



Networked Environment for Music Analysis (NEMA)

Andrew Shirk, IMIRSEL, GSLIS, UIUC



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

GRADUATE SCHOOL OF **LIBRARY AND
INFORMATION SCIENCE**
The iSchool at Illinois



Credits and Acknowledgements

- Funding Provided By
 - Andrew W. Mellon foundation
- Co-authors
 - Kris West, Amit Kumar, Guojun Zhu, J. Stephen Downie, Andreas Ehmann, Mert Bay
- IMIRSEL
 - International **M**usic **I**nformation **R**etrieval **S**ystems **E**valuation Laboratory
 - The goal of the IMIRSEL is to advance the state-of-the-art in Music Information Retrieval research



Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



Agenda

- **Definitions**
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



Definitions

- **MIR**
 - Music Information Retrieval
- **MIR Task**
 - A specific, well-defined MIR problem, for example:
 - Artist Identification
 - Mood Classification
- **Music Collection**
 - A large body of audio tracks from a common source
- **Dataset**
 - Datasets aggregate sets of audio tracks, metadata about the tracks, and machine learning parameters needed to execute a particular task.



Definitions

- **MIR**
 - Music Information Retrieval
- **MIR Task**
 - A specific, well-defined MIR problem, for example:
 - Artist Identification
 - Mood Classification
- **Music Collection**
 - A large body of audio tracks from a common source
- **Dataset**
 - Datasets aggregate sets of audio tracks, metadata about the tracks, and machine learning parameters needed to execute a particular task.



Agenda

- Definitions
- **Background**
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



Collection Access Restrictions

- Copyright restrictions prohibit sharing of large music collections between researchers
- Licenses are prohibitively expensive
- Use of creative commons collections is not a good solution
 - Small scale collections
 - Widespread use within the MIR community leads to overfitting of models



MIREX

- Music collections challenges led to the establishment of MIREX
 - **M**usic **I**nformation **R**etrieval **E**valuation **eX**change
 - An annual competition since 2005
- Researchers submit (to IMIRSEL) MIR algorithms to be *manually* executed against large music collections, and evaluated using standardized measures
- MIREX drives MIR research by encouraging the community to define new tasks, algorithms and evaluation procedures
- MIREX has required enormous effort by the IMIRSEL team in the execution, debugging, and validation of submitted code



Agenda

- Definitions
- Background
- **Project Goals**
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



Project Goals

- Service-oriented infrastructure for the submission and *automated* execution of MIR compute tasks
- Movement and execution of arbitrary, user-submitted code
- Support heterogeneous execution environments
- Support the use of distributed music collections
- Enforce security of music collections
- Promote sharing of techniques and algorithms

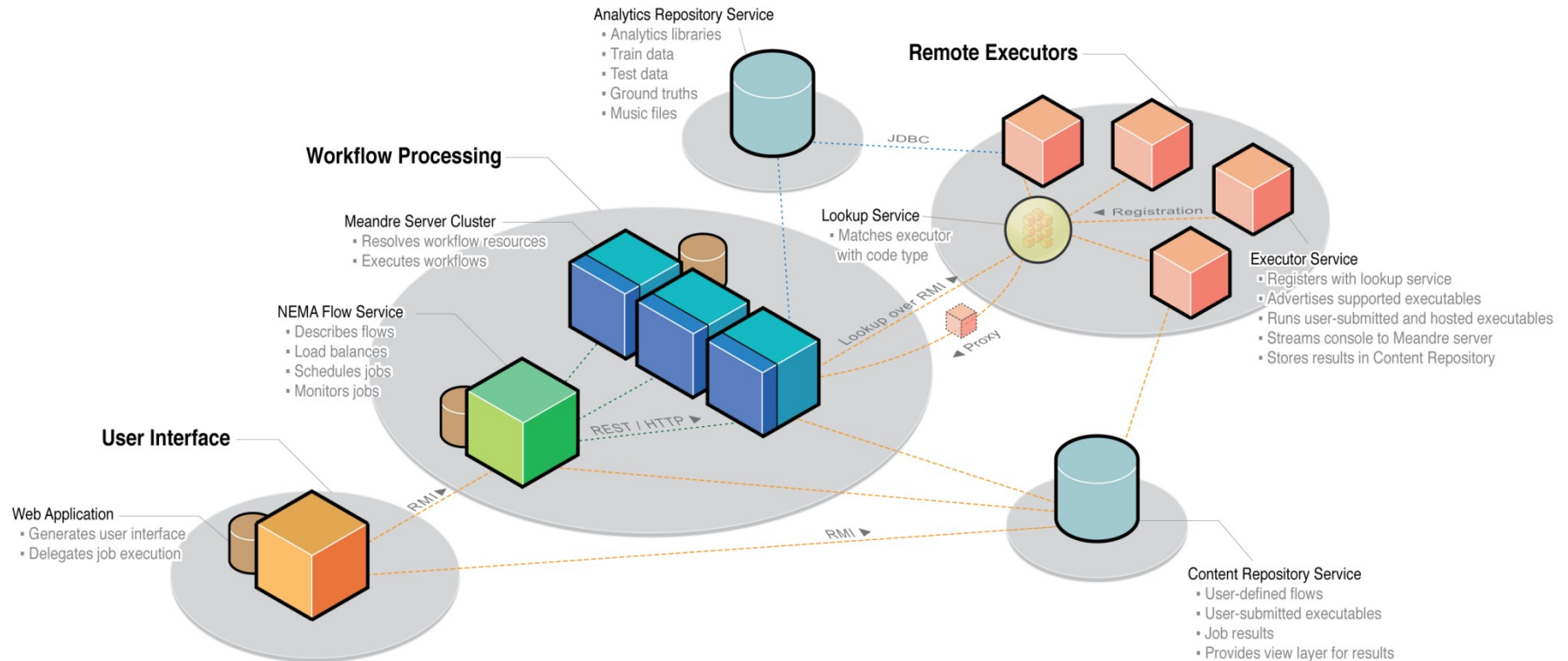


Agenda

- Definitions
- Background
- Project Goals
- **Implementation**
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



NEMA System Overview (Logical)





Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - **MIR Tasks as Workflows**
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



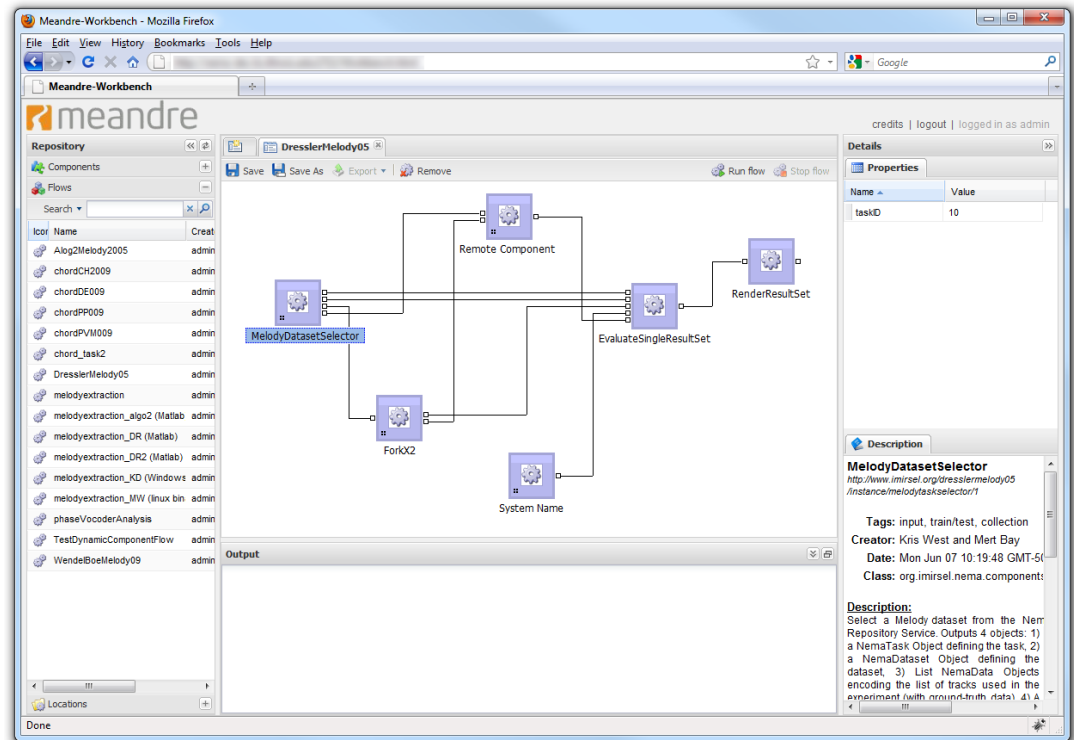
Tasks as Workflows

- MIR tasks can be broken down into discrete units of work
 - Task “components”
 - e.g. dataset selection, dataset analysis, result evaluation
- Component execution can be delegated to remote executors to satisfy collection access restrictions
- Workflow-based computing facilitates repeatable experiments and valid comparisons between runs



Meandre

- NEMA uses Meandre (NCSA) to implement its workflow processes
- Meandre provides basic infrastructure for creating and executing flows
- RDF-based models for executable components, flows, and repositories
- Meandre Workbench for visually wiring together components to create flows
- Meandre Server for executing flows



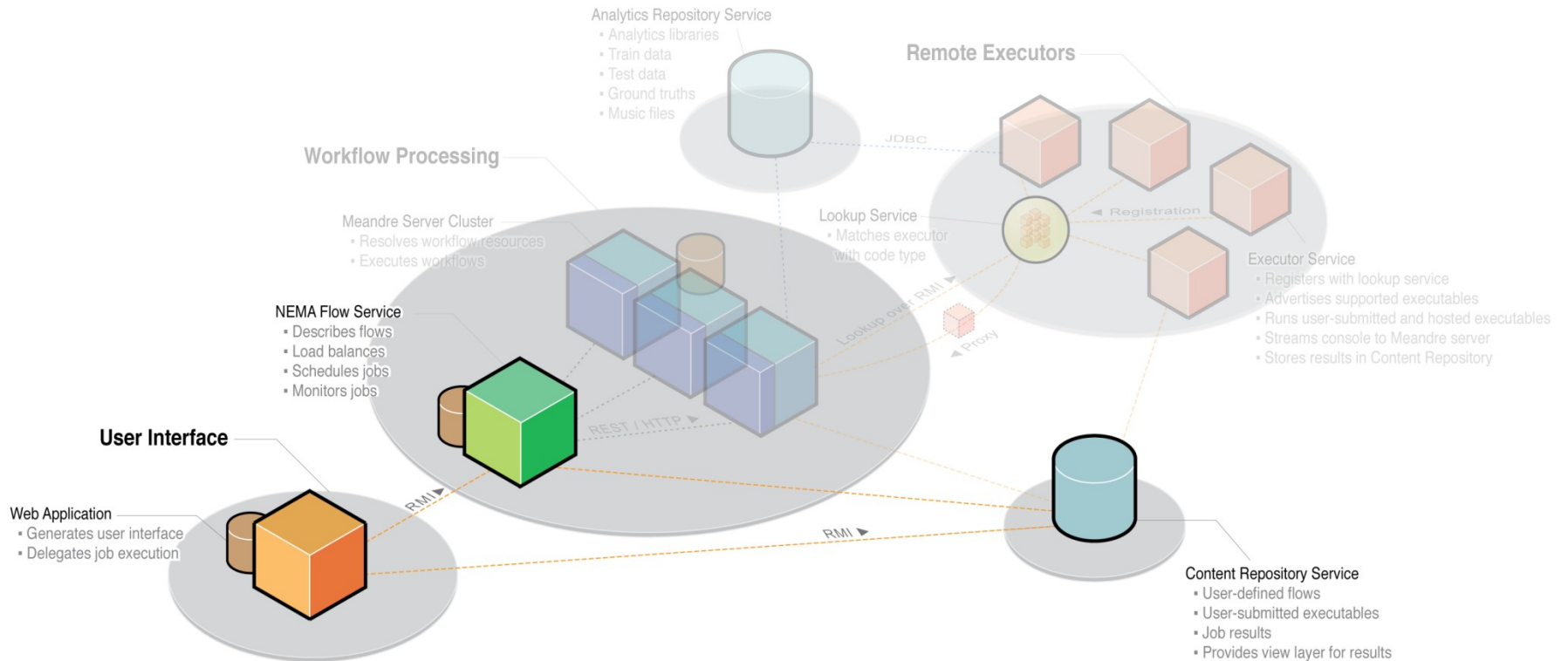


Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - **User Interface**
 - Workflow Processing
 - Remote Executors
 - Job Results
- Future Directions



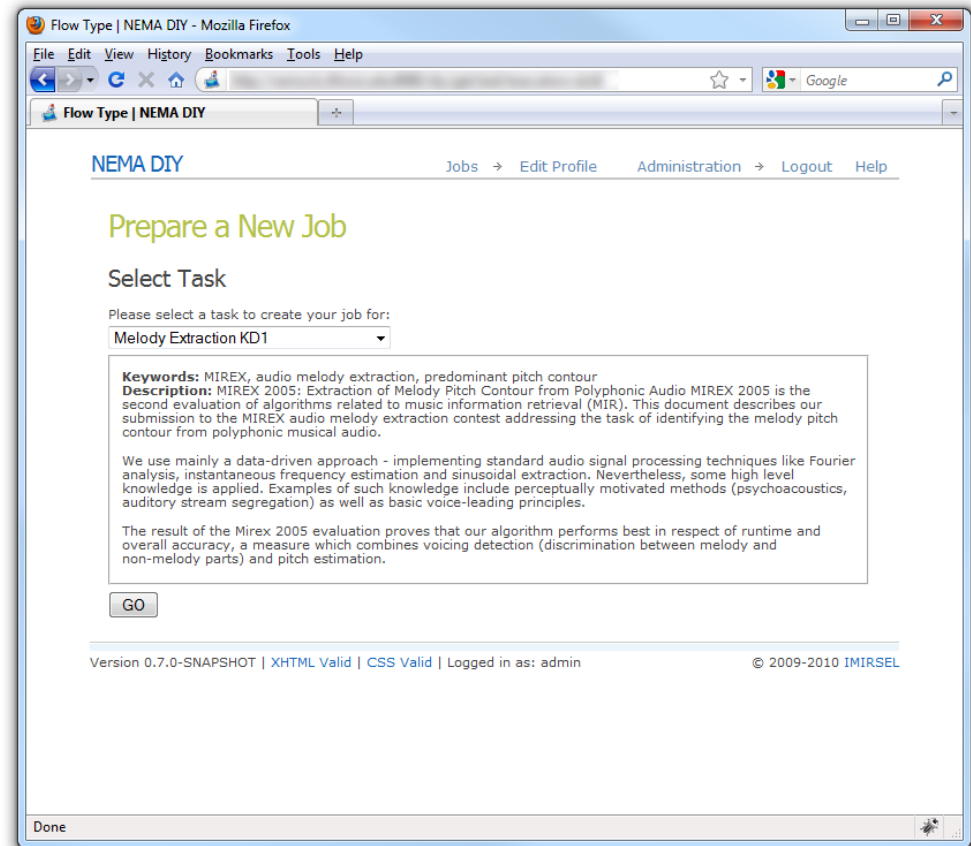
User Interface





Selecting a Task

- Users select from the list of available MIR tasks in order to run a new job
- “Task templates”
- Each task template corresponds to a flow in our central Meandre repository





Configuring a Task

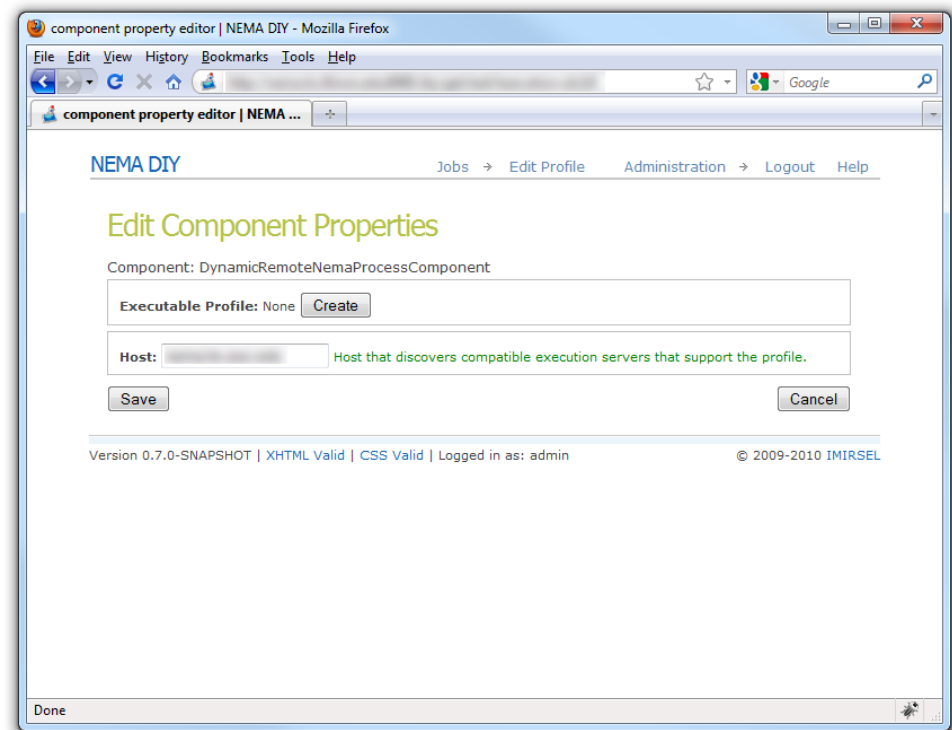
- NEMA Flow Service is queried for the components that make up the flow
- User edits component properties to configure the flow instance

The screenshot shows a web browser window titled 'Flow Type | NEMA DIY - Mozilla Firefox'. The browser's address bar shows 'http://localhost:8080/nema-diy/'. The page has a navigation bar with links: 'NEMA DIY', 'Jobs', 'Edit Profile', 'Administration', 'Logout', and 'Help'. The main heading is 'Edit Task Properties: Melody Extraction KD1'. Below this is a message: 'Please enter the Job details, and edit the properties of the task components accordingly.' There are three input fields: 'Enter the Job Name: Melody Extraction KD1', 'Enter the Job Description:', and 'Component Name: SubmissionComponent'. Below the 'Component Name' field is a description: 'Description: Matches this job with a MIREX submission.' and an 'Edit Properties' button. There are two more component sections. The first has 'Component Name: RemoteNemaProcessComponent' and a description: 'Description: Remote execution component that converts input NEMA models into file formats (specified in the command formatting template of the chosen process), using file paths that will be valid on the remote machine, transfers the input files to the remote machine and executes the process. The output of the process is harvested from the remote machine and read back into NEMA models and output.' and an 'Edit Properties' button. The second has 'Component Name: MelodyDatasetSelector' and a description: 'Description: Select a Melody dataset from the Nema Repository Service. Outputs 5 objects: 1) a NemaTask Object defining the task, 2) a NemaDataset Object defining the dataset, 3) List NemaData Objects encoding the list of tracks used in the experiment (with ground-truth data), 4) A Map of test NemaTrackList Objects to a List NemaData Objects encoding the test set data' and an 'Edit Properties' button. At the bottom are two buttons: 'Review Job and Task Settings' and 'Cancel'. The footer shows 'Version 0.7.0-SNAPSHOT | XHTML Valid | CSS Valid | Logged in as: admin' and '© 2009-2010 IMIRSEL'.



Configuring Remote Components

- Shown is a “Dynamic Remote Component”
- The “Executable Profile” property fully describes the executable that needs to be run
- The “Host” property specifies the address of the remote executor lookup service





Configuring Remote Components

- User-submitted executables can be coded in Java, MATLAB, Binary (C, C++, etc.), Perl, Python, Wine, Shell or Ruby
- User uploads an archive containing the executable entry point, supporting libraries and resources

The screenshot shows a web browser window titled "Upload the Executable | NEMA DIY - Mozilla Firefox". The page is titled "NEMA DIY" and has navigation links for "Jobs", "Edit Profile", "Administration", "Logout", and "Help". The main heading is "Create Executable Profile: Step 1 of 3". Below this, the section "Upload the Executable" contains several form fields:

- "Specify the executable type:" with a dropdown menu set to "Java".
- "Executable JAR or ZIP file containing JARs:" with a text input field containing "sources\helloArchive.zip" and a "Browse..." button.
- "Main class including the package:" with a text input field containing "org.imirsel.nema.HelloW".
- "Operating system required to run the executable:" with a dropdown menu set to "Unix".
- "Select the group:" with a dropdown menu set to "imirsel".

At the bottom of the form are "Next" and "Cancel" buttons. The footer of the page includes "Version 0.7.0-SNAPSHOT | XHTML Valid | CSS Valid | Logged in as: admin" and "© 2009-2010 IMIRSEL".



Configuring Remote Components

- After uploading the executable, the user can supply further configuration information
 - System properties
 - Arguments
 - Environment variables
 - Input & output file formats

The screenshot shows a web browser window titled "fill the arguments | NEMA DIY - Mozilla Firefox". The page is titled "NEMA DIY" and has navigation links: "Jobs", "Edit Profile", "Administration", "Logout", and "Help". The main heading is "Create Executable Profile: Step 2 of 3". Below this, the text "Specify the Arguments for the Java Process" is followed by a form with the following fields:

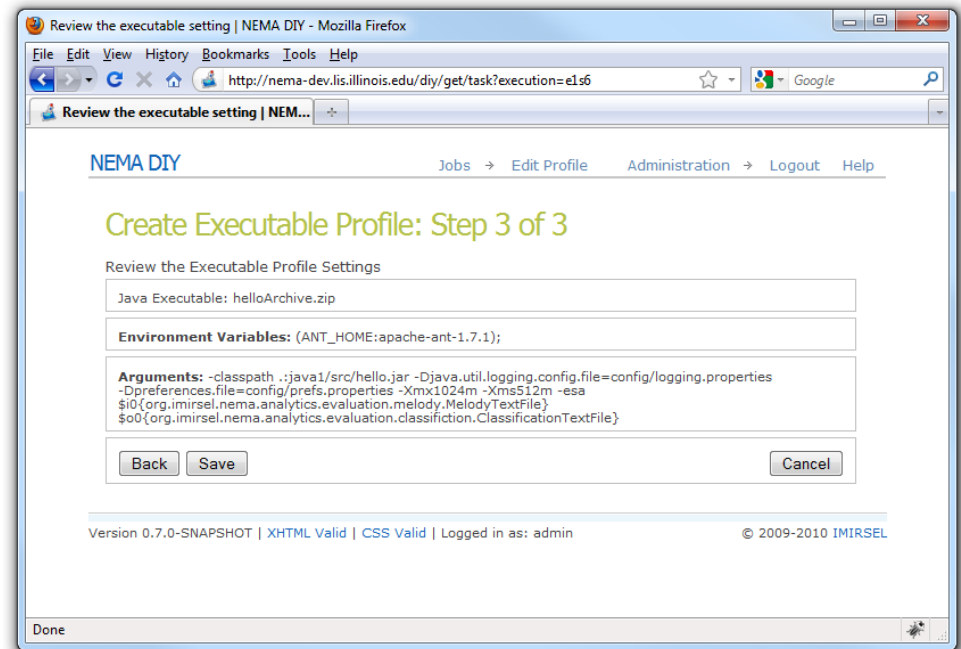
- Uploaded Executable Archive:** helloArchive.zip
- Verbose Execution GC:** yes ☐ no ☒
- Verbose Execution Class:** yes ☐ no ☒
- Verbose Execution JNI:** yes ☐ no ☒
- System Assertions:** enable ☒ disable ☐
- Enable Assertion Packages:** org.imirsel...
- Disable Assertion Packages:** org.imirsel.nema.test...
- Memory:** max:1024m,min:512m
- System Properties:**
 - java.util.logging.config.file = config/logging.properties
 - preferences file = config/prefs.properties
- Environment Variables:**
 - ANT_HOME = apache-ant-1.7.1
- Input Files:**
 - Melody Text File
- Output Files:**
 - Classification Text File
- Other Argument Flags:**

At the bottom of the form are "Back", "Next", and "Cancel" buttons. The footer of the page includes "Version 0.7.0-SNAPSHOT | XHTML Valid | CSS Valid | Logged in as: admin" and "© 2009-2010 IMIRSEL".



Configuring Remote Components

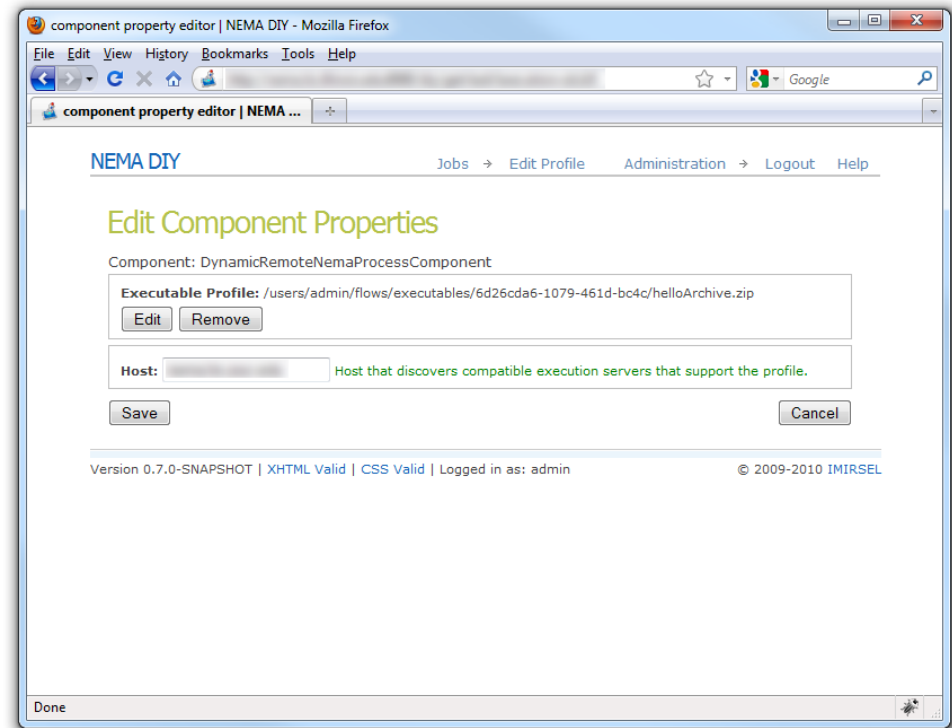
- A preview of the complete executable profile is displayed prior to submission
- When a save is triggered, the completed profile is persisted to the remote Content Repository Service





Configuring Remote Components

- The “Executable Profile” property is now populated with the location of the profile in the Content Repository





Submitting Jobs

- After the user submits a job for execution, the web application creates a new “flow instance” preserving the user-configured flow state
- The web application then issues a job request to the Flow Service with a reference to the new flow instance

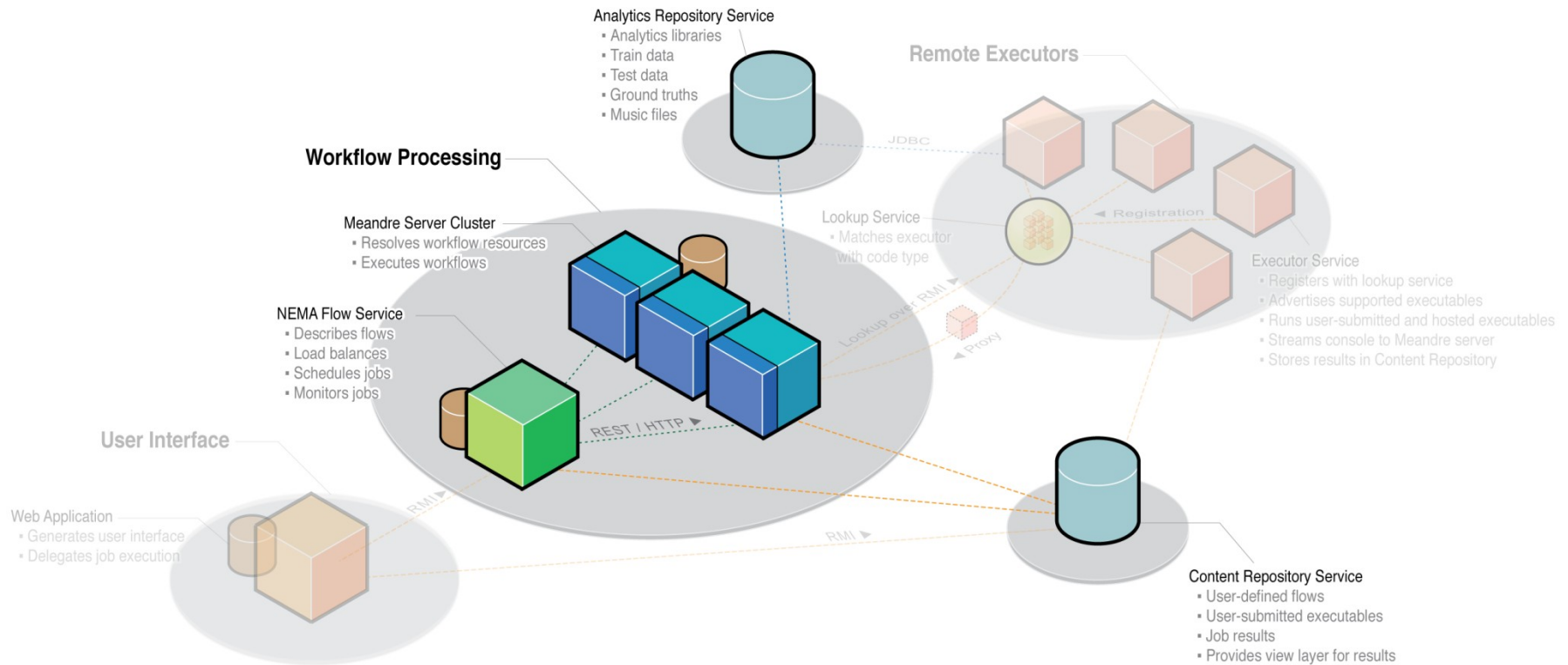


Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - **Workflow Processing**
 - Remote Executors
 - Job Results
- Future Directions



Workflow Processing





Flow Service

- Abstraction layer between UI and Meandre Servers
 - Java application w/RMI service interface
- Implements functionality for:
 - Load balancing
 - Job execution
 - Job monitoring
 - Job status notifications
 - Server monitoring
- Also provides operations for:
 - Describing installed NEMA flow templates
 - Describing components and properties



Meandre Server

- Provides a RESTful web service API
 - Operations for adding, querying & removing:
 - Repositories, components & flows
 - Operations for running and monitoring flows
- NEMA Meandre Server cluster
 - 1 head node
 - Flow Service uses for querying flows & components
 - N worker nodes
 - Flow Service uses for job execution



Meandre Server

- Meandre repositories, components and flows are stored in RDF
 - Independent repositories can be dynamically merged
 - Flows may refer to components that reside at different URLs on the web
- Execution-time component resolution
 - Given a flow model to execute, Meandre will resolve the components at runtime



Datasets

- The dataset, along with the larger NEMA data model, supports all MIR scenarios experienced with MIREX
- Other key objects used in our flows are contained by, or derived from, datasets
 - Audio tracks
 - Track metadata
 - Training and testing sets
 - Ground truths
 - Machine learning parameters
- Datasets contain paths to the physical audio tracks

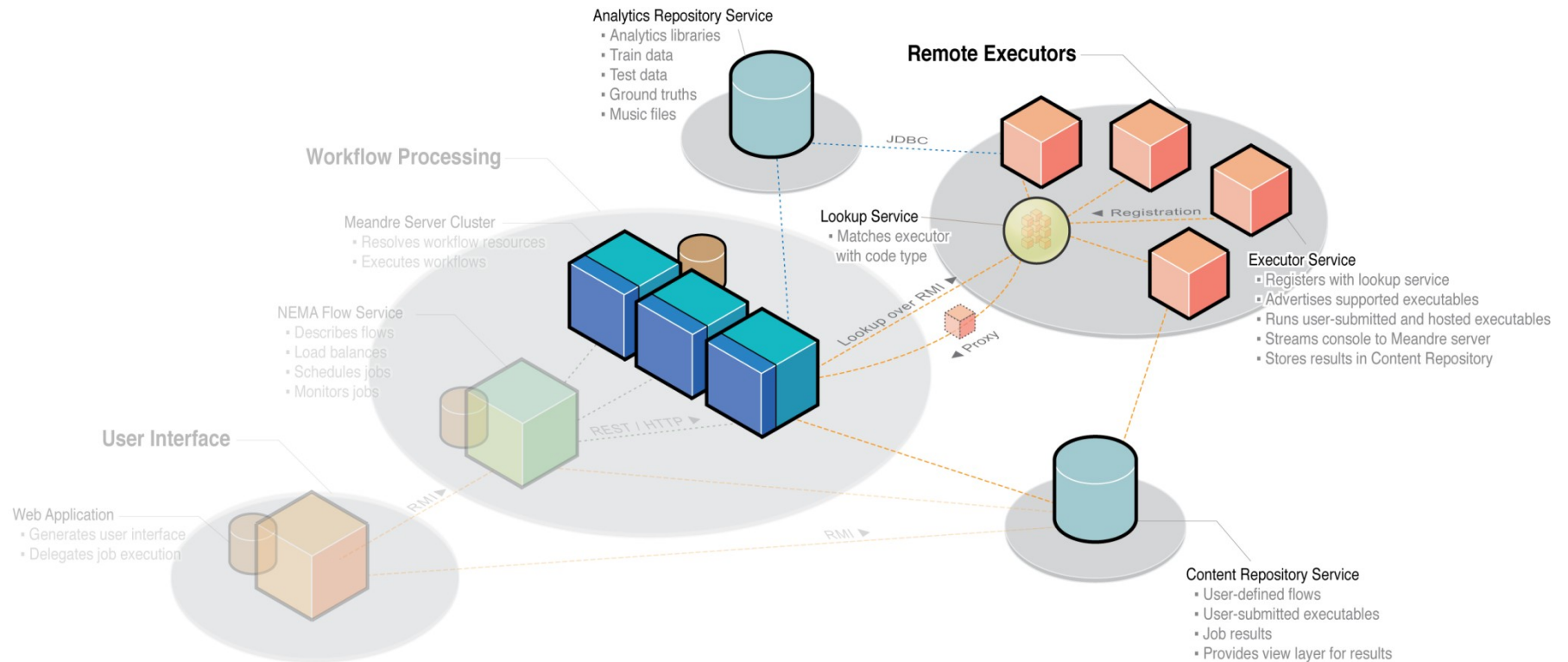


Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - **Remote Executors**
 - Job Results
- Future Directions



Remote Executors





Remote Executors

- Client side
 - Implemented as a Meandre component
 - Gets the executable profile stored in the Content Repository
 - Queries the lookup service for Remote Executors capable of running the executable
 - Delegates processing to the Remote Executor
 - Remotely monitors process lifecycle, output/error streams, and process results



Remote Executors

- Client side
 - Implemented as a Meandre component
 - Gets the executable profile stored in the Content Repository
 - Queries the lookup service for Remote Executors capable of running the executable
 - Delegates processing to the Remote Executor
 - Remotely monitors process lifecycle, output/error streams, and process results

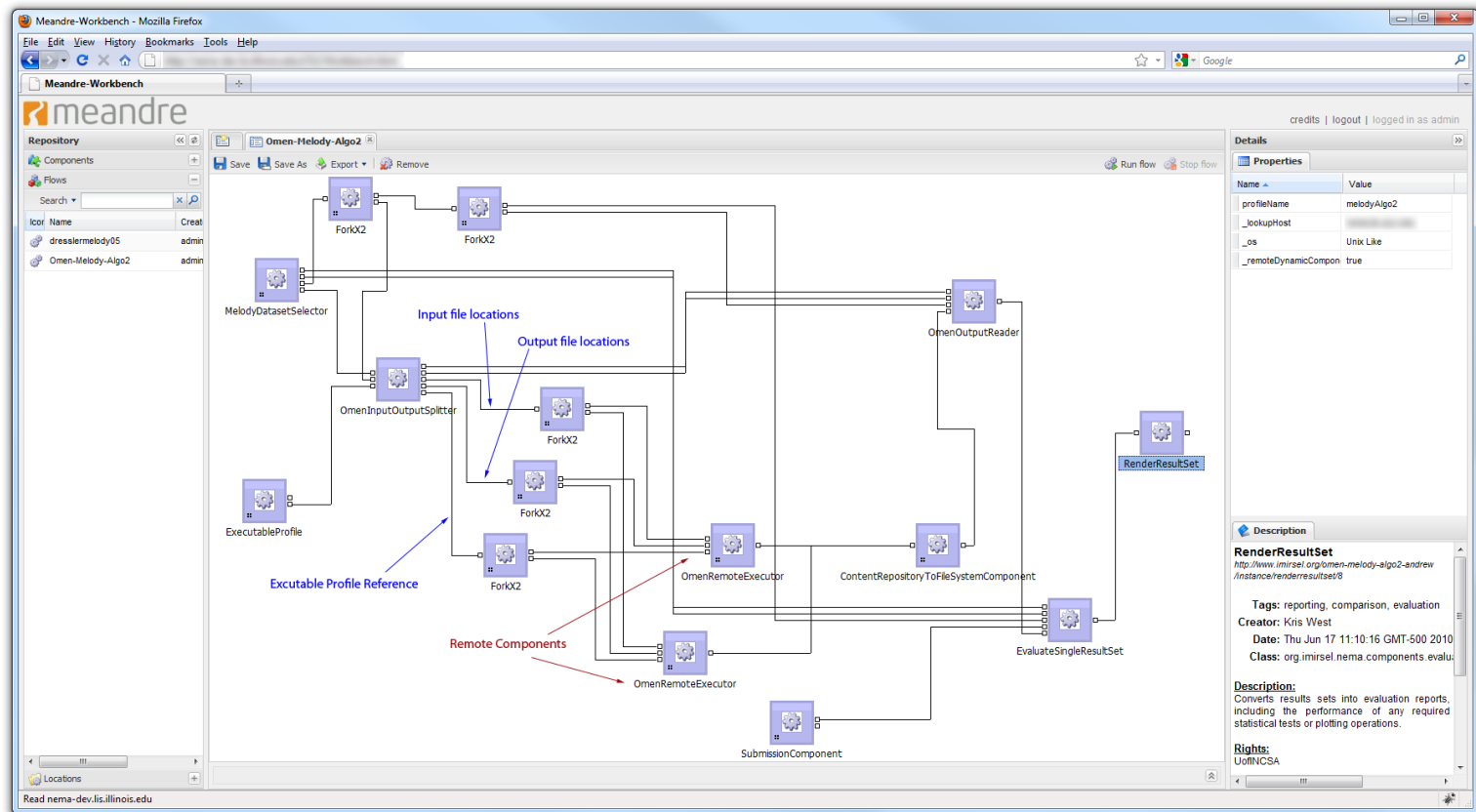


Remote Executors

- Service Side (placed at the data)
 - JINI-based service
 - Registers itself with the lookup service
 - Advertises supported executable types, host OS, etc.
 - For dynamic executables, downloads executable archive from the Content Repository Service
 - Queues process requests if server is already processing maximum number
 - Runs processes using Java ProcessBuilder class
 - Stores results in the Content Repository Service



Multiple Remote Executors





Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - **Job Results**
- Future Directions



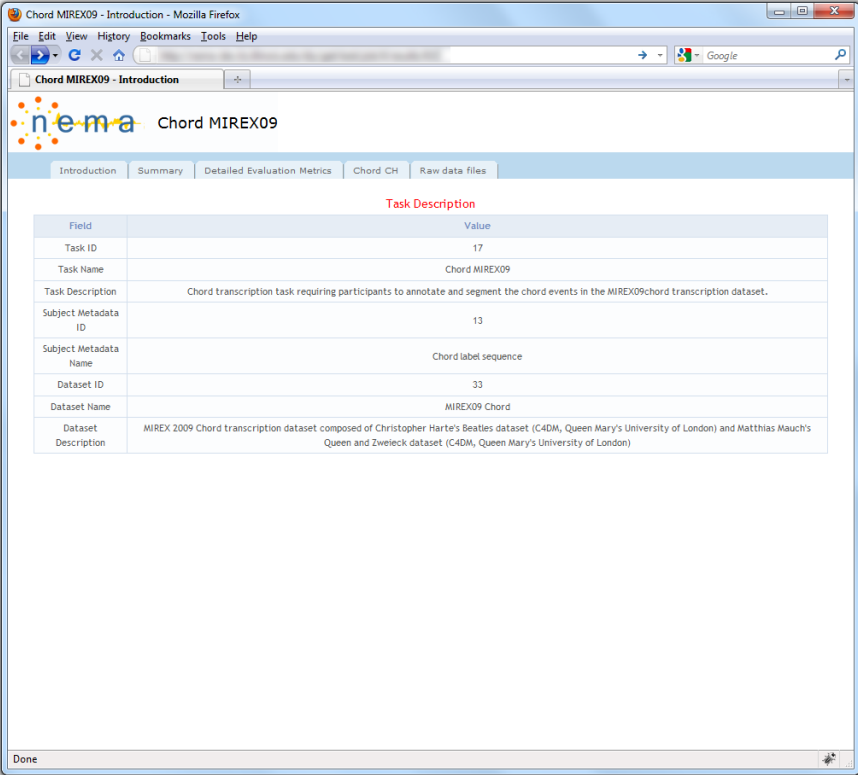
Job Results

- Results are generated by user code in a format specified in the executable profile
- Remote Executors save results to the Content Repository
- Downstream components, running on the Meandre Server, copy the results locally for use by an evaluation component
- After the evaluation component runs, a report is generated that summarizes the evaluation



Evaluation Reports

- Evaluation reports are made available in the UI along with other job details
- Shown is a chord transcription report



Chord MIREX09 - Introduction - Mozilla Firefox

Chord MIREX09

Introduction Summary Detailed Evaluation Metrics Chord CH Raw data files

Task Description

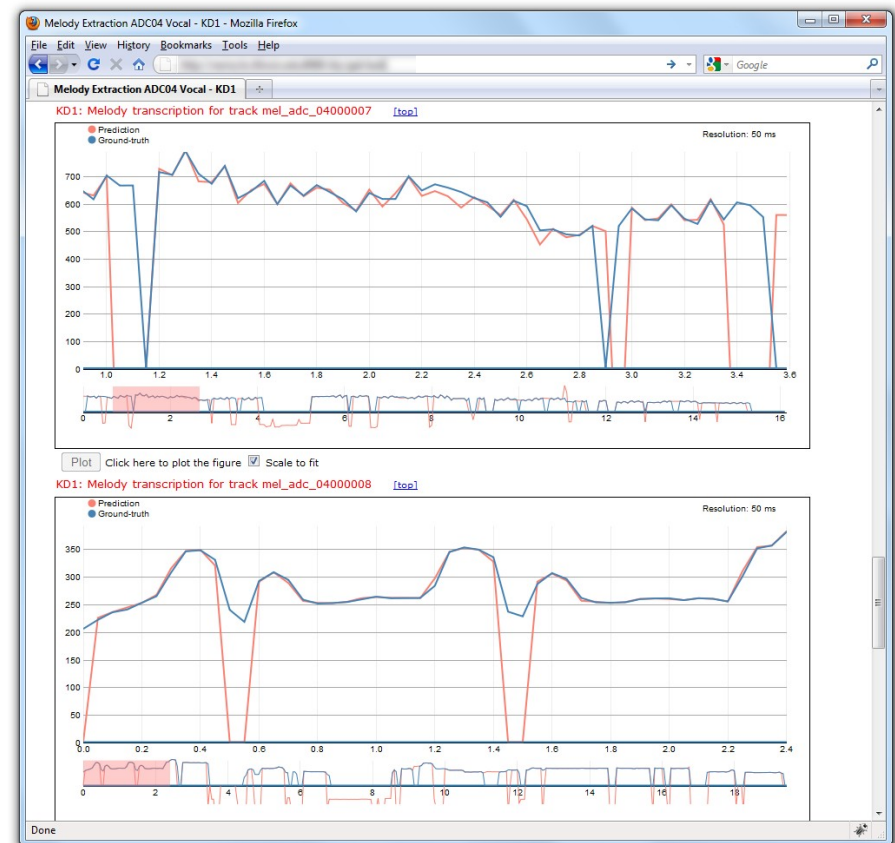
Field	Value
Task ID	17
Task Name	Chord MIREX09
Task Description	Chord transcription task requiring participants to annotate and segment the chord events in the MIREX09chord transcription dataset.
Subject Metadata ID	13
Subject Metadata Name	Chord label sequence
Dataset ID	33
Dataset Name	MIREX09 Chord
Dataset Description	MIREX 2009 Chord transcription dataset composed of Christopher Harter's Beatles dataset (C4DM, Queen Mary's University of London) and Matthias Mauch's Queen and Zweleck dataset (C4DM, Queen Mary's University of London)

Done



Evaluation Reports

- Melody extraction result plots
- Each plot represents the system's performance for a single song in the dataset
- Blue lines are ground truth
- Pink lines are predictions
- X axis is time in seconds
- Y axis is frequency in Hertz





Agenda

- Definitions
- Background
- Project Goals
- Implementation
 - MIR Tasks as Workflows
 - User Interface
 - Workflow Processing
 - Remote Executors
 - Job Results
- **Future Directions**



Future Directions

- Additional partner collection sites
- Integration with myExperiment
- UI improvements
- User-configurable VMs
- Year-round access



Future Directions

- Additional partner collection sites
- Integration with myExperiment
- UI improvements
- User-configurable VMs
- Year-round access

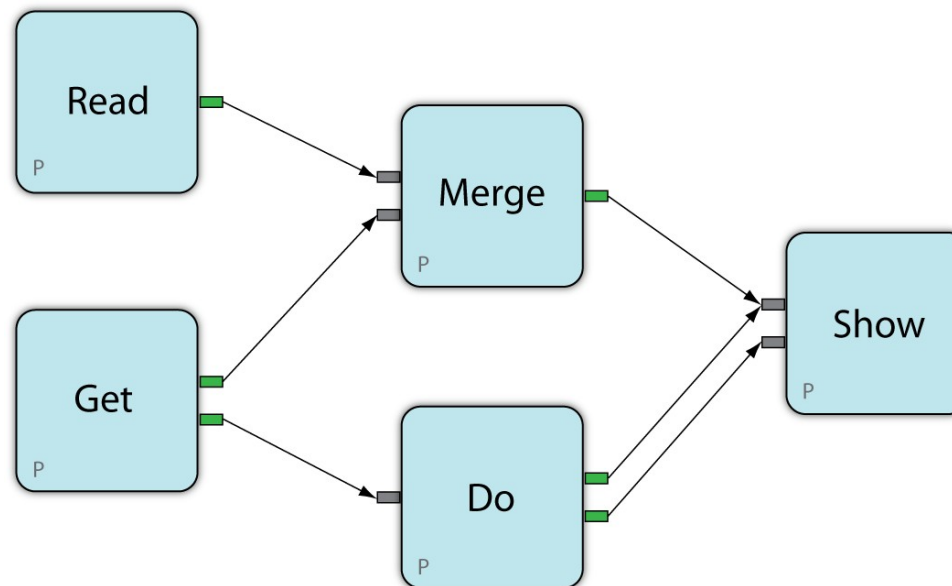


Thank you

- For more information:
 - Andrew Shirk
 - shirk@illinois.edu

Tasks as Workflows

- A task is a collection of connected components with inputs, outputs and configurable properties





Meandre Repository RDF

```
<http://www.imirsel.org/omen-melody-algo2/>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://www.meandre.org/ontology/flow_component> ;
  <http://purl.org/dc/elements/1.1/creator>
    "admin"^^<http://www.w3.org/2001/XMLSchema#string> ;
  <http://purl.org/dc/elements/1.1/date>
    "2010-06-27T11:39:47"^^<http://www.w3.org/2001/XMLSchema#dateTime> ;
  <http://purl.org/dc/elements/1.1/description>
    "\n\n&#pMEN Melody Extraction<br>"^^<http://www.w3.org/2001/XMLSchema#string> ;
  <http://purl.org/dc/elements/1.1/rights>
    ""^^<http://www.w3.org/2001/XMLSchema#string> ;
  <http://www.meandre.org/ontology/components_instances>
    <http://www.imirsel.org/omen-melody-algo2/components/set> ;
  <http://www.meandre.org/ontology/connectors>
    <http://www.imirsel.org/omen-melody-algo2/connector/set> ;
  <http://www.meandre.org/ontology/name>
    "Omen-Melody-Algo2"^^<http://www.w3.org/2001/XMLSchema#string> ;
  <http://www.meandre.org/ontology/tag>
    "omen"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.imirsel.org/omen-melody-algo2/instance/omenoutputreader/2>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://www.meandre.org/ontology/instance_configuration> ;
  <http://purl.org/dc/elements/1.1/description>
    "Receives a datastructure that defines a list of expected output file locations and then receives a stream of ProcessArtifacts that represent entries in that datastructure (files) being fulfilled by a process execution and
  <http://www.meandre.org/ontology/instance_name>
    "OmenOutputReader"^^<http://www.w3.org/2001/XMLSchema#string> ;
  <http://www.meandre.org/ontology/instance_resource>
    <meandre://seasr.org/components/omenoutputreader> ;
  <http://www.meandre.org/ontology/property_set>
    <http://www.imirsel.org/omen-melody-algo2/instance/omenoutputreader/2/property/wb_top_pix_pos> , <http://www.imirsel.org/omen-melody-algo2/instance/omenoutputreader/2/property/wb_left_pix_pos> .

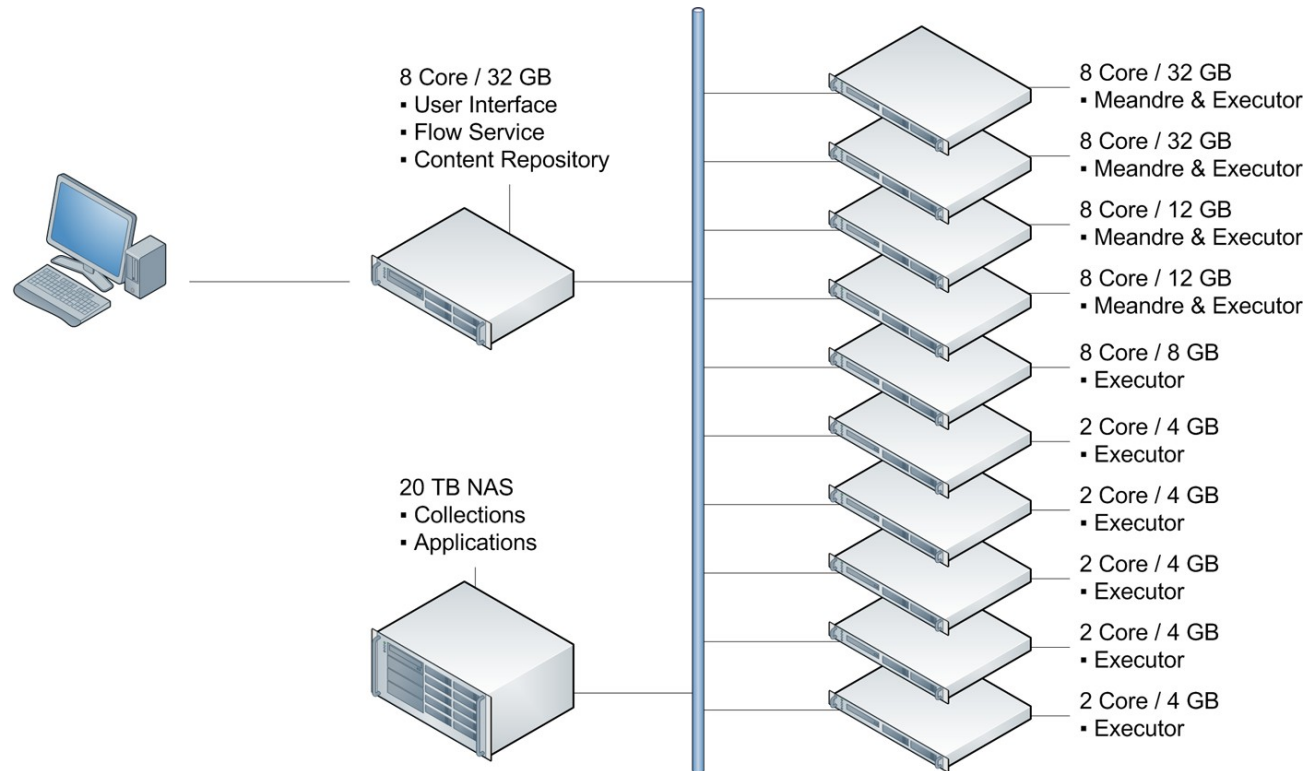
<http://www.imirsel.org/omen-melody-algo2/components/set>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://www.meandre.org/ontology/instance_set> ;
  <http://www.meandre.org/ontology/executable_component_instance>
    <http://www.imirsel.org/omen-melody-algo2/instance/profiletopprocesstemplatecomponent/0> , <http://www.imirsel.org/omen-melody-algo2/instance/contentrepositorytofilesystemcomponent/12> , <http://www.imirsel.org/omen-melody-a

<http://www.imirsel.org/omen-melody-algo2/connector/4>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://www.meandre.org/ontology/data_connector_configuration> ;
  <http://www.meandre.org/ontology/connector_instance_data_port_source>
    <meandre://seasr.org/components/melodytaskselector/output/testsets> ;
  <http://www.meandre.org/ontology/connector_instance_data_port_target>
    <meandre://seasr.org/components/omeninputoutputsplitter/input/datatoprocess> ;
  <http://www.meandre.org/ontology/connector_instance_source>
    <http://www.imirsel.org/omen-melody-algo2/instance/melodytaskselector/1> ;
  <http://www.meandre.org/ontology/connector_instance_target>
    <http://www.imirsel.org/omen-melody-algo2/instance/omeninputoutputsplitter/4> .

<http://www.imirsel.org/omen-melody-algo2/connector/24>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://www.meandre.org/ontology/data_connector_configuration> ;
  <http://www.meandre.org/ontology/connector_instance_data_port_source>
    <meandre://seasr.org/components/forKx2/output/output_object_1> ;
  <http://www.meandre.org/ontology/connector_instance_data_port_target>
    <meandre://seasr.org/components/evaluatesinglereultset/input/nematask> ;
  <http://www.meandre.org/ontology/connector_instance_source>
```



NEMA System Overview (Physical)





- Chord transcription result plots
- Each plot represents the system's performance for a single song in the dataset
- Blue is ground truth
- Pink is the prediction
- X axis is time
- Vertical lines delineate chord instances



