# Composing and Realising a Game-like Performance for Disklavier and Electronics

Maria Kallionpää[1], Chris Greenhalgh[2], Adrian Hazzard[2], David M. Weigl[3] and Kevin R. Page[3] and Steve Benford[2]
[1]Music and Sound Knowledge Group, Aalburg University, Denmark, kallionpaa@hum.aau.dk
[2]Mixed Reality Laboratory, University of Nottingham, Nottingham, {first name.last name}@nottingham.ac.uk
[3]Oxford e-Research Centre, University of Oxford, Oxford, {first name.last name}@oerc.ox.ac.uk

## ABSTRACT
"Climb!" is a musical composition that combines the ideas of a classical virtuoso piece and a computer game. We present a case study of the composition process and realization of "Climb!", written for Disklavier and a digital interactive engine, which was co-developed together with the musical score. Specifically, the engine combines a system for recognising and responding to musical trigger phrases along with a dynamic digital score renderer. This tool chain allows for the composer's original scoring to include notational elements such as trigger phrases to be automatically extracted to auto-configure the engine for live performance. We reflect holistically on the development process to date and highlight the emerging challenges and opportunities. For example, this includes the potential for further developing the workflow around the scoring process and the ways in which support for musical triggers has shaped the compositional approach.

## Author Keywords
Composition, musical codes, score, music information retrieval.

## ACM Classification
H.5.2 [Information Interfaces and Presentation] User Interfaces–Input devices and strategies, H.5.5 [Information Interfaces and Presentation] Sound and Music Computing.

## 1. INTRODUCTION
In this paper we describe the composition and technical realisation of "Climb!", a non-linear musical work for Disklavier and electronics by Maria Kallionpää. This charts an ongoing collaboration between the composer and two research institutions to develop an interactive system to support this composition.

Computers can have a profound impact on music composition and performance, the works of Iannis Xenakis (1922-2001), Karlheinz Stockhausen (1928-2007), Pierre Boulez (1925-2016), and their many other contemporaries are illustrative examples of this. Techniques dependent on electronics and computers such as sound synthesis, algorithmic composition, the realization of stochastic arrangements and score following have been widely explored, as witnessed by the almost unlimited variety of interactive systems seen in contemporary performance. However, composers' and performers' wishing to engage in such musical works must invest significant time and resources in becoming 'computer-literate' or alternatively share their artistic vision with technically-minded collaborators, both of which may present barriers to use.

A key aim in our collaborative development of "Climb!" is to take steps to lower some of the technical barriers that can limit composers' and performers' use of computers in their musical practice. The interactive engine and supporting tools and processes that we are developing seek to provide a flexible but usable combination of features and capabilities. Specifically, they include an interactive real-time score and support for non-linear compositions that respond directly to the performer's playing. We begin by positioning our work against related work in this area. We then describe the specific requirements of the composition and our technical response to these, and chart the parallel process of scoring and system development. We highlight the key challenges arising and opportunities encountered. We conclude with some further challenges.


*NIME'17*, May 15 – 18, 2017, Aalborg University Copenhagen, Copenhagen, Denmark.

## 2. Related Work
### 2.1 Performance Interactions and Systems
The work in this paper is positioned in contemporary music composition and performance, which often incorporates a mixture of human and computer performers, and the exploration of electroacoustic, stochastic and open forms. When such works also include a computer accompaniment the *"live"* system resembles *"an invisible chamber music partner"* [1] and, just like a human performer, an electronic system is equally susceptible to making mistakes [2].

Dannenberg et al. [3] observes two forms of interaction with computers in live performance, namely autonomy and synchronization. Our interest lies in the intersection of these forms, where the performer sets in motion and then interacts with actions that the system runs. Many such works employ score following systems, which are numerous [4]–[6]. They aim to synchronize the electronic accompaniment with the human performer. Score following systems can be challenging for both the performer and the listener, being prone to error, which can disrupt synchronization [6]. Our interest is not to use score following in performances of "Climb!", but rather employ other methods of computer listening less reliant on continuous synchronization between performer and system.

Electronic or electroacoustic repertoire may require the performer to manage other activities other than just playing their instrument, such as controlling facets of the system. A range of contrasting solutions have previously been employed, from the use of motion sensors to enable hands free control of *"physical [or virtual] knobs, sliders and switches"* to more mundane solutions, such as foot pedals, or moving to the next setup in the piece by pressing the space bar of a laptop [8].

### 2.2 Real-time Score Display
Printed scores are inherently problematic when the performed music is non-linear in structure. Unsurprisingly, the integration of new technologies into the field of music performance has driven the exploration of similarly interactive solutions for the display of scores. Winkler outlines a set of characteristics for their application [8], stating they need to be *"generated in real-time"*; *"projected directly onto a computer screen"*; that *'the musicians can interactively influence the evolution of the*

*piece"*; and finally *"that each performance creates only one possible version of many"*. Winkler also distinguishes between a control score (real-time feedback on system state) and a playing score (i.e. traditional notation) [8]. Winkler's characteristics of real-time score application neatly describe our composer's compositional intent.

Hope and Vickery [9] classify real-time scores into four types: 'the scrolling score', where the notation is rendered in time under a fixed playhead; the 'permutative score', where discreet musical fragments can be displayed in open arrangements; the 'transformative score', where an already displayed score can be transformed in real-time; and finally the 'generative score', which is constructed in real-time.

There are numerous systems and approaches for creating and rendering real-time scores, one such example can found in InScore [10] an interactive score viewer whose rendering actions are driven by OSC messages. In addition to displaying standard western notation, InScore "*extends the traditional music score to arbitrary heterogeneous graphic objects"*[10]. For instance this might include rendering graphical elements such as the coloured highlighting of specific measures.

The "Climb!" composition is constructed around a series of musical fragments designed to be navigated through in a variety of directions, which suggests a primarily permutative approach, although with transformative elements where key phrases in the score can be dynamically highlighted.

## 2.3 Structuring Music Information

While music is typically delivered in a linear rendition, its rich conceptual structure can be viewed as hyperstructures which can encode branching compositions and incorporate other multimedia elements and annotations [11]. Earlier systems expressed relationships anchored to surface level representations: music is typically anchored to a timeline expressed in milliseconds, beat instances, or MIDI clock ticks, or to a sequence of (pitch-derived) chroma features e.g. [12]; digitized images of musical notations are referenced by spatial coordinates describing relevant regions, perhaps obtained via Optical Music Recognition [13].

The Synchronized Multimedia Integration Language[1] (SMIL) provides a means of specifying hypermedia linking structures, including grouping of elements and temporal ordering but requires the use of coordinated timelines ultimately expressed in terms of milliseconds. The commonly used MIDI format encodes pitch sequences, durations, and other control information, but is impoverished from a musicological standpoint; for example it cannot represent simple structural features, such as repeats or jump instructions, or note stem directions, beams, slurs, and other aspects of musical notation. MusicXML [14] can handle all these features but is predominantly designed to support score rendering consistency across different software tools, and consequently valuable musicological information is omitted [16].

The Music Encoding Initiative (MEI) [16] provides an alternative XML schema with a focus on capturing the semantics of musical content. This enables the clean separation of content from presentation [17] and provides a rich, cohesive, and detailed representation of musical structure in which identifiers can be assigned to notational elements at any level of the musical hierarchy. The Verovio[2] renderer generates beautiful engravings of the musical score, closely mirroring the MEI input structure. It also perpetuates the identifiers in the MEI into its SVG output such that each visual element is universally addressable.

[1] https://www.w3.org/TR/REC-smil/

[2] http://www.verovio.org/index.xhtml

# 3. Design

We now describe "Climb!" from the perspective of the composer and the audience. In the subsequent sections we will consider its technical realisation.

## 3.1 Composer's Vision

From the composer's perspective "Climb!" explores how best to maximise the capabilities of both the performer and their musical instrument. The goal was to contribute to the ongoing development of contemporary piano repertoire by creating an innovative virtuoso composition that could offer new perspectives both for the concert audience and for the musicians who play it. The original intention was to design a system, an interactive 'music engine', that could be used in the context of various compositions. This would then provide a means of multiplying the performer´s instrumental skills by providing them access to technical possibilities that would not be achievable by a human performer on a regular concert instrument (including, for example, faster tempi, changing the tuning in the middle of the performance, playing multiple octave ranges simultaneously, or one performer playing "a duet" by him/herself). Furthermore, the composer also wanted to challenge the concept of 'form' in a classical composition, which is why they decided to compose this piece in the form of a game. This approach references the tradition of musical dice games in the works of C.P.E. Bach and W.A. Mozart or John Cage's compositions based on chess play.

## 3.2 Performance Elements

"Climb!" is written for a Disklavier grand piano and electronics. The Disklavier is played by both a human performer and at times automatically, alongside the pianist, as driven by a MIDI input. Electronics here denotes digital signal processing effects (e.g. filtering, reverberation, delay) applied to the Disklavier's audio signal. Accompanying projected visuals complete the range of media envisaged for the work. A key purpose of the supporting visuals is to give the audience a view into the interactive nature of the performance, something that may be otherwise hidden within the system.

The stage setup consists of the piano, with two microphones capturing its sound. This is routed through an audio interface into the 'music engine' laptop. MIDI in and out is also routed between the Disklavier and the system laptop. The traditional printed score is replaced with a tablet or second display attached to the system laptop showing the real-time score. This is connected to a foot pedal that actions the next page of the score. The decision to place control of digital page turns over to the pianist means they can control the timing of turns within the performance of continuous musical passages and transitions between sections, where a pause may be placed (commonplace practice between movements of a piano concerto). Visuals are displayed on a large screen behind the piano and performer.

## 3.3 Open [game] form

"Climb!" uses an open form, designed so that the performer´s real-time actions define the progression through a branching compositional structure (see Figure 1). This describes the game like approach to the work: a demanding climb up a mountain where the pianist is faced with a number of challenges that steer the trajectory of their climb. "Climb!" consists of three 'macro compositions' each of them symbolizing a path leading from 'Basecamp' to the 'Summit' (i.e. Path A, Path B, Path C). These 'paths' contain a number of 'events' (i.e. micro compositions, see Figure 1) that may branch to other paths depending on how the performer meets their particular challenges. These challenges are realized as specifically scored musical material that is technically demanding for the pianist,

such as phrases, sequences or rhythms. If played correctly the pianist will remain on course, continuing along the same path, but if articulated inaccurately the system will then steer the pianist over to another path.

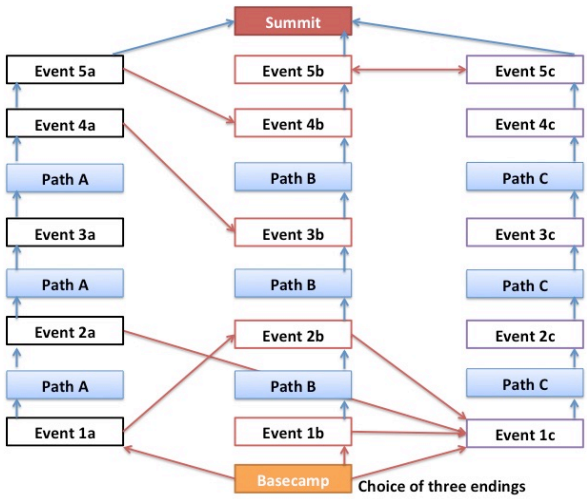


**Figure 1: Branching structure of "Climb!"**

All the micro-compositions are fully scored for both the pianist and automated Disklavier using traditional western notation, with the exception of occasional requests stated in the score for improvised, or semi-improvised musical statements.

## 3.4 Example Interactions

By way of an example, we now offer a walk-through description of a series of composed performance interactions from "Climb!". Each performance commences at 'Basecamp' (see Figure 1). At the end of this micro-composition the performer is presented with a choice of one of three alternative endings to play. Their choice will determine which path they commence their climb along (e.g. ending 1 = Path A or Ending 2 = Path B). The musical material in these endings, as is the case with key phrases in the other micro-compositions have an extra-musical functionality. Specifically they are codes that when played by the pianist trigger subsequent interactions, such as determining the next micro-composition along the path, or triggering of a Disklavier accompaniment, or changing the audio effects or cueing visual images.

If, for example, the performer chooses Ending 2 and performs it accurately the system (music engine) will queue up the next micro-composition, in this case 'Event 1b – The Stones' (see Figure 1). The performer initiates the start of 'The Stones' by advancing the score via the foot pedal. This triggers an automated Disklavier accompaniment. In the 'The Stones' our climber is faced with falling stones on the mountainside. This event finds the pianist dodging a flurry of phrases performed automatically by the Disklavier, and the audio signal is treated with a digital delay effect to mimic the rattling sound of the falling stones. Towards the end of 'The Stones' the performer is again presented with another musical challenge, which also functions as a musical code. If performed accurately they will then progress to the next micro-composition along that path entitled 'Path B' (Figure 1). Alternatively, if they fail the challenge with an inaccurate performance of the musical code they branch off to 'Event 1c – Echo'.

An additional interaction comes in the form of variable weather conditions experienced by the climber as represented by audio effects (i.e. MAX/MSP patch). Although partially random, there are also controlled by the software system so that some effects can only occur at certain stages of the piece.

These examples illustrate the nature of the interactions contained in a "Climb!' performance. Given the open form and elements of indeterminacy (i.e. branching structure and random weather conditions) it is anticipated that each performance of "Climb!" will deliver a unique reading.

# 4. Implementation

We now present the technical realisation of "Climb!" and specifically the interactive engine that has been created to support it. The architecture of the system is shown in figure 2, and integrates and extends a combination of bespoke and established software. The key software components are: Muzicodes [18], which recognises and responds to the musical trigger phrases played on the Disklavier and coordinates the entire runtime engine; MELD, which handles and renders the dynamic digital score; and MAX/MSP, which implements the audio effects and sequences the Disklavier parts. These are described in more detail below.

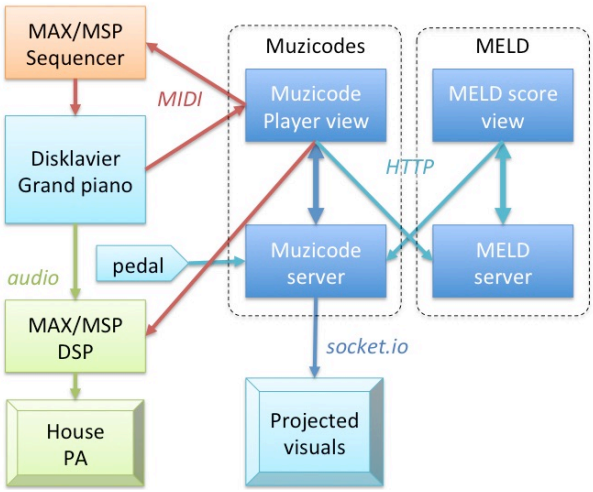


**Figure 2: The "Climb!" runtime engine**

## 4.1 Muzicodes

Muzicodes is a research system that enables composers and performers to bestow musical gestures (e.g. melodies, rhythms, or combination of both) with an extra-musical functionality, namely to trigger a range of media interactions. A previous version of the software is charted in [19], which provides a system overview.

The Muzicodes approach is distinct from Score Following in that it does not seek continuous synchronization between performer and system. Rather Muzicodes listens for pre-defined musical statements, or musical gestures, we refer to as 'codes'. These codes can be defined as pitch strings (i.e. melody), a group of delays between onsets (i.e. rhythms), by velocity or a combination there of. Thus, musical motifs embedded within the score can be identified as codes and then mapped to actions that function as triggers for other performance media. This approach can offer a great deal of flexibility for the composer/performer. For example, the presentation of codes can be performed at will, as there is no reliance on keeping to a fixed sequence of events or time code. Furthermore, codes can be absolute or, by using Regular Expressions (RegEx), 'loose' and flexible. RegEx characters can permit matching for ranges of notes or repetitions or any inputted pitch or delay, thus enabling a performer to extemporize around them. Finally, conditional matching enables codes to become active under prescribed conditions, for instance deactivated after first presentation, or only active after other preceding codes have been 'heard'.

Muzicodes can work with either audio or MIDI as input. The latter is used in this instance as the Disklavier outputs all keyboard interactions as MIDI events. MIDI provides a clean signal for the system as it bypasses the feature extraction step. Once a code is heard and any pre-conditions are fulfilled they are then mapped to 'actions'. These can be MIDI, OSC, URL or text based messages outputted to other software or hardware.

## 4.2 MELD

The next core functionality required for "Climb!" is that of a real-time score renderer. For this purpose we turned to another piece of research software, the Music Encoding and Linked Data (MELD) framework. The hierarchy of musical structure encoded by an MEI version of the score provides a semantic spine, through which elements of musical notation or aggregations thereof – for instance, groupings of individual notes and measures – are associated and annotated using Web Annotations[4]. This enables MELD to associate information with specific elements of the music regardless of performance characteristics (e.g. tempo, structural decisions) or layout decisions (e.g. page size, number of systems per page).

MELD stores this information as a set of dynamically updated RDF triples, external to the MEI source; then combines the two on-demand in a web-based renderer, augmenting the SVG score view produced by the Verovio renderer with custom presentation or actions. The RDF triples encode semantic annotations targeting notation element URIs. Action handlers corresponding to each semantic tag are modularly defined, for example to display the name of the next fragment in the queue or to instruct Verovio to render the MEI corresponding to the next segment. The rendering state is tracked using MELD RESTful API, which also ensures that actions such as rendering a new MEI source only occur at the appropriate time and that refreshing the web-browser does not "break" the performance.

Provenance information is generated during the creation of each annotation, and on its actioning by the renderer. This captures MELD's 'perspective', allowing us to determine the sequence of actions performed. We can use this information to generate real-time or post-hoc descriptions of the performance, which can be shared with the audience and performer.

The digital score view contains two views in alignment with Winkler [8], i.e. the playing score (MELD view) and a control score (Muzicodes view). The former displays the traditional western notation including the Disklavier part (where appropriate). The latter displays the state of the Muzicodes system: the incoming stream of note events, candidate codes which, when performed, are progressively highlighted in red and a preview/action window that display actioned codes. However work is also in progress to integrate the control information directly into the playing score, for example dynamically highlighting triggered codes.

## 4.3 Other Components

The well-known commercial product MAX/MSP provides supporting capabilities to the interactive engine. A simple patch loads the Disklavier parts exported from the score and plays (sequences) these when triggered by Muzicodes over an internal MIDI connection. The treatment of audio effects on the Disklavier are realized by a second MAX/MSP patch hosting a number of Virtual Studio Technology (VST) effect plugins (using the vst~ object). This patch is also controlled by MIDI messages from Muzicodes that define the routing of the audio signal through the VST's. The projected visuals are intended to show animated sequences, although to date system tests have used static images. This display is directly controlled by Muzicodes (using its "channel" system [18]).

# 5. Challenges and Solutions

Having presented the overall implementation of the interactive engine that supports "Climb!", we now consider some of the key challenges that have emerged during development and how they have been addressed.

## 5.1 System Integration

The first challenge was to achieve an effective integration of the main technical elements, in particular Muzicodes and MELD: the two systems need to coordinate closely throughout a performance. While Muzicodes was capable of issuing a range of actions in response to the recognition of performed codes it lacked any other form of control input. So we extended Muzicodes with a 'Controls' section that permits for actions in Muzicodes to be associated with other inputs including buttons, MIDI control messages and most importantly HTTP requests. We also extended the range of actions available in Muzicodes to include the ability to make HTTP requests to other software.

Using these new HTTP interfaces we were able to integrate and synchronise the operation of Muzicodes and MELD. Specifically, MELD is responsible for displaying the digital score, and it informs Muzicodes (over HTTP) each time the performer advances the display to a new micro-composition. Meanwhile, Muzicodes is responsible for detecting the musical codes or trigger phrases and responding appropriately. In particular Muzicodes informs MELD (again over HTTP) when a new micro-composition should be queued as the next fragment to display. The Muzicodes system also acts as a bridge between the foot pedal and MELD, responding to pedal presses by sending a page-turn annotation to MELD.

These extensions to Muzicodes have begun to expand its role beyond a music recognition system to one of a performance management system, capable of integrating other diverse sub-systems.

## 5.2 System Expressiveness

One of the key functions of this integrated system is the triggering of actions to control "Climb!'s" various media elements: the real-time interactive score display, automated Disklavier accompaniment, routing of the audio signal through digital signal processing and the synchronisation of visuals to transitions between micro-compositions. These actions need to respond to a number of competing considerations, namely to compliment performance aesthetics, bestow system control to both the performer and system in different circumstances, and present a real-time view on the system state, across a number of linked sub-systems. This required careful thought and development so that in the performance system each musical code or change of score can trigger any or all of the following actions, initiated by Muzicodes:

- Queuing of the next appropriate score fragment in MELD.
- Asking MELD to highlight measures in the score in order to give real-time feedback to the performer as to whether the corresponding Muzicode has been triggered.
- Starting an automated Disklavier accompaniment (MAX patch 1).
- Triggering a specific (predefined or random) audio effect in MAX Patch 2.
- Switching the visual display to a new image or animation (specified via a URL).

The development process also revealed a number of other extensions to Muzicodes' functionality required to realise the composition as envisioned. In addition to the general support for integration described above, it was also necessary to

[4] https://www.w3.org/ns/oa

introduce *delayed* actions in Muzicodes. Previously actions were performed as soon as a code was recognized. Thus composers and performers, when defining codes, had to balance the needs of the music and the placement of the codes against the desired moment of initiation of any resultant action. By introducing a function to queue future actions (i.e. to delay actions by a certain number of seconds), the composer has more flexibility about where codes can be embedded within the music. Note that this is effective when timings can be prescribed and are consistent across multiple performances, but less effective if this is uncertain. In the context of "Climb!" those micro-compositions with Disklavier accompaniments (driven by MIDI files) are fixed and therefore consistent in tempo and duration, thus appropriate for delayed actions.

In addition to delayed actions, support for randomized actions has also been added. This was in response to the requirement for random 'weather conditions' manifest by particular choices of effect processing in MAX, so that each micro-composition can trigger one of several possible effects at random.

## 5.3 Scoring and Annotation

The composition of "Climb!" has for the most part taken place independently from the rest of team. Periodically the composer presents new fragments and these are integrated into the music engine. One theme of discussion has revolved around score markings and annotations, specifically how to notate the codes and other interactions beyond the piano keyboard. To this end, the Disklavier has been notated on a separate staff, in keeping with traditional practices. Codes are marked in two ways. First, they are placed on an ossia, a breakout staff that appears above or below the piano part in the score. Traditionally, ossias are used to notate an alternative passage that can be played, or to assist a performer in navigating their score after a period of rest by indicating other currently playing instrument lines. Our composer has adopted this engraving practice to draw attention to the location of codes. Secondly, text has been used to provide performance instructions (e.g. for 'loose' or semi-improvised codes) and subsequent actions. These ossias are only present in the full score, and as such they speak to the composer, technician and the performer in rehearsal. They do not appear in the real-time score used in a performance setting, due in part to limited display space on the performer's screen.

Some of the audio effects have also been marked in the full score. As with the above example, existing traditional score markings have been re-purposed for this task, such as using 'hairpins' (normally used to express dynamics) to indicate the fade in and out of audio effects. To clarify the function of the marking the composer has again used accompanying text descriptions. When scoring new or novel musical interactions contemporary composers typically repurpose existing markings, rather than invent new ones.

Our composer's approach to the nature of codes embedded within the musical material has taken on a notable relevance. In principle these codes do not need to be musically remarkable, and can be indistinct to the human ear as long as they are recognised by the system. Nonetheless, the composer aligned the extra-musical codes with the principle musical themes of the work, describing their function as the 'keys' by which the piece is structured. While this is an artistic decision, as opposed to one of necessity, it highlights how the Muzicodes approach has promoted interdependency between the composition of music and composition of interaction; specifically Muzicodes has become a defining element of the compositional process.

## 5.4 Score Encoding and Rendering

Generating the version of the score for use in MELD presented a number of challenges, specifically the conversion from Sibelius to MEI. With no native MEI export, an available third-party plugin created MEI files with many errors (sometimes doubling note events and omitting many markings). We compared these to musicXML encodings from Sibelius which demonstrated a number of improvements. We discovered that opening and re-exporting these as musicXML files in MuseScore2 improved matters. These were then converted to MEI, however further manual editing was still required to rectify outstanding errors. This highlights a current challenge for this and other such systems using the MELD approach, and consequently represents an area for future work.

## 5.5 Workflow of Composition and Performance

To date the collaborative realisation of "Climb!" has spanned a 7 month period including a number of focused development sessions. Here we reflect on the process or workflow that has been developed to support composition and performance. Figure 3 illustrates an integrated workflow of three distinct stages: composition and production, performance and post-performance. Central to these stages is notation and annotation.

The composition process is undertaken directly to a digital score (i.e. in Avid Sibelius). Once codes are identified in the score these are then added to the "Climb!" Muzicodes experience (the file that contains all of the configuration parameters for Muzicodes, such as codes, conditional matching and actions). This is not necessarily a one-way process: the needs of the extra-musical functionality of the codes can push back on the creative decisions the composer makes. The creation or configuration of other performance media, such as the audio effects (DSP) and visuals also come into play here.

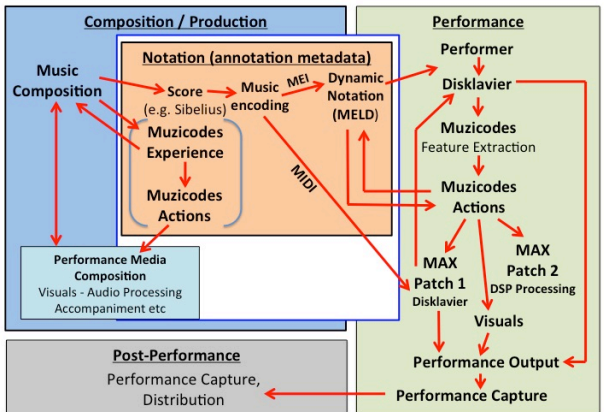


**Figure 3: "Climb!" Music Engine Workflow**

Once scored, the notation is then converted into MEI for MELD to use in the real-time rendering of score fragments. The Disklavier parts in the score are also exported as MIDI files to be sequenced by MAX Patch 1 (see Figure 3). The performance stage is as described in section 4 and above.

MELD also maintains a complete record of the history of the dynamic score, allowing us to determine, for instance, the exact time at which a Muzicode was triggered and when the next piece of the composition was determined and subsequently transitioned to. This information can be captured for every performance of "Climb!" allowing each individual performance to be documented and compared.

Part of our ongoing intention is to make it easier for the composer/performer to set up and configure the complete performance system, especially valuable when a performance is being reproduced by a different set of performers or when a work is being adapted or modified. To this end we have created an initial software tool to semi-automate the configuration process. "Climb!" is a large-scale composition comprising

about 20 different micro-compositions, each with its own musical code(s), accompaniment, visuals and choice of effects. We have defined a simple Excel spreadsheet in which all of these performance options can be specified. If the score is changed or the settings for a micro-composition are modified then the system can re-process the score MEI files, the spreadsheet and the previous Muzicodes experience file to create a new experience file with the correct codes, conditions and actions. This tool also auto-generates all of the specific controls and actions needed to link Muzicodes and MELD.

# 6. CONCLUSIONS

The integration of Muzicodes and MELD has created an interactive performance system capable of supporting a broad range of performance interactions linked to a dynamic score. This is successfully supporting the composition and performance of "Climb!", whose musical challenges, branching structure and stochastic elements draw inspiration from computer games. Our long-term aim is to lower some of the technical barriers to the integration of technologies in such interactive performance works. This clearly references a wide spectrum of tailoring and the work charted here represents an initial exploration of a single mid-point example. We have begun to develop additional software tools to support and automate mechanical elements of the process of setting up and configuring the software system. These permit for some authoring and re-configuration outside of the software's 'code'. There is still more to be done about extracting further detail from the score, where the composer's annotations can auto-configure other system interactions. Furthermore, our MAX/MSP patches require significant handcrafting, which, given our aim, raises a question as to whether and how one might automate this process. Our vision is that a future version of the system will be able to operate entirely from the composer's score notations together with a suitably structured form of programme note, requiring minimal technical expertise.

There are three further related challenges that are brought to the fore by compositions such as "Climb!" which we are now beginning to address. First, it is relatively difficult to rehearse a piece such as this. Disklavier pianos (like many other unusual musical instruments and interfaces) are not universally available. Consequently initial testing of the system was done using a MIDI keyboard which can mimic the sound of the auto-accompaniment but not the mechanical and visual aspect of keys being depressed. But even with this adjustment it can be quite difficult and time-consuming to set up the performance system, especially on a new computer. Second, it is relatively difficult to perform the piece, for essentially the same reasons. Dobrian and Koppelman [19] observe that it is commonplace in contemporary performances, especially those that employ novel music systems and interfaces for performers to have limited time to get to grips with the instrument, with a direct impact on performance virtuosity. Furthermore, if problems crop up with the hardware or software then trouble-shooting typically requires a high level of familiarity and expertise. Third, it is relatively difficult to record a performance in all of its richness, for example including the score notations and visual elements in addition to the sound. And where such multi-faceted recordings *are* created they are difficult to pass on, view, interpret or analyse. We envisage that a set of related Digital Music Objects (DMOs) could be defined that encapsulate the diverse aspects of such a composition and/or its performances. Associated software tools could then scaffold and at least partially automate configuring, controlling, recording, repurposing and reproducing such performances. We are using the performances of "Climb!" to motivate, inform and drive the initial development of such tools.

# 7. ACKNOWLEDGEMENTS

This work was supported by the UK Engineering and Physical Sciences Research Council [grant number EP/L019981/1] Fusing Semantic and Audio Technologies for Intelligent Music Production and Consumption, and the Kone Foundation.

# 8. REFERENCES

[1] E. McNutt, 'Performing electroacoustic music: a wider view of interactivity', *Organised Sound*, vol. 8, no. 3, pp. 297–304, Dec. 2003.

[2] S. Berweck, 'It worked yesterday: On (re-) performing electroacoustic music', University of Huddersfield, 2012.

[3] N. E. Gold *et al.*, 'Human-computer music performance: From synchronized accompaniment to musical partner', 2013.

[4] 'antescofo [Antescofo]'. [Online]. Available: http://repmus.ircam.fr/antescofo. [Accessed: 27-Jan-2017].

[5] M. Puckette and C. Lippe, 'Score following in practice', in *Proceedings of the International Computer Music Conference*, 1992, pp. 182–182.

[6] N. Orio, S. Lemouton, and D. Schwarz, 'Score following: State of the art and new developments', in *Proceedings of the 2003 conference on New interfaces for musical expression*, 2003, pp. 36–41.

[7] Q. Yang and G. Essl, 'Augmented piano performance using a depth camera', *Ann Arbor*, vol. 1001, no. 2012, pp. 48109–2121, 2012.

[8] G. E. Winkler, 'The realtime-score: A missing link in computer-music performance', *Sound and Music Computing*, vol. 4, 2004.

[9] C. Hope and L. Vickery, 'Screen scores: New media music manuscripts', 2011.

[10] D. Fober, Y. Orlarey, and S. Letz, 'Augmented interactive scores for music creation', in *Korean Electro-Acoustic Music Society's 2014 Annual Conference*, 2014, pp. 85–91.

[11] D. Byrd and T. Crawford, 'Problems of music information retrieval in the real world', *Information processing & management*, vol. 38, no. 2, pp. 249–272, 2002.

[12] V. Thomas, C. Fremerey, M. Meinard, and M. Clausen, 'Linking Sheet Music and Audio - Challenges and New Approaches', *Wadern: Schloss Dagstuhl - Leibniz-Zentrum für Informatik GmbH.*, vol. 3, pp. 1–22, 2012.

[13] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marcal, C. Guedes, and J. S. Cardoso, 'Optical music recognition: state-of-the-art and open issues', *International Journal of Multimedia Information Retrieval*, vol. 1, no. 3, pp. 173–190, 2012.

[14] M. Good and others, 'MusicXML: An internet-friendly format for sheet music', in *XML Conference and Expo*, 2001, pp. 3–4.

[15] T. Crawford and R. Lewis, *Review: Music encoding initiative*. University of California Press Journals, 2016.

[16] A. Hankinson, P. Roland, and I. Fujinaga, 'The Music Encoding Initiative as a Document-Encoding Framework.', in *ISMIR*, 2011, pp. 293–298.

[17] L. Pugin, J. Kepper, P. Roland, M. Hartwig, and A. Hankinson, 'Separating Presentation and Content in MEI.', in *ISMIR*, 2012, pp. 505–510.

[18] Chris Greenhalgh, S. Benford, and A. Hazzard, '^muzicode$: Composing and Performing Musical Codes.', presented at the Audio Mostly, Norrköping, Sweden, 2016.

[19] C. Dobrian and D. Koppelman, 'The 'E' in NIME: musical expression with new computer interfaces', in *Proceedings of the 2006 conference on New interfaces for musical expression*, 2006, pp. 277–282.