How to find your data - six months later

The MPO System for Automatic Workflow Documentation

John Wright
for The MPO Team

International Conference on Numerical Simulation of Plasmas
12-14 August 2015
Acknowledgments

- **Metadata, Provenance and Ontology project**
  - Year 3 of 3 year project.
  - Software in Beta release

- **MPO team members**
  - Gheni Abla, Liz Coviello, Sean Flanagan, Xia Lee, David Schissel – **GA/DIII-D**
  - Alex Romosan, Arie Shoshani, John Wu – **LBNL**
  - Martin Greenwald, Josh Stillerman, John Wright – **MIT/C-Mod**

- **Our colleagues**
  - Members of SWIM, AToM, CASCADE teams
  - Staff at GA, LBNL, MIT

- **This work is supported by US Department of Energy**
  - Office of Advanced Scientific Computing Research
  - Office of Fusion Energy Sciences
Where is my data?

- `ls -l`
  - Run1/
  - Run2_a=3/
  - Run2_a=3_b=4/
  - Run3_good-one/

- How did my (long-gone) student/post-doc generate this result?

- What *am* I going to put in my data management plan?
Documenting Data and Processes is important

- Data from research activities is expensive to produce and may be critical for follow-on research.
- It is not the mere existence of data that is important, but our ability to make use of it.
- The context and metadata makes the data more usable:
  - Hypotheses
  - Pre-process activities
  - Experiments
  - Computational process
  - Reflections
- Documenting the process is not an easy task.
Throughout History, Scientists Generated Handwritten Logbooks to Keep Track of Data
Documenting Research Process and Scientific Workflows Met Challenges in the Modern Era

• Personal computers and mobile devices helped electronic logbooks replace handwritten ones – brought conveniences
  – Multi-media and hypertext support
  – Storing and sharing
  – Search
• However, the content creation and log entry remained as a manual activity in the electronic logbooks
• As the pace of scientific research accelerated, documenting the process & data became more challenging & time consuming
  – Increased precision of scientific instruments
  – Rise of exa-scale computing
  – Arrival of Big Data
  – Result: fragmentation of data, processing, and documentation
MPO Objectives: Document Scientific Data & Workflow

- **Provenance:** Preserve meaning of data by documenting all of the steps taken to produce the data
  - Capture both data and process – automate as much as possible
  - Support more systematic management of analysis and simulation data
- **Provide and preserve answers to two key questions:**
  - *Where did a particular piece of data come from?*
    - Assumptions, inputs, parameters used for calculation
    - The origin of inputs; reasons for assumptions & parameters
  - *Where was this data used?*
    - Other calculations
    - Publication or presentation
    - Contributions to databases
Use Cases

• How did I get the data plotted in Fig.6 of my 2014 Phys. Plasmas article?

• A calibration error was found in Thomson Scattering data taken during 2011 - the data has now been recalculated.
  – Where was the old data used?
  – What publications used that data?
  – Did we contribute that data to an international database?

• A recently graduated PhD student left behind output from thousands of gyrokinetic simulations
  – Which of these were used in her thesis?
  – Which might be useful in the future?
  – What were the inputs and parameters used in the interesting runs?
Demo of MPO search

<table>
<thead>
<tr>
<th>CompositeID</th>
<th>Description</th>
<th>Creation Time</th>
<th>Comments</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 mpmgr / EFIT / 3</td>
<td>example of using python interface to API.</td>
<td>2015-07-29 11:12:19</td>
<td>1</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>2 <a href="mailto:jowright-ocf@MIT.EDU">jowright-ocf@MIT.EDU</a> / TORICS / 1</td>
<td>Reference case used in initial coupling</td>
<td>2015-07-28 19:19:10</td>
<td>1</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>3 <a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 15</td>
<td>test idl mpo interface</td>
<td>2015-07-10 12:57:14</td>
<td>7</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>4 <a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 14</td>
<td>test idl mpo interface</td>
<td>2015-07-10 12:48:08</td>
<td>6</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>5 <a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 13</td>
<td>test idl mpo interface</td>
<td>2015-07-10 12:46:19</td>
<td>6</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>6 <a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 12</td>
<td>test idl mpo interface</td>
<td>2015-07-10 12:23:03</td>
<td>6</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>7 <a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 11</td>
<td>test idl mpo interface</td>
<td>2015-07-10 12:17:41</td>
<td>6</td>
<td>🟢🟢🟢</td>
</tr>
<tr>
<td>8 <a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 10</td>
<td>test idl mpo interface</td>
<td>2015-07-10 12:17:41</td>
<td>6</td>
<td>🟢🟢🟢</td>
</tr>
</tbody>
</table>
Capabilities of the MPO system

• Support all types of the scientific workflow – both experimental and computational
  – Typically involves processing of raw data, with small or large codes whose output requires processing as well
• Allow users to record as much or as little info as they need
• Function in a heterogeneous environment and interoperate with workflow tools people are already using
  – Researchers use many different languages (Shell scripts, Python, IDL, Matlab, etc.) and tools to get their work done
  – Many different computing platforms – laptop to supercomputer
  – Data is stored in different formats (MDSplus, HDF5, NetCDF, ASCII)
  – It would be futile to insist that researchers change all of that to get the benefits that we propose
• Once set up, needs to work as automatically as possible
  – Best suited for scripted rather than one-time use
The MPO System is Based on a Multi-Tier Software Architecture
Basic Components of the MPO System

• Database
  – Captures metadata, location of data & all processing steps
• API server
  – Mediates all communication with database
  – Uniform language-independent interface for clients
• Interactive UI Server
  – Provides interactive interface to discover and explore workflows
  – Allows users to enter new comments about any object
• Event Server
  – Enables automatic updates of workflow information
• Clients
  – Instrument MPO calls
MPO System Entities and Data Model

- **Data Object**: Description of a data including a pointer **URI** (Uniform Resource Identifier)
- **Activity**: Any process that creates, moves or transmutes data from one form to another
- **Workflow**: A series of connected Data Objects and Activities, which can be organized as a Directed Acyclic Graph (**DAG**)
- **Connection**: Internal data model to represent DAGs.
- **Metadata**: Text-based, arbitrary name-value pairs
- **Comment**: User annotation as unstructured text
- **Collection**: Simple lists of Data Objects, Collections, Workflows
MPO Objects Are Uniquely Identified

- **Each MPO object is given a global unique numerical identifier**
  - UID (GUID/UUID) – Universally Unique ID (we use UUID v.4)
  - 128 bit, pseudo random numbers \( \sim 10^{37} \) possible values
  - Example: 08d2f5db-97d8-49c9-bd99-85303b07f9e2

- **Data objects are also given a URI (Uniform Resource Identifier)**
  - The URI is the pointer to the data object - implies a **persistent storage**.
  - URI includes the data protocol name and the path to the data.
  - Examples: mdsplus:///magnetics/550335\&path=\magnetics::ip, file://host.edu/path/to/file.h5

- **Workflows also can be identified by composite ID that is easy to remember (as opposed to UIDs)**
  - Examples: doej/EFIT/52, smitha/OMFIT/1002

- **Searching is enhanced by defining a “controlled vocabulary”**
  - User-defined, hierarchical ontology
API uses a RESTful interface

- Clients only need HTTPS POST and GET operations to access the MPO.
- Ease of implementation but puts complexity in API server.
- Oriented around construction of URI resources
- Examples:
  - GET /workflow?user=jwright
  - POST /comment
    ```
    {
    'content': 'This is a comment',
    'parent_uid': '3d55-4...
    }
    ```
  - GET /workflow/:uid/graph
DAGs define Direction of Provenance

- Directed Acyclic Graphs provide a mathematical structure to trace inheritance and ancestry of provenance
- Ancestry: “What simulation results used a particular input?”
- Inheritance: “What results were affected by a bad calibration?”
# MPO Demo of DAG Navigation

## MPO Workflows

<table>
<thead>
<tr>
<th>CompositeID</th>
<th>Description</th>
<th>Creation Time</th>
<th>Comments</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 2</td>
<td>test idl mpo interface</td>
<td>2015-07-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: 10285555-250a-444b-8606-05883a82fe</td>
<td></td>
<td>14:20:21</td>
<td></td>
<td></td>
</tr>
<tr>
<td><a href="mailto:mjgreen@MIT.EDU">mjgreen@MIT.EDU</a> / Gyro / 1</td>
<td>test idl mpo interface</td>
<td>2015-07-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: 58b44a3-e349-f4f0-b64d-47f8e8a5c</td>
<td></td>
<td>14:17:47</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mpodemo / Zipfit / 1</td>
<td>Zipfit Electron Temperature for 160124</td>
<td>2015-05-12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: 9b19a5a7-8e1a-4e2b-99fe-b64ddab</td>
<td></td>
<td>21:58:18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mpodemo / EFIT / 2</td>
<td>EFIT02 for 160124</td>
<td>2015-05-12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: 6b2d9d0c-9e8a-4b2e-999a-669d30b265</td>
<td></td>
<td>21:57:35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mpodemo / EFIT / 1</td>
<td>EFIT01 for 160124</td>
<td>2015-05-12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: 040711e-5c9b-4509-86cc-f442c5d5ab</td>
<td></td>
<td>21:57:21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mpodemo / OMFIT / 31</td>
<td>OMFIT workflow</td>
<td>2015-02-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: 7f6b6e6-022b-4f33-90ac-86b1a17741</td>
<td></td>
<td>19:43:51</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mpodemo / SWIM / 113</td>
<td>29841 : fastran_30001Z2</td>
<td>2014-03-18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID: aea8004-629a-4250-a59f-a6547bc40f</td>
<td></td>
<td>00:00:04</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Workflows can be recorded from HPC Systems

- CASCADE climate ensemble simulation
- This workflow was executed on Edison@Nersc and recorded on an external MPO server using the API over HTTPS.
Future Work

• Data and workflow discovery in an environment where a modeling code or an exp can generate large collections of heterogenous data of up to $10^{10}$ objects.

• Metadata evolution with particular attention paid to schema and ontology evolution.

• Import/export workflows in standard formats (e.g. PROV).

• Provenance and structure inside more complex data objects. E.g. array inside Na HDF5 file or a profile in the ITER IMAS database.

• Fine grained authorization.

• UI methods for very large workflows and real time monitoring.
Summary

• There is no free lunch, but maybe there are cheap lunches.
  – MPO software for documenting scientific workflows and data provides a way to instrument scripts to record provenance.
  – Non-intrusive. Use what features you want.
  – Augments your existing workflows - where ever they are.
• Production workflows have been MPO instrumented
  – Tokamak Experiment Between Pulse Workflow at DIIIID, Fusion Simulations in SWIM and AToM frameworks, CASCADE project from climatology
  – Approach seems valid and general
  – Lessons learned and improvements being made
• MPO team seeks partners and Beta users!
  – Test the capability by instrumenting your workflows
How to get started with MPO.

- mailto: mpo-info@fusion.gat.com

https://mpo.psfc.mit.edu
Select “Try MPO”, read the tutorial and try it out:

Get it:
Server: virtual machine distributed public servers available
Client: simple python class installation
BACKUP SLIDES
Command Line Client For Use in Scripts

- Client uses ‘meta’ commands and method names
- Shell scripts and batch scripts can be instrumented
- User can make queries & comments via command line
- Example script or command line session:
  - wid = mpo init --name=EFIT --desc=test`
  - oid = mpo add $wid --parent=$wid --name=shot --
    desc="Plasma shot number" --uri=150335`
  - oid2 = mpo add $wid --parent=$wid --name="Snap file" --
    desc="EFIT input file" --uri="\efit01:namelist"`
  - aid = mpo step $wid --input=$oid --input=$oid2 --name="EFIT
    exec"
    --desc="Fit equilibrium and compute plasma parameters" --
    uri=EFIT`
  - cid = mpo comment $aid "This program is the only one in this
    workflow"
Script example

- Example PBS file:

```
#!/bin/bash
#PBS -N TORIC5_imp
#PBS -l nodes=10:ppn=8

export MPO=/home/jwright/bin/mpo.py

wid=`$MPO init -d "Testing mode coupling for COMSOL." \
    -t TORIC -n r505`

echo The work_uid of this job is $wid.

PROGRAM=/home/jwright/bin/Ptoric5_svn_path.e
mpiexec $PROGRAM
aid1=`$MPO step $wid $wid --input $oid1 --input $oid2 --name $PROGRAM`

$MPO meta $aid1 host loki.psfc.mit.edu
$MPO meta $aid1 exec $PROGRAM
$MPO meta $aid1 nproc $NPROCS
$MPO meta $aid1 jobid $PBS_JOBID
$MPO meta $aid1 workdir $PBS_O_WORKDIR
$MPO meta $aid1 git-version `cat $TORIC_HOME/.git/refs/heads/master`

oid3=`$MPO add $wid $aid1 --name toric.sol --desc "TORIC ascii solution file"`
```
We support several distribution methods.

- VM image for servers
- System wide install of mpo clients
- User level install of command line client in python virtual environment
- Soon:
  - containers (Docker)
  - `pip install mpo`
MPO and Persistent Data Store

• Underlying model is an assumption that data objects will be maintained
  – If the underlying data are allowed to change in untracked ways, the descriptions and provenance are corrupted
  – Data can be moved to a new location or converted to a new format – as long as this is written down into MPO database
  – MPO does not dictate the implementation of the persistent store
    • Data objects can be a reference to a user’s file system
    • Data objects can be a description of how to retrieve the item from a database or record store or file

• A set of methods is available to manage data in a persistent store in a manner consistent with maintaining the integrity of MPO database
Interactive UI Page Example: Workflow List

- Enhanced “Controlled Vocabulary” Search: User-defined, Hierarchical Ontology
- Comments can be inserted/viewed directly on this listing page
Interactive UI Page Example: Workflow Details

List of nodes and their corresponding details: UID, URI, metadata, comments, other linked workflows

Blue nodes are used in other workflows
## Interactive UI Page Example: Collections List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Username</th>
<th>Creation Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMRT kinetic EPIT</td>
<td>OMRT kinetic EPIT runs for shots 158634-158640</td>
<td>smitha</td>
<td>2015-03-11 14:26:19:223008</td>
</tr>
</tbody>
</table>

Select to view details
Interactive UI Page Example: Collection Details

Sample collection including multiple workflows, multiple data objects and another collection
Current RESTful API Supports Workflow Instrumentation

- Routes for workflow creation and annotation
  - /workflow, /activity, /dataobject, /comment, /metadata
  - Each route supports POST for object creation and GET:uid for object retrieval
  - Objects are encoded in JSON for POSTing and GETting
    - POST /workflow
      BODY: { "name":"GYRO", "description":"Important ITER run" }
    - GET /metadata?work_uid=f20b23ec-aefb-481c-8c08-6443f
      Returns: {
        "target_uid": f20b23ec-aefb-481c-8c08-6443f,
        "key": "Te(kev)",
        "value": 3,
        "uid": "e1b13f63-97ca-490d-9218-15c8f5cae1d5",
        "time": 2013-03-14 19:44:34.235565,
        "uri": http://mpohost/metadata/e1b13f63-97ca-490d-9218-15c8f5cae1d5"
MPO technology stack

- “PostgreSQL” database used for current implementation
- Both API server and Interactive UI server use “Flask.py”, a lightweight python web application framework
- Twitter “Bootstrap” to create standardized Web front-end
- DAGs rendered by “Graphviz” software
- Authentication via x.509 certificates (currently support OSG, MIT & MPO certs)
- MDSplus event services