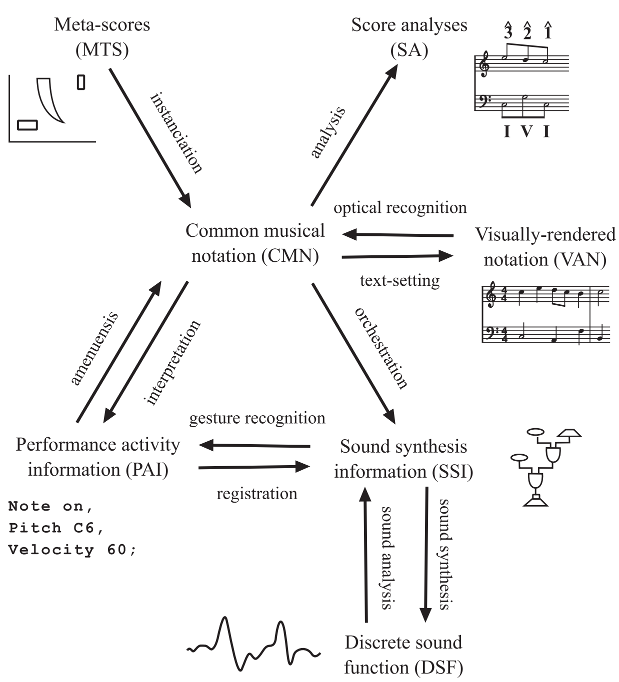

Figure 4. From Huron (1992).

the ways in which they can be related. This possibility is alluded to in a figure provided by Baratè et al. (2019) who, although limited to the six domains previously outlined in Baratè et al. (2016), demonstrate the possibility of multiple, fluid domain relationships. They illustrate not only the hierarchical and hub-and-spoke models already described, but also a more free-form graph model (Figure 5, middle example).

3. Related work

Two main strands of research have led to a critical understanding of musical representation. The first is the fundamental concept of applying the techniques of linguistics to musical analysis, namely generative grammars, and the problems that arise from this approach. Lerdahl and Jackendoff (1983) decomposed musical structure into a finite grammar whereby CWMN could be divided into phrases, bars, beats, and other structures. Two distinct problems immediately stem from this: firstly, that any music other than the simplest monophonic melodies contains an 'inherent parallelism' (Roads, 1987, p. 417), which Lerdahl and Jackendoff began to remedy by creating multiple parallel grammars to analyse both melody and bass simultaneously (Lerdahl & Jackendoff, 1983, p. 275). This is despite Lidov (1975) having already demonstrated by this time that there is always more than one way to parse musical phrases. It was also around this time that Dannenberg (1986, p. 153) began describing the accompanying need for music representations that could support 'multiple hierarchical structures'.

The second strand comes from the desire to represent music in the computer for purposes such as typesetting and analysis. Early representation systems such as DARMS mixed domains freely (Erickson, 1976), but the work of Byrd (1984) placed a key focus on the need for representation systems to separate the concerns of layout (a graphical domain) from the meaning of the symbolic notation itself (a logical domain), and broaden their horizons when modelling even CWMN. Byrd cited numerous examples in famous works from the classical canon that cause issues of encoding when using overly simplistic models of CWMN, such as two clefs simultaneously active on one staff, slurs that cross staves or even instruments, and collisions of pitches and stems. Many of these examples are still difficult, if not impossible, to encode satisfactorily using modern typesetting systems.

On the broader level, many issues related to hierarchy and domain were highlighted by Wiggins et al. (1993) who put forward a projection of the encoding landscape in the early 1990s along the twin axes of 'expressive completeness' ('the range of raw musical data that can be represented') and 'structural generality' ('the range of high-level structures that can be represented and manipulated') (Wiggins et al., 1993, p. 31). However, this work is now outdated for two reasons: firstly, very few of the systems surveyed in that paper are recognisable today, and secondly the issue of 'expressive completeness' is no longer pressing for music encoding systems. The reason for this is that older music encoding formats such as those surveyed in Wiggins et al. (1993) were constrained by being able to express only a limited subset of CWMN, or being domain-specific modelling languages for the encoding issues apparent within particular repertoires such as plainchant (Stinson & Stoessel, 2014). Fortunately, today all serious encoding formats

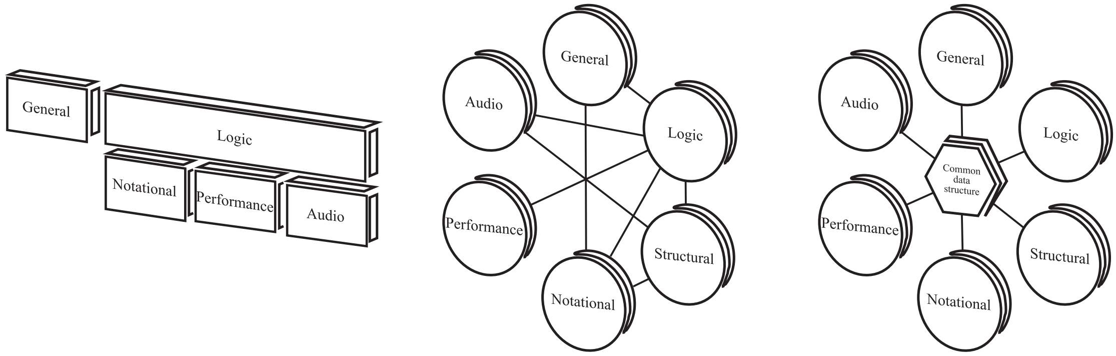

Figure 5. Adapted for clarity from Baratè et al. (2019).

for musicological research have extensibility built in as standard, and so new elements and attributes can be added to these formats with relative ease.[2] Therefore, if in our encoding we believe that there is something that causes the representation to be lacking in expressivity, it can quickly be added. However, the issue of structural generality has still not been adequately addressed and requires further work, and this is what this paper examines directly.

Wiggins (2009a, p. 19) took a more abstract and reflective view, presenting 'a sort of shopping list of desirable properties of generic music representation systems', once again reiterating the important principles for hierarchy and domain. Wiggins' list of wishes is clearly well considered but is solely 'specification' (Wiggins, 2009a, p. 9). Although such specifications are indispensable for the creation of new encoding schemata or for the improvement of current ones, they do not help to answer the question of what decisions should be made today for those engaged in creating digital encodings of music. The aim of this paper is to inform those engaged in musicological study within the digital turn on this crucial and somewhat neglected strand of digital music encoding. Although many considerations for the human and cultural aspects that play a key part in music encoding decisions can be neatly encapsulated in the FAIR and CARE principles (Neumann et al., 2024), we cannot ignore the further role of technological determinism in how that data is encoded in data structures. This paper stands at the mid level between Wiggins et al. (1993) and Wiggins (2009a), in part to update the 1990s survey of encoding data structures but also to take a more pragmatic stance on encoding. Far from being simply a technical decision, the data structure used for encoding can fundamentally alter the representation's features, priorities, and ultimately its uses.

[2] For example in *Humdrum* see Huron (2002, p. 11), in IEEE 1599 see Baggi & Haus (2013, p. 2), and in the *Music Encoding Initiative* see Crawford & Lewis (2016, p. 277).

4. Methodology

This paper will show that, rather than one data structure being 'better' than the others, the selection of any one structure for a representation format creates a trade-off between structural generality and another key axis that has not yet been seriously considered – structural readability – and that this has knock-on effects for the issues of domain and representation. Structural readability is defined here as the combination of two separate issues that contain similar requirements: firstly there is the perceived ability for a human to understand the structure and content of a representation when encoded in a text format, and secondly there is the complexity for a computer to parse and store features from the representation in memory. This is especially important when it comes to large musical data sets that may require streaming from disk rather than instant access in RAM. A data structure encoded in a representation that can be parsed over a common axis is much simpler to process than a structure that contains links to disparate parts of the encoding that may or may not yet have been parsed.

I will consider three key data structures through this lens: tables, trees (i.e. single hierarchies), and graphs (sometimes termed 'networks'). These three common formats reflect the three typical structures for data modelling more generally (Jannidis & Flanders, 2019, p. 55), and this takes an important step towards considering music data modelling within the emerging field of critical code studies (Marino, 2020). Many of these structures are apparent in commonly used music encoding formats: for example *Humdrum* can be viewed as a tabular representation (Huron, 1994, p. 10), and the *Music Encoding Initiative* (MEI) and IEEE 1599 as hierarchical trees (Crawford & Lewis, 2016, p. 275; Baggi & Haus, 2013, p. 102). However, to avoid overburdening the non-technical reader with syntactical details, this study will remain detached at the level of domain and data structure rather than the differences between particular encoding schemata and styles.

Similarly, focus is pulled away from the digital tools that are used for encoding and shifted instead to the underlying data structures employed in the encodings processed by those tools. This is because, in modern digital musicological study, digital editing tools are typically created in order to process already-existing representation formats more efficiently, rather than representations created to accommodate tools, and music is still commonly encoded and edited by hand for accuracy. For example, in MEI the *mei-friend* graphical tool aims mostly to 'alleviate the difficulty of learning and working with MEI' (Goebl & Weigl, 2024, p. 2), and the typesetting library *Verovio* closely models MEI's structure in its SVG output (Pugin et al., 2014). The issues of data structure discussed here are therefore fundamental to the concerns addressed by such digital editing tools as, rather than defining new structures, digital editing tools serve mostly as methods of enhancing productivity in managing, organising, and manipulating existing data structures.

These three structures are defined here through the classification between two types of link that each element within the structure affords. A first-class link I will define as an implicit link that is fundamentally part of the data structure's operation: the elements are directly connected to each other through either adjacency or by a precise syntactical structure that could be described by a context-free grammar. For example, two elements appearing next to each other in a file are described as adjacent, and a set of elements enclosed by an S-expression or XML tag are grammatical. Conversely, a second-class link is explicit in saying what it links to and how the two items are related: a URI or link to a unique ID within a larger system would be an example of this.

Through the axes of structural generality and structural readability, and by a focus on first- and second-class links, each of these three data structures will be critiqued on their ability to support multiple domains or modes of representation, i.e. how easy is it within these data structures to add new representations or domains to the encoding? As demonstrated above, the separation of domains is often considered key to creating accurate music representations, and the intersections between domains are most often the foci of musicological study. It follows therefore that we must ask not just how the domains themselves can be represented but also how flexibly those domains can be processed together in order to study their relationships.

5. Tables

Using this study's vocabulary of links, a tabular data structure is one that has three first-class links from a specific event: the previous event in the stream (such as the previous note in time), the next event, and the set of events in other representations that are concurrent with that event (the order of simultaneous events is assumed not to matter). This structure is tabular as it can be visualised as a table where the columns constitute parallel representations of musical data such as individual voices, and the rows a series of time instants. Indeed, music has been modelled this way using spreadsheet software (Moll, 1995). Such a simple layout is arguably the greatest strength of tabular systems: the two-dimensional arrangement reflects at once the quasi-xy coordinate system of a CWMN score (Buxton et al., 1985), and the data can therefore be easily displayed and edited on a computer screen purely through the presentation of its first-class links.

Table 2 encodes the first phrase of the Bach chorale in Figure 6 as a hypothetical tabular format. Each column of soprano, alto, tenor, and bass contain data points of musical events occurring at each vertical sonority, and each row is a discrete time slice. Starting at one event, it is very easy to traverse the data structure to find all other events happening at that instant (the row), and all of that representation's other events (the column). More generally, it could be argued that each voice should in fact consist of two columns: one for pitch and one for duration, or perhaps even further splitting each pitch into octave, note name, and chromatic inflection. However, this would distribute the data for each voice across multiple columns, and split the sense of the musical 'event' into simultaneous pitch and duration events.

The final chord of the phrase has a fermata attached to it, so we must add another Boolean column indicating whether that instant has a fermata or not. The reason why we cannot add this information only to the last row and must add a new column in this instance is because in a tabular format we should keep our datatypes uniform: to add an optional 'fermata' flag to each pitch would be to begin to construct a rudimentary grammar within each column (i.e. a hierarchy). Such a grammar would then also make it more difficult to query the system to find all fermatas. The first issue that emerges with a tabular structure, then, is that as the number of features we want to capture increases, new columns must be added (e.g. barring, phrasing, clefs, etc.). The data structure must always maintain a one-to-one relationship between instants and representations.

A second issue is exemplified by the records that read 'null'. A Bach chorale was chosen as it is perhaps the archetypal case for dividing music vertically into coincident sonorities and is therefore perhaps the fairest example to tabular representations. However, in all but the simplest cases, there are complex points of movement

Table 2. Bach chorale encoded as a table.

Row number	Slice length	Soprano	Alto	Tenor	Bass	Has fermata?
1	♪	A4 ♩	F4 ♩	D4 ♩	D4 ♪	False
2	♪	null	null	null	C4 ♪	False
3	♪	D5 ♩	F4 ♪	D4 ♪	B♭3 ♩	False
4	♪	null	G4 ♪	E4 ♪	null	False
5	♩	C5 ♩	A4 ♩	F4 ♩	A3 ♩	False
6	♪	B♭4 ♩	D4 ♪	G4 ♩	G3 ♩	False
7	♪	null	E4 ♪	null	null	False
8	♩	A4 ♩	F4 ♩	C4 ♩	F3 ♩	False
9	♩	G4 𝅗𝅥	F4 ♩	D4 ♩	B♭2 ♩	False
10	♩	null	E4 ♩	C4 ♩	C3 ♩	False
11	♩	A4 ♩	F4 ♩	C4 ♩	F3 ♩	True

Figure 6. First phrase of *O Haupt voll Blut und Wunden*, BWV 244/54.

such as passing notes and suspensions. Once again, merging cells would create a form of grammar, so that all that can be said about the soprano at row 10 is that there is no event beginning there. To understand what note is currently sounding in the soprano at that point, we must go back a row to see that there is a minim.

Far from being free of grammar, there are actually two grammars at work here: the first is that to understand 'true' vertical sonority and not just the events that begin at that instant, there must always be some traversal of the data structure; the second is that the number of 'null' spaces inserted must exactly match the duration of the initial note and length of each slice. For example, replacing the soprano minim in row 9 with a semibreve would be a grammatical error. As can be seen by the added column 'Slice length', each row does not map to a fixed duration. The rate at which we must parse the rows of the table depends on the busyness of the texture: lots of passing notes and syncopations will cause many unique vertical sonorities to be created, and therefore many rows. Such structures are therefore weak at representing complex counterpoint and polyrhythms.

6. Trees

A tree data structure is defined here as one where every musical event can be decomposed into a hierarchy according to a formal musical grammar specifying musical structure. For example, a score may be divided into instruments and voices, then into bars, beats, and notes. Such hierarchical arrangements are today perhaps the most commonly used structures for music encoding. In these grammars, it is often understood that unless otherwise specified by a grammatical construct indicating simultaneity, adjacent elements in the encoding will follow each other in time. From a single event, there are first-class type links to sub-events (children), and a second-class link to the parent event (as this link requires the entire structure to be parsed first). The Bach chorale example can be more easily encoded this way using a hypothetical tree encoding. This particular example makes no grammatical assumptions regarding which events are sequential and which are simultaneous, but treats both cases as explicit syntaxes.[3] It is important to note, however, that many encoding schemata do not take this into account.

```
(Score
  (Attributes
    (Key F Major)
    (Time C)
  )
  (GrandStaff
    (Staff
      (SimultaneousMusic
         (Voice "Soprano"
           (SequentialMusic
             (Bar
               (SequentialMusic
                 (Note (Pitch A 4) (Duration
                   Crotchet))
               )
             )
             (Bar
               (SequentialMusic
                 (Note (Pitch D 5) (Duration
                   Crotchet))
                 (Note (Pitch C 5) (Duration
                   Crotchet))
                 (Note (Pitch B 4 Flat) (Duration
                   Crotchet))
                 (Note (Pitch A 4) (Duration
                   Crotchet))
               )
```

[3] In this way, and also in terms of syntax, the following hypothetical example shares many similarities with the internal *Lisp* representation of the typesetting software package *LilyPond* (LilyPond, 2022).

```
            )
            (Bar
              (SequentialMusic
                (Note (Pitch G 4) (Duration
                    Minim))
                (Note
                   (Pitch A 4) (Duration
                     Crotchet)
                   (Additions Fermata)
                )
              )
            )
          )
        )
        (Voice "Alto"
          ...
        )
      )
    )
    (Staff
      (SimultaneousMusic
        (Voice "Tenor"


          ...
        )
        (Voice "Bass"
          ...
        )
      )
    )
  )
)
```

There are many immediate advantages to such a structure. The relationships between elements are not so constrained as in the tabular structure. A first-class link can mean adjacency or concurrency according to the grammatical construct the items are enclosed within. The structure is also much denser: there is no need for empty records as each element can have any number of children. Features need not be present on every element, but only where they are necessary: for example the fermata element is only instantiated for the note that has the original symbol, all the other notes are understood by the grammar not to have a fermata attached unless otherwise instructed.

The first difficulty comes with continuous features that do not respect the hierarchy that has already been encoded. For example, ties are most useful across bar lines, but it is not immediately obvious how to target one note at the end of one bar and another note later, as the tie is a construct attached to a note but must escape the enclosure of the bar in the hierarchy to reach its end point. The same is true for all lines in CWMN such as phrase marks, gradual dynamic changes such as hairpins, and beaming. For an even more extreme example, we can consider one of the examples from Byrd (1984, p. 45): the cross-instrument slurs from the opening of Alban Berg's *Violin concerto* (Figure 7). The true meaning of these slurs aside, in our hypothetical tree encoding the two end points of these slurs would be nested far inside

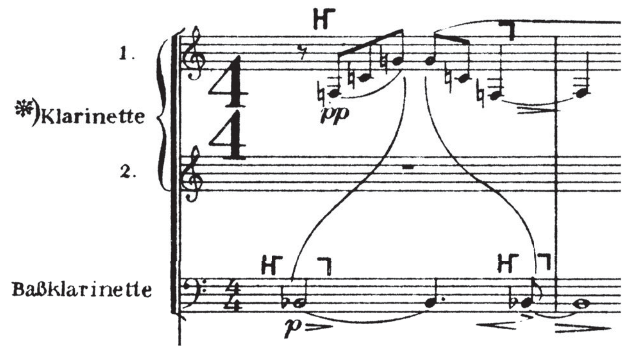

Figure 7. Extract from Berg, *Violin concerto*, b. 1 (Berg, 1936).

other elements, and so it is impossible to make a first-class link between the two. The intrinsic grammar of the hierarchy has prioritised time and instrumentation over other concerns such as phrasing.

These complications require the single hierarchy formed of purely first-class links to be broken, and second-class links added. Each element could contain an attribute indicating a second-class link of type 'slur' to another note. However, this would transform the single hierarchy into a graph structure, as it would be possible to now reach these two notes from two different directions: both from the typical descent of the syntax tree in the first-class links, and also by following the second-class link of the slur. This is a non-trivial alteration, as the loosening of the syntax causes the data structure to become much more difficult to parse and read. Rather than being able to be parsed simply in file order, a second-class link may require the parsing of another portion of the file before that element can be fully parsed, and cycles can be created, requiring each element's state to be held in memory.

Nonetheless, the first-class hierarchy retains the dominant logical domain links within this data structure, privileging the typical division of music into voice and time, and sidelining elements that do not respect this taxonomy, such as slurs. This choice of dividing first by voice then by time is arbitrary: the encoding could equally have decided to split by time first, then by voice.[4] By design, single hierarchies privilege one division of music above all others, and in this case the aspects of time and part have been placed as more important than other features. However, there is no inherent reason why this should always be the case. For example, we could equally design an equivalent encoding that privileged the representation of pitch classes in first-class links, and used second-class links for time and part attributes. Indeed, such a flipping

[4] *MusicXML* does precisely this, having two equivalent representations: one that is split first by voice (< `score-partwise` >), and another split first by time (< `score-timewise` >) (Good, 2021).

of the hierarchy could well be useful for some forms of analysis.

Both tables and trees create rigid domain representations notwithstanding, where one principal view, typically of the logical domain, is placed at the centre of the encoding much like the hierarchical or hub-and-spoke models shown in Figures 4 and 5, and attributes from other domains must be tagged using second-class links. This makes it easy for a system to query aspects of the central privileged domain, but more difficult to extract data from the other parts of the structure, enhancing readability for the central domain at the expense of structural flexibility.

7. Graphs

To escape the inherent constraints and biases of table and tree structures, a more equitable representation can be created by using a graph data structure. In this structure, there are no natural first-class links, and the structure consists solely of a uniform surface of second-class links between elements, creating an ontology. As a result, any element can link to any number of other elements, and each of these links are treated equally: there is no distinction made between common relationships such as sequence, and other more esoteric links such as complex set membership rules. Whereas the tree structure privileged one set of domain relationships above others – typically within the logical domain – the representation of domains in graph relationships are all considered as a single set. This is not to say that graph structures are incapable of creating hierarchies: the tree structure could be represented as a hierarchical graph, and the tabular structure as a lattice graph. However, unlike the previous structures considered, the flexibility of a graph structure does not force the data into a particular model.

Figure 8 demonstrates how two notes from our Bach chorale example could be encoded as part of a wider graph structure. The advantages of such a structure are clear in the variety of topologies that can be created: this model cannot be represented either as a table or hierarchical structure. For example, the 'Crotchet' node has two parents: both of the 'Note' nodes. This representation would be impossible in a hierarchical model as each element must have one parent only. We could imagine other notes that are not crotchets that would have other duration links. In a tabular or hierarchical structure, if we wanted to find all crotchets – and our hierarchy did not privilege this relationship through first-class links – we would have to scan through the entire structure, picking out all crotchet instances. However, in this hypothetical graph ontology, there is only one Platonic ideal of a crotchet duration, and all notes that have that duration must link to it. Therefore, finding all crotchets is simply a process of fetching the nodes that link to the 'Crotchet' node. Byrd's example of the cross-instrument slur could be encoded just as easily as a slur to another note in the same instrument, or one note following another.

The trade-off with such a structure is in its readability. The table and tree data structures took advantage of the easily read and comprehended features of rows and columns of text files. The table data structure could

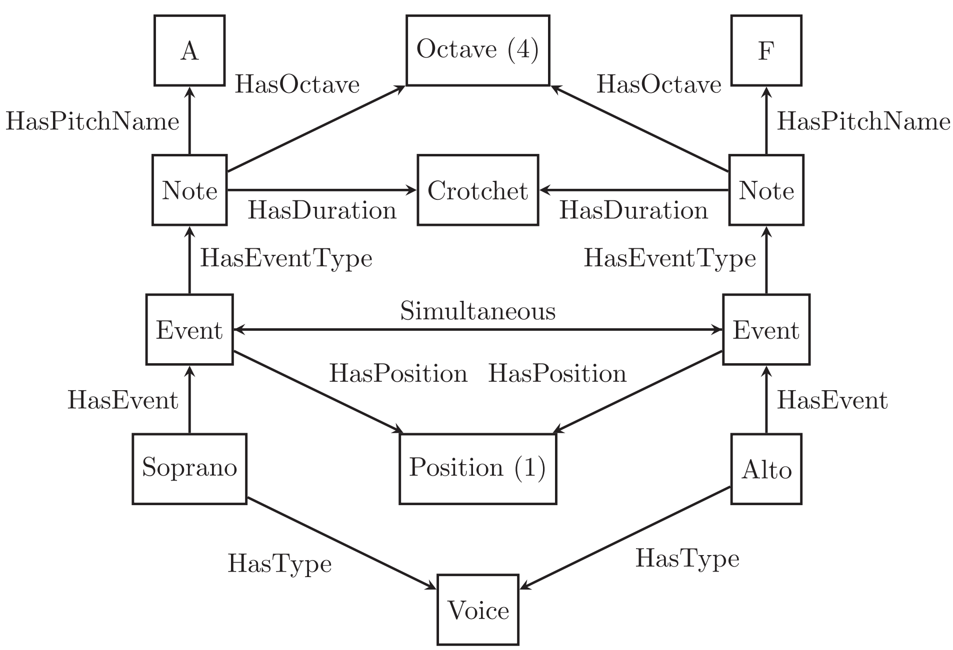

Figure 8. Hypothetical network of two notes in a graph representation.

be characterised as a graphical layout, where whitespace such as carriage returns and indentation have semantic meaning. Similarly, although many hierarchical text formats do not rely on whitespace, they commonly use indentation to make their structure clear (Goldfarb, 1990, p. 200). In contrast to the tangled skein of connections in a graph structure, tabular and tree structures are eminently human readable and simple to parse line-by-line in a computer, but this comes at the expense of structural inflexibility and the imposition of hierarchy. In the encoding of a graph structure, the opposite is true: the structure can be much more fluid, but cannot take advantage of the layout of text files (and it is particularly telling that Figure 8 managed to encode only two notes of two voices in the same space that Table 2 encoded the entire phrase in all four parts). In the file itself, the data model of Figure 8 must be encoded by an out-of-order textual representation along the lines of:

```
Soprano HasType Voice
Alto HasType Voice
Soprano HasEvent EventA
Alto HasEvent EventB
EventA HasPosition 1
EventB HasPosition 1
EventA Simultaneous EventB
EventB Simultaneous EventA
EventA HasEventType NoteA
EventB HasEventType NoteB
NoteA HasDuration Crotchet
NoteB HasDuration Crotchet
NoteA HasOctave 4
NoteB HasOctave 4
NoteA HasPitchName A
NoteB HasPitchName F
```

As the graph becomes more complex, following the layout of the graph becomes more difficult, as each link could jump to any other place in the structure. This is not a structure that is easily encoded or parsed, as the lack of first-class links necessitates all relationships to be made explicitly, and the entire graph must be parsed first in order to understand it correctly. This can very easily become overwhelming for a user who is trying to understand or edit the content of the encoding, but it is also more difficult for a computer to parse and then visualise in a user interface.

Due to the challenges associated with encoding and parsing complex graph structures, there are remarkably few encoding schemata that commit to this form, despite its potential for multiple representative hierarchies and domains, as graph representations often require specialised tooling and user training. Where they are used, graph representations are most frequently introduced as ad hoc solutions to other tasks, and music is rarely encoded as a graph directly. For example, Karystinaios and Widmer (2022) use a graph structure to perform cadence detection, but the resulting graphs are simply an 'intermediate means' rather than an independent representation (Karystinaios & Widmer, 2022, p. 917), and the scores are converted directly from *Humdrum*, that is, a tabular representation. As the genesis of this data is tabular and the conversion direct, there is nothing encoded in the graph structure that was not already present in the tabular representation. Nonetheless, text-based encoding systems gain a significant bootstrapping effect for tabular and tree representations, as these structures rarely require any specific tooling and can be edited more easily using text editors. For example, it is far simpler for a user to begin encoding using a particular schema if certain domains in the music can be readily understood by the layout of the text file.

8. Limitations and future work

One of the reasons that the preceding discussion has focused on the abstract concept of data structures is that real-world encoding schemata rarely adhere to just one of these paradigms, but use one of the above models as a starting point from which to create musical structure. For example, although a format such as *Humdrum* can broadly be aligned with a tabular data structure, it introduces hierarchy through the grammar of its `**kern` representation (Huron, 1994, pp. 96–106), and a graph structure in `!!!RDF` fields (Huron, 1994, p. 34). However, mixing models this way does not help the cause of structural generality highlighted by Wiggins et al. (1993). End users – both human readers as well as programmers writing code to parse these structures – are more likely to rely on the domain attributes most accessible in first-class links (typically the logical domain) over others (such as structural, performative, and analytical domain representations). Although an encoding standard may include extrinsic aspects of other models, the predominant model of the format will be the primary structure through which the representation is viewed. This may be precisely as intended, but all encodings are created with a purpose that should be critically examined; encodings created for one project are not automatically suitable for another. Not only may these other encodings not capture the requisite data, but the data structure they are encoded in may create a forced perspective on the musical data by privileging one domain above all others. This is not to say that encodings created within a certain schema and using a certain data structure should be automatically considered defunct, but that the requirements of those older encodings should be critically examined before being imported wholesale into new projects. Further work is needed to create frameworks and processes capable of evaluating the particular perspectives and biases of digital music encodings at the individual level rather than the relative advantages and disadvantages of schemata more generally.

This paper has also only focused on prescriptive representations: encoding music artefacts as static and immutable documentation. However, purely prescriptive representations are rare in music: even CWMN must be placed within a context of description where performers understand how to execute common descriptive features (Ferand, 1961). Prescriptive notation can also be analysed in a descriptive way: Antoni and Haus (1982) modelled the *Canon perpetuus* from J.S. Bach's *Musical Offering* as a Petri net, and Hudak and Quick (2018, pp. 67–72) demonstrated the capabilities of their Haskell-based music description language by encoding Chick Corea's *Children's Song No.6* as a series of programmatic ostinati. Further work should therefore concentrate on how to reconcile both the prescriptive and descriptive strategies, as early music and modern music in particular both rely heavily on descriptive graphical notations outside of the realm of CWMN, as well as complex questions that concern the interaction between domains.

9. Conclusions

This paper has demonstrated that none of the three typical data structures in use today are wholly capable of representing all domains of music adequately but, when confronted with the richness of possible musical expression, must limit their concerns to a trade-off between privileging structural generality or structural readability. It is unlikely that the scale and scope of domains listed in literature is exhaustive, as it is philosophically difficult to limit what constitutes musical experience into a comprehensive taxonomy. By privileging certain musical domains above others, the structure of encodings can become more readable by humans and parseable by computers, but they lose flexibility in the breadth and variety of knowledge that they can represent. However, a purer and more neutral structure such as a graph that does not privilege any links above any others can easily become excessively complex, overwhelming for users that are aiming to understand the structure directly, as well as more difficult to parse and store.

Therefore, those involved with creating digital editions of music should carefully examine the data structures that they are creating through their chosen encoding format or importing from elsewhere, and weigh up this generality-readability trade-off, considering exactly what musical data they wish to capture and what should be left out. Domain analysis should form a key part of this process in order to ascertain which, if any, musical domains are being privileged as part of the encoding process, and therefore the biases towards certain domains that may be implicitly encoded.

Disclosure statement

No potential conflict of interest was reported by the author(s).

References

Antoni, G. D., & Haus, G. (1982). Music and causality. In *Proceedings of the 1982 International Computer Music Conference* (pp. 279–296). International Computer Music Association. https://hdl.handle.net/2027/spo.bbp2372.1982.025

Babbitt, M. (1965). The use of computers in musicological research. *Perspectives of New Music*, *3*(2), 74–83. https://doi.org/10.2307/832505

Baggi, D. L., & Haus, G. M. (2013). *Music navigation with symbols and layers: Toward content browsing with IEEE 1599 XML encoding*. Wiley. https://doi.org/10.1002/9781118494455

Baratè, A., Haus, G., & Ludovico, L. A. (2016). A critical review of the IEEE 1599 standard. *Computer Standards & Interfaces*, *46*, 46–51. https://doi.org/10.1016/j.csi.2016.02.001

Baratè, A., Haus, G., & Ludovico, L. A. (2019). State of the art and perspectives in multi-layer formats for music representation. In *2019 International Workshop on Multilayer Music Representation and Processing* (pp. 27–34). IEEE. https://doi.org/10.1109/MMRP.2019.00014

Berg, A. (1936). *Violinkonzert*. Universal Edition.

Blasius, L. (2002). Mapping the terrain. In T. Christensen (Ed.), *The Cambridge history of western music theory* (pp. 27–45). Cambridge University Press. https://doi.org/10.1017/chol9780521623711.003

Buxton, W., Sniderman, R., Reeves, W., Patel, S., & Baecker, R. (1985). The evolution of the SSSP score-editing tools. In C. Roads & J. Strawn (Eds.), *Foundations of computer music* (pp. 376–402). The MIT Press.

Byrd, D. A. (1984). *Music notation by computer* [Unpublished doctoral dissertation]. Indiana University.

Chew, G., & Rastall, R. (2014). Notation §III. History of western notation, 6. Non-mensural and specialist notations. In *Grove music online*. https://doi.org/10.1093/gmo/9781561592630.article.20114

Cook, N. (2002). Epistemologies of music theory. In T. Christensen (Ed.), *The Cambridge history of Western music theory* (pp. 78–105). Cambridge University Press. https://doi.org/10.1017/chol9780521623711.005

Crawford, T., & Lewis, R. (2016). Review: Music encoding initiative. *Journal of the American Musicological Society*, *69*(1), 273–285. https://doi.org/10.1525/jams.2016.69.1.273

Cross, I., & Tolbert, E. (2021). Epistemologies. In T. McAuley, N. Nielsen, & J. Levinson (Eds.), *The Oxford handbook of Western music and philosophy* (pp. 265–282). Oxford University Press. https://doi.org/10.1093/oxfordhb/9780199367313.013.14

Dannenberg, R. B. (1986). A structure for representing, displaying, and editing music. In *Proceedings of the 1986 International Computer Music Conference* (pp. 153–160). International Computer Music Association. https://hdl.handle.net/2027/spo.bbp2372.1986.031

Dannenberg, R. B. (1989). Music representation issues: A position paper. In *Proceedings of the Fifteenth International Computer Music Conference* (pp. 73–75). International Computer Music Association. https://hdl.handle.net/2027/spo.bbp2372.1989.018
Erickson, R. (1976). *DARMS: A reference manual* [Computer software manual]. Retrieved December 12, 2023, from https://esf.ccarh.org/ccarh-wiki/DARMS-1976.pdf
Ferand, E. T. (1961). *Improvisation in nine centuries of western music: An anthology with a historical introduction*. Arno Volk Verlag.
Goebl, W., & Weigl, D. M. (2024). mei-friend: An interactive web-based editor for digital music encodings. *The Journal of Open Source Software*, *9*(98), 6002. https://doi.org/10.21105/joss.06002
Goldfarb, C. F. (1990). *The SGML handbook* (Y. Rubinsky, Ed.). Clarendon Press.
Good, M. (2021). *MusicXML 4.0: Final community group report* [Computer software manual]. Retrieved December 12, 2023, from https://www.w3.org/2021/06/musicxml40/
Goodman, N. (1968). *Languages of art: An approach to a theory of symbols*. The Bobbs-Merrill Company, Inc.
Haus, G., & Longari, M. (2005). A multi-layered, time-based music description approach based on XML. *Computer Music Journal*, *29*(1), 70–85. https://doi.org/10.1162/comj.2005.29.1.70
Hudak, P., & Quick, D. (2018). *The Haskell school of music: From signals to symphonies*. Cambridge University Press.
Huron, D. (1992). Design principles in computer-based music representation. In A. Marsden & A. Pople (Eds.), *Computer representations and models in music* (pp. 5–39). Academic Press.
Huron, D. (1994). *UNIX tools for musical research: The Humdrum toolkit* [Computer software manual]. Retrieved December 12, 2023, from https://ccarh.org/publications/manuals/humdrumreference/humdrumreference-1995.pdf
Huron, D. (2002). Music information processing using the Humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, *26*(2), 11–26. https://doi.org/10.1162/014892602760137158
Jannidis, F., & Flanders, J. (2019). A gentle introduction to data modeling. In J. Flanders & F. Jannidis (Eds.), *The shape of data in digital humanities: Modeling texts and text-based resources* (pp. 26–95). Routledge. https://doi.org/10.4324/9781315552941-2
Kania, A. (2008). The methodology of musical ontology: Descriptivism and its implications. *British Journal of Aesthetics*, *48*(4), 426–444. https://doi.org/10.1093/aesthj/ayn034
Kania, A. (2023). The philosophy of music. In E. N. Zalta & U. Nodelman (Eds.), *The Stanford encyclopedia of philosophy* (Spring 2023 ed.). Metaphysics Research Lab, Stanford University. Retrieved December 12, 2023, from https://plato.stanford.edu/archives/spr2023/entries/music/
Karystinaios, E., & Widmer, G. (2022). Cadence detection in symbolic classical music using graph neural networks. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference* (pp. 917–924). International Society for Music Information Retrieval. https://archives.ismir.net/ismir2022/paper/000110.pdf
Kassler, M. (1965). A sketch of the use of formalized languages for the assertion of music. *Perspectives of New Music*, *1*(2), 83–94. https://doi.org/10.2307/832106
Leman, M. (1993). Symbolic and subsymbolic description of music. In G. Haus (Ed.), *Music processing* (pp. 119–164). Oxford University Press.
Lerdahl, F., & Jackendoff, R. (1983). *A generative theory of tonal music*. The MIT Press.
Lidov, D. (1975). *On musical phrase*. York University.
LilyPond (2022). *LilyPond 2.24: Internals reference* [Computer software manual]. Retrieved December 12, 2023, from https://lilypond.org/doc/v2.24/Documentation/internals/index
Lindsay, A., & Kriechbaum, W. (1999). There's more than one way to hear it: Multiple representations of music in MPEG-7. *Journal of New Music Research*, *28*(4), 364–372. https://doi.org/10.1076/0929-8215(199912)28:04;1-o;ft364
Marino, M. C. (2020). *Critical code studies*. The MIT Press. https://doi.org/10.7551/mitpress/12122.001.0001
Matheson, C., & Caplan, B. (2011). Ontology. In T. Gracyk & A. Kania (Eds.), *The Routledge companion to philosophy and music* (pp. 37–47). Routledge.
Maxwell, J. T. (1981). *Mockingbird: An interactive composer's aid* [Unpublished master's thesis]. Massachusetts Institute of Technology.
McLuhan, M. (1964). *Understanding media: The extensions of man*. McGraw-Hill Book Company.
Moll, K. N. (1995). Vertical sonorities in renaissance polyphony: A music-analytic application of spreadsheet software. *Computing in Musicology*, *10*, 59–78.
Neumann, J., Richts-Matthaei, K., & Beer, N. (2024). Toward a community model of scholarly editing: FAIR/CARE, research ethics, & labour visbility. *Journal of New Music Research*, 1–19. https://doi.org/10.1080/09298215.2024.2373049
Pace, I. (2009). Notation, time and the performer's relationship to the score in contemporary music. In D. Crispin (Ed.), *Unfolding time: Studies in temporality in twentieth-century music* (pp. 149–191). Leuven University Press. https://doi.org/10.2307/j.ctt9qdxmr.8
Pazel, D. P. (2022). *Music representation and transformation in software: Structure and algorithms in Python*. Springer. https://doi.org/10.1007/978-3-030-97472-5
Pesce, D. (2011). Theory and notation. In M. Everist (Ed.), *The Cambridge companion to medieval music* (pp. 276–290). Cambridge University Press. https://doi.org/10.1017/ccol9780521846196.018
Pugin, L. (2015). The challenge of data in digital musicology. *Frontiers in Digital Humanities*, *2*, 1–3. https://doi.org/10.3389/fdigh.2015.00004
Pugin, L., Zitellini, R., & Roland, P. (2014). Verovio: A library for engraving MEI music notation into SVG. In *Proceedings of the 15th International Society for Music Information Retrieval Conference* (pp. 107–112). International Society for Music Information Retrieval. https://archives.ismir.net/ismir2014/paper/000221.pdf
Ridley, A. (2003). Against musical ontology. *The Journal of Philosophy*, *100*(4), 203–220. https://doi.org/10.5840/jphil200310046
Roads, C. (1987). Grammars as representations for music. In C. Roads & J. Strawn (Eds.), *Foundations of computer music* (pp. 403–442). The MIT Press.
Roland, P., Hankinson, A., & Pugin, L. (2014). Early music and the music encoding initiative. *Early Music*, *42*(4), 605–611. https://doi.org/10.1093/em/cau098

Selfridge-Field, E. (1997). *Beyond MIDI: The handbook of musical codes*. The MIT Press.

Sloan, D. (1993). Aspects of music representation in HyTime/SMDL. *Computer Music Journal*, *17*(4), 51–59. https://doi.org/10.2307/3680544

Steyn, J. (2013). *Structuring music through markup language*. Information Science Reference.

Stinson, J., & Stoessel, J. (2014). Encoding medieval music notation for research. *Early Music*, *42*(4), 613–617. https://doi.org/10.1093/em/cau093

Wiggins, G. (2009a). Computer representation of music in the research environment. In T. Crawford & L. Gibson (Eds.), *Modern methods for musicology: Prospects, proposals, and realities* (pp. 7–22). Ashgate.

Wiggins, G. (2009b). Semantic gap?? Schemantic schmap!! Methodological considerations in the scientific study of music. In *Proceedings of the 11th IEEE International Symposium on Multimedia* (pp. 477–482). IEEE. https://doi.org/10.1109/ISM.2009.36

Wiggins, G., Miranda, E., Smaill, A., & Harris, M. (1993). A framework for the evaluation of music representation systems. *Computer Music Journal*, *17*(3), 31–42. https://doi.org/10.2307/3680941