Narratives and exploration in a musicology app: Supporting scholarly argument with the Lohengrin TimeMachine

David Lewis
Kevin R. Page
david.lewis@oerc.ox.ac.uk
kevin.page@oerc.ox.ac.uk
Oxford e-Research Centre, University of Oxford
Oxford, UK

Laurence Dreyfus
Faculty of Music, University of Oxford
Oxford, UK
laurence.dreyfus@magd.ox.ac.uk

ABSTRACT

We present the Lohengrin TimeMachine web application, consisting of video and textual musicological essays supported by an interactive digital companion. The digital companion allows a user to browse and compare all the occurrences of a motive in the opera Lohengrin, viewing them by text, vocal score and orchestration, with detailed views, segment labelling, audio excerpts and textual commentaries supporting the exploration. The video and essay modes show live links into the companion as the viewer or reader progresses through the narrative. This application is built on Linked Data technology and demonstrates the viability of such an approach, with the knowledge graph being traversed in the user’s browser to gather the materials for display. It uses the Music Encoding and Linked Data (MELD) framework, which provides the basis for a range of music-related Linked Data applications.

In this paper, we describe and illustrate the application in use, its technological underpinnings, as well as the motivation and implementation experience.

KEYWORDS

digital musicology, linked data, web applications, music encoding

1 INTRODUCTION

Musicology is often presented in linear, textual form, even though its materials are complex and multimodal. Scholarship must necessarily simplify and dramatically reduce complexity in order to make an ordered prose argument that others can follow. The process of producing these more constrained narratives is a central part of scholarly communication. The ability to step from such texts into a more detailed interaction with the scholar’s observations, and their intellectual and evidential bases, is also an important research skill.

In the digital realm, it becomes possible to bring these two sides closer together – the directed, expert narrative can be made available alongside its marshalled evidential materials. This supports a richer experience of scholarly narrative, but also the freedom to explore counter-narratives and alternative paths through the information provided.

In this paper, we introduce an application that is intended for just this purpose. We present the Lohengrin TimeMachine, a web application in which two pieces of linear musicological scholarship – a textual essay and a 30-minute video – are augmented by a digital companion that supports user-driven interactive, multiple-path exploration of related material. This material, which includes musical notation, audio, analytical content, and textual quotation, can be explored with or independently from the narrative.

The application is intended to be accessible to enthusiastic amateurs as well as scholars, and concerns Richard Wagner’s use of motives in his early opera, Lohengrin, advocating a more sophisticated understanding than Leitmotiv guides often communicate. It is optimised for use with a tablet, without excluding other means of browsing. The musicological content was conceived, written and, for the video, presented by co-author Laurence Dreyfus, a Wagner specialist, with the companion realised through a multi-disciplinary collaboration.

The application combines and presents web-published resources connected using Linked Data, maximising the ability for rich information to be reused by different parts of the system or, indeed, for other applications seeking to use the same material. The app thereby also serves as a practical demonstration of the strength and viability of building user-facing applications directly upon web-published Linked Data.

Central to the value of this digital companion is the combination of rich resources with an expert guide through them. The richer and more complex the data that we make available, the more helpful narrative structures become, particularly as we begin an investigation. These structures act as golden threads leading us through the complexity, giving perspectives and priorities for future exploration. We believe it is a necessary consequence of this mix that musicological, technical, and design expertise are all required for a successful result, leading inevitably to a significant joint endeavour.

Improving software libraries and tooling can greatly reduce the time and effort required, but not remove them – instead we should...
look to software to maximally convey the valuable knowledge and skills encapsulated in the digital resource.

2 BACKGROUND AND MOTIVATION

Although hypertext is a well-established concept, and HTML and related technologies have long been capable of supporting rich multimedia applications, they have relatively rarely been used to support scholarly musical narratives. When they have been used, they are usually bespoke applications with little reuse possible of code or data. In library and museum contexts, web content is usually produced by content management systems, enriched with plug-ins for browsing digitised material from the organisation’s collection. For example, the British Library system, SiteCore, powers sites such as Discovering Literature and Discovering Music. These are sites with an educational purpose, which are navigable by multiple paths and can be explored by topics (such as those required by UK schools’ curricula). Such sites are hypertextual in nature, but the articles within them remain linear texts, even though augmented with media plugins where figures might be, including page turning and zooming of images, viewing of videos and so on.

At the other end of the spectrum, less narrative-led, more exploratory digital library publishing systems such as Greenstone[1] can also provide powerful ways of search and discovery in large digital collections. A rich, music-specific browsing experience has also been realised using Open Linked Data as part of the DoReMus project, bringing together media, and music-analytical and historical information[6].

Going below the level of whole works and metadata, to incorporate musical extracts into the hyperstructure can produce richer hypermedia[9], but integration can be a challenge in terms of connecting temporal sub-units[3]. Often a spine is created in terms of clock ticks or absolute time slices (e.g. [11]) or, for music notation, spatial co-ordinates ([8]), while a more structural approach has been used for HyTime[2] and the Synchronized Multimedia Integration Language (SMIL)t. Once resources are linked into a hypermedia publication, many applications become possible – as does the possibility of building multiple very different applications on the same hyperlinked data. Whether for want of commercial interest or lack of resourcing, compelling examples are rare (although, see [10] for some exceptions). Some attention has been given to the specifics of making musical extracts web accessible[12], but much modelling work remains to be done.

3 USING THE APPLICATION INTERFACE

The Lohengrin TimeMachine digital companion is presented as a multimedia web application, optimised for tablet – specifically an iPad Pro – but fully functional on a desktop machine with mouse control. Using the application can be divided into two types of activity: (i) following a provided musico-logical narrative and (ii) exploring a digital companion to those narratives. In the sections that follow, we refer to these as narrative-led and exploratory modes.

3.1 Narrative-led modes

We provide two modes of narrative-led interaction – a textual essay and a video. Both narratives assess the way in which Wagner transforms motives for dramatic purposes throughout the opera. These narratives discuss many musical elements, such as key region and orchestration, along with the structure of the motives themselves, particularly the motive known as Frägerverbot.

This motive is associated with Lohengrin’s prohibition, banning his betrothed, Elsa, from asking about his identity or past. It is divided into two structures which can occur together or separately, an x-segment that introduces the ban (‘Nie sollst du mich befragen’, ‘Never shall you ask of me’) and a softer y-segment that alludes to Lohengrin’s secret past (‘woher ich kam der Fahrt, noch wie mein Nam’ und Art!’,” whence I came, my name or my kind’). These structures are indicated and labelled in the score pane on the left in figure 1. The antagonist character Ortrud aims to defeat them both by persuading Elsa to break the prohibition, and the conflict between Ortrud’s influence and Lohengrin’s is played out in transformations of the motive in a way that the essays characterise as the ‘magical realm’ (Ortrud) and the ‘grail realm’.

The textual essay. The text of the essay is presented to the user in the central pane of the screen (see figure 2). Below, as on all screens in the application, is an automatically-generated timeline. The timeline is divided into acts and scenes, and each iteration of the Frägerverbot motive is marked on the timeline as a vertical line, giving an overview of its distribution through the opera. The colour of line used reflects the character of that iteration – which of the two realms (‘grail’ or ‘magical’) dominates that occurrence. One motive iteration that was cut before the first performance is represented as a dashed line.

The application provides two dynamic elements to the user, both in the right-hand pane. An index of motive iterations provides jumping points for the essay itself, with short quotes from the prose helping the user choose where to jump to. An alternative tab in the pane (shown in figure 2) responds to iterations mentioned in the visible text and provides recordings of them, along with navigable links into the relevant free-exploration part of the app.

The reader can thus either read the essay in a fully linear manner, with the added visual support of the timeline and audible support of the sidebar, or they can jump more freely around the essay itself, exploring the parts that discuss a particular iteration, or they can leave at any point to explore the application itself, returning to the point they left off.

The video essay. The video provides a very similar experience for the user, if they watch it from within the application – it is also available on YouTube. A similar pane layout (see figure 3) puts the video in the centre with the timeline below. As the video is viewed, the right hand pane again provides audio clips from the opera and links into the more self-driven parts of the application.

Unlike the textual essay, the video also serves as a guide to the app, explicitly showing and referring to it, when the viewer is encouraged to click on links or listen to particular motives. As with the essay, on their return, the viewer who followed an outward link will come back to the same point in the video at which they left it.
Narratives and exploration in a musicology app: Supporting scholarly argument with the Lohengrin TimeMachine

3.2 Exploratory modes

A user visiting the application website is presented with a landing page with links to the essay and video, but also offering two entry points into comparatively unguided exploration of the data – inspector and TimeMachine modes. These are two of the three exploratory modes in the application; the third, comparison mode, is less useful as a starting point. Figure 4 illustrates the implemented navigation paths a viewer can take to move between modes.

The Lohengrin score is thousands of bars long (the second act alone has over two thousand bars), so it becomes crucial to support the reader in gaining an overview of change over time. The timeline introduced earlier provides a very high level abstracted overview of the opera, which also acts as an index – clicking on motives in the timeline jumps the application to that motive. We provide a second, more detailed, overview in one of the exploratory modes – the TimeMachine mode (figure 5). This mode summarises the sequence of motive iterations within the opera as an intuitive carousel-like interface, in which users can flick left or right to ‘time travel’ their way through the opera. The iterations in this mode can be visualised as score, libretto (or ‘poem’) or as an illustration of their orchestration (described below). Commentary on each iteration is displayed to the side.

Selecting an iteration in the TimeMachine mode, or following a link from the video or essay, will bring the user to the detailed inspector mode (figure 1). This mode offers the most information about the motive, including key and structure, vocal score with structural segments labelled, text underlay and stage markings (in German and English) and a visualisation of the orchestration of the extract. The layout is designed to make it easy to see these visualisations side by side, rather than one at a time.

From the inspector mode, the user can choose to compare motive iterations, bringing the TimeMachine mode back so that they can select another iteration. These are then displayed side by side in the comparison mode (figure 6), which is functionally very similar to the inspector mode. The narrative-led modes can also link straight to a comparison, but in practice, only the video uses this capability.
Figure 2: The musicological essay provides a traditional narrative that the reader can use as the basis for their exploration of the app. This mode shows the opera timeline (below) and links to other views (right). The Motifs tab on the right allows navigation within the essay text by motif iteration number, in a similar manner to a conventional index.

Even if their combination and deployment is novel, most of the visualisations illustrated above, taken individually, draw on relatively well-established practice in both conventional and digital modes of presentation. One visualisation is less conventional, and warrants attention. It is an important part of the musicological narrative that it is not simply the notes in the motive iterations that supports the dramatic logic, but the timbre and orchestration. For each motive iteration, we link to encodings of both the vocal score and orchestral edition, but a Wagnerian orchestral score presents challenges within the constraints of the application. We do not target established academic musicologists as the primary audience for the application — and even those users might struggle with a full orchestral score, especially a Wagnerian one fitted into the dimensions of a tablet screen. To sidestep this problem, but still to support an understanding of the orchestration, we use a new, more abstract visualisation of an orchestral score, simplifying its visual complexity. In our orchestration pane, each instrument playing at a particular time is shown as a coloured ribbon, with the instrument’s section of the orchestra providing the colour (seen in the right hand pane of figure 1). This visualisation, inspired in part by views in some music sequencer software, highlights differences in orchestration that may be invisible in a vocal score.
Narratives and exploration in a musicology app: Supporting scholarly argument with the Lohengrin TimeMachine

**Figure 4:** Diagram showing the main application pages and the flows between them. Links to exploratory modes can come from the essay or video modes, from the landing page, or from each other.

**Figure 5:** The TimeMachine mode, set to browse by vocal score. Flicking sideways scrolls through all the iterations in the order in which they appear in the opera, with the exact location highlighted in the timeline below. Browsing is also possible by orchestration and libretto (or 'Poem'). Each iteration is accompanied by commentary text and a recorded extract.
A MELD application can be seen as operating in two phases: graph building, and interaction. In its graph-building phase MELD traverses Linked Data graphs, ‘following its nose’ by successively preparing the URL for each. The MELD traverser creates agents to visit the digital companion. The sum of these forms an independent, repurposable, and open Research Object[7]. Once the traverser has obtained enough of the knowledge graph locally to start the interaction phase, the application dynamically constructs views based on the information and resources it has discovered. As more is received, it can dynamically reload and redraw as necessary.

MELD is written in JavaScript (as are its apps, including the Lohengrin TimeMachine). It uses the React framework, and operates over resources and data structured using Web standards. Linked Data for the application both uses and extends those structures expected by core MELD libraries, drawing on ontologies including the Music Ontology, Dublin Core, FRBR and, crucially, Web Annotations. FRBR is used here not only at the level of the complete work, but also to relate the abstract concept of, for example, the Frageverbot motive to the iterations of that motive that occur in the original score and, from those, to the editions and recordings that we include (see figure 7). This helps bridge the semantic gap between addressable resources on the one hand (represented by URLs or collections of URLs with fragment identifiers) and musical ideas that would be named, discussed and manipulated in a musicological investigation.

For textual and musical materials, we benefit from the existence of stable XML-based standards for representing content in machine-readable ways: TEI3 and MEI4. In both cases, the use of XML means that, putting a document on the web at a particular URL also makes all the elements within the document available to be specified as URIs. More directly, any element with an xml:id can be directly specified within our linked data using a fragment identifier (# in the URL). In the Lohengrin TimeMachine, all textual content, including the essay, commentary and historical texts, is served as TEI, and rendered to the screen using CETEIcean5. Music notation is encoded as MEI, and either rendered using a MELD component that calls the Verovio toolkit6 (in the case of the vocal scores) or using our new orchestration viewer. The decision of which renderer to use is made dynamically, based on RDF indicating whether the score is for orchestra or for piano and voices. Both Verovio and

4 IMPLEMENTATION

4.1 Technical overview

The TimeMachine is constructed as a MELD application – that is, it uses the MELD 2.0 (Music Encoding and Linked Data) framework[14]. A MELD application can be seen as operating in two phases: graph building, and interaction. In its graph-building phase MELD traverses Linked Data graphs, ‘following its nose’ by successively preparing the URL for each. The MELD traverser creates agents to visit the digital companion. The sum of these forms an independent, repurposable, and open Research Object[7]. Once the traverser has obtained enough of the knowledge graph locally to start the interaction phase, the application dynamically constructs views based on the information and resources it has discovered. As more is received, it can dynamically reload and redraw as necessary.

MELD is written in JavaScript (as are its apps, including the Lohengrin TimeMachine). It uses the React framework, and operates over resources and data structured using Web standards. Linked Data for the application both uses and extends those structures expected by core MELD libraries, drawing on ontologies including the Music Ontology, Dublin Core, FRBR and, crucially, Web Annotations. FRBR is used here not only at the level of the complete work, but also to relate the abstract concept of, for example, the Frageverbot motive to the iterations of that motive that occur in the original score and, from those, to the editions and recordings that we include (see figure 7). This helps bridge the semantic gap between addressable resources on the one hand (represented by URLs or collections of URLs with fragment identifiers) and musical ideas that would be named, discussed and manipulated in a musicological investigation.

For textual and musical materials, we benefit from the existence of stable XML-based standards for representing content in machine-readable ways: TEI3 and MEI4. In both cases, the use of XML means that, putting a document on the web at a particular URL also makes all the elements within the document available to be specified as URIs. More directly, any element with an xml:id can be directly specified within our linked data using a fragment identifier (# in the URL). In the Lohengrin TimeMachine, all textual content, including the essay, commentary and historical texts, is served as TEI, and rendered to the screen using CETEIcean5. Music notation is encoded as MEI, and either rendered using a MELD component that calls the Verovio toolkit6 (in the case of the vocal scores) or using our new orchestration viewer. The decision of which renderer to use is made dynamically, based on RDF indicating whether the score is for orchestra or for piano and voices. Both Verovio and

---


---

The Text Encoding Initiative, https://tei-c.org

The Music Encoding Initiative, https://music-encoding.org

https://github.com/TEIC/CETEIcean

https://verovio.org/
will usually be configured by the calling application. The component based on spreadsheets of information provided by the musicologist and whiteboard prototypes. Meanwhile, modelling work began, between Professor Dreyfus and MELD researchers, with use of pa-

The orchestration viewer is implemented as a reusable React component within the main MELD module (meld-clients-core) and generates an SVG image which is placed directly into the DOM. Ribbons are drawn live from an MEI encoding of an orchestral score, extracting the locations and durations of notes, and instrument names. Instrument labels are derived from those in the file, but can be overridden in the component configuration. This configuration also allows the definition or redefinition of orchestral sections – defaults are provided, but unusual instruments and singer names will usually be configured by the calling application. The component is capable of merging instruments that play together throughout an extract, but limited manual merging (such as for the first and second flutes) was preferred for the Lohengrin TimeMachine app.

4.2 Implementation experience

As part of the Transforming Musicology project (2013-17), some explorations were made into innovative ways of presenting scholarship around the theme of Wagner and the Leitmotiv. This work also gave rise to the first version of the essay that forms the intellectual basis of the app. Co-author Laurence Dreyfus wrote this with the idea that it could be accompanied by a ‘digital companion’. Further funding in the form of the Unlocking Musicology project (2018-19) presented the opportunity to develop the idea into this app.

The development process had co-creation across the team at its core. The ideas for the presentational and interactive aspects of the application were developed through intensive workshopping between Professor Dreyfus and MELD researchers, with use of paper and whiteboard prototypes. Meanwhile, modelling work began, based on spreadsheets of information provided by the musicologist and Sibelius files of music examples were converted to MEI (via MusicXML, the MuseScore package and the Verovio toolkit) and corrected.

Development of prototype software and versions of the modes described above could then begin. Again, this required collaboration between the researchers and the musicologist. At this point, it was felt that there was enough material to take to web designers. This order of working is extremely unusual for web development, and it proved difficult to find practitioners willing to provide design work and graphic design for a system that, largely, already existed, and which had fixed technical and resource constraints already in place. The chosen designer took briefing documents and screenshots and met with the entire team at each significant stage. They provided low-fidelity wireframes and page flow diagrams that clarified the flow structures of the site and simplified the layout. Finally, they provided high-fidelity mock ups, with the necessary digital assets and specification documents. With a modular system and sensitive, precise design, implementation of the redesign proved quick and, for the most part, easy.

We have found that processes such as this, where conception and initial collaboration is carried out in the absence of a visual designer, and perhaps without a clear idea of the final goal, are not uncommon. It may prove useful to others in similar situations to note that bringing a designer in at an early stage can still produce a successful result, given a willingness to engage in collaborative discussions and development, and to deviate from norms of industry practice.

During the design process – even before the visual designer had become involved – it became clear that the richness of information about each motive given in the spreadsheet was more than could be displayed to a user while retaining an intuitive approachable interface. As a result of this some information was simply not encoded, and other information was removed. This again was a set of collaborative decisions taken by the whole team, which acts as a reminder that even in the ‘exploratory modes’ of the app, the

![Figure 7: A small (and slightly simplified) extract from the Lohengrin TimeMachine knowledge graph. Numbered boxes represent RDF resources at different URLs (this diagram shows 5 such documents).](image-url)
paths and the information presented are informed by the demands of narrative priorities.

4.3 Resource use and application responsiveness

The two phases of operation – graph building and interaction – are performed serially for the Lohengrin TimeMachine. The application traverses the Linked Data graph once on loading, with all subsequent interactions performed without any further synchronous page loads. Once the local knowledge graph has been built and processed, the interaction phase starts, and the application becomes fully available and interactive. Once this second phase has started, the running of the application is fast, smooth and responsive, with most redraws in the 10-100ms range. The graph building phase does have a cost, though, with an initial load time of 10-15s on a 2020 iPad Pro. Although this is perceived as a ‘loading’ time, the 78 documents requested and downloaded represent about 408KB of data and loading occupies a tiny fraction of this. Traversal itself is also very quick in this case. Instead, the time is taken by the JSON-LD library converting, flattening and then reframing the graph, ready for consumption by the application and its components. Prior versions of this application had loading times in excess of 200 seconds, and the reduction of this has come about primarily through updates to the JSON-LD library itself (there is less memory use and garbage collection during this time than was previously the case).

Once the application is loaded, the system is responsive and generally neither processor nor memory hungry. The processor generally reaches about 25% use for most mode changes, except for the TimeMachine mode, for which Verovio draws all its scores at once, resulting in a reported short-term peak of 168% CPU use on our test iPad. Since this drawing is asynchronous, the user interface is available quickly, even though the score drawing itself takes approximately 4 seconds. Since the parameters given to Verovio do not change as much on a tablet as they might on a PC (where windows are more often resized), more use of caching for Verovio’s SVGs should reduce this overhead. Since Verovio is not a native React component, the SVG that it generates is less efficiently tracked by the framework, and this may also add to overhead.

5 DISCUSSION: AUTHORING DATA, AUTHORING NARRATIVES

The Lohengrin TimeMachine is neither a pure data set nor a pure musico-logical narrative – it combines these two elements in support of each other. Primarily, it is a set of gathered evidence that supports a musicological argument. This argument appears in two linear components. Prior versions of this application had loading times in excess of 200 seconds, and the reduction of this has come about primarily through updates to the JSON-LD library itself (there is less memory use and garbage collection during this time than was previously the case).

Once the application is loaded, the system is responsive and generally neither processor nor memory hungry. The processor generally reaches about 25% use for most mode changes, except for the TimeMachine mode, for which Verovio draws all its scores at once, resulting in a reported short-term peak of 168% CPU use on our test iPad. Since this drawing is asynchronous, the user interface is available quickly, even though the score drawing itself takes approximately 4 seconds. Since the parameters given to Verovio do not change as much on a tablet as they might on a PC (where windows are more often resized), more use of caching for Verovio’s SVGs should reduce this overhead. Since Verovio is not a native React component, the SVG that it generates is less efficiently tracked by the framework, and this may also add to overhead.

It is inevitable that such an approach will require collaboration between musicologist, developer and designer, but it is perhaps less inevitable that it should be as labour-intensive. Some of the effort expended on this project, such as component development for MELD and ontology development, will reduce the load for future work, but one important hurdle remains. Currently, we have no software for authoring the information underlying the application – all Linked Data was assembled by hand, all MEI and TEI was converted and corrected individually, and the time indexes for audio and video were found and manually coded into annotations using generic tools and a text editor. This approach is possible within a research project, but extremely limiting if this were proposed as a viable means of publishing scholarship in the future. Whether a sufficiently adaptable and powerful authoring tool could be developed, and what form such a tool would take, is a crucial question for future research.

ACKNOWLEDGMENTS

The Lohengrin TimeMachine was developed by the UK Arts and Humanities Research Council (AHRC) funded Unlocking Musicology project (AH/R004803/1) as a companion to the essay “Asking a Forbidden Question: A new view of musical themes in Wagner’s Lohengrin” by Prof. Laurence Dreyfus (University of Oxford Faculty of Music). Unlocking Musicology builds on research undertaken during the Transforming Musicology project (AH/L006820/1) with
important contributions from Carolin Rindfleisch. The MELD framework was created, with David Weigl as lead developer, as part of the Fusing Audio and Semantic Technologies for Intelligent Music Production and Consumption project, funded by the UK Engineering and Physical Sciences Research Council (EP/L019981/1). We thank Will Elsom for his work on the *Lohengrin TimeMachine* design specification, Steve Pierce and Oxford Filmshed Productions for the video tutorial, and Ralph Woodward for his music engravings.

**REFERENCES**


