

Proposal for a common multimedia ontology framework: Information management for music analysis systems

Samer A. Abdallah, Yves Raimond, Mark Sandler

Centre for Digital Music, Queen Mary, University of London
{samer.abdallah,yves.raimond,mark.sandler}@elec.qmul.ac.uk

1 Introduction

In this proposal, we discuss requirements for and potential applications of a music information management system that represents in a richly structured way not only the media objects themselves and associated metadata, but also the computational systems by which new features and representations of given media objects can be derived.

In the first instance, this is intended to support the activities of researchers, who may be developing new algorithms for analysis audio or symbolic representations of music, or may wish to apply methodically a battery of such algorithms to a collection or multiple sub-collections of music. For example, we may wish to examine the performance of a number key finding algorithms on a varied collection, grouping the pieces of music along multiple dimensions by, say, instrumentation, genre, and date of composition. The knowledge representation should support the definition of this experiment in a succinct way, selecting the pieces according to given criteria, applying each algorithm, perhaps multiple times in order to explore the algorithms' parameter spaces, adding the results to the knowledge base, evaluating the performance by comparing the estimated keys with the annotated keys, and aggregating the performance measures by instrumentation, genre and date of composition. The outputs of each algorithm should be added to the knowledge base in such a way that each piece of data generated is unambiguously associated with the function that created it and all the parameters that were used, so that the resulting knowledge base is fully self-describing. Finally, a statistical analysis could be performed to judge whether or not a particular algorithm has successfully captured the concept of 'key', and if so, to add this to the ontology of the system so that the algorithm gains a semantic value; subsequent queries involving the concept of 'key' would then be able to invoke that algorithm even if no key annotations are present in the knowledge base.

One of the key ideas here is to blur the distinction between a 'relation' or 'property' in a database or ontology and a 'function' provided by a software library—in many respects, a function is like a relation whose tuples are not stored explicitly, but are computed on demand when enough attributes (i.e. the inputs to the function) have been supplied. By making a complete record of

each function evaluation, that function can be used exactly like a relation in a database, being usable in arbitrary queries and potentially having semantic weight by virtue of its relationships with concepts defined in the ontology.

For further elaboration of these ideas, see [1] (attached as it is currently unpublished). We also used our current system to run an experiment in song segmentation [2].

2 Requirements

2.1 Scope

A basic requirement for a music information system is to be able to represent all the ‘circumstantially’ related information pertaining to a piece of music and the various representations of that piece such as scores and audio recordings; that is, the information pertaining to the circumstances under which a piece of music or a recording was created. This includes physical times and places, the agents involved (like composers and performers), and the equipment involved (like musical instruments, microphones etc.). To this we may add annotations like key, tempo, musical form (e.g. symphony, sonata etc.).

To support the kind of analysis and experimentation we are interested in also requires that the library of available computations be represented at some level of granularity. Each computation would be annotated with information about the types of its arguments and returns, its implementation language (so that it can be invoked automatically), whether it behaves as a ‘pure’ function (deterministic and stateless) or as a stochastic computation, which is useful for Monte Carlo-based algorithms, and whether or not the computation should be ‘tabled’ or ‘memoised’, as described below.

In the current implementation of our system, Whenever a computation marked for tabling is performed, the system makes a record of the computation event, storing the inputs and outputs, the time and duration of the computation, and the name of the computer used. For pure functions, these computation records save repeated evaluation of the same function with the same arguments, so, for example, if many algorithms use an audio spectrogram as an intermediate processing step, the spectrogram is computed just once the first time it is required.

With these elements in place, various procedures can be put in place to reason about the contents of the knowledge base and expand it in a structured way. For example, we can combine a function with its table of previous evaluations to create a sort of ‘virtual relation’ or ‘view’, which can answer queries by looking up previous evaluations or, if all the inputs to the function are supplied, by triggering new evaluations. This means that the results of a computation can be retrieved using the same query that triggered the computation the first time round.

Alternatively, if a function is very cheap to compute, we may choose not to table it, in which case it can only take part in queries where all its inputs are supplied.

Once a function has been ‘installed’ into the ontology as a relation with the same logical status as other predefined relations, it may be given semantic value, for example, by stating that it is equivalent to or a subproperty of some existing property like ‘key’ or ‘tempo’. This would enable it to take part in general reasoning tasks such as user level queries or experiment design.

2.2 Important ontology concepts

In this section, we list some of the important concepts to be represented in a music information ontology. Since we have already implemented a prototype system, some of the text below is phrased as a description of our current system, but these also stand as requirements or recommendations for a common multimedia ontology.

Music is above all a time-based phenomenon. We would like to see the temporal logic at heart of this formalised in a set of concepts which will be useful for describing any temporal phenomenon. Many relevant ideas have been discussed in the AI, logic and knowledge representation literature [3–5]. In particular, the idea of multiple *timelines*, both continuous and discrete, is relevant for signal processing systems where multiple continuous-time and discrete-time signals may co-exist, some of which will be related (conceptually co-temporal) and some of which will be unrelated. Each timeline can support its own universe of time points, intervals and signals. However, timelines of different topologies can be related by maps which accurately capture the relationship implied when, for example, a continuous timeline is sampled to create a discrete timeline, or when a discrete timeline is sub-sampled or buffered to obtain a new discrete timeline.

Closely related to temporal logic is the representation of events, as addressed in the literature on *event calculi* [6–8]. The ontology of events has also been addressed in the semantic web literature [9–11]. In a music information system, the notion of ‘an event’ is a useful way to characterise the physical processes associated with a musical entity, such as a composition, a performance, or a recording. Extra information like time, location, human agency, instruments used and so on can be associated with the event in an extensible way.

Music is also a social activity, so the representation of people and groups of people is required, as suggested above in the requirement to represent the agents involved in the occurrence of an event.

The ontology of computation requires the notion of a ‘callable computation’, which may be a pure function, or something more general, such as a computation which behaves non-deterministically. By encoding the types of all the inputs and outputs of a computation, we gain the ability to reason about legal compositions of functions. In addition, to manage the results of computations, we need a concept of ‘evaluation’ to represent computation events, recording inputs, outputs, and other potentially useful statistics like computation time.

The computation ontology we are currently developing includes a concept of ‘mode’ inspired by the *Mercury* language¹. This allows relations to be declared

¹ see www.cs.mu.oz.au/research/mercury

as strictly functional when particular attributes are treated as ‘inputs’. For example, the relation `square(x, y)`, where $y = x^2$, is functional when treated as a map from x to y , but not when treated as a map from y to x , since a real number has two square roots. Representing this information in the computation ontology will allow us to reason about legal ways to use the relation and how to optimise its use by tabling previous computations.

We aim to extend the mode system to allow for a class of stochastic computations, where the output is defined by a conditional probability distribution, that is $p(\text{outputs}|\text{inputs})$. This will be useful for representing algorithms that rely in an essential way on random number generation.

Specifically musical concepts include specialisations of concepts mentioned above, such as specifically musical events (compositions, performances), specifically musical groups of people (like orchestras or bands), specifically musical conceptions of time (as in ‘metrical’ or ‘score’ time, perhaps measured in bars, beats and subdivisions thereof), and specifically musical instruments. To these we must add abstract musical domains like pitch, harmony, key, musical form and musical genre.

2.3 Modularity and cross-domain links

We have no particular requirements for cross-domain links, but we would like to point out that a music information system covers a broad range of concepts which are not just specific to music; for example, people and social bodies with varying memberships, time and the need to reason about time, the description of physical events, signals and signal processing in general and not just of music signals, the relationship between information objects (like symbolic scores and digital signals) and physical manifestations of information objects (like a printed score or a physical sound), the representation of computational systems, and finally, the representation of probabilistic models including any data used to train them. In fact, once these non-music-specific domains have been brought together, only a few extra musical concepts need be defined in order to have a very comprehensive system.

2.4 Languages and standards

In our current system, we use OWL (DL) for ontologies and XSB-Prolog (tabled Prolog) for DL reasoning. We store relational data models using an RDBMS accessed via SQL or RDF managed by JENA. We use Matlab as a computational engine via an extension to SWI Prolog², which we use as a binding between some of the other components and as a command-line environment for driving the system. We would like to support other (especially open source) computational engines such as Octave, LISP, Java and Haskell.

² see www.swi-prolog.org

3 Harmonisation

The representation of physical events has also been addressed in other ontologies, notably ABC [9], DOLCE [10], and SUMO [11]. We would be cautious about using an *overly* rich conceptualisation due to the risk of making unnecessary metaphysical commitments (e.g. arguments between those who advocate endurants and perdurants as fundamental distinctions and those who prefer to think in 4-dimensional space-time). We would generally take the view that an ontology should be as simple as possible while still addressing the concrete needs of systems that are going to use it. However, the upper ontologies cited above may be a useful reference when designing multimedia ontologies, especially where they help to identify which concepts are so general that they transcend particular domains like music, multimedia, computation etc. In addition, we found the *OntoClean* methodology and meta-ontology [12] provided some valuable insights when trying to clarify the role of each concept in an ontology.

Using the modularisation of domain ontologies defined in [13], we can draw clear links between the different domains of our ontology, but also between one of our domain and an other ontology. In our current system, we have such explicit links to two ontologies. The first one is the *MusicBrainz* ontology. *MusicBrainz* is a semantic web service [14], describing CDDB-style informations, such as artists, songs and albums. The second one is the Dublin Core ontology. It handles some common general properties like ‘title’, ‘creator’ etc..

References

1. S. Abdallah, Y. Raimond, and M. Sandler, “An ontology-based approach to information management for music analysis systems,” 2005, unpublished.
2. S. Abdallah, K. Noland, M. Sandler, M. Casey, and C. Rhodes, “Theory and evaluation of a bayesian music structure extractor,” in *Proceedings of the Sixth International Conference on Music Information Retrieval*, J. D. Reiss and G. A. Wiggins, Eds., 2005, pp. 420–425.
3. J. Allen, “Towards a general theory of action and time,” *Artificial Intelligence*, vol. 23, pp. 123–154, 1984.
4. Y. Shoham, *Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence*. Cambridge, MA: MIT Press, 1988.
5. A. Galton, “The logic of occurrence,” in *Temporal Logics and their Applications*, A. Galton, Ed. London: Academic Press, 1987, ch. 5, pp. 169–196.
6. R. Kowalski and M. Sergot, “A logic-based calculus of events,” *New Generation Computing*, vol. 4, pp. 67–95, 1986.
7. A. Galton, “Reified temporal theories and how to unreify them,” in *Proceedings of IJCAI’91*, 1991.
8. L. Vila and H. Reichgelt, “The token reification approach to temporal reasoning,” *Artificial Intelligence*, vol. 83, no. 1, pp. 59–74, 1996. [Online]. Available: citeseer.ist.psu.edu/vila93token.html
9. C. Lagoze and J. Hunter, “The ABC ontology and model,” *Journal of Digital Information*, vol. 2, no. 2, 2001. [Online]. Available: <http://jodi.ecs.soton.ac.uk/Articles/v02/i02/Lagoze/>

10. C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari, "Wonderweb deliverable d18: Ontology library (final)," Laboratory for Applied Ontology - ISTC-CNR, Trento, Italy, Tech. Rep., 2003.
11. A. Pease, I. Niles, and J. Li, "The Suggested Upper Merged Ontology: A large ontology for the semantic web and its applications," in *Working Notes of the AAAI-2002 Workshop on Ontologies and the Semantic Web*, Edmonton, Canada, 2002.
12. C. Welty and N. Guarino, "Supporting ontological analysis of taxonomic relationships," *Data and Knowledge Engineering*, vol. 39, pp. 51–74, 2001.
13. A. L. Rector, "Modularisation of domain ontologies implemented in description logics and related formalisms including owl," in *Proceedings of the international conference on Knowledge capture*. ACM Press, 2003, pp. 121–128.
14. A. Swartz, "Musicbrainz: A semantic web service." *IEEE Intelligent Systems*, vol. 17, no. 1, pp. 76–77, 2002.