The Digital Music Lab: A Big Data Infrastructure for Digital Musicology

Samer Abdallah, University College London
Emmanouil Benetos, Queen Mary University of London
Nicolas Gold, University College London
Steven Hargreaves, Queen Mary University of London
Tillman Weyde, City University London
Daniel Wolff, City University London

In musicology and music research generally, the increasing availability of digital music, storage capacities and computing power both enable and require new and intelligent systems. In the transition from traditional to digital musicology, many techniques and tools have been developed for the analysis of individual pieces of music, but large scale music data that are increasingly becoming available require research methods and systems that work on the collection-level and at scale. Although many relevant algorithms have been developed during the last 15 years of research in Music Information Retrieval, an integrated system that supports large-scale digital musicology research has so far been lacking.

In the Digital Music Lab (DML) project, a collaboration between music librarians, musicologists, computer scientists, and human-computer interface specialists, the DML software system has been developed that fills this gap by providing a platform for large-scale music analysis with a user-friendly interactive interface supporting musicologists in their exploration and enquiry. The DML system empowers musicologists by addressing several challenges: distributed processing of audio and other music data, management of the data analysis process and results, remote analysis of data under copyright, logical inference on the extracted information and metadata, and visual web-based interfaces for exploring and querying the music collections. The DML system is scalable and based on Semantic Web technology, and it integrates into Linked Data with the vision of a distributed system that enables music research across archives, libraries and other providers of music data. A first DML system prototype has been set up in collaboration with the British Library and I Like Music Ltd. This system has been used to analyse a diverse corpus of currently 250,000 music tracks. In this article we describe the DML system requirements, design, architecture, components, available data sources, explaining their interaction. We report use cases and applications with initial evaluations of the proposed system.

CCS Concepts: •**Information systems** → **Multimedia information systems; Digital libraries and archives;** •**Applied computing** → **Sound and music computing;**

Additional Key Words and Phrases: Digital Musicology, Music Information Retrieval, Big Data, Semantic Web

ACM Reference Format:

Samer Abdallah, Emmanouil Benetos, Nicolas Gold, Steven Hargreaves, Tillman Weyde, Daniel Wolff, 2016. The Digital Music Lab: A Big Data Infrastructure for Digital Musicology. *ACM J. Comput. Cult. Herit.* X, X, Article XXXX (January 2016), 20 pages.
DOI: 0000001.0000001

Equally contributing authors listed in alphabetical order. This work was supported by the UK Arts and Humanities Research Council-funded projects 'Digital Music Lab - Analysing Big Music Data' (grant no. AH/L01016X/1) and 'An Integrated Audio-Symbolic Model of Music Similarity' (grant no. AH/M002454/1). EB is supported by a UK Royal Academy of Engineering Research Fellowship (grant no. RF/128).

Author's addresses: Samer Abdallah and Nicolas Gold, Department of Computer Science, University College London; Emmanouil Benetos and Steven Hargreaves, Centre for Digital Music, Queen Mary University of London; Tillman Weyde and Daniel Wolff, Department of Computer Science, City University London.

DOI: 0000001.0000001

1. INTRODUCTION

Musicology has traditionally relied on data of many kinds, such as scores and recordings representing music composition and performance as well as representations of other aspects of music, e.g. lyrics and metadata. Systematic musicology and ethnomusicology have often used quantitative methods, but typically on small datasets. In the last two decades the digitisation of communication and media has started an ongoing methodical shift in the humanities, which goes beyond the use of computational analysis of cultural artefacts and leads to new research questions and practices, generally described as Digital Humanities [Hughes et al. 2016]. In this context, Digital Musicology addresses not only the computational tools for analysing digital audio, scores and metadata but also the methods for musicological research in this context. This development has attracted increasing attention in recent years, e.g. from the European Science Foundation [Dahlig-Turek et al. 2012], the IMS study group on Digital Musicology[1] in the UK AHRC funded projects *Transforming Musicology*[2] and *A Big Data History of Music*[3] (the latter focusing on printed music), the Oxford Summer School workshop on Digital Musicology [4] as well as publications, e.g. [Pugin 2015; Duval et al. 2015; Ng et al. 2014].

Digital datasets in music are smaller than in some other domains, and according to [Burgoyne et al. 2016], of the openly accessible music datasets only the Million Song Dataset[5] qualifies as "truly 'big'" with 280GB of feature data extracted from 1 million audio tracks. However, the quantity of music data is growing and even the smaller data sets available now are big in the sense that the traditional musicological method, where the researcher closely inspects every work, is no longer applicable. In the Humanities in general, [Schöch 2013] claims that velocity and volume do not have the same role or requirements (respectively) as in the natural sciences or economics, but that the key shift is in the methodological change from studying individual artefacts to collection-level analysis. This position is well-characterised by Moretti's distant reading paradigm [Moretti 2013], i.e. the use of reduction and abstraction by quantitative analysis to understand culture, as if watching from a distance to observe large-scale structure. The work reported here might be considered a kind of 'distant audition' wherein music itself is not audited but instead features derived from it are analysed. This approach views not the details of individual works, but selections of pieces with respect to their aggregated properties, and compares similarities and differences between selections by specific aspects. Examples of these aspects of music are key or chord frequencies, harmonic progression patterns, or similarity structures, which can enable the understanding of commonalities, differences and trends in data collections across historic, geographic and cultural dimensions. An important aspect in this context is the goal of analysing music from multiple libraries and archives, which is necessary as specific recordings are often only available in specific institutions, and which will lead to joint datasets significantly larger than what is currently available. Since scholars in the humanities are not typically trained in the development or even the use of computing technology, there is a gap to bridge in order to make systems accessible to music researchers and to enable scholars to develop questions and seek answers that can be approached with the growing digital datasets and emerging computational tools.

In this paper we present the Digital Music Lab (DML) system, which addresses this gap by providing an environment for musicologists to explore and analyse large-scale music data collections, offering a range of tools and visualisations. We bring computation to the data, in order to enable the remote analysis of copyright-restricted material, and enable scalable interactive processing with large-scale

[1] http://ims-international.ch/content/index.php/study-groups/digital-musicology
[2] http://www.transforming-musicology.org
[3] https://www.royalholloway.ac.uk/music/research/abigdatahistoryofmusic/home.aspx
[4] http://dhoxss.humanities.ox.ac.uk/2015/digitalmusicology.html
[5] http://labrosa.ee.columbia.edu/millionsong/

parallelisation. The DML system is available as open source software[6], so that additional installations can be set up and connected via Semantic Web interfaces to create a distributed musicological research environment across institutions. Our first installation has access to a collection of over 1.2 million audio recordings from multiple sources (cf. Section 4 for details) across a wide range of musical cultures and styles, of which over 250,000 have been analysed so far, producing over 3 terabytes of features and aggregated data. Although the DML framework has been developed for musicology, it can be extended to digital data in other areas such as of visual arts or multi-modal linking and indexing of cultural artefacts. A preliminary version of the DML system is described in [Abdallah et al. 2016].

This paper presents requirements, design and technical architecture, as well as an implementation and initial evaluation of the DML system, showing how it addresses the needs of musicologists. Specifically, we make the following novel contributions:

— A set of requirements for music analysis as elicited from musicologists
— A system architecture for distributed analysis of large music collections
— A multi-level computation management system enabling interactive data exploration
— A system implementation that is available as open source software
— An installed and publicly available system
— Initial results from technical and user-based evaluations

The remainder of this paper is organised as follows: Section 2 describes related existing work. Section 3 introduces the requirements and concepts of the DML system. Section 4 gives an overview of the system architecture. Section 5 describes the middle-tier information management and gives some performance results. Section 6 provides information on the back-end compute service. Section 7 introduces the front-end interface and Section 8 describes the user evaluation and applications. Section 9 discusses achievements and challenges and Section 10 presents the conclusions of this article.

2. RELATED WORK

In order to develop Digital Musicology, research and technology from Music Information Retrieval (MIR) should be highly relevant. However, MIR has mostly focused on commercial use cases, and there has been relatively little interaction between musicology and MIR as shown by citation analysis of ISMIR papers [Neubarth et al. 2011]. On the other hand, musicology has applied computational methods for decades (e.g. see [Hewlett and Selfridge-Field 1991], [Volk et al. 2011]), typically on encoded musical scores, such as CCARH's kern and musedata collections[7]. The use of MIR at different scales for musicology has been addressed by [Rhodes et al. 2010] in the context of an experimental system which is no longer available. Larger amounts of data are nowadays stored in the collections of major libraries or commercial vendors, particularly audio, video and scanned documents, but these data are not normally accessible for academic analysis due to copyright restrictions.

For analysing large datasets the parallelisation approaches of MapReduce [Dean and Ghemawat 2008] and more recently in-memory and graph-based computing [Zaharia et al. 2012] have become widely used. This approach has proven popular as it provides a good trade-off between ease of development, efficiency of processing, and cost of setting up and running a system. For interoperability between multiple systems over the Internet, a natural approach is the Semantic Web [Berners-Lee et al. 2001]. The development of semantic tools and standards enables the description and linking of music information in the Semantic Web, e.g. see [Abdallah et al. 2006]. In projects like OMRAS 2 [Cannam et al. 2010], efforts were made to provide Semantic Web technology to music information systems. The OMRAS 2 project was working "Towards a Distributed Research Environment for Music Informatics and Com-

[6]See: http://dml.city.ac.uk/code
[7]http://kern.ccarh.org, http://www.musedata.org

Table I. High priority requirements of music researchers.

User Interface
Finding different ways to browse big music data collections
A user interface to define parameters (location, time, genre, etc.)
Usability for non-experts in technology
Metadata
Availability of metadata and annotations including genre tags

Analysis
Big data analysis of trends over time
Big Data analysis of styles
Similarity metrics
Harmonic analysis

putational Musicology"[Dixon et al. 2010], which, although not realised as an integrated system for humanities scholars, resulted in a set of technologies, several of which are used in the DML. An earlier approach to a distributed audio analysis and music information retrieval was proposed by [Al-Shakarchi et al. 2006] using peer-to-peer networking with a local application interface. Although technically interesting, this approach did not attract much support and web interfaces have generally been seen as a more effective way of providing interfaces to distributed systems.

More recently, several systems have been developed for presenting music collections, particularly of ethnomusicological recordings, on the web. The Telemeta system [Fillon et al. 2014] provides a function-rich framework for presenting music audio archives on the web. A similar approach is followed by the Dunya system, which supports browsing one of several collections in specific music cultures, each with a specific interface [Porter et al. 2013]. These systems are focused on searching and viewing audio recordings individually rather then analysing collections of recordings. For analysing data, the AcousticBrainz project [Porter et al. 2015] uses a crowdsourcing approach collecting feature data from audio that private contributors have on their computers. The extracted features are accessible via an online API[8].

3. REQUIREMENTS AND CONCEPTS OF THE DIGITAL MUSIC LAB

An initial workshop was held within the DML project on 19th March 2014, in which 48 musicologists and music researchers with varying degrees of computational expertise participated, discussing and collecting research questions and requirements for Big Data systems in musicology. Generally this was perceived as a novel topic for musicology, which was reflected in the diverse range of approaches and subjects suggested and discussed during the workshop. Table I lists the main points that musicologists brought up in the discussion: the user interface that is needed to use the system, the metadata that enables formulation of hypotheses and queries, and the content analysis that is required to answer the queries. Use cases that emerged in the workshop include the analysis of

— historical developments (e.g. tonality, chord progressions, tuning levels)
— music reception (e.g. playlists, concert programs, 'Hit Song Science')
— style analysis of players, composers, and genres

The discussion at the workshop included more specific topics, e.g. the analysis of the relation between guitar technology and guitar sounds or the relation between audio recordings and related texts (liner notes, reviews). The breadth of these topics indicated that the requirements are not yet specific enough to be immediately testable, but they were used as guidance in development over several iterations with musicologists, and evaluated toward the end of the project in a second workshop.

3.1 Big Data Collection Analysis

Based on the user input we developed our approach for the analysis of large music collections. In the following we describe the main concepts we used throughout the development and in this paper.

[8]http://acousticbrainz.org

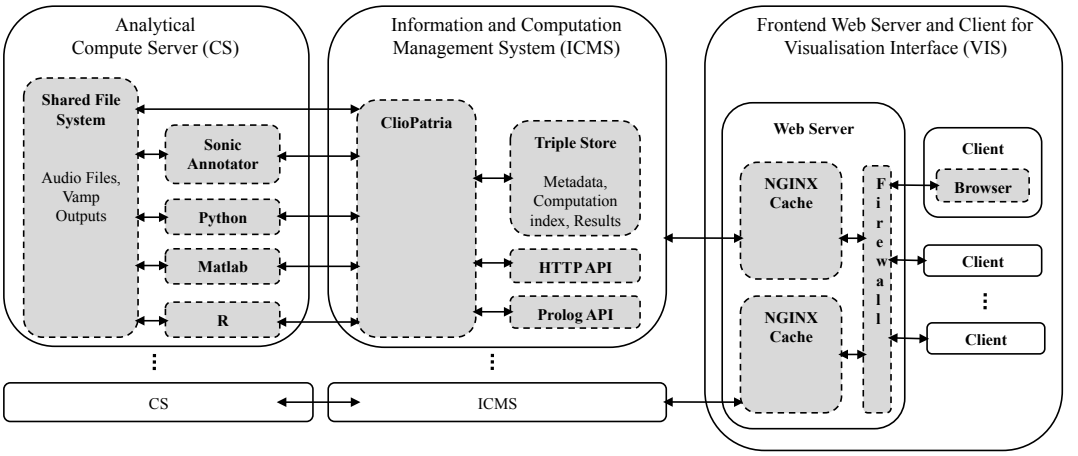

Fig. 1. Overview of the DML system. The web front-end (VIS) is connected to one or more information and computation management systems (ICMS) which request and store metadata as well as analysis results that have been computed by the back-end compute servers.

3.1.1 *Content.* The content accessible through the DML system is organised into *libraries*, *works* and *recordings*. The *library* information identifies the provider of the data, such as the British Library (BL), which may also provide physical access or further information, e.g. if the access is restricted by copyright. A *work* reflects a composition, which may have a digital score or other information associated with it. Works can be associated with one or more digital audio *recordings*, which are currently the main objects of analysis in the DML system. The DML manages internal files as needed and provides, if available, URLs for public download or streaming of audio. Based on metadata, recordings are grouped by the user into *collections*, which form the basis for analysis and inspection. Users can create collections by different criteria, e.g. music from Uganda vs. Ethiopia, and compare the analytic results.

3.1.2 *Analysis.* In the DML system we distinguish *recording-level analysis* (RLA) and *collection-level analysis* (CLA). An analysis on either level is defined as a triple of *perspective*, *parameters* and *target*. The target to be analysed can be either a recording or a collection. The perspective specifies a transformation of this data which may be based on multiple sub-transformations. If a transformation supports parameters, their values are defined with each instance of the transformation. Default values are used if no parameter information is provided. The triple of perspective, parameters and target identifies each analysis for storage and retrieval of previously computed results.

Even if the collections may not be as big as in some other domains, there are challenges caused by their size: first, each audio recording consists of millions of data points with needed to be analysed for their musical content, e.g. by automatic music transcription, which needs approximately as much processor time as the duration of the recording. Second, the data is richly structured: there is not just one analysis needed, but several different ones, e.g. for melody, harmony, rhythm, timbre, similarity structure, etc. These create a rich set of relations within and between works and collections, that is more complex than for instance typical textual data.

4. THE DML SYSTEM

4.1 Architectural Overview

As displayed in Figure 1, the DML system consists of three main components: the *Analytical Compute Servers (CS)* and the *Information and Computation Management Systems (ICMS)* in the back-end and the *Visualisation Interface (VIS)* in the front-end.

The computation of feature data is distributed to the CS servers, which are placed at the content providers for in-place processing. The CS instances extract features from media data and pre-compute aggregate statistics as far as possible. This model reduces network load and addresses copyright restrictions so that analysis of copyrighted audio material can be conducted, which has rarely been possible so far even in public libraries such as the BL.

An ICMS organises the available media and related information. It addresses data diversity with the use of Semantic Web technology and links the local contents to a graph of related internal and external data. Extracted features and aggregate data become part of this information graph. The ICMS schedules the computation and makes efficient use of existing information.

The VIS visualisation provides an end-user interface to define musicological queries and explore music datasets. It focuses on collection-level analysis, and presents individual and comparative perspectives of distinct collections to the user. For an analysis perspective requested via the VIS interface, a RESTful API call is made to an ICMS system (described in Section 5). This returns data that has been computed before or the ICMS determines the necessary analysis steps and triggers the corresponding computations on the CS server. In the latter case, the VIS notifies the user about the ongoing computation and displays the result when it becomes available. The CS performs the recording-level and collection-level analysis in parallelised processes. The result is returned to the ICMS which saves it to an RDF triple-store and forwards it to the requesting client interface.

The DML system supports scaling with the number of users and queries, the size of music collections and the number of connected DML installations. The client-side JavaScript implementation of the DML visualisation reduces the load on the server side. Multiple CS, each using multi-core processing, can analyse audio in different locations, and feed into the ICMS. A core contribution of the DML system is that it combines these parallelisation techniques with a fine-grained result caching system in the ICMS that allows the reuse of intermediate results amongst multiple queries. This feature together with multi-level caching enables visualisation response times of a few seconds for analyses that have previously been requested and reduced response times for queries requiring new computation.

4.2 The DML installation

We have created a running installation of the system, publicly available at http://dml.city.ac.uk/vis, with the CS components hosted at the BL and I Like Music Ltd (ILM) and the ICMS at City University London. The CS at ILM is not permanently connected for security reasons; pre-computed low-level features are extracted and regularly synchronised with the ICMS at City. The technical components are described in more detail in Sections 5 and 6.

4.3 Datasets and Music Collections

Four collections of audio recordings have been integrated to our DML installation, spanning many music cultures and genres, with scope to include more collections as the project grows. Over 250,000 recordings are currently available.

The BL holds over 3 million music recordings, of which approximately 10% are digitized, and over 49,000 were imported so far, which mostly originate from the Classical Music collection (∼19k recordings) and World and Traditional Music Collection (∼29k recordings). Each recording is also accompa-

nied by a metadata file in METS/XML format, which contains several identifiers on title, collection ID, composer, performer, recording date, geographic information, as well as audio file information. For a subset of these recordings the BL Sounds website provides audio streams, which can be listened to in the VIS interface.

The CHARM database [Beardsley and Leech-Wilkinson 2009][9]. was published by the AHRC Research Centre for the History and Analysis of Recorded Music. It contains digitised versions of 4,882 copyright-free historical recordings of classical music transferred from 78rpm discs, dated between 1902-1962, as well as metadata describing both the provenance of the recordings and the digitisation process.

The Mazurka database[10] was created in the UK AHRC-funded 'Mazurka Project', and contains 2,732 recorded performances for 49 Mazurkas by Frédéric Chopin, ranging from 1902 until recent years. The collection is accompanied with metadata on opus, key, performer, year, duration, and recording label.

I Like Music [11] has a repertoire of over 1 million commercial music recordings, of which we have yet analysed a selection of 6 music genres: jazz, rock & roll, reggae, classical, blues, and folk. Recording dates span from 1927, with the vast majority from the last two decades. We have so far analysed 216,523 audio recordings. The largest subset consists of folk music (approx 62k recordings), followed by jazz (approx 39k recordings). The collection metadata includes title, artist name, album title, genre, and release year.

5. THE INFORMATION AND COMPUTATION MANAGEMENT SYSTEM

The information and computation management (ICMS) sub-system of the DML has the job of organising and keeping track of the recordings, their metadata, and the details of any computations done on them. Its API serves results and the status of the computation to the interactive Vis interface. In addition, it is responsible for triggering new computations when required, and so must keep information about the functions available for application in new computations. These requirements are realised using a relational data model based on Semantic Web technologies [Berners-Lee et al. 2001] combined with a system for managing computations based on memoisation [Norvig 1991]. The approach derives from previous work on knowledge representations for music analysis and logic based systems for music analysis [Abdallah et al. 2006; Pastor et al. 2008], but supports web-interaction, collaborative working, and the essential building blocks for distributed data and computation. By integrating the DML data into the Semantic Web and it's conventions for e.g. representation of entities such as artists, the ICMS can not only scale to multiple ICMS instances, but is also able provide or draw information from external sources such as Musicbrainz or DBpedia[12]. In the digital humanities we find it a typical case that data is distributed, making such integration across services and sites necessary to scale to large datasets spanning multiple data sources.

5.1 Data Representation

The Semantic Web has at its core the Resource Description Framework (RDF): a data model built from sets of *triples*, which are simple logical statements of the form $(\langle Subject\rangle, \langle Predicate\rangle, \langle Object\rangle)$, such as $(\langle J.\ S.\ Bach\rangle, \langle composed\rangle, \langle The\ Goldberg\ Variations\rangle)$. The entities, or *resources*, referenced in these particular triples are not literal text, but atomic unique global identifiers, called *uniform resource identifiers* or URIs, denoting various entities such as people, events, places, musical works, etc.

[9] http://www.charm.kcl.ac.uk/sound/sound_search.html
[10] http://www.mazurka.org.uk/
[11] http://www.ilikemusic.com
[12] http://musicbrainz.org, http://wiki.dbpedia.org

For example, using real URIs, the resource ⟨*http://dbpedia.org/resource/Sun_Ra*⟩, is described by other triples, such as (now using standard prefix abbreviations)

(⟨*dbr:Sun_Ra*⟩, ⟨*rdf:type*⟩, ⟨*foaf:Person*⟩),
(⟨*dbr:Sun_Ra*⟩, ⟨*foaf:name*⟩, "Sun Ra"),
(⟨*dbr:Sun_Ra*⟩, ⟨*foaf:givenName*⟩, "Herman Poole Blount"),
(⟨*dbr:Space_Is_The_Place_(album)*⟩, ⟨*dbo:artist*⟩⟨*dbr:Sun_Ra*⟩),

where "Sun Ra" and "Herman Poole Blount" are literal character strings, and ⟨*foaf:Person*⟩ is a resource denoting a class of entity. Even the predicates (sometimes called 'properties') themselves can be described by other triples, such as

(⟨*dbo:artist*⟩, ⟨*rdfs:domain*⟩, ⟨*dbo:MusicalWork*⟩),
(⟨*dbo:artist*⟩, ⟨*rdfs:range*⟩, ⟨*dbo:Agent*⟩),

which indicate that the 'artist' property links musical works to agents (such as people or groups). In this way, a complex self-describing knowledge representation can be built up, as a collection of triples, referred to as an RDF database or a triple store. Because a triple can be thought of as a labelled *edge* connecting two *nodes*, an RDF database is also called a *graph*. A set of triples describing the classes and predicates for a particular application domain is an *ontology*; for example, the Music Ontology and Event Ontology [Raimond et al. 2007] define a broad range of terms useful for describing music, including many classes of entity involved (people, works, recordings, composition and performance events, etc.) and their relationships and are thus used throughout the ICMS.

5.2 Implementation framework

The ICMS is implemented in Prolog, a language whose data representation and programming model makes it a very effective match for managing relational data, avoiding the so-called 'object-relational impedance mismatch' [Ireland et al. 2009]. Furthermore, the implementation we used, SWI Prolog [Wielemaker et al. 2012], provides a substantial set of libraries for managing an RDF database, communicating with external databases, and building websites [Wielemaker et al. 2008]. ClioPatria [Wielemaker et al. 2015], is a Semantic Web application framework written in Prolog and brings together many components required to build a semantic web application into a modular and extensible system. The DML ICMS is written as a ClioPatria add-on package that provides many facilities for managing, exploring, and presenting music-related data.

5.3 Data import

The first step in making a music library available to the DML system is the translation of its metadata and audio files to a set of triples in the RDF database. To this end, a set of importers was written to handle the METS/XML [Gartner 2002] metadata (BL), MySQL databases supplied with CHARM and ILM data,] and the SQLite database created by the *beets*[13] music library management tool, and reference records in the Humdrum file format for symbolic scores where available. In most cases, the original metadata consists of plain text representations of recordings, artists, works, dates etc. rather than unique identifiers; for example, the CHARM library refers to the composer 'Bach', which, without further information, could refer to J. S. Bach, J. C. Bach, C. P. E. Bach, or indeed any other 'Bach'. In the interests of fidelity and to avoid the need for user intervention during the import process, such metadata is imported as is, without any attempt to resolve the names of artists and works to unique

[13] http://beets.radbox.org/

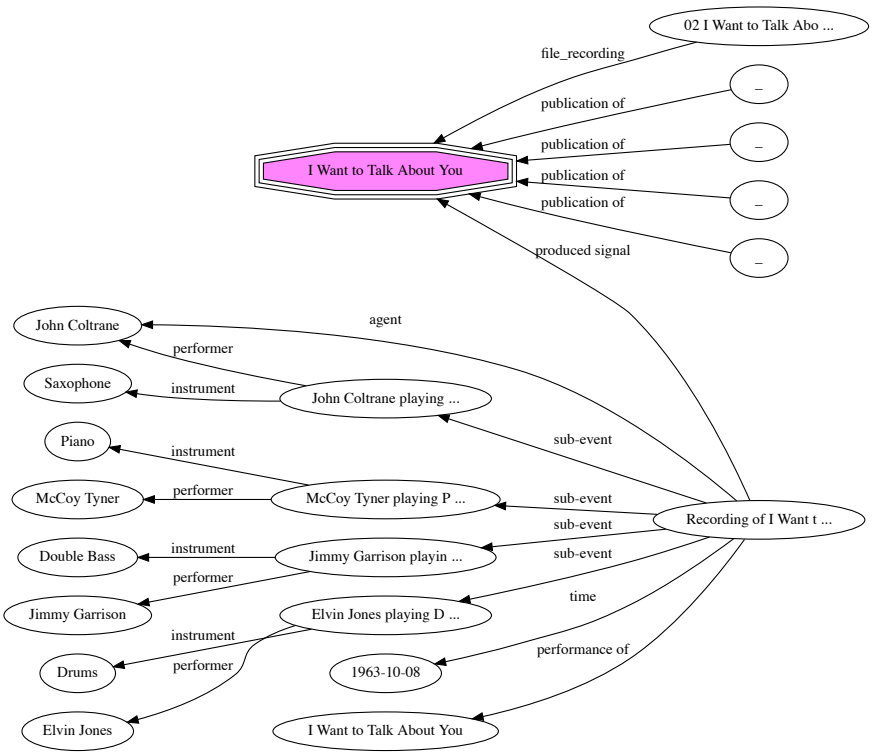

Fig. 2. Local RDF graph surrounding a recording of Eckstein's 'I want to talk about you' performed by the John Coltrane Quartet. Each node (except the recording date) represents an RDF resource that can be clicked on to find more information about that entity.

identifiers. Although such an automatic entity resolution (or *alignment* [Crawford et al. 2014]) process could be made relatively reliable in some cases (e.g. the names of classical music composers), it would in general require an inferential process based on all the information available about a recording, potentially followed by a user-guided review and correction process. This will be the subject of future work.

The one exception to this is the *beets* library importer: it can make use of audio fingerprinting and metadata matching to identify certain commercially available recordings and to match them to MusicBrainz (see Sec. 5.4) identifiers.

5.4 Musicbrainz-Music ontology integration

MusicBrainz[14] is a large online database of information about commercially available recordings. It uses a relational data model, with unique identifiers for all the main entities, such as artists, recordings, albums and works. Given a Musicbrainz identifier of a recording, the DML-ICMS can dynamically retrieve information surrounding that recording and add it to the RDF database using Music Ontology vocabulary. This includes detailed information about the recording such as dates, the roles of the people involved, who played what instrument, which musical works were recorded, etc. For example, a DML

[14]http://musicbrainz.org

page describing a recording of Billy Eckstein's 'I want to talk about you' by John Coltrane contains the derived triples and a graphical view of the RDF graph surrounding this resource (see Fig. 2 or online[15]). The graph can be used to reach (via the 'mo:performance_of' predicate) the DML page for the musical work 'I Want To Talk About You' and a complex graph showing all known recordings of it[16]. These data can be programmatically accessed through the SPARQL, Pengine and API endpoints of the ICMS.

The system for dynamically obtaining information from Musicbrainz is supported by an extension to ClioPatria, which also supports obtaining information from public SPARQL endpoints, such as *LinkedBrainz* (http://linkedbrainz.org/), or by dereferencing URIs that conform to Linked Open Data standards [Yu 2011], such as those published by DBpedia, the Semantic Web version of Wikipedia [Auer et al. 2007].

5.5 Computation management

Computation management in the ICMS is based on the idea that the relational data model is ideal for recording information about each computation performed by the system: a relational tuple or set of triples describing information about a computation can record which function was used, what each input parameter was set to, what output or outputs were produced, and metadata about the computation, such as whether or not it was successful, any error information, and its time and duration. When computation results are requested, the database can be checked to see if the result is already known. This process is known as *memoisation*, and the memo database contains all the information needed for a full exploration and further analysis of the results.

The memoisation process is augmented with a multi-threaded job control system to enable *parallel asynchronous memoisation*. If the results of a computation are required, the memo database is consulted to see if that particular computation (function+parameters) has been done before. If so, the result is returned. If not, the job control system is consulted to see if that computation is already in progress, in which case the status of the job, possibly including progress information and partial results computed on the basis of items processed so far, is returned. Finally, if the computation is not found, then a new job is submitted. On completion, the results of a job are recorded in the memo database. If the computation fails, information about that failure is recorded in the memo database. Thus, if a request is made for a previously failed computation, information about the failure is available and there is an option to retry the computation in case the implementation has been changed in the meantime.

Computations done using the VAMP system of audio analysis plugins [Cannam et al. 2006] are managed using the RDF database, as VAMP plugins are already described by RDF documents (e.g., the Silvet transcription plugin is described here[17]. Information about which VAMP plugins are available on the current system is obtained from Sonic Annotator (a program for running VAMP plugins). A VAMP 'transform' is a binding of a plugin to some specific parameter values. For example, this[18] is a page for a particular transcription transform which includes at the bottom a table of all the computations that were done using it. The resources in this table can be followed to find out more about the computation and its result.

The bulk of the VAMP analysis results currently in DML were pre-computed off-line using general purpose parallelisation frameworks (see Section 6) and imported into the ICMS. This approach is useful for low-level feature extraction on large collections which can take significant amounts of processing time. However, if the result of applying a particular transform to a particular recording is not available, then the computation is triggered automatically: the system manages the files involved in an

[15] http://dml.city.ac.uk/pub/JOCCH16/cp1.htm
[16] http://dml.city.ac.uk/pub/JOCCH16/cp2.htm
[17] http://dml.city.ac.uk/pub/JOCCH16/cp3.htm
[18] http://dml.city.ac.uk/pub/JOCCH16/cp4.htm

invocation of Sonic Annotator, puts the results file in a managed directory tree, and adds an entry to a dedicated VAMP memo database so that the results can be retrieved in future. Any memoised computations are furthermore accessible to other systems via the ICMS public interfaces described in Section 5.6.

Other analysis functions, mainly those that rely on the results of the primary VAMP-based analysis, can be written in several languages: Prolog, Matlab, R, or Python. The results are memoised in a persistent Prolog database, which is more suitable for computations that may have many input and output arguments. A table of all such memoised functions in the DML is available online, and the results of previous computations can be viewed[19].

Access to Matlab is provided via the Prolog-Matlab interface library *plml*[20], which uses the Matlab Engine API to start and communicate with a long-running separate Matlab process. Access to Python is via a separate invocation of the Python executable, with data exchange in JSON format over standard input and output. Access to R is via the Prolog-R interface library *Real* [Angelopoulos et al. 2013], which uses the R API to run an embedded instance of R within the same process and memory space as SWI Prolog itself. These mechanisms are used to implement a variety of secondary analysis methods that use the results of the various VAMP computations, both at the recording and collection levels (see Sec. 6.2).

Most computation is run in external processes such as Python, Sonic Annotator, Matlab, sox, lilypond, fluidsynth (see below for details about the latter), which means that in the event of an error, those processes can crash without bringing down the central server. This is an important consideration for a service which is expected to be available continuously. In contrast, computations in R (using the Real Prolog package) use an embedded R instance—this process is faster but leaves the server more vulnerable to errors generated by the R subsystem. Therefore R is preferred for smaller computation tasks, such as collection level statistics, as the overheads for invocation and data exchange are far lower than for comparable computations with the Matlab or Python interfaces, which are preferred for lower level task running of VAMP output or audio files.

5.6 Supported APIs for Presentation and Distributed Processing

The ICMS supports several web APIs for providing access to the information it contains to other systems and users. We use web APIs for obtaining rendered Matlab and R graphics, for obtaining audio streams in several formats, for obtaining symbolic scores in several formats including standard music notation rendered using the music typesetting system Lilypond, and for obtaining synthetic performances of symbolic scores using the fluidsynth MIDI synthesis program.

To support the VIS web application, the ICMS provides a RESTful web API for defining and managing datasets, obtaining information about recordings, and for requesting results for analysis on both recording and collection level. This API interacts with the asynchronous memoisation system and can retrieve progress information and partial results from computations in progress.

The DML system can be scaled with multiple ICMS using the Pengine (Prolog Engine) interface that allows several instances to interact using high-level Prolog language and consistent representation of data entities. This way, potentially pre-aggregated information for a specific analysis can be drawn from several sources before final aggregation in a single ICMS.

The integrated SPARQL endpoint provides a basis for data access by external clients and systems as well as other ICMS. The ClioPatria SPARQL endpoint supports this scenario with *Federated Queries*, which integrate queries on multiple remote service endpoints - in our case ICMS - and combine their

[19] http://dml.city.ac.uk/pub/JOCCH16/cp5.htm, http://dml.city.ac.uk/pub/JOCCH16/cp6.htm
[20] http://www.swi-prolog.org/pack/list?p=plml

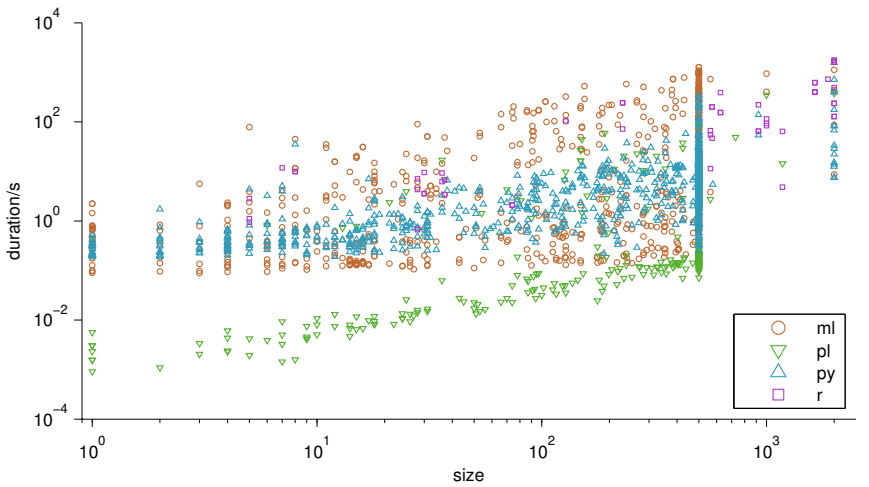

Fig. 3. Relationship between collection size and computation duration for collection level analyses, grouped by implementation language.

results. This further provides potential of scaling and allows for a strong interoperability with the Semantic Web.

5.7 Collection level performance statistics

Fig 3 shows some statistics summarising the performance of the ICMS running on the server based at City University over a total of 1918 collection level analyses (CLAs), implemented in Prolog, Python, Matlab and R. The data for this plot was gathered from the computation memo database which includes information about the times and duration of each computation. Most of the computations were triggered during the final DML workshop, where over 40 participants used the visualisation front end concurrently. The CLAs depend on RLAs which may or may not have been pre-computed and the RLA may have been implemented in a different language than the CLA. The small number of computations performed in R is due to the fact that support for R was added after the workshop, and so the statistics for these computations are not as informative as for the other languages.

Fig. 3 shows the relationship between collection size and computation duration. From this we can see that the overhead for computations implemented in Prolog is very much lower than for the other languages. Analysis done in Python requires the starting of new operating system processes and the communication of input and output data over Unix streams. Computations done in Matlab or R must be serialised due to the inherently single-threaded nature of both the Matlab engine API and the embedded R API. The results thus show that the strategy of integrating feature extraction into the database query system helps reduce time for data channelling and initialisation of the Python, R, or Matlab environments. However, they also show that the system is flexible enough to incorporate a range of analysis implementation languages, allowing for quick prototyping using existing code, with the option to improve performance by re-implementation subsequently.

6. BACK-END PROCESSING

A prerequisite for performing collection-level analysis (see Section 6.2), is the extraction, by means of VAMP plugins, of low and mid-level audio features (see Section 6.1) for the audio recordings under

consideration. This process, carried out using the batch tool *Sonic Annotator*[21], takes mostly a few seconds up to a few minutes for a typical audio file. In order to speed up the process for large collections, this back-end processing is parallelised. We used one server physically located at the premises of ILM (24 cores @2.4GHz, 128GB RAM, 14TB HDD) and a second at the BL (20 cores @3 GHz, 128GB RAM, 14TB HDD). This in-place access to data is at the core of the DML system design, enabling analysis on datasets that cannot be copied off-site due to copyright and licensing.

Lists of audio file URIs are sent in batches to the parallelisation engine, which calls sonic-annotator with a set of VAMP transform definitions to extract audio features. We tested different approaches and found Apache Spark[22] to provide flexibility for multiple servers and distributed multi-level analyses (e.g. including collection level analysis) using the map-reduce architecture. Processing the ILM dataset with 7 VAMP transforms using 20 CPU cores on the ILM server took 30 days.

6.1 Feature Extraction

A primary goal of the back-end system is to automatically extract and store low- and mid-level descriptors from individual audio recordings. The following list shows the low and mid-level features extracted in the DML (information on VAMP plugins is available online[23]):

(1) *Spectrograms* provide time-frequency content of the recordings, using the short-time Fourier transform or the constant-Q transform[Schörkhuber and Klapuri 2010].

(2) *Mel-frequency Cepstral Coefficients* (MFCCs) offer a compact representation of the frequency content of an audio signal; 20 MFCCs per time frame were extracted using the QM Vamp Plugin Set.

(3) *Chroma* projects the entire spectrum onto 12 semitone bins. Two implementations were used: QM Chromagram and NNLS Chroma Vamp [Mauch and Dixon 2010].

(4) *Onsets* represent the beginning of a musical note or other sounds in an audio signal using the QM Onset plugin [Bello et al. 2005].

(5) *Speech / music segmentation* on ethnographic and radio recordings was done using the BBC Speech/Music Segmentation plugin[24].

(6) *Chords* provide a concise description of musical harmony; we used the Chordino Vamp Plugin [Mauch and Dixon 2010].

(7) *Beats* were extracted using: Beatroot [Dixon 2007], Marsyas [Tzanetakis and Cook 2000], and Tempotracker [Davies and Plumbley 2007].

(8) *Tempo* following is strongly related to beats. We use the Tempotracker [Davies and Plumbley 2007] and Tempogram [Grosche et al. 2010].

(9) *Keys* are detected (in a Western tonal music context) with the QM Key plugin [Noland and Sandler 2007].

(10) *Melody* is estimated by the MELODIA Vamp plugin [Salamon and Gomez 2012]

(11) *Note transcription* from audio to music notation uses the Silvet vamp plugin [Benetos and Dixon 2012] two different settings: 12-tone equal temperament (for Western tonal music), and 20 cent resolution (for World & Traditional music).

[21] http://vamp-plugins.org/sonic-annotator/
[22] http://spark.apache.org
[23] http://www.vamp-plugins.org
[24] https://github.com/bbcrd/bbc-vamp-plugins

6.2 Collection-Level Analysis

Based on the low and mid-level features listed above, the DML system computes *collection-level* features for large-scale musicological analysis as shown below.

(1) *Key-relative chord sequences* combine information from the chord and key extraction features, extending [Barthet et al. 2014] to sequential pattern mining (CM-SPADE) on key-relative chord sequences.
(2) *Mean tempo curve* summarises tempo changes over the duration of a recording. The curve displays average normalised tempo vs. the normalised track length.
(3) *Pitch class histogram* summarises detected pitches from the (semitone-scale) *Note Transcription* in a histogram with octave-equivalent pitch classes 0-11 (C-B). The individual histograms are averaged across the collection based on the notes' duration. A tonic-relative variant of the pitch class histogram is also available.
(4) *Pitch histogram* aggregates all detected pitches over a collection of recordings, (without the octave-wrapping of the *pitch class histogram*). Information from *Note Transcription* is used in two versions: *semitone resolution* histogram (pitch on MIDI scale 0-127) and *fine resolution*, a fifth semitone (20 cent) summarised in 200 bins.
(5) *Similarity matrix* contains the pairwise feature similarity of the recordings in a collection, using a distance metric (Euclidean as in [Pampalk 2006] or normalised compression distance [Li et al. 2004]). The user can select any combinations of the following features: chords, chromagram, MFCCs.
(6) *Similarity plane* arranges recordings on a two-dimensional pane according to their similarity: similar recordings more closely together, dissimilar recordings farther apart. The spatial arrangement is determined using Multidimensional Scaling (MDS) [Borg and Groenen 2005] on the basis of the Similarity Matrix.
(7) *Tempo histogram* summarised all tempi detected using the QM tempo tracker Vamp plugin across the entire collection.
(8) *Tonic histogram* shows the tonic (i.e. key for Western tonal music) over all recordings as estimated by the QM key detector. In tonal music the last tonic detected is considered a good estimate for the entire piece.
(9) *Tuning stats* summarise the reference pitch distribution based on the 20 cent resolution *Note Transcription* feature in a histogram plus average and standard deviation. The tuning frequency is estimated per recording based on the precise F0 for all detected A, E and D notes.

7. FRONT-END INTERFACES

7.1 Data-Management Web Interface

ClioPatria provides a web interface for browsing and managing the RDF database. The core RDF concepts of triples, resources, predicates and classes are exposed, so that users can see all the triples for a given subject, or for a given predicate, and traverse the RDF graph by following links associated with resources, predicates, or literal values.

For example, a recording of 'Blackthorn Stick' is described in a page[25], which lists the predicate-object pairs for that subject as shown in figure. Clicking on the word 'Irish' next to the property ⟨*dc:language*⟩ retrieves all the recordings with that value of the language property. Clicking on the property ⟨*marcrel:Performer*⟩ yields information about the definition of that term, and scrolling down

[25]http://dml.city.ac.uk/pub/JOCCH16/cp7.htm

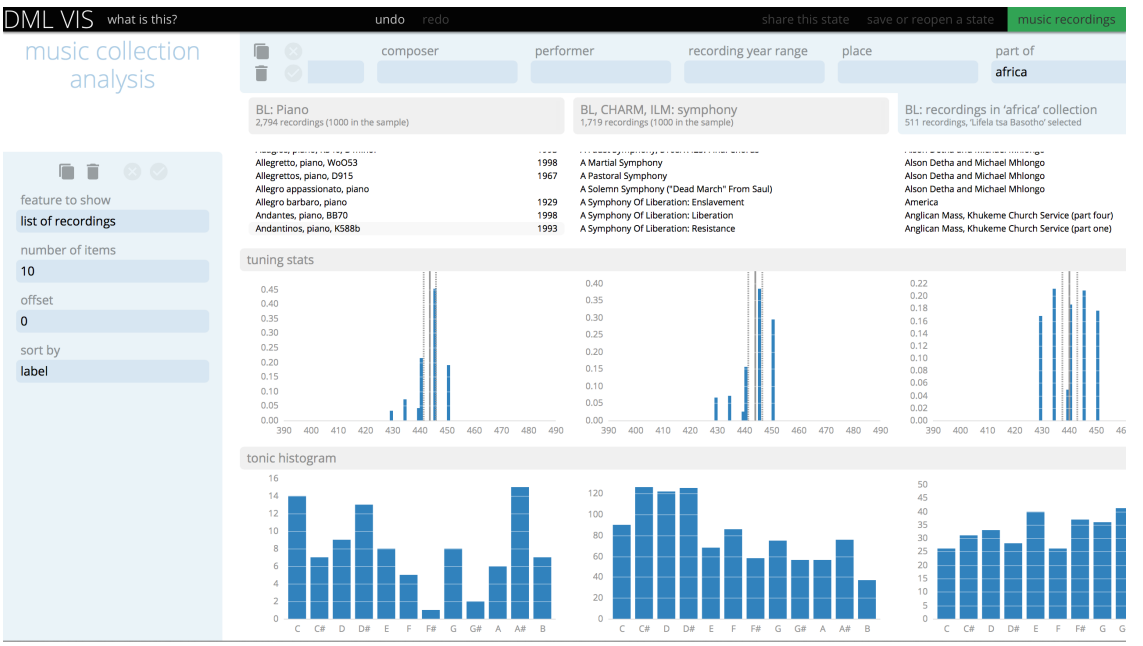

Fig. 4. Screenshot of the VIS interface: Analysis on user-defined collections (specified through the menu on the top). Three perspectives (list view, tuning statistics and tonic histogram) are shown as rows.

and clicking on the number under '#Distinct objects' in the 'Predicate statistics' table yields all of the values of the performer property[26]. From there, one can find all the recordings featuring a given performer.

ClioPatria also provides several hooks by which the display of information can be customised. We use them to enhance the presentation of music-related data in several ways. For example, if the audio data for a recording is available, then the page describing that recording is extended with an audio spectrogram, (computed and rendered dynamically in Matlab). If the audio is available publicly, then references to that resource are decorated with an HTML5 audio player. References to symbolic scores are decorated with an audio player and a link to a page[27] displaying the score in standard music notation (created dynamically using *Lilypond*), and an interface that allows the score to be played back with control over tempo and transposition (using the synthesis program *fluidsynth*). Similarly, the page representing the results of an automatic transcription is augmented with a sonification interface and a piano-roll view of the transcription[28].

7.2 Collection-level Visual Interface (VIS)

Our VIS interface supports musicologists in exploratory data analysis on the collection level. The VIS interface is publicly available[29] and a screenshot is shown in Figure 4. The VIS interface runs in a web browser, and the visualisations are created using an array of state-of-the-art web technologies.

[26] http://dml.city.ac.uk/pub/JOCCH16/cp8.htm
[27] http://dml.city.ac.uk/pub/JOCCH16/cp9.htm
[28] http://dml.city.ac.uk/pub/JOCCH16/cp10.htm
[29] http://dml.city.ac.uk/vis

From a systems perspective, this approach makes heavy use of processing in the browser, which reduces the central system load and makes it, together with HTTP level caching using NGINX, more scalable and responsive than server-based visualisation.

The VIS design is based on a grid of collections in the columns and analyses in the rows. In each cell of the grid, a visual representation, such as a bar chart, graph or histogram, is shown, including tool tips for individual values at mouse point. The user defines collections by selecting libraries and providing search terms in metadata fields at the top of the screen. Analyses are selected from a list and parametrised on the left. A comparison visualisation can be generated between columns to view differences between collections, e.g. composers or regional styles. Further discussion of the specifics of the technology and the human interaction design are out of the scope of this article.

8. USER EVALUATION AND CASE STUDIES

8.1 User Evaluation

A user-based evaluation took place during a second DML workshop on 13 March 2015; in attendance were 40 participants with interests in digital music and musicology (academic, commercial, and public sector). Participants were asked to carry out two tasks from the following list in pairs using the VIS interface.

—*Tuning Frequency*: identify trends in orchestral pitch over time

—*Pitch Profile*: identify, compare, and explore pitch class sets and pitch hierarchies

—*Tempo Curves*: identify historical trends in classical music through tempo summaries

We asked participants to rate their level of agreement with the statement *these kinds of tools could help with the task in hand*. In total, we received 28 partial and 16 complete responses from participants. Out of the 16 complete responses, 9 participants strongly agreed that the developed tools can help with the task in hand, while 6 participants agreed (on a scale of 1 to 5, with 5 indicating strong agreement).

Most suggestions for improvements were about usability of the user interface (server messages, data access speeds), many of which have been addressed in the current version. Apart from those, the most frequent requests were the addition of more genres of music (e.g. electronica, US country music), integration with other services (e.g. Echonest), and of more metadata (e.g. duration information) to support more powerful search and selection. During the user study, several musicological interpretations and reflections were made by the participants, e.g. “Uganda’s music uses a wider range of pitches than Dinka”, “it seems that the tuning of symphonies was more accurate in 1980-2010 than in the earlier period” (cf. Section 8.2).

8.2 Musicological Analysis

The DML system aims to enable musicological analysis of large collections, providing musicologists with descriptors and tools based on music information retrieval, expanding on previous work on technologies for musicological research (such as Sonic Visualiser and the Sonic Annotator Web Application) [Fazekas et al. 2010].

More specifically, the DML system can be used for empirical/systematic musicological research focusing on *comparative* analysis between collections of recordings, using various cues, such as: spatial/geographical information, pitch/tonality/harmony (e.g. pitch histograms, chord sequences, tonic, tuning), rhythm (tempo curves/histograms), as well as similarity research across a music collection. The modular structure of the proposed system also enables the insertion of additional descriptors and tools that can be of benefit to musicological research.

8.2.1 *Application examples.*

8.2.1.1 *Tuning frequencies.* The development of tuning frequencies over time and between instrumentations is a question that can be addressed with the DML. We performed exploratory experiments, as exemplified in Figure 4, which shows a comparison of symphonic recordings vs. piano recordings, with the latter showing lower spread. Regarding pitch levels, we found that already in the 1950s there was an elevated pitch level of 444Hz, despite the then recent standardisation of 440Hz, but the results vary depending on instrumentation. These initial findings justify further studies from a musicological perspective[30].

8.2.1.2 *Traits of successful musicals.* The DML system was used in a study commissioned by *Sky Arts* regarding characteristics of successful musical theatre music, as compared to less successful musicals (commercially or by critics' judgement). Harmonic, dynamic and tempo-related patterns were found and used by a team that then created a computer-generated musical that will be performed in London's West End[31].

8.3 Music Archive Management

The information management system presented in Sec. 5 can be used for integrating music data, derived features, and metadata across heterogeneous collections and archives, as exemplified by the integration of several databases and archives, as described in Section 4. It is also worth noting that while the public DML interface focusses on displaying/exploring audio collections, the ICMS additionally supports the use of machine-readable music notation in formats such as MIDI and kern, and linking audio and symbolic notation for a given music piece or performance. This, combined with the MusicBrainz integration (Section 5.4) offers additional metadata resources, which are not provided with the original metadata.

8.4 Music Similarity

The use of content-based similarity descriptors can be directly useful for music recommendation applications [McFee et al. 2012]. The similarity features of the DML system, as presented in Section 6.2, can be used to that end. The modular use of the features (supporting any combination of input low-level features, as well as various distance measures) also allows for a thorough evaluation of music similarity estimators, across several music collections. Expanding the concept of music similarity beyond audio recordings is part of the ASyMMuS project on audio-symbolic music similarity[32], which relies on the DML infrastructure.

9. DISCUSSION

Overall, the feedback from the final workshop and from musicologists we interacted with was very positive. The system was seen as generally providing the required functionality and providing great potential for advancing Digital Musicology. The main shortfall that was mentioned is the lack of consistent metadata and search facilities to enable more context-related queries. There were many suggestions regarding the visual interface, which are outside the scope of this paper as it focuses on the software system. There are some issues relating to the system design that have not fully been resolved during the DML project or have emerged as new problems requiring future work.

Extended and linked metadata The resolution of metadata entities to integrate with the Semantic Web has not been fully achieved. This requires mapping the names of things to URIs as described above, and needs substantial work that is outside the scope of our project. Having metadata fully linked to

[30]We would like to thank Prof Stephen Cottrell for musicological contribution and guidance.
[31]http://www.broadwayworld.com/article/VIDEOS-Computer-Generated-Musical-BEYOND-THE-FENCE-Prepares-For-Opening-20160216
[32]http://dml.city.ac.uk/asymmus/

the Semantic Web would be extremely useful, as musicologists did ask for very specific queries that require extensive linking to outside sources of knowledge, including authority files, DBpedia and other data sources.

Size of metadata and results Currently, the results of collection level analyses are stored using the SWI Prolog *persistency* library to manage these memo tables, which uses the in-memory Prolog database backed by an on-disk journal in a text-based format. While this means that accessing the database is fast, it also means that initially reading the database is getting slower as the database gets larger, and the pressure on system memory will eventually become a problem. Alternatives currently being developed are to use an SQL-based database or the Berkeley DB format for these memo tables. In addition, the use of files to store large values instead of writing them literally in the database, as explored in [Pastor et al. 2008], would alleviate this problem.

Parallelisation Prolog's declarative nature should make it relatively straightforward to parallelise collection level analyses which follow the map-reduce paradigm, though to make effective use of this it will be necessary avoid the use of Matlab or R for numerical computations, as both of these libraries are single-threaded. Computations done in Python or Sonic Annotator avoid this problem, but incur the overhead of starting an operating system process for each item. Alternatively, parallelisation could be done in Matlab or R using their own parallelisation mechanisms.

Distributed data and computation Musicologists could greatly benefit from a distributed system that enables analyses on data held by different libraries and archives. In a system distributed over several ICMS, analysis that requires computations needs to be triggered on the server that does have access to the audio, and the results made available to other systems. The built-in Pengine (Prolog engine) API already provides means for multiple ICMS to interoperate at the level of remote Prolog calls. This way we plan to scale up the DML system and integrate further content via new ICMS and CS at other libraries. For integration with existing Linked Data systems, federated SPARQL queries across multiple servers provide a standardised way to integration. If more scaling is required because of large numbers of clients, triple fragments, as recently proposed by [Verborgh et al. 2014], may be a potential solution.

10. CONCLUSIONS

We have proposed the DML system as an approach to bridge the gap between musicology, music information retrieval and Big Data back-end technology. The first instantiation of the system enables music researchers to explore and analyse substantial music audio collections and datasets. Our evaluations showed that the combination of different facets of music, audio analysis, musical structure, and metadata is of value to musicology, as it enables researchers to understand music in its context and conduct comparative analyses between music collections.

To support fast and interactive exploration and retrieval of results we developed an integrated system using intelligent computation management, including result-caching on several levels. This allows for efficient analysis of large music collections, based in remote locations where they are locally analysed by compute servers. Combining existing technology with intelligent knowledge management techniques, we developed new collection-analysis tools that are now accessible to end-users through the interactive graphical VIS interface. The DML system can process copyrighted music data by bringing the computation to the data, which enables computational analysis of recordings at the BL for the first time. Our use of standardised data formats renders both the analysis back-end and the visualisation extensible to new analyses and datasets. Given the outcome of the development and the positive feedback from the users, the creation and connection of multiple DML systems at other libraries, archives or commercial music providers is a very interesting prospect, as it can potentially create unprecedented coverage and insights in music research. Finally, the DML system can serve as a paradigm for digital

humanities applications outside the realm of music and musicology, as an open framework for linking and providing access to raw and derived data with metadata in copyright-restricted collections of any digital or digitised cultural artefact.

REFERENCES

S. Abdallah, E. Benetos, N. Gold, S. Hargreaves, T. Weyde, and D. Wolff. 2016. Digital Music Lab: A Framework for Analysing Big Music Data. In *24th European Signal Processing Conference (EUSIPCO)*. EURASIP, Budapest, Hungary.

S. Abdallah, Y. Raimond, and M. Sandler. 2006. An ontology-based approach to information management for music analysis systems. In *Proc. 120th AES Convention*. Paris, France.

E. Al-Shakarchi, I. Taylor, and S.D. Beck. 2006. DART: A Framework for Distributed Audio Analysis and Music Information Retrieval. (2006).

N. Angelopoulos, V.S. Costa, J. Azevedo, J. Wielemaker, R. Camacho, and L. Wessels. 2013. Integrative functional statistics in logic programming. In *Practical Aspects of Declarative Languages*. Springer, 190–205.

S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. 2007. *Dbpedia: A nucleus for a web of open data*. Springer.

M. Barthet, M. Plumbley, A. Kachkaev, J. Dykes, D. Wolff, and T. Weyde. 2014. Big Chord Data Extraction and Mining. In *Conf. on Interdisciplinary Musicology*.

R. Beardsley and D. Leech-Wilkinson. 2009. A Brief History of Recording. http://www.charm.kcl.ac.uk. (2009). Centre for the History and Analysis of Recorded Music.

J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler. 2005. A tutorial on onset detection of music signals. *IEEE Trans. on Audio, Speech, and Language Process.* 13, 5 (Sept. 2005), 1035–1047.

E. Benetos and S. Dixon. 2012. A Shift-Invariant Latent Variable Model for Automatic Music Transcription. *Comput. Music J.* 36, 4 (2012), 81–94.

T. Berners-Lee, J. Hendler, O. Lassila, and others. 2001. The semantic web. *Sci. Amer.* 284, 5 (2001), 28–37.

I. Borg and P.J.F. Groenen. 2005. *Modern multidimensional scaling* (2 ed.). Springer, New York, USA.

J.A. Burgoyne, I. Fujinaga, and J.S. Downie. 2016. Music Inform. Retrieval. In *A New Companion to Digital Humanities*, S. Schreibman and R. Siemens (Eds.). Wiley, 213–228.

C. Cannam, C. Landone, M.B. Sandler, and J.P. Bello. 2006. The Sonic Visualiser: A Visualisation Platform for Semantic Descriptors from Musical Signals.. In *ISMIR*. 324–327.

C. Cannam, M. Sandler, M.O. Jewell, C. Rhodes, and M. d'Inverno. 2010. Linked data and you: Bringing music research software into the semantic web. *J. of New Music Res.* 39, 4 (2010), 313–325.

T. Crawford, B. Fields, D. Lewis, and K. Page. 2014. Explorations in Linked Data practice for early music corpora. In *Digital Libraries (JCDL), 2014 IEEE/ACM Joint Conf. on*. IEEE, 309–312.

E. Dahlig-Turek, S. Klotz, R. Parncutt, and F. Wiering (Eds.). 2012. *Musicology (Re-) Mapped: Discussion Paper*. European Sci. Foundation.

M.E.P. Davies and M.D. Plumbley. 2007. Context-Dependent Beat Tracking of Musical Audio. *IEEE Trans. on Audio, Speech, and Language Process.* 15, 3 (March 2007), 1009–1020.

J. Dean and S. Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. of the ACM* 51, 1 (2008), 107–113.

S. Dixon. 2007. Evaluation of the Audio Beat Tracking System BeatRoot. *J. of New Music Res.* 36, 1 (2007), 39–50.

S. Dixon, M. Sandler, M. d'Inverno, and C. Rhodes. 2010. Towards a Distributed Research Environment for Music Informatics and Computational Musicology. *J. of New Music Res.* 39, 4 (2010), 291–294.

E. Duval, M. van Berchum, A. Lentzsch, G.A. Parra, and A. Drakos. 2015. Musicology of Early Music with Europeana Tools and Services. In *Proc. of the 16th Internat. Soc. for Music Inform. Retrieval Conf., ISMIR 2015, Málaga, Spain, October 26-30, 2015*. 632–638.

G. Fazekas, Y. Raimond, K. Jacobson, and M. Sandler. 2010. An Overview of Semantic Web Activities in the OMRAS2 Project. *J. of New Music Res.* 39, 4 (2010), 295–311.

T. Fillon, J. Simonnot, M.F. Mifune, S. Khoury, G. Pellerin, and M. Le Coz. 2014. Telemeta: An Open-source Web Framework for Ethnomusicological Audio Archives Management and Automatic Analysis. In *Proc. of the 1st Internat. Workshop on Digital Libraries for Musicology (DLfM '14)*. ACM, New York, NY, USA, 1–8.

R. Gartner. 2002. METS: Metadata Encoding and Transm ission Standard. *JISC Techwatch report TSW* (2002), 02–05.

P. Grosche, M. Müller, and F. Kurth. 2010. Cyclic tempogram - a mid-level tempo representation for music signals. In *Proc. of the 2010 IEEE Internat. Conf. on Acoustics, Speech, and Signal Process. (ICASSP 2010)*.

W.B. Hewlett and E. Selfridge-Field. 1991. Computing in musicology, 1966–91. *Comput. and the Humanities* 25, 6 (1991), 381–392.

L. Hughes, P. Constantopoulos, and C. Dallas. 2016. Digital methods in the humanities: understanding and describing their use across the disciplines. In *A New Companion to Digital Humanities*, S. Schreibman and R. Siemens (Eds.). Wiley, 150–170.

C. Ireland, D. Bowers, M. Newton, and K. Waugh. 2009. A classification of object-relational impedance mismatch. In *Advances in Databases, Knowledge, and Data Applications, 2009. DBKDA'09. First Internat. Conf. on*. IEEE, 36–43.

M. Li, X. Chen, X. Li, B. Ma, and P.M.B. Vitanyi. 2004. The similarity metric. *IEEE Trans. on Inform. Theory* 50, 12 (Dec. 2004), 3250–3264.

M. Mauch and S. Dixon. 2010. Approximate Note Transcription for the Improved Identification of Difficult Chords. In *Proc. of the 11th Internat. Soc. for Music Inform. Retrieval Conf. (ISMIR 2010)*. 135–140.

B. McFee, L. Barrington, and G. Lanckriet. 2012. Learning Content Similarity for Music Recommendation. *IEEE Trans. on Audio, Speech, and Language Process.* 20, 8 (Oct. 2012), 2207–2218.

F. Moretti. 2013. Conjectures on World Literature. In *Distant reading*. Verso, London ; New York.

K. Neubarth, M. Bergeron, and D. Conklin. 2011. Associations between Musicology and Music Information Retrieval.. In *ISMIR*. 429–434.

K. Ng, A. McLean, and A. Marsden. 2014. Big Data Optical Music Recognition with Multi Images and Multi Recognisers. In *Proc. of the EVA London 2014 on Electron. Visualisation and the Arts (EVA London 2014)*. BCS, 215–218.

K. Noland and M. Sandler. 2007. Signal Processing Parameters for Tonality Estimation. In *Audio Engrg. Soc. Convention 122*.

P. Norvig. 1991. Techniques for automatic memoization with applications to context-free parsing. *Comput. Linguistics* 17, 1 (1991), 91–98.

E. Pampalk. 2006. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. Ph.D. Dissertation. Vienna University of Technology, Vienna, Austria.

D. Pastor, Y. Raimond, S. Abdallah, and M. Sandler. 2008. A Logic-Based Framework for Digital Signal Process. (2008).

A. Porter, D. Bogdanov, R. Kaye, R. Tsukanov, and X. Serra. 2015. AcousticBrainz: a community platform for gathering music information obtained from audio. In *16th Internat. Soc. for Music Inform. Retrieval (ISMIR) Conf.*

A. Porter, M. Sordo, and X. Serra. 2013. Dunya: A system for browsing audio music collections exploiting cultural context. In *Proc. of 14th Internat. Soc. for Music Inform. Retrieval Conf. (ISMIR 2013), Curitiba, Brazil*, Vol. 11.

L. Pugin. 2015. The challenge of data in digital musicology. *Frontiers in Digital Humanities* 2 (2015), 4.

Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson. 2007. The Music Ontology. In *Internat. Conf. on Music Inform. Retrieval, ISMIR 2007*.

C. Rhodes, T. Crawford, M. Casey, and M. d'Inverno. 2010. Investigating music collections at different scales with AudioDB. *J. of New Music Res.* 39, 4 (2010), 337–348.

J. Salamon and E. Gomez. 2012. Melody Extraction From Polyphonic Music Signals Using Pitch Contour Characteristics. *IEEE Trans. on Audio, Speech, and Language Process.* 20, 6 (Aug 2012), 1759–1770.

C. Schöch. 2013. Big? Smart? Clean? Messy? Data in the Humanities. *J. Dig. Hum.* 2, 3 (2013), 2–13.

C. Schörkhuber and A. Klapuri. 2010. Constant-Q transform toolbox for music processing. In *7th Sound and Music Comput. Conf.* Barcelona, Spain.

G. Tzanetakis and P. Cook. 2000. MARSYAS: a framework for audio analysis. *Organised Sound* 4 (12 2000), 169–175. Issue 03.

R. Verborgh, O. Hartig, B. De Meester, G. Haesendonck, L. De Vocht, M. Vander Sande, R. Cyganiak, P. Colpaert, E. Mannens, and R. Van de Walle. 2014. Querying Datasets on the Web with High Availability. In *Proc. of the 13th Internat. Semantic Web Conf. (Lecture Notes in Comput. Sci.)*, P. et al Mika (Ed.), Vol. 8796. Springer, 180–196.

A. Volk, F. Wiering, and van P. Kranenburg. 2011. Unfolding the potential of computational musicology. In *Proc. of the 13th Internat. Conf. on Informatics and Semiotics in Organisations*.

J. Wielemaker, W. Beek, M. Hildebrand, and J. van Ossenbruggen. 2015. ClioPatria: A SWI Prolog Infrastructure for the Semantic Web. *Semantic Web Journal* (2015).

J. Wielemaker, Z. Huang, and L. Van Der Meij. 2008. SWI-Prolog and the Web. *Theory and practice of logic programming* 8, 03 (2008), 363–392.

J. Wielemaker, T. Schrijvers, M. Triska, and T. Lager. 2012. SWI-Prolog. *Theory and Practice of Logic Programming* 12, 1-2 (2012), 67–96.

L. Yu. 2011. Linked open data. In *A Developer's Guide to the Semantic Web*. Springer, 409–466.

M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M.J. Franklin, S. Shenker, and I. Stoica. 2012. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proc. of the 9th USENIX conf. on Networked Syst. Design and Implementation*. USENIX Assoc., 2–2.