The Networked Environment for Music Analysis (NEMA)

Kris West*, Amit Kumar†, Andrew Shirk† and J. Stephen Downie †
*Widget Works Ltd
Norfolk, UK
Email: kris.west@gmail.com
†IMIRSEL, The Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
Email: mirproject@lists.lis.illinois.edu

***Abstract*—Conducting valid comparative evaluations of techniques in the field of Music Information Retrieval (MIR) presents particular challenges to MIR researchers due to issues of copyright and data sharing. Further, the interdisciplinary nature of MIR research and multi-faceted nature of human music perception make the sharing and reuse of techniques and implementations for particular facets of music perception and music information retrieval tasks highly desirable. However, the field makes use of a diverse range of file formats, software environments and toolkits for extracting, encoding and accessing MIR data and services, making reuse extremely challenging and often forcing the reimplementation of standard procedures significantly increasing the complexity of implementing linked data systems.**

The NEMA project aims to provide the MIR field with a high-quality, secure and extensible workflow environment to facilitate: computation over remote audio and resource collections; optimal code reuse, interoperability, sharing and dissemination; standardised, high-quality evaluation procedures; and the encoding of metadata, data and results in a format suitable for use in a detailed linked data system for MIR.

I. Motivation

Unlike many other fields of multimedia research, Music Information Retrieval (MIR) suffers from extreme issues of data sharing and comparative evaluation. Owing to the current climate in the music industry datasets (for example, those drawn from the set of western commercial music) cannot be freely shared between researchers. Further, the purchase of licenses to the constituent tracks and reconstruction of a dataset is often prohibitively expensive.

One solution to this problem is the use of creative commons collections, particularly those with research friendly licensing, such as the Magnatune collection [1]. However, such collections are often relatively small scale and they rapidly become well known to the research community (risking the over-fitting of multiple techniques and models to those collections). Further, multiple authors [2], [3] have reported significant experimental errors introduced by poorly chosen experiment criteria, such as allowing an *artist* to appear in both testing and training sets for *genre* classification experiments (as it may be easier to fit characteristics of artists rather than genres of music). Hence, experiments must be both carefully constructed and exactly repeatable to facilitate valid statisitcal comparisons between techniques.

A. The Music Information Retrieval Evaluation eXchange

These challenges led to establishment of an annual 'Music Information Retrieval Evaluation eXchange' or MIREX [4] hosted by the IMIRSEL lab at the University of Illinois at Urbana-Champaign. MIREX is based on the TREC approach [5], where algorithms and applications are submitted by the MIR community to up to 26 separate, community defined tasks and evaluated using standardised queries, collections and evaluation measures. The large number of tasks and datasets involved necessitates the sending of the code for each submission to the data, rather than the reverse as is the case in most if not all other fields.

Since the establishment of MIREX in 2005 (and its predecessor, the 'Audio Description Contest' hosted by the Music Technology Group at Universitat Pompeu Fabra in 2004), the annual evaluation campaign has been a strong driver of research in MIR by encouraging the community to clearly define tasks and research goals, develop and gain consensus on standardised evaluation procedures and provide detailed and statistically valid comparisons of techniques for a given task. Finally, in order to run MIREX a series of extremely simple Unicode file format standards have been developed to encode data for each highly specific task context, in order to keep the implementation cost for participating systems as low as possible.

Given that MIREX submissions must be delivered to MIREX and executed by the IMIRSEL team on the task datasets, the running of a MIREX evaluation is no small task. In 2009 alone IMIRSEL received submissions from 138 individuals, in 26 tasks, generating a total of 289 runs. The association of MIREX with the annual ISMIR conference, where the results are presented, can also be problematic as this means that most or all of the submissions will be received and must be executed within the same eight week period each year. Further, the research-grade code received is often problematic owing to missing dependencies or

platforms, variance from the defined IO file formats, poor error handling/reporting, lack of runtime feedback (is it working?) and extraordinary runtimes. Finally, the reporting and dissemination of results is also a challenge as detailed statistical analyses must be rapidly conducted and then presented for each of the hosted tasks in time for the ISMIR conference. This presently takes the form or a report encoded in page on that years MIREX wiki[1].

MIREX has brought about standardised evaluation procedures and some limited, but standardised file formats. However, MIREX has so far failed to facilitate easy cross-pollination of research systems and code. Owing to the relative wealth of development environments, programming languages, software frameworks and toolkits used or developed for MIR research, such collaborations are often challenging as there exist few standards for encoding the wide range of music and music informatics data. Where standards do exist (such as the Music, Audio Feature and other ontologies developed by the OMRAS2 project for encoding MIR data in RDF [6]) uptake is as yet relatively low. Further, although MIREX provides a snapshot of the state-of-the-art in each MIREX task, it does not facilitate access to a wide range of published MIR techniques for less technical members of the MIR community such as Musicologists and LIS researchers (the natural consumers of many MIR techniques). Hence, the NEMA project was established to address many perhaps all of these issues.

II. Solution

The stated goals of the NEMA project include the establishment of a computational infrastructure to facilitate submission of MIR compute tasks against remotely held content and metadata collections, interoperability between almost arbitrary systems for well defined MIR tasks, standardised and automated experimentation/evaluation and the exposure of metadata and results in a standardised form suitable for integration with a full linked data architecture for MIR.

Though the NEMA project is not intended to replace MIREX, MIREX offers an excellent starting point for a field-wide computational infrastructure and interoperability project as defines standardised tasks, experiment formats, collections and low-level file formats that are already support by a very large number of implementations across the research community.

A. Key concepts

1) The NEMA Data Model: The NEMA infrastructure includes the definition of an extensible data model and data structures that can encode data relating to all aspects of music and, music metadata, music informatics and MIR evaluation that have been encountered at MIREX, in the jMIR framework [7] or in the MIR related ontologies developed by the OMRAS2 project [6]. In order to facilitate interoperability between diverse components forming part of or submitted to NEMA by third parties, this data model is exclusively used to communicate between components of the system. Each third party component application used with the NEMA service is wrapped in a component harness that uses contextual information about both the experiment and third party component code to raise or lower the data model to or from a suitable file format in order to supply data to and harvest results from the application, construct valid calls to the application to achieve the desired result and satisfy resource and dependency requirements in order to run the application.

2) Workflow and Dependency infrastructure: The NEMA infrastructure is initially intended to support component harnesses to execute Java, Matlab, VAMP API plugins [8] and arbitrary binary executables and shell scripts (that launch processes compatible with the host operating system and architecture). This provides an immediate path to integration of most, if not all software types and offers a model and examples for building more convenient hosts for particular languages or plugin environments. By using a workflow approach to defining experiment tasks, those tasks are naturally broken down into discrete units or components which may be independently scheduled and distributed to satisfy collection access restrictions, software/architecture dependencies or IO file format requirements. For example a user of the NEMA service would be able to conduct an experiment in which:

1) a collection of tracks held at a remote site, which cannot be distributed, is analysed by Matlab code outputting into Weka's ARFF file format [9],
2) the resulting audio analysis based features are raised into the data model and transmitted to another NEMA site with greater compute infrastructure,
3) one or more machine learning frameworks or procedures are launched using automatically constructed shell commands, on specified Virtual Machine infrastructures, to predict a metadata field or transcription for each track, which is then serialized in a MIREX file format,
4) the MIREX format file is read and interpreted in the context of the experiment into the NEMA data model,
5) the results of the experiment are evaluated using Java-based MIR evaluation components provided by the NEMA project,
6) and finally both the prediction and evaluation data is exposed by the NEMA service in the standardised data model and eventually will be integrated into a linked data repository.

3) Experiment and Workflow Sharing: The workflow sharing approach designed for NEMA has been modelled after the myExperiment Virtual Research Environment [10] and directly addresses many of the same issues. It is expected that the NEMA service will fully integrate with myExperiment at some stage in the future. This approach facilitates the easy sharing and reuse of experiment setups and partially or fully-specified implementations of those experiments. For example, a MIREX task may be defined in terms of a component to provide an input dataset and task description (e.g. classify music

[1] e.g. the MIREX 2009 wiki is at http://www.music-ir.org/mirex/2009

by genre), and number of required but unspecified process components (e.g. perform feature extraction, train classifier, apply classifier to test dataset) and an evaluation component. A participating lab would then configure components to perform each of the specified processes and execute the configured workflow to perform the task and receive evaluation results. Similarly a researcher might publish an experiment definition (a fully defined task) and fully configured workflows or simply computed results representing experiments in published paper or thesis, allowing other researchers to accurately compare their work or build on the results.

Further, if users choose to publish their components for re-use by other researchers, NEMA can drive the cross-pollination of research mitigating the need to re-implement techniques or learn additional languages/frameworks/applications. Further, this would lower the bar of entry in particular areas of research, allow researchers to focus on their particular area rather than supporting technologies, provide recommendations and know-how on ways to address particular problems and will facilitate higher-level Musicological research in more reasonable time-scales, skill-sets and levels of effort than is presently possible. For example, a researcher might be able to leverage existing chord transcription systems in implementations of techniques for tasks such as score following, music description or search. Similarly, existing audio event-onset detection procedures might be combined with dominant F0 (melody) estimators and machine learning/IR frameworks to produce an index of melodies. Hence, the NEMA service may eventually form a ideal teaching tool for use in many fields including Musicology, Machine-learning, Library Information Science and MIR.

Finally, the provision of automated and standardised evaluation tools and result collections should facilitate, as MIREX has done in the past, better analysis, reporting and dissemination of results in the field of MIREX than would otherwise be possible. The evaluation tools and IO/Interoperability framework developed to implement the NEMA service is also made available as an open-source toolkit for MIR experimentation and software building and, it is hoped, will be a significant resource to the community in of itself[2]. Out of the box, the toolkit will enable the interoperability of AudioDB [11], jMIR [7], sonic annotator and the VAMP API [8], ACE [12] and the Weka toolkit [9].

4) Resource Resolution, Security and Attribution: Given that security is a prime concern for any NEMA-hosted collections, it is vital that the analysis code is sent to the data and that result data is only returned in a controlled model in which the transmission of the original audio content (or a reconstructable form thereof) may be prevented. Format dependencies may also mean that multiple different encodings of the content may need to be maintained or conversion facilities provided. The model for the extraction of MIR information on remote collections used in NEMA, which protects the NEMA-hosted music collections, is based on the On-demand Metadata Extraction Network (OMEN) [13] developed at the Schulich School of Music, McGill University.

The ability to represent multiple sources for the content also facilitates better access to the resource as, for example, NEMA might not be able to provide streaming access to the controlled content for auditioning but could refer to a publicly or commercially accessible version of the content.

The security of code submitted to the NEMA service is also prime concern, as these components may or may not be under IP restriction or may not be ready for wide-spread distribution. Hence, the NEMA service allows users contributing component code to control the level of access of other users to their components (implement a 'Some rights reserved' sharing model that may help promote academic work and experimentation that would otherwise be difficult or impossible), limited only by the fact that NEMA will never facilitate the download of component code to non-NEMA managed servers. Additionally, full author attribution information and citation information is captured with each component profile in order to facilitate correct attribution in cases of reuse and the viewing of the components themselves as collections of documents and metadata.

B. Using and re-using the NEMA service

The existing approach to the running of MIREX evaluations involves the hand execution of extremely diverse pieces of code by around a half dozen research students over short period of time and may therefore be considered expensive in terms of human resources. One of the key advantages of the NEMA 'self-service' infrastructure is that it exposes, in a controlled manner, the configuration of component execution to the user. This enables the user to solve their own issues, in such a way that, they can rapidly make corrections/alterations to their own codebase or configuration to produce a successful run, avoiding the often drawn out testing, email communication, bug report and update cycles that incurred each year at MIREX.

Further, this issue is often exacerbated by the difficulty of providing test datasets for tasks, as even these are often impossible to share due to copyright issues or the impact it would have on already limited dataset sizes. However, NEMA is of course able to provide test datasets within the NEMA service to facilitate rapid testing of component code before long runs are executed.

Finally, the NEMA service embraces a Virtual Machine Infrastructure for the execution of component code, where necessary. Throughout the hosting of five years of MIREX evaluations the IMIRSEL team has encountered a wide variety of dependency requirements and has developed/maintains a linux based compute cluster node image satisfying as many of these dependencies as possible within a single image. However, there are always special cases, particularly including

[2]source code and documentation for the NEMA project infrastructure is available from google code: http://code.google.com/p/nemadiy

platform dependencies on Windows operating systems and Visual Studio libraries. Hence, a range of VM images will be maintained for use by third party components. Eventually, it is hoped, a user customisable VM container system will be exposed (allowing a user to login to an existing image, modify it for their use in a limited fashion and save it for later reuse). Obviously, such modified VMs or indeed any VM running potentially unknown code must be suitably sand-boxed and will be permitted: read access only to its component code and data required for a particular experiment or task; write access only to specified scratch and output locations and will be required to return its output in certain specified formats, such that it may be appropriately inspected or interpreted; and no access to the outside world.

C. Exposing Results, Metadata, Predictions and Provenance

The NEMA service infrastructure provides each task execution with a unique identifier and persistent storage of logs, execution information / provenance, predictions and evaluation results. This data may be published by a user within NEMA so that it maybe accessed by the community through either a provided digital library framework (Greenstone [14]) or linked data repository/SPARQL endpoint. The design of the linked data system or *repository* for the NEMA project is beyond the scope of this paper. However, it is hoped that the NEMA service infrastructure will one day be used to provide on-demand processing required to satisfy queries submitted via the repository endpoint (i.e. extracting and/or predicting metadata for content that is not already known).

D. Implementation

NEMA uses Meandre [3] to implement it's worklow processes. Meandre is a semantic-web-driven data-intensive flow execution environment. Meandre provides basic infrastructure for data-intensive computation. It provides a high-level language to describe flows, and multicore and distributed execution environment based on a service-oriented paradigm. The Meandre Infrastructure provides a plugin model similar to the servlets in a Java Enterprise Enviornment where custom functionality can be implemented, NEMA uses plugins to make data sources available as a JNDI [15] resource to all the tasks.

The Meandre Infrastructure allows NEMA to Monitor the lifecycle of a task. The task is submitted to the NEMA Flow Service, that schedules the experiment to run on one of the Meandre Servers. The Meandre Servers assembles the experiment by downloading the work flow definitions and code from possibly distributed locations and starts the execution.

The flow execution can depend upon a Virtual machine with a preset execution environment, such as a particular version of python, JVM or other libraries The NEMA Executor Service runs on each virtual machine and handles the process execution and the lifecycle of the process. It reports back to the flow asynchronously the success or failure of the process. The artifacts produced by the process are serialized and moved to the flow server as input to the next the component. We are currently working on an implementation that would store the results in a central NEMA Repository service based on Flexible and Extensible Digital Object and Repository Architecture (Fedora) [16] with a client interface that would allow tasks to retrieve the artifacts.

[3]http://seasr.org/meandre/

ACKNOWLEDGMENT

The NEMA project is funded by the Andrew W. Mellon foundation. The authors would also like to acknowledge the contributions of the NEMA project team at the IMIRSEL laboratory at UIUC and the project partners from the Universities of McGill, Southampton, Waikato, Goldsmiths University of London and Queen Mary's University of London.

REFERENCES

[1] J. Buckman, "Magnatune: Mp3 music and music licensing," http://magnatune.com, April 2006.

[2] Y. Kim, D. Williamson, and S. Pilli, "Towards quantifying the album effect in artist identification," in *Proceedings of ISMIR 2006 Seventh International Conference on Music Information Retrieval*, September 2006.

[3] E. Pampalk, "Computational models of music similarity and their application in music information retrieval," Ph.D. dissertation, Johannes Kepler University, Linz, March 2006.

[4] J. Downie, "The Music Information Retrieval Evaluation eXchange (MIREX)," *D-Lib Magazine*, vol. 12, no. 12, pp. 1082–9873, 2006.

[5] J. Tague-Sutcliffe and J. Blustein, "A Statistical Analysis of the TREC-3 Data," *Overview of the Third Text Retrieval Conference (Trec-3)*, 1995.

[6] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson, "The music ontology," in *Proceedings of the International Conference on Music Information Retrieval*, 2007, pp. 417–422.

[7] C. McKay and I. Fujinaga, "jMIR: Tools for automatic music classification," in *Proceedings of the International Computer Music Conference*, 2009.

[8] C. Cannam, C. Landone, M. Sandler, and J. Bello, "The sonic visualiser: A visualisation platform for semantic descriptors from musical signals," in *Proceedings of the International Conference on Music Information Retrieval*. Citeseer, 2006, pp. 324–7.

[9] I. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunningham, "Weka: Practical Machine Learning Tools and Techniques with Java Implementations," *ICONIP/ANZIIS/ANNES*, pp. 192–196, 1999.

[10] D. De Roure, C. Goble, and R. Stevens, "Designing the myexperiment virtual research environment for the social sharing of workflows," in *Proceedings of the Third IEEE International Conference on e-Science and Grid Computing*. IEEE Computer Society Washington, DC, USA, 2007, pp. 603–610.

[11] M. Casey, "AudioDB: Scalable approximate nearest-neighbor search with automatic radius-bounded indexing." *The Journal of the Acoustical Society of America*, vol. 124, p. 2571, 2008.

[12] C. McKay, R. Fiebrink, D. McEnnis, B. Li, and I. Fujinaga, "ACE: A framework for optimizing music classification," in *Proceedings of the International Conference on Music Information Retrieval*. Citeseer, 2005, pp. 42–9.

[13] D. McEnnis, C. McKay, and I. Fujinaga, "Overview of OMEN," in *Proceedings of the International Conference on Music Information Retrieval*. Citeseer, 2006, pp. 7–12.

[14] I. Witten, S. Boddie, D. Bainbridge, and R. McNab, "Greenstone: a comprehensive open-source digital library software system," in *Proceedings of the fifth ACM conference on Digital libraries*. ACM New York, NY, USA, 2000, pp. 113–121.

[15] R. Lee and S. Seligman, *The JNDI API tutorial and reference: building directory-enabled Java applications*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2000.

[16] S. PAYETTE and C. LAGOZE, "Flexible and extensible digital object and repository architecture(FEDORA)," *Lecture notes in computer science*, pp. 41–59, 1998.