Storage Lifecycle Management Workflow Scripting Training

NAVY August 2013
ARL September 2013

DOD HPC MODERNIZATION PROGRAM

SLM Workflow Scripting Training

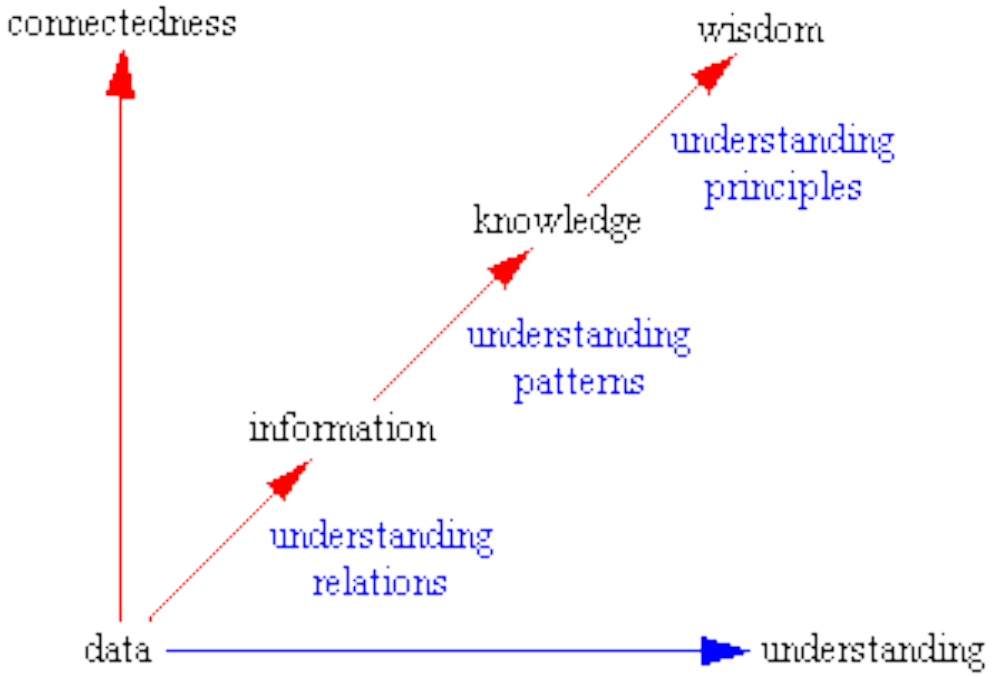
- **Location: NAVY and ARL**
  - Tuesday, August 27 2013 through Thursday, August 29 at NAVY
  - Tuesday, September 10, 2013 through Thursday, September 12, 2013 at ARL
- **Presenters:**
  - Constantin (Tino) Scheder General Atomics
  - Lawrence A. Murakami HEUE SLM CEW
- **Training Materials:**
  - https://help.ccac.hpc.mil/heue
  - http://people.arsc.edu/~murakami/SLM

Agenda

- Use cases and corresponding script solutions
- CEW Sretain, Sdata, Sreview, … implementations (Murakami (+ Ianni))
- Break
- Use case (Schraml case), analysis & solution (Martin commentary)
- Use case (Wallcraft case), analysis & solution (Martin commentary)
- Lunch
- Use case team exercises
- Setup & team work
- Presentation of analysis & proposed solution
- Break
- Demonstration of solution
- Wrap up

A Successful Information Management Should Facilitate the Transition to Wisdom

- **Data**
  - Raw, in and of itself
- **Information**
  - Processed data
  - Relational connections
  - Who, what, where, when
- **Knowledge**
  - Applying data and information
  - How
- **Understanding**
  - Appreciation of why
  - Cognitive and analytical
- **Wisdom**
  - Evaluated understanding
  - Deals with the future



G. Bellinger

R.L. Ackoff, "From Data to Wisdom,"
Journal of Applies Systems Analysis, Volume 16, 1989 p 3-9.

Evidence Suggests that Much of the Data From Experiments and Simulations Remains Untouched

- **For files: 66% re-opened once, 95% fewer than five times***
  - As file systems grow in size, just too much to sift through
- **Provenance**
  - Lineage of data products
  - Workflow: data objects, parameters of each process step
- **Metadata**
  - Information about each process step
  - A set of data that describes and gives information about other data
- **Ontology**
  - Structure that captures common terms to describe object properties
  - Controlled-vocabulary or classification structures

*Andrew Leung, et al., "Measurement and Analysis of Large-Scale Network File System Workloads," Proceedings of the 2008 USENIX Annual Technical Conference, Boston, Massachusetts, June 2008.

Scientific Metadata Can Accelerate Science Discovery

- **Metadata has accelerated understanding of large data repositories**
  - Rapid browsing, search, and discovery
  - Facilitates data comparison: both experiments and simulations
  - Creates a rich knowledge base of the science for newcomers
- **Metadata and provenance tracking should be embedded in workflows**
  - Automated as much as possible to not burden the scientist
- **Metadata can go beyond simple scalars**
  - Includes text for comments or an electronic logbook
  - Dates & times for historical logging
- **Able to be used in a variety of UIs**
  - Web browsers, scripts, interactive graphics, custom code, etc

SLM Provides For User Metadata

- **SLM MCAT is very powerful**
  - This power can be used for accelerating science
- **One goal of the Navy Pilot is to ID several "metadata interested" scientists**
  - Do one or two proof of concepts to show value
  - Requires scientist to work with us to create something of value
- **For your science, where can metadata add value?**
  - Are there workflow bottlenecks that can be alleviated?
  - Are there things desired to be done that can not be done now that metadata might help?
- **A variety of avenues can be investigated**
  - Web portals, custom GUIs (IDL, Python, etc.), scripts
  - Our thinking is to not require scientists to know SLM specific commands

Workflow

- Seamless integration between science and logistics for archiving
- Trying to place archive activities using SLM behind the scene
- Customizable and modular to support wide range of scientific discipline
- Could be input oriented or output oriented (put / get)

Challenges

- Advanced SLM syntax too complex
- SLM has steep learning curve
- Try to remove manual interactive steps which prove to be error prone and repetitive

Methods

Aspect	Script Based	API Based
Difficulty	Easy	Medium
Language	Any scripting language	C, Java, .Net
Knowledge of SRB concepts	Minimal	Proficient
Customizability	SRB CLI as is	High
Speed	Interpreted script	Compiled program
Interpretation effort	CLI text output parsing	Binary data structures

The implementation language choice needs to consider not only the developer's skill set but also maintenance by others across time and possibly end-user preference.

Where Scripts Can Help:
SLM Ingest Assistance

- Script to ingest or load files into SLM system based on some rules
- Setting up scientific or administrative metadata key-value pairs information to SLM system
- Can be implemented as module for reuse and compounded/complex workflow

Where Scripts Can Help:
SLM File Retrieval Or Data Analysis

- Script to download files/directories from archive
- Script to generate a report from SLM Metadata Catalog (MCAT) regarding files/directories owned by you
- Can be implemented as a module for reuse and compounded / complex workflow

How To Start:
Problem Statements

- Start here
- Problem has to be well defined and bounded
- Could be science driven problem or systems driven problem (archive/computer science) problem, or combination of the two

SLM Functionalities

Metadata

- Provenance
- Scientific metrics
- Schemes (grouping of Metadata) e.g. Dublin Core, User Value, Table of Content (TOC)
- Post processing (analysis and virtualization)

Archive and Migration

- Retention period
- Migration rules
- Replication rules
- Synchronization rules
- Disaster Recovery Set and multi copy
- Ingestion (put)
- Download (get)

Sharing

- Ticket (one time token to grant SLM users conditional access to files without creating group membership or ACLs)
- Data Object and Collection ownership and Access Control List (ACL)
- Virtual Collection

Strategy

- **Break down problems into multiple components**
  - The Output Problem – what metadata do I need to solve my problem and how do I retrieve it and use it?
  - The Input Problem – how do I develop metadata and tag files with it to solve the output problem?
- **Handle input arguments or input files**
- **Verify and validate inputs**
- **Authenticate to SLM**
- **Begin:**
  - Construct SLM command based on input
  - Parse SLM command outputs
  - Error checking
  - Save output buffer and optionally dump it to screen
- **End:**
- **For compounded workflow, repeat Begin**

CEW Scommands: Sretain

- **Problem: HPCMP wishes users to be responsible for setting Retention Period, HPCMP Project ID and other critical attributes. SLM syntax deemed to be too complex and requires too a steep learning curve.**
- **Proposed Solution: Create a script "wrapper" that accepts a simpler syntax for critical attribute updates**
- **Advantages: Relatively easy with minimal knowledge of SRB concepts**
- **Challenges: Changes in SRB native commands and syntax can "break" script "wrapper"**
  - This creates future maintenance and regression testing workload.

CEW Scommands: Sretain

- **The Sretain command will accept a date in many formats and an object or objects to act on.**
- **It was recommended that Sretain also include flags for the Archive scheme Booleans for archive, DR and purge.**
  - The -archive, -noarchive, -dr, -nodr, -purge, -nopurge flags were included to address this.
- **The man page was the first deliverable**
- **The Sretain command man page was completed nearly a year before "volunteers" were requested to create Sretain and Sdata.**

CEW Scommands: Sretain

- **Lines 56-139 - Define the Admin scheme attributes this script works with.**
- **Lines 160-164 - This script uses the Sscheme native SLM Command possibly with the recursive argument.**
  - $CMD = 'Sscheme'; # Use native SRB Sscheme command
- **Lines 174-176 - Handle Data Object arguments**
- **Lines 320-342 - Handle Days Date argument**
- **Lines 659-749 - convert_dwmy subroutine to convert integer days, weeks, months to date.**
- **Lines 586-657 - srb_date subroutine to format date for SLM**

CEW Scommands: Sreview

- **Problem: HPCMP wishes users to be responsible for setting Admin.Last_Review_Time attribute. SLM syntax deemed to be too complex and requires too a steep learning curve.**
- **Proposed Solution: Create a script "wrapper" that accepts a simpler syntax for critical attribute updates**
- **Advantages: Relatively easy with minimal knowledge of SRB concepts**
- **Challenges: Changes in SRB native commands and syntax can "break" script "wrapper"**
  - This creates future maintenance and regression testing workload.

CEW Scommands: Sreview

- **Lines 58-63 - This script uses the Sscheme native SLM Command possibly with the recursive argument.**
  - $CMD = 'Sscheme -w -val Admin.Last_Review_Time::[sysdate]';
- **Lines 65-71 Handle Data Object arguments**
- **Special processing to handle wildcard.**
- **Native SLM Command is "shelled out" and wildcard must be protected from expansion.**
  - This processing is also in Sretain.
- **No date or other attributes are handled by Sreview which was created by throwing away a lot of Sretain and a modification of the command being invoked.**

CEW Scommands: Sdata

- **Problem: HPCMP wishes users to be able to add arbitrary metadata to support enhancing science. SLM syntax deemed to be too complex and require a steep learning curve. SLM included a Character Large Object attribute in the users scheme advertised to be able to handle arbitrary metadata encoded in XML.**
- **Proposed Partial Solution: Add the Name_Value scheme allowing arbitrary metadata pairs without requiring XML parsing and editing.**
- **Advantages: Relatively easy setting of metadata without the requirement to develop XML templates, identify XML editor, etc.**
- **Challenges: Querying against multiple Name_Value attributes for a single object requires complex syntax.**

CEW Scommands: Sdata

- Problem: HPCMP wishes users to be able to add arbitrary metadata to support enhancing science. SLM syntax deemed to be too complex and require a steep learning curve. Querying against multiple Name_Value attributes for a single object requires complex syntax.
- Challenge: Simpler query syntax of Sdata not supported by other native SLM commands such as Sget and Srm to define object to be acted upon.
- Proposed Solution: Create a script "wrapper" that accepts a simpler syntax for working with arbitrary metadata. Include a --verbose option that reports the native syntax of a query that can be used with other native SLM commands.
- Advantages: Relatively easy with minimal knowledge of SRB concepts. The --verbose option helps users with more complex native SLM syntax.

CEW Scommands: Sdata

- lines 11-724 - a number of internal functions
- lines 795-928 - Parse input and perform some requested actions
- lines 936-1036 - perform remaining requested actions
- line 696 Sdata uses Spwd to validate Scommand availability and Sput to test functionality in an undocumented test function accessed with "-t|--test”
- Sdata uses Sscheme for updating metadata
- Sdata uses Sls for the search function
- Sdata creates a temp file which requires ${WORKDIR} to be set and point to a place the user can write.

SLM Navy Pilot scripts

Case 1: Managing Container Files More Efficiently
Using SLM (Wallcraft)

Managing Container Files More Efficiently Using SLM (Wallcraft)

- **Goal: To preserve, manage, and query offline containerized files without having to excessively stage the archived Table of Contents or pull entire container file from tertiary offline storage**
- **Side benefits:**
  - Online table of content query
  - Ability to detect corrupted simulation run (inherent from absence of critical files in container Data Object)
  - Discover trending from containerized Data Object
  - Forecast storage usage by file size accumulation trending from a year's simulation run results

Analysis: The Output Problem

- **Searching container file content**
  - Locate ‘files within container files’
- **Insight:**
  - Storing a table of contents into SLM Metadata Catalog allows users to pinpoint containerized files in the archive that meet his/her criteria (e.g. file_size, file_name, uid) using online metadata search rather than lengthy retrieval of containerized files from the archive
- **Desired SLM command**
  - Sscheme
  - Sls
  - Sput

TOC Metadata Scheme

- **What metadata is needed?**

row_id	unix_mode	unix_uid	unix_gid	file_size	date_modify	file_name
0	-rw-rw-r--	501	501	519	2013-06-06 16:41	file1.txt
1	-rw-rw-r--	501	501	560	2013-06-06 16:41	file2.txt
2	-rw-rw-r--	501	501	519	2013-06-06 16:41	dir2/file3.txt
3	-rw-rw-r--	501	501	519	2013-06-06 16:41	dir2/file4.txt

Analysis: The Input Problem

- **Data Source**
  - User supplies Python script SrbTarManifest.py script with a absolute or relative path to one or a group of GNU tar compatible container formats (e.g. .tar, .tar.gz, .tgz)
  - Python script SrbTarManifest.py extracts manifest of POSIX tar container file using Python Tar library and saves Table of Content (TOC) metadata information in batches of 5000 rows from container files, after container file(s) is/are successfully ingested to SLM. SrbTarManifest.py (line 155- 166)
- **A new metadata scheme called TOC was added to SLM to solve this use case.**
  - Scientist and Application Developer worked closely with CEW and GA to propose new SLM scheme with site wide access

Solution

- **Input/ingest side script features**
  - Parse command line argument, verification, and validation. See SrbTarManifest.py line 97-132
  - Ingest container files to SLM/SRB. See SrbTarManifest.py line 129-135
    - cmd = "Sput "
    - cmd = cmd + "{0} .".format(' '.join(args))
    - (rc,out) = execSMe(cmd)
  - Generate and save metadata to TOC scheme.See SrbTarManifest.py line 147- 172
    - (rc,out) = execMe("Sscheme -w -scheme TOC -file {0} {1}".format(path, realfile))
- **Search file in archived container using metadata**
  - Search and output search result. See SrbTarSearch.py line 83
    - Sls -R -policy “TOC.file_name like ‘to_be_found.dat’" /home/margom/test

Case 2: Scientific Metadata Management With SLM (Schraml)

Scientific Metadata Management With SLM (Schraml)

- **Goal: To preserve, manage, and discover flexible metadata key values without referring to manual notes or readme files, and reduce the number of files (and storage space) that need retention and continuing management over long periods**
- **Side benefit**
  - Parameter sweep analysis is easier
  - Reproduce older simulation run
  - Preserve ontology and terminology
  - Ability to query a series of important metadata values to produce meaningful charts and plots without lengthy archive file retrieval

Analysis: The Output Problems

- **Scenario -- Exploration**
  - Discover metadata keys (vocabulary) on current SLM Collection path (script argument: --explore –name)
  - Discover metadata values for a specific metadata key (e.g. runID) on current SLM Collection path (script argument: --explore “runID”)
- **Scenario -- Collection Search**
  - Find Collection(s) that matches your criteria
  - Can search with multiple criteria (e.g. runID=10450, RHAWidth=104 mm)
- **Scenario -- Plot Mode**
  - Create 2 dimensional data plot file from current SLM Collection path

Analysis: Scenario - Exploration

- **Insight:**
  - Discover the terminology used in metadata keys
  - Based on discovered vocabulary/terminology, select possible values of the key. For example: listing runID on current SLM Collection path shows count of completed experiments and corresponding runID values; output could quickly identify duplicate runID metadata values
  - Name_Value scheme (key value pair) is tagged/associated to a parent Collection containing one simulation run artifacts

Analysis: Scenario - Collection Search

- **Insight:**
  - From knowledge gathered in Exploration step, It is possible to search and get/download Collections (folders) based on a Metadata key (e.g. –coll “StudyName::WHA RHA Study” will give you a list of matching Collections to get/stage from archive)

Analysis Scenario - Plot Mode

- **Insight:**
  - With knowledge from explore and collection stages, user can group matching metadata criteria, X-axis metadata, and Y-axis metadata
  - Without expensive archival commands, user is able to plot a two dimensional chart using a third party tool such as Microsoft Excel or GNUPlot.

Name_Value Metadata Scheme (Partial)

Row_id	Name	Value
0	RunID	20120522002
1	Comment	This simulation is part of a parameter study to determine the influence of constitutive model and model para….
2	PenLength	203.8 mm
3	PenDiameter	6.8 mm
4	PenMass	130 g
5	PenNoseShape	Hemispherical
6	PenMaterial	93W-5Ni-2Fe
7	PenStrModel	Johnson-Cook
8	PenStrModelPara	Weerasooriya
9	RHAThickness	152.4 mm
10	RHAWidth	152.4 mm

Analysis: The Input Problem

- **Data Source**
  - Manually created Readme.xml containing metadata key value pair. See partial output below
    - <simulation>
    - <RunID>20120716000</RunID>
    - <Comment>This simulation is part of a parameter study to determine the influence of constitutive model and model parameter selection on the penetration of semi-infinite RHA targets by monolithic WHA rods. This simulation ran to completion and the results were compiled into the overall study.</Comment>
    - <StudyName> WHA RHA constitutive model parameter study</StudyName>
    - <PenLength>203.8 mm</PenLength>
    - <PenDiameter>6.8 mm</PenDiameter>
  - Ruleset dictionary structure controlling Disaster Recovery and Retention Time attributes
    - Example:
      - <rule name="Output file">
      - <name>Output files for CTH</name>
      - <pattern>oct*</pattern>
      - <attributes>
      - <Admin.Retention_Period>365</Admin.Retention_Period>
      - <Admin.DR_behavior>yes</Admin.DR_behavior>
      - </attributes>
      - </rule>

Solution: Scenario – Input

- **Matching runID command line argument with runID value from Readme.xml. See SrbRegisterUtil.py line 325-327**
  - if dElements['RunID'] != RunID:
  - print "Error! Argument RunID ({0}) does not match XML file RunID ({1})".format (RunID,dElements['RunID'])
  - sys.exit()
- **Ingest directory and its content to SLM. See SrbRegisterUtil.py line 358**
  - Sput –Rf /home/source /My/Collection
- **Set administrative metadata attributes (DR_Behavior and Retention_Period) to /My/Collection. See SrbRegisterUtil.py line 381-388**
  - Sscheme –w –R –val Admin.Retention_Period::365 /My/Collection/*.hth
- **Parse and store Readme.xml key-value metadata pairs. See SrbRegisterUtil.py line 421-441**
  - Sscheme –w scheme Name_Value –val “comment::a new comment, runID:: 1234” /My/Collection

Solution: Scenario – Explore

- **See SrbQueryUtil.py line 429**
  - `Sscheme -l -scheme Name_Value  | egrep -w 'Name|Value' | grep 'string'`

Solution: Scenario – Collection Search

- See SrbQueryUtil.py line 280
  - SgetD -R –policy “DATA_OBJECT.data_id IN (select DATA_OBJECT.data_id where Name_Value.Name = ‘runID' AND Name_Value.Value like ‘1234')“

Solution: Scenario - Plot

- **See SrbQueryUtil.py line 155-165:**
  - SgetD –R –policy “DATA_OBJECT.data_id IN (select DATA_OBJECT.data_id where Name_Value.Name = RHAThickness AND Name_Value.Value like ‘152.4 mm’”

Use Case 1 Compare Metadata

- **Problem: A Chemist wishes to compare her simulation run output from last week simulation run output and generate basic comparative error analysis**
  - Step 1: Evaluate Target collection and source directory, error check
  - Step 2: Run Sput from source directory to target Collection
  - Step 3: Generate metadata metrics and save them using Sscheme call (now_metrics)
  - Step 4: Retrieve last week's simulation metadata using Sscheme –l call and save metrics to (past_metrics). Avoids retrieving entirety of last week's files from tape.
  - Step 5: Evaluate diff between past_metrics and now_metrics
  - Step 6: Print basic comparative error analysis

Use Case 2 Tar files

- **Problem: A Meteorologist wishes to archive his simulation run output artifacts numbering in thousands in a tar file, but he wants to access tarfile's Table of Contents (TOC) online**
- **Note: A tool had been developed for this purpose called SrbTarManifest.py**
  - Step 1: Evaluate Target collection and source directory, error check
  - Step 2: Run POSIX tar command to create archive
  - Step 3: Run POSIX tar –tvf command to list resulting tar file
  - Step 4: Use Sscheme call to save tar –tvf command to TOC scheme
  - Step 5: Print success/failure report

Use Case 3 Post Processing

- **Problem: A Mathematician modeling scientist wishes to run post processing visualization using intermediate output file from his some of the runs completed in the previous week**
  - Step 1: Use a query script to locate desired runs. They have already been tagged with key result characteristics as metadata param_q value of 5
  - Step 2: Use query script to locate previous simulation results with param_q in the range of 4 <= q <= 6 and date range of (today minus 7 days until today) and list the Collection names
  - Step 3: Extract selected Collections to Center-wide File System and set proper permission so the DACC collaborators could access them
  - Step 3: Contact DACC about visualizing all the related output files for differences in the result space

Group Exercise Setup

- **Break into up to three groups**

Group Exercise #1

- **Find missing artifacts from a set of HPC run using Name_Value metadata in archive**
- **Hints**
  - Each HPC run has metadata key “runID”. runIDs have sequential numbering, thus user is able to detect missing (due to failure) runs
  - E.g. runID of science.0001, science.0002, science.0003, science.0004
- **Setup**
  - Create three sub-Collections, with following metadata: Name_Value.name[0] = runID, and Name_Value.value[0]={science.0001, science.0002, science.0005}

Group Exercise #2

- **Goal: To detect a HPC simulation run failure using TOC scheme**
- **Hint:**
  - Failures to run is indicated by <1000 bytes size in some stages/intermediate files in TOC
- **Setup**
  - Create two directories, one directory named 2run1 containing a 500 bytes file *mydata.1*. Create another directory name 2run2 containing a 2 MB file *mydata.1*.
  - Archive both directories so you will have two .tar.gz files.
  - Ingest both .tar.gz files to SLM

Group Exercise #3

- **Goal: Performance Degradation detection and performance improvement quantification in next generation HPC machines**
- **Hints:**
  - Linpack is a benchmark application for parallel computers. Results from Linpack is typically stored in TOC metadata scheme, including its start time and end time. From the two times, compute time difference (ΔT) in seconds and insert its value to Metadata Scheme Name_Value as “delta_t” key
  - Use computed delta_t to compare multiple Linpack simulation run.
- **Setup:**
  - Create two directories 3run1, 3run2, each should have two files *linpack.start* and *linpack.end*. Ensure that 3run1’s linpack.end has modification time of (now + 5 minutes) and 3run2’s linpack.end has modification time of (now + 3 minutes)
  - Create tar.gz archive of 3run1 and 3run2
  - Ingest 3run1.tar.gz and 3run2.tar.gz to SLM/SRB

Closing Remarks and Questions