XML Music Performance Description: Reflections on and Future Developments of the Music Performance Markup Format

Axel Berndt
Center of Music and Film Informatics, University of Music Detmold, Detmold, Germany
berndt@hfm-detmold.de

Benjamin W. Bohl
Institut für Musikwissenschaft, Johann Wolfgang Goethe-Universität Frankfurt, Frankfurt am Main, Germany
bohl@em.uni-frankfurt.de

Abstract

This paper introduces the `Music Performance Markup` format for encoding expressive performance parameters. The main focus is the general data organization in global vs. local data and header vs. dated data, illustrated by some musical examples. Moreover future work and use cases are envisioned.

Motivation and related work

Haus and Longari [6] distinguish six layers of music information: general, structural, music logic, notation, performance, and audio. Regarding the performance layer, most music formats provide either very technical information (e.g., MIDI) or very rough information (MusicXML, MEI). While the technical information is rich in details, precise and unambiguous it lacks a musically meaningful abstraction. Abstract representations, on the other hand, are easy to read and write into music scores but are ambiguous and incomplete. E.g.: how loud is “forte”? How fast is “allegro”?

Throughout the past two decades, several research projects contributed to answering such questions in order to bridge the gap between abstract information and technical implementation. While analytical projects investigated the phenomena of music performance and their shaping by human musicians [7], generative projects developed systems that (semi-)automatically create expressive performances (see [8], [10], [4], [5]). The data models to describe these performances were rarely a matter of deeper discussions. Hence, they never reached any relevance beyond the scope of their corresponding systems.

The XML-based Music Performance Markup (MPM) model for performance description initially had been developed for exclusive usage within a specific performance rendering engine [1]. Since their publication however, the format and the engine have been used in several different contexts, e.g. research on video.

General concept

An expressive performance of a piece of music transforms the musical raw material (typically notes) into sounding output. One piece of music can be performed in many different ways. In our approach this is reflected by—as in SMDL [9]—generally separating logical and gestural domain data. While a MIDI file provides the musical "raw material" (and should not contain anything else but the note-on and note-off events) a corresponding MPM file provides the description of the performative aspects. Our software implements the techniques to render this information into expressive MIDI sequences that are output directly or sent to a *Digital Audio Workstation* for music production.

The MPM format is based on the formal models that were introduced by Berndt [2], and thus allows descriptions of rich, nuanced music performances that reach down to a variety of subtle details. These cover a wide range of performance phenomena, including tempo (macro timing) and dynamics curves (macro dynamics), articulations (micro dynamics and micro timing), metrical accentuation (micro dynamics), asynchrony, rubati, and quasi-random imprecision (micro timing). The models are based on a series of studies, experiments with musicians, and measurements of music recordings. From the observed characteristics we deduced mathematical functions and developed a parameterization that is meaningful in the musical context and provides access to the full range of variability. For example, continuous tempo transitions, i.e. accelerando and ritardando, can feature different shapes depending on various factors such as the musical context or the abilities of the musicians The tempo model provides all these shapes of tempo curves so that they can be used to recreate existing performances or simulate hypothetic performances.

Furthermore, the format defines global information, applying to all musical parts, and local information, applying to single parts. Technically, the parts correspond with MIDI channels. Local information may be defined for each part individually, and if present, local information dominates the global. This pays off e.g. in solo-plus-accompaniment constellations. Global performance instructions are executed by the whole ensemble; only the solo part features its own local instructions that "overwrite" the global.

Both, global and local information, are subdivided into header and dated information. Header information applies to the whole movement, e.g. the definition of articulation styles and MIDI controller mappings. Dated information is organized in sequential lists, so-called *maps*, one for each type of performance features. Each element in a map demands a date attribute (in MIDI ticks). Discrete performance features, e.g. articulation instructions, are applied at the respective time position in the MIDI sequence. Most performance features, however, extend over a certain timeframe. A dynamics instruction, like *forte* for instance, lasts from its date until the date of a succeeding instruction in the same dynamics map. Other penomena like rubato instructions or metrical accentuations define a certain timeframe of their application—in case of metrical accentuations typically one measure. The schemes are then repeatedly applied (e.g., measure-wise) until the succeeding instruction in the same map defines a new scheme and frame length.

The tempo and dynamics models define macro curves, i.e. absolute timing and loudness curves with no other details than rough ritardandi, accelerandi, crescendi, and decrescendi, etc. All other models are called micro features. They are defined as relative features, i.e. they are mathematically added onto the macro curves and, thereby, introduce rich details to the performance description. One requirement to the model design of micro features was self-containedness. Even though many features affect the same domain (e.g. the timing domain is addressed by the micro features rubato, random imprecision and asynchrony) the features should be self-contained, i.e., give the user independent control. E.g., editing a Viennese waltz timing is achieved only via rubato, other feature types do not interfere with it and it is not necessary to balance rubato and several other features to achieve the desired rubato timing. This self-containedness is not only of practical value when defining and editing music performances. It is also relevant to analytical applications as it provides guidance for the analytical decomposition of complex timing and dynamics curves.

The software is also capable of rendering seamless transitions between different performance styles, which is, for instance, used for adaptive game scoring. In a listener study we used this interactivity for an analysis-by-synthesis approach where the participants could adjust certain performance parameters according to the tasks they were given [3].

Reflections and perspectives

The interplay of global and local information is the key to complex polyphonic performance structures. Each part can feature its own performance plan. One part may perform a decrescendo while another part plays a crescendo. One part may perform a swing timing, another an even timing, while they all follow the same basic tempo. A key feature of our models to allow such multifarious performance representations is the self-containedness of each single model. This gives precise and independent control of each performance detail without any distorting interference from another detail. It is possible, for instance, to change the macro timing (tempo curve) completely while keeping all micro timing features unaltered.

But where do these performance details come from? The performance instructions that the format conveys are explicit and numerically precise. For instance, a dynamics instruction *forte* corresponds with a clearly defined numeric value, even if each part defines a different such value.[1] The models underlying the format act as interfaces, the user adjusts their parameters. However, the creation of a fully fledged performance is, nonetheless, an excessive amount of work. This must be supported by editing tools, such as music notation software plug-ins. Reduced editing workload can also be achieved by semiautomatic approaches. Rough performances may be generated automatically. The human editor then corrects and incorporates further details. The analysis of preexistent performances is another way to generate a detailed performance in MPM. Ultimately, this makes the computer play the music as the human musician did. Such formalized human music performances might become a subject of closer analysis to learn more about performance styles, e.g. in the context of historically informed performance practice and its research.

The format’s close connection to the MIDI standard reflects the direct coupling to music production as the core application domain. This may imply some disadvantages such as bad readability. A better compatibility with other formats also demands a flexible numeric basis of the attributes, for instance by user-defined reference systems. We therefore decided to make the timing, dynamics and pitch domain floating point with the next development iteration. Although this precision might be lost when the performance is rendered to an expressive MIDI sequence, other (for instance analytical) application contexts can benefit from higher numeric resolution and accuracy. To break the reliance on the MIDI standard even further, we will generalize the relation to musical score data. Via converters we plan to add support for formats such as MusicXML and MEI.

Currently, MPM undergoes a drastic redesign addressing issues of generalization and cross-format references (e.g. between MPM and score data). We plan to extend the library of models in many respects. For instance, delay and (de-)tuning are added to the articulation model. Random imprecision—thus far available only on the timing axis—are added to the dynamics and articulation domain. Several alternative imprecision models are offered, from a simple uniform distribution, Gaussian and triangular distribution over correlated noise types such as pink noise up to an imprecision table that can be created from measurements of human performances.

1. The *forte* of, e.g., a flute can be defined different to that of a trumpet.

Further potential features include playing techniques (e.g. fingering, breathing), ornamentations, ensemble communication and how it is influenced by seating order and acoustics. A formal representation of a more general “aesthetic agenda” could capture specifics of a certain musician, school or musical era and might be used to generate expressive performances in an automatic or semi-automatic fashion.

The redesign will also make a redevelopment of the rendering engine necessary which we plan to add to the *meico* converter framework.[2] With this we hope to support interaction of the two musicological disciplines performance research and (digital) music edition. We see further potential applications of MPM in computer-based music production and interactive media scoring [1].

Works cited

[1] Berndt, Axel. *Musik für interaktive Medien: Arrangement- und Interpretationstechniken*. Munich: Hut, 2011.

[2] Berndt, Axel. “Formalizing Expressive Music Performance Phenomena” in *Works in Audio and Music Technology*. Dresden: TUDpress, 2015, 7-128.

[3] Berndt, Axel and Hähnel, Tilo. “Studying Music Performance and Perception via Interaction” in *Works in Audio and Music Technology*. Dresden: TUDpress, 2015, 129-53.

[4] Flossmann, Sebastian, et al. “Expressive Performance Rendering: Introducing Performance Context” in *Proceedings of the 6th Sound and Music Computing Conference (SMC). Porto, Portugal, 23-25 July 2009*, 155-60 (http://smc2009.smcnetwork.org/proceedings)

[5] Friberg, Andreas et al. “Overview of the KTH Rule System for Musical Performance” in *Advances in Cognitive Psychology* 2.2-3 (2006): 145-61.

[6] Haus, Goffredo and Longari, Maurizio. “A Multi-Layered, Time-Based Music Description Approach Based on XML” *Computer Music Journal* 29.1 (2005): 70-85.

[7] Hähnel, Tilo. *Baroque Performance: a Research Study on Characteristic Parameters of 18th-Century Music*. Studies in Cognitive Musicology. Osnabrück: Electronic Publishing Osnabrück, 2013.

[8] Mazzola, Guerino. *The Topos of Music: Geometric Logic of Concepts, Theory, and Performance*, in collaboration with Stefan Göller and Stefan Müller. Basel: Birkhäuser, 2002.

[9] Newcomb, Steven R. “Standards: Standard Music Description Language Complies with Hypermedia Standard” *Computer* 24.7 (1991): 76-79.

[10] Widmer, Gerhard, and Werner Goebl. “Computational Models of Expressive Music Performance: The State of the Art” *Journal of New Music Research* 33.3 (2004): 203-16.

2. https://github.com/cemfi/meico, accessed 1 September 2017.