

APS Data Management System



Siniša Veseli

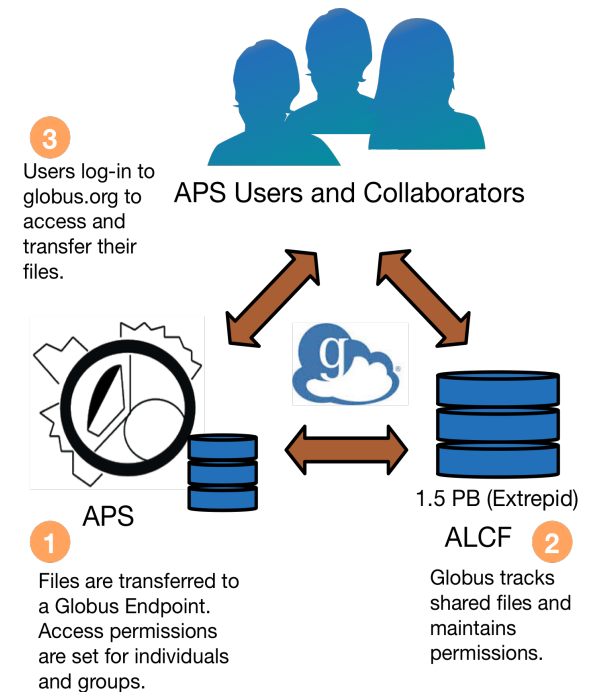
D.P. Jarosz, J.P. Hammonds, N. Schwarz (SDM)

R. Sersted, D. Wallis (IT)



A Bit Of History

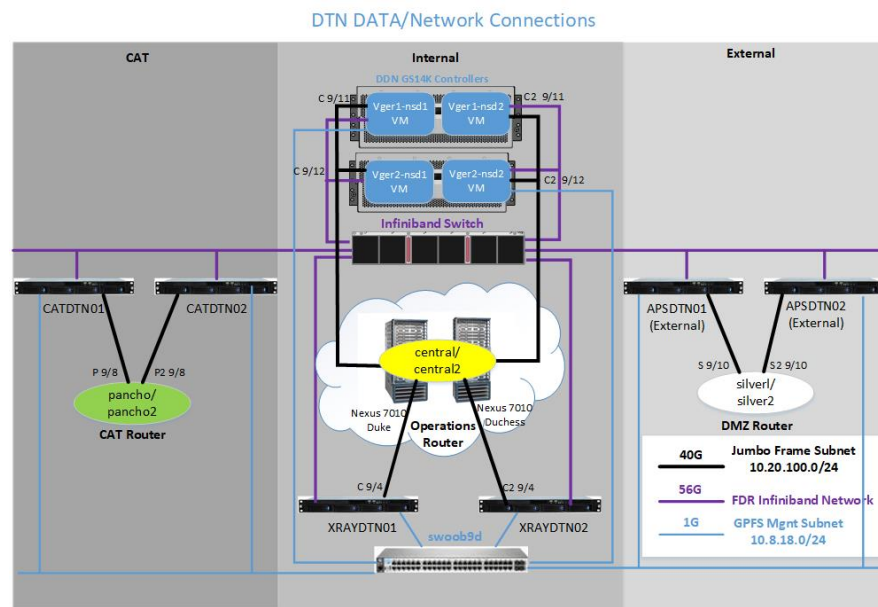
- 2013: Tao Fusion project (LDRD) acquired XSTOR storage (250 TB)
- September 2014: APS Data Management project started
 - Goal: provide APS users with means to easily access their data remotely using Globus Online
- October 2015: First successful software deployment at 6-ID-D (D. Robinson)



- January 2017: Transition to DDN storage (1.5PB) with high-performance GPFS file system, data redundancy, and 2x10Gbps network links
- March 2018: New VM cluster used exclusively for DM virtual machines
 - Total of 512GB RAM, 144 CPU cores (72x2 due to hyper-threading), 2x10Gbps network links

A Bit Of History - Current Storage Hardware

- September 2018: Transition to Voyager storage from DataDirect Networks (DDN)
 - GridScaler GS14KE file storage appliance: 2 controllers, each controller running 2 VMs (GPFS servers)
 - 6 DTNs: 2 for XRAY beamlines, 2 for CAT beamlines, 2 for Globus Endpoint
 - Each DTN has a 1 Intel Xeon 6144 3.5GHz CPU, 128GB RAM, 40 Gbps ethernet link, 56 Gbps Infiniband, 2 x 400GB SSD drives (mirrored)
 - Storage configuration: 5 90-bay disk enclosures, 1 72-bay enclosure with embedded controllers, 90 12TB SAS drives for data, 12 1.9 TB SAS drives for metadata
 - Initial usable storage space: 837 TB
 - Can add up to 20 enclosures with max storage capacity of 16.5 PB
 - Used ALCF storage to backup existing data
- January 2019: Expanded storage capacity to 3.7 PB
 - Restored experiment data from ALCF backup
- January 2020: DM VM cluster expansion: 4 new hypervisors, additional memory



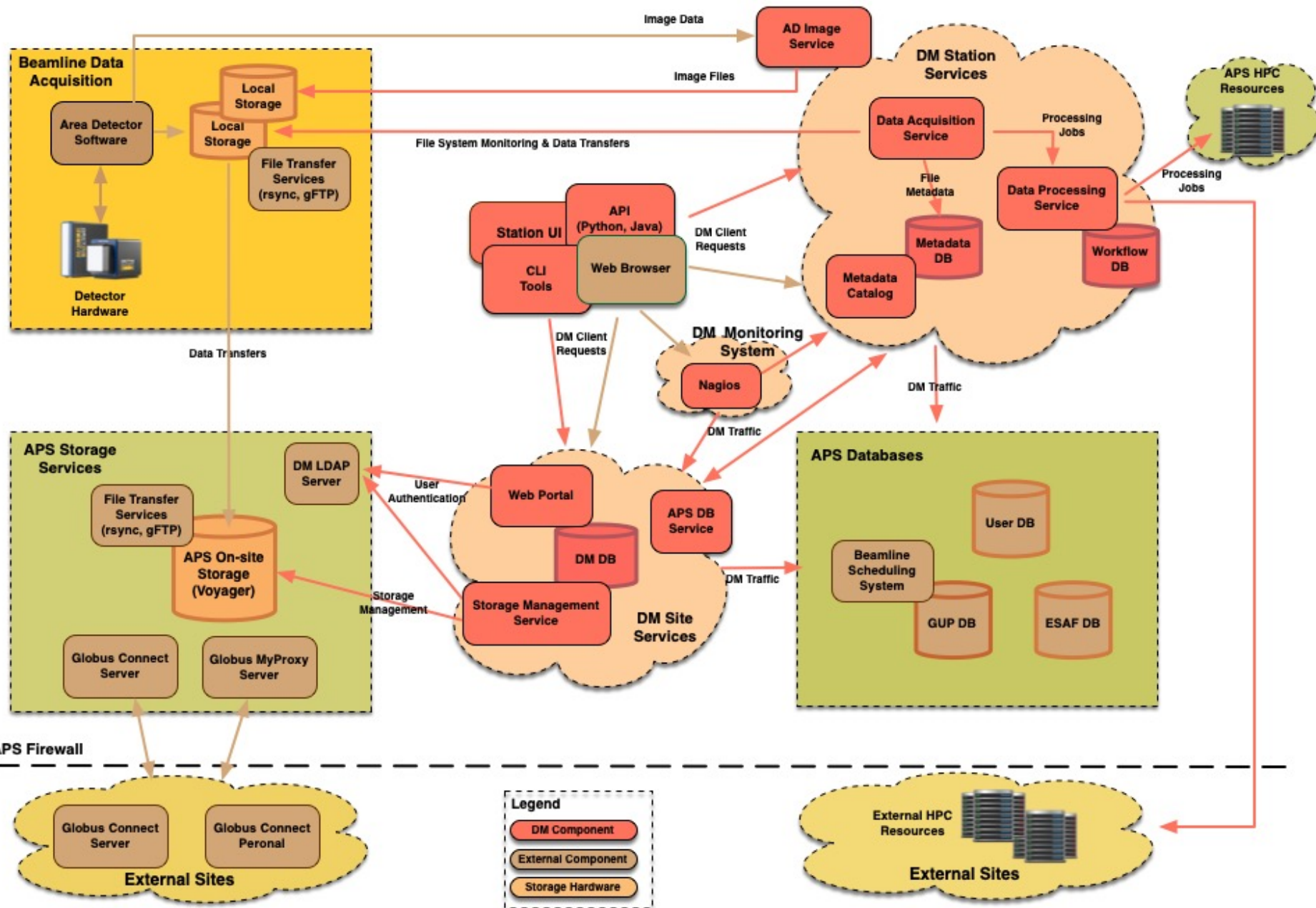
14 Sep 2018
RPS,DWL,MLW

APS Data Management Goals

- Specific data management needs typically vary from beamline-to-beamline, mostly depending upon the types of detectors, X-ray techniques, and data processing tools in use
- However, most of the data management requirements are related to a set of tasks common to all beamlines:
 - *Storage area management* (e.g. movement of acquired data from local storage to a more permanent location, data archival, etc.)
 - *Enabling users and applications to easily find and access data* (metadata and replica catalogs, remote data access tools)
 - *Facilitating data processing and analysis with automated or user-initiated processing workflows*
- APS Data Management (DM) project strives to help with these tasks

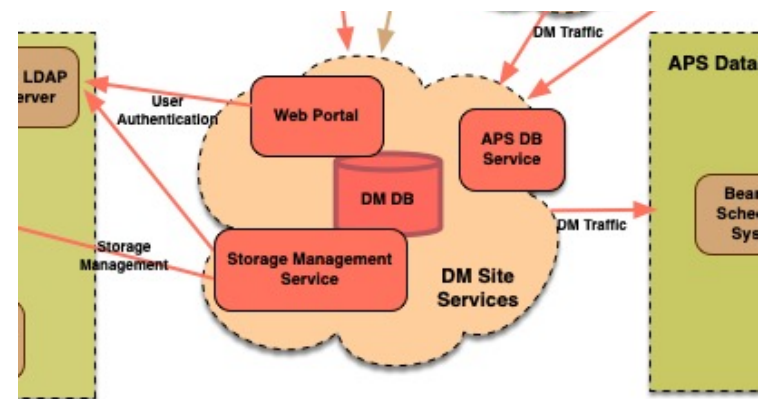


APS Data Management System



Site Services

- **DM Database (PostgreSQL)**
 - Maintains information about users, experiments, and beamline deployments
- **Storage Management Service (Python, REST)**
 - Runs on the storage head node
 - Provides experiment management services
 - Interacts with LDAP and APS Databases
 - Controls storage file system permissions, which enables data access for remote users
- **APS DB Service (Python, REST)**
 - Enables easy access to ESAF and GUP information
- **Web Portal (Java EE, Glassfish)**
 - Experiment management
 - Support for beamline deployments
- **Automated utilities for synchronizing DM user information with APS User Database**



Argonne NATIONAL LABORATORY Data Management Portal

Home Experiments Experiment Stations Experiment Types Role Types Users Logout

Experiments

Id	Name	Type	Station	Description	Start Date	End Date	Actions
51	Smith2017	MPE	34IDE	Pressure Induced Dynamics in Colloidal Suspensions	10/08/2017 10:39 PM CDT	10/18/2017 08:52 PM CDT	
52	Wilson2017	XMD	8IDI	Study of Nanocrystal Lattices	11/08/2017 11:11 PM CST	11/13/2017 12:05 AM CST	
53	Brown2017	FSD	28M	Lipid Degradation Studies	12/13/2017 01:44 PM CST	12/25/2017 02:25 PM CST	
54	Johnson2017	XMD	6IDB	Cryogenic Solvents in Real Environments	05/27/2017 01:46 PM CDT	06/07/2017 12:25 PM CDT	
55	Jones2017	SP	33ID	Polymer Time Correlations	01/07/2017 08:35 AM CST	01/15/2017 01:43 PM CST	
56	Williams2018	XRIM	ID3	PDF Solvent Measurements	01/05/2018 08:32 AM CST	01/23/2018 11:07 PM CST	
57	Taylor2018	EDD	11IDB	Temperature Variant Superconductor Analysis	01/16/2018 09:24 AM CST	01/19/2018 01:20 PM CST	
58	Evans2018	EDD	11IDC	DNA Degradation Probes	01/26/2018 01:59 PM CST	02/02/2018 11:35 AM CST	
59	Brown2018	MPE	7ID	X-ray Measurements of Biological Surfactants	01/30/2018 03:53 PM CST	02/07/2018 05:15 AM CST	
60	Davies2018	FSD	68M	Micro-diffraction of Porous Building Materials	02/15/2018 05:06 PM CST	02/27/2018 07:24 PM CST	
61	Wilkinson2018	XMD	1ID	Nano-composite Materials Under Stress	03/01/2018 07:31 AM CST	03/24/2018 08:22 AM CDT	
62	Johnson2018	XRIM	ID3	Rare Earth Ions Near Interfaces	02/01/2018 10:31 AM CST	02/14/2018 05:33 PM CST	

Station Services: DAQ Service, Metadata Catalog

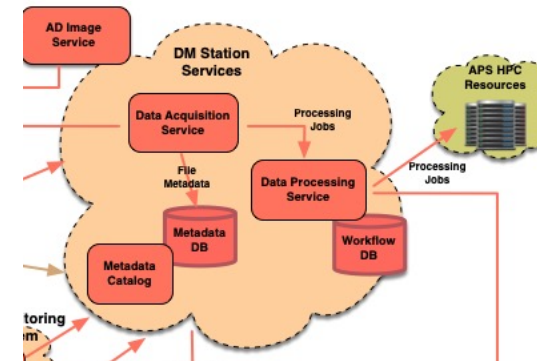
- Each beamline deployment (“DM Station”) includes several Python services accessible via REST interfaces: DAQ Service, Metadata Catalog and Processing Service

- Data Acquisition Service

- Responsible for data uploads and for monitoring local file storage
- Customizable, plugin-based processing framework
- Plugins handle file transfers, metadata cataloging, interaction with other services

- Metadata Catalog (MongoDB)

- Metadata are arbitrary key/value pairs
- Each experiment has its own file metadata collection
- File metadata can be retrieved using command line or API tools, DM Station GUI, or via the Mongo Express application



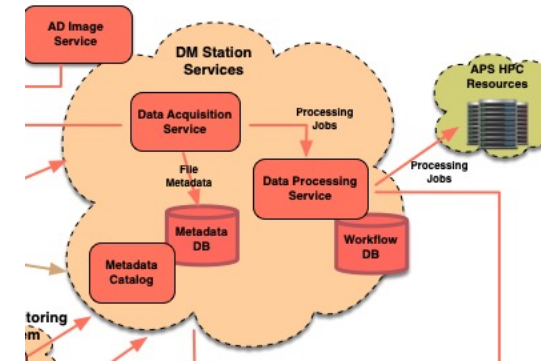
```
dmadmin@sXiddm> dm-get-file-collection-stats --experiment=Smith-Mar18 --display-keys=ALL --display-f
ormat=key-per-line
experimentName=Smith-Mar18
stdDevFileSize=21275858.208
queryDict={}
maxFileSize=251666432
minFileSize=14
collectionSize=1.63133244728e+12
averageFileSize=9634554.76451
nFiles=169321

dmadmin@sXiddm> dm-list-experiment-files --experiment=Smith-Mar18 --display-keys=experimentFilePath,
fileSize,md5Sum
experimentFilePath=pix/macing_000001.tif fileSize=823569 md5Sum=9a87108d60bb3bdd16d58b0c7cd6855
experimentFilePath=pix/macing_000002.tif fileSize=823569 md5Sum=7bef341f3625b6db350e4d7060394ca5
experimentFilePath=pix/macing_000003.tif fileSize=823569 md5Sum=ad7399871ce9b4f6dab637d64b2be2fe
experimentFilePath=pix/macing_000004.tif fileSize=823569 md5Sum=d74b11f00438761209e81ac72525cad0
experimentFilePath=pix/macing_000005.tif fileSize=823569 md5Sum=309e7402992ff870f1d0bd7445eb2c39
experimentFilePath=pix/macing_000006.tif fileSize=823569 md5Sum=53e16027b5c2fe3d4ff640c2a847a4ff
experimentFilePath=pix/macing_000007.tif fileSize=823569 md5Sum=5b4526978e63a8eea035362309b134dc
experimentFilePath=pix/macing_000008.tif fileSize=823569 md5Sum=3b503018bcae697f7b9cf35a0c7b9120
experimentFilePath=pix/macing_000009.tif fileSize=823569 md5Sum=7d27d4842c7f312742ae6e67c44d284e
experimentFilePath=pix/macing_000010.tif fileSize=823569 md5Sum=b0671eeb63507b5668c8a2ff914e8281
experimentFilePath=pix/macing_000011.tif fileSize=823569 md5Sum=66fa93dab6b96bb5f4aff78c6b5ee9
experimentFilePath=pix/macing_000012.tif fileSize=823569 md5Sum=05267d7a2184e69affbfb3aeef6638b
experimentFilePath=pix/macing_000013.tif fileSize=823569 md5Sum=48dcf9cd278be3c76cb2cd7176b9a95b
experimentFilePath=pix/macing_000014.tif fileSize=823569 md5Sum=b7f3fcfb5bd2e2304422a3ab17ad418
experimentFilePath=pix/macing_000015.tif fileSize=823569 md5Sum=b11a0f6383b3795854b5cfea2d79a5af
experimentFilePath=pix/macing_000016.tif fileSize=823569 md5Sum=9bda2ac5161c825d94298d50b0cb8176

dmadmin@sXiddm>
```


Station Services: Workflow Engine

- Processing Service provides support for managing user-defined workflows, as well as for submitting and monitoring processing jobs based on those workflows
- DM workflow is a collection of processing steps executed in order:
 - Workflow definitions are described as Python dictionaries
 - Each processing step must be associated with an (arbitrary) executable
 - Support for input/output variables
 - Processing steps are automatically parallelized if possible
 - Support for batch jobs
 - Support for processing multiple files with a given workflow



```
{
  'name': 'nersc-example',
  'owner': 'biduser',
  'description': 'NERSC Example Workflow',
  'stages': {
    '01-START': {
      'command': '/home/dm/workflows/nersc-example/start-workflow.sh $filePath $id',
      'outputVariableRegexList': [
        'INPUT DIRECTORY: (?P<inputDir>.)',
        'WORKFLOW DIRECTORY: (?P<workflowDir>.)',
        ...
        'NERSC SLURM SCRIPT: (?P<nerscSlurmScript>.)',
      ],
    },
    '02-AUTH': {
      'command': '/home/dm/workflows/nersc-example/get-nersc-ssh-proxy.sh $nerscUsername $nerscPasswordFile $nerscMfaSecretFile $nerscSshProxyFile',
    },
    '03-PREP-JOB-DIR': {
      'command': 'ssh -q -i $nerscSshProxyFile $nerscUserAccount "mkdir -p $nerscResultsDir"',
    },
    '04-COPY-SLURM-SCRIPT': {
      'command': 'rsync -ar -e "ssh -q -i $nerscSshProxyFile" $workflowDir/$nerscSlurmScript $nerscDtnAccount:$nerscJobDir',
    },
    '05-COPY-INPUT-HDF5': {
      'command': 'rsync -ar -e "ssh -q -i $nerscSshProxyFile" $orthosResultsDir/$inputHdf5File $nerscDtnAccount:$nerscResultsDir',
    },
    '06-COPY-INPUT-IMM': {
      'command': 'rsync -ar -e "ssh -q -i $nerscSshProxyFile" $immDir/$immFile $nerscDtnAccount:$nerscJobDir',
    },
    '07-SUBMIT-CORR': {
      'command': 'ssh -q -i $nerscSshProxyFile $nerscUserAccount "sbatch -J $nerscJobName -e $nerscJobDir/$nerscJobName.err -o $nerscJobDir/$nerscJobName.out $nerscJobDir/$nerscSlurmScript $nerscResultsDir/$inputHdf5File $nerscJobDir/$immFile"',
      'outputVariableRegexList': [
        'Submitted batch job (?P<nerscJobId>.)',
      ],
    },
    '08-MONITOR-CORR': {
      'command': '/home/dm/workflows/nersc-example/get-nersc-job-status.sh $nerscSshProxyFile $nerscUserAccount $nerscJobId $nerscJobName',
      'outputVariableRegexList': [
        'Job . state: (?P<nerscJobState>.)',
        'Job . exited: (?P<nerscJobExited>.)',
        'Job . exit status: (?P<nerscJobExitStatus>.)',
      ],
      'repeatPeriod': 60,
      'repeatUntil': '"$nerscJobExited" == "True"',
      'maxRepeats': 120,
    },
    '09-COPY-AUTH': {
      'command': 'ssh -q $orthosUserAccount "mkdir -p $orthosJobRunDir" && rsync -ar $nerscSshProxyFile $orthosUserAccount:$orthosJobRunDir',
    },
    '10-COPY-RESULTS': {
      'command': 'ssh -q $orthosUserAccount rsync -ar -e "ssh -i $orthosSshProxyFile" $nerscDtnAccount:$nerscResultsDir $orthosUserDataDir',
    },
    '11-CLEANUP': {
      'command': 'ssh -q -i $nerscSshProxyFile $nerscUserAccount "rm -rf $nerscJobDir"',
    },
  },
}
```


Station Services: Workflow Engine

- Workflows are owned by beamline DM users and kept in the beamline Workflow Database (MongoDB)
- Processing Service provides REST and OMQ interfaces for adding/updating/deleting workflow definitions, as well as for submitting/monitoring processing jobs

```

sveseli — ssh bluegill2 — 80x31
dmadmin@s8ididm> dm-update-workflow -h | more
Usage:
  dm-update-workflow --py-spec=PYSPECFILE|--json-spec=JSONSPECFILE

Description:
  Update workflow described in the given python (or JSON) spec file.

Options:
  --py-spec=PYSPECFILE  Python spec file.
  --json-spec=JSONSPECFILE
                        JSON spec file (used only if python spec file is not
                        provided).

Common Options:
  -h, --help            Show this help message and exit.
  -?                    Show this help message and exit.
  -v, --version         Print version and exit.
  -d CONSOLELOGLEVEL, --debug=CONSOLELOGLEVEL
                        Set debug level (valid values: CRITICAL, ERROR,
                        WARNING, INFO, DEBUG). Console log level can also be
                        set via DM_CONSOLE_LOG_LEVEL environment variable,
                        --display-format=DISPLAYFORMAT
                        Display format for output objects. Possible options

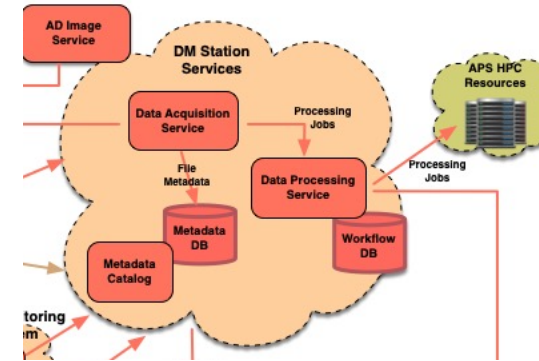
dmadmin@s8ididm>
dmadmin@s8ididm> dm-list-workflows --owner=8iduser
id=5a6035b23441dc3d3fff3a96 name=xpcs8-01 owner=8iduser
id=5a6649e83441dc4e95f1c321 name=xpcs8-02 owner=8iduser
id=5a6a77203441dcc337ab9575 name=xpcs8-03 owner=8iduser
id=5b90571384231d0a7aa74a48 name=xpcs8-04-NERSC owner=8iduser
dmadmin@s8ididm>

```

- Users can access the service via the Python API (REST, OMQ), or CLI
- CLI supports HTML output, which enables easy job monitoring

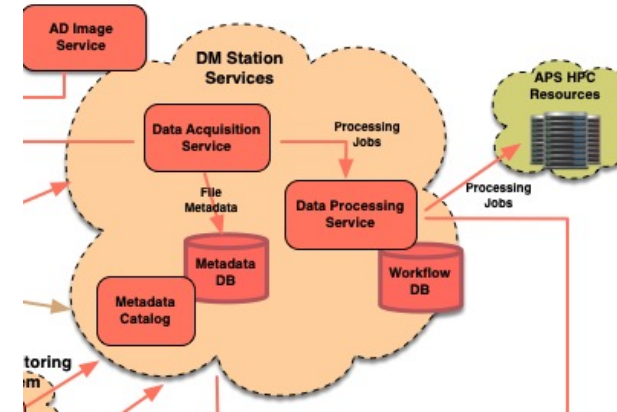
Station Services: Workflow Engine

- Generic framework that can handle both automated real-time processing, as well as user-initiated processing and analysis of data
- Seamless integration with the rest of the DM system components
- User scripts become much simpler, as system takes care of tasks common to most processing and analysis workflows
- System promotes reuse of user scripts/tools by making those easily available to everyone
- Processing jobs are run on a set of designated servers, rather than on user desktops
- Better tools lead to higher productivity, shorter development/testing times, easier deployment, better user experience, etc.
- *Processing Service can be used standalone, or together with other DM Station services in support of fully automated beamline data acquisition and processing pipelines*



Station Services: AD Image Service

- Attaches a PV Access monitor to Area Detector PVA Server plugin channel
- Can save images in JPEG, HDF5, SDDS formats; more can be added if needed
- Uses PvaPy (EPICS7 Python API that wraps C++ libraries)
 - Can keep up with a fully saturated 10Gbps link
 - Actual data rates depend on storage capabilities and the chosen image format (i.e., the efficiency of the underlying Python libraries)
- Image receiver can also be run as a local process, via the *dm-start-image-receiver* command line:



```
dm-start-image-receiver --image-channel=C2:SIM2:Pva1:Image \
--image-format=sdds --n-frames-to-receive=10 \
--output-directory=./%Y/%m/%d --file-prefix=f.%id.%y.%m.%d
```


Monitoring

- Every DM service has a set of monitoring interfaces that enable external applications to find out about its state
- These are used by the custom Nagios plugins that provide up-to-date information about the health of the DM station deployments



Nagios®

General
Home
Documentation

Current Status
Tactical Overview
Map (Legacy)
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Quick Search:

Reports
Availability
Trends (Legacy)
Alerts
History
Summary
Histogram (Legacy)
Notifications
Event Log

System
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Configuration

Current Network Status
Last Updated: Mon Dec 18 08:57:23 CST 2017
Updated every 90 seconds
Nagios® Core™ 4.3.4 - www.nagios.org
Logged in as dmadmin
View Service Status Detail For All Service Groups
View Status Summary For All Service Groups
View Service Status Grid For All Service Groups

Host Status Totals
Up: 21, Down: 0, Unreachable: 0, Pending: 0
All Problems: 0, All Types: 21

Service Status Totals
Ok: 69, Warning: 0, Unknown: 0, Critical: 0, Pending: 0
All Problems: 0, All Types: 69

Service Overview For All Service Groups

Host	Status	Services	Actions
s11dbdm.xray.aps.anl.gov	UP	3 OK	[Icons]

DM 11IDB Services (dm-11idb-services)

Host	Status	Services	Actions
s11ddm.xray.aps.anl.gov	UP	3 OK	[Icons]

DM 11D Services (dm-11d-services)

Host	Status	Services	Actions
s32iddm.xray.aps.anl.gov	UP	3 OK	[Icons]

DM 32ID Services (dm-32id-services)

Host	Status	Services	Actions
dcspc53.dcs.aps.anl.gov	UP	3 OK	[Icons]

DM 35ID Services (dm-35id-services)

Host	Status	Services	Actions
s6bmdm.xray.aps.anl.gov	UP	3 OK	[Icons]

DM 6BM Services (dm-6bm-services)

Host	Status	Services	Actions
s6idbmdm.xray.aps.anl.gov	UP	3 OK	[Icons]

DM 6IDB Services (dm-6idb-services)

Host	Status	Services	Actions
s7bmdm.xray.aps.anl.gov	UP	3 OK	[Icons]

DM 7BM Services (dm-7bm-services)

Host	Status	Services	Actions
apsdata.aps.anl.gov	UP	1 OK	[Icons]

DM Storage Services @ apsdta (dm-apsdata-storage-services)

Host	Status	Services	Actions
bluegill2.aps.anl.gov	UP	1 OK	[Icons]

DM TEST Services @ bluegill2 (dm-bluegill2-test-services)

Host	Status	Services	Actions
cleo.aps.anl.gov	UP	3 OK	[Icons]

DM TEST Services @ cleo (dm-cleo-test-services)

Host	Status	Services	Actions
bmw.cars.aps.anl.gov	UP	3 OK	[Icons]

DM CHMCARS Services (dm-chmcars-services)

Host	Status	Services	Actions
bmw.cars.aps.anl.gov	UP	3 OK	[Icons]

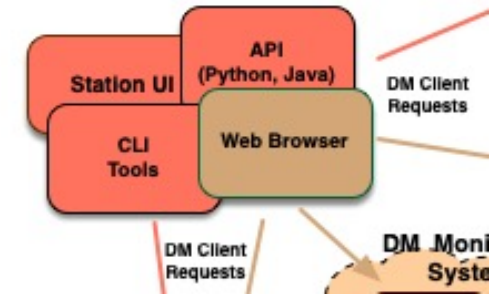
DM BIOCARS Services (dm-biocars-services)

Host	Status	Services	Actions
bmw.cars.aps.anl.gov	UP	3 OK	[Icons]

DM GSECARS Services (dm-gsecars-services)

User Interfaces

- Web browser access to DM Web Portal, Nagios web pages, beamline Metadata Catalog, and Globus Online (for remote data access)
- Python REST services are accessible via DM Python and Java APIs
 - DM Python modules available as Conda packages (*aps-anl-tag/aps-dm-api*)
 - Easy access to ESAF DB and Beamline Scheduling System
- Processing Service provides OMQ interfaces
- Extensive set of command line tools
 - Built on top of Python APIs
 - Session based
 - Fully scriptable
 - Online usage documentation (--help option)
- DM Station GUI (C. Schmitz, D.P. Jarosz)
 - Implemented in PyQt
 - Uses Python REST APIs
 - Easiest way to start using the system



```

dmadmin@6bmdm> ssh bluegill2 -- 80x45
dmadmin@6bmdm> dm-6bm-daq -h
Usage:
  dm-6bm-daq --experiment=EXPERIMENTNAME --data-directory=DATADIRECTORY
  [--duration=DURATION]
  [--dest-directory=DESTDIRECTORY]
  [--upload-data-directory-on-exit=UPLOADDATADIRECTORYONEXIT]
  [--upload-dest-directory-on-exit=UPLOADEDSTDIRORYONEXIT]
  [--process-hidden]
  [--process-existing]
  [--workflow-name=WORKFLOWNAME --workflow-owner=WORKFLOWOWNER]
  [--workflow-job-owner=WORKFLOWJOBOWNER]
  [--workflow-args="key1:value1, key2:value2"]
]
  [--skip-plugins=SKIPPLUGII]
  [--type=TYPENAME]
  [--description=DESCRIPTION]
  [--start-dates=STARTDATE]
  [--end-dates=ENDDATE]
  [--users=USERS]
  [--esaf-id=ESAFID]
  [--proposal-id=PROPOSALID]
  [--run=RUNNAME]
Description:
  Run DAQ for experiment on sta
  be added to the DM database. If
  command will also add roles for :

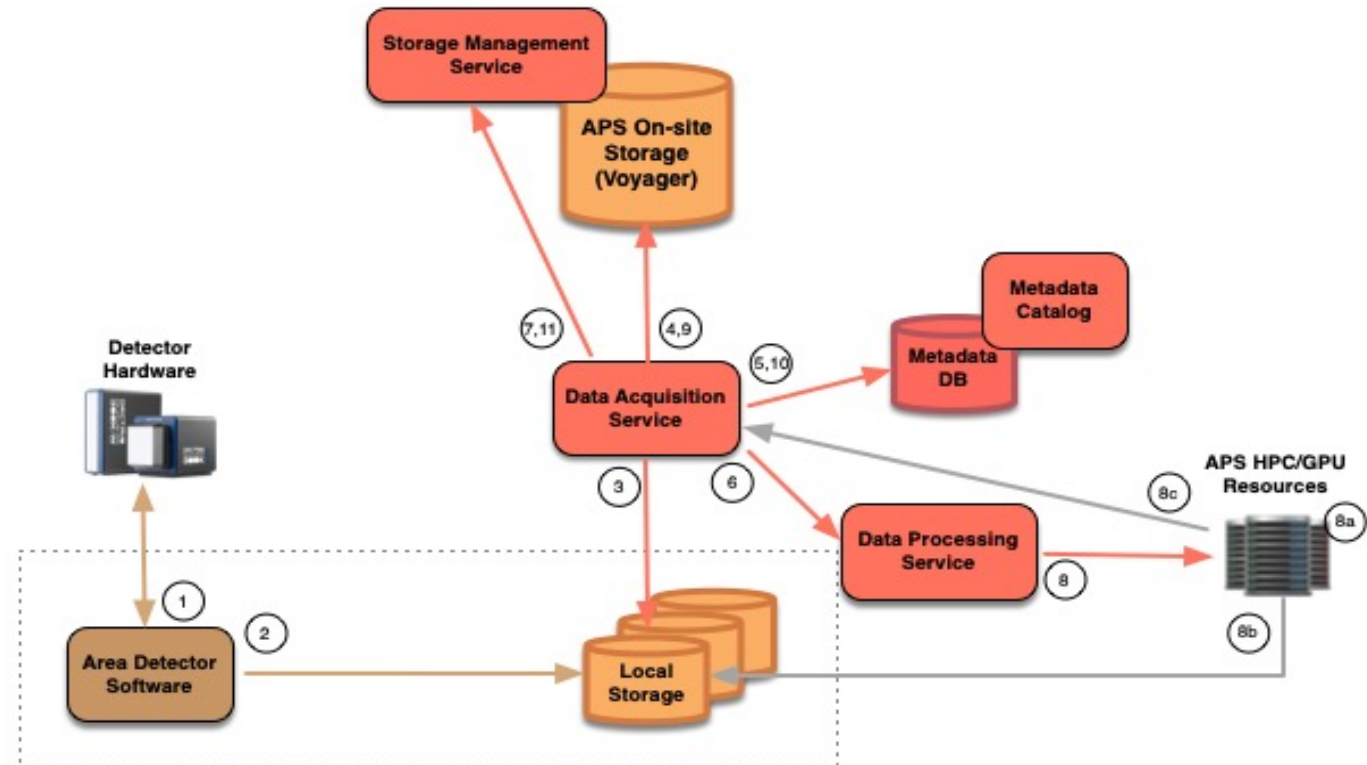
Options:
  --experiment=EXPERIMENTNAME      Experiment
  --data-directory=DATADIRECTORY    Destination
  --duration=DURATION              DAQ durat
  --upload-data-directory-on-exit=upload-data-directory-on-exit
  --upload-dest-directory-on-exit=upload-dest-directory-on-exit
  --process-hidden
  --process-existing
  --workflow-name=WORKFLOWNAME     Workflow name
  --workflow-owner=WORKFLOWOWNER   Workflow owner
  --workflow-job-owner=WORKFLOWJOBOWNER
  --workflow-args="key1:value1, key2:value2"

DAQ Options:
  --dest-directory=DESTDIR        Destination
  --duration=DURATION             DAQ durat
  --upload-data-directory-on-exit=upload-data-directory-on-exit
  
```


Use Case: Data Acquisition & Real-Time Processing

Data Acquisition & Real-Time Processing Flow

- 1 : Receive Image Data
- 2 : Write Raw Image File to Local Storage
- 3 : Monitor DAQ Folder
- 4 : Transfer Image File to APS Storage
- 5 : Catalog Image File Metadata
- 6 : Submit Processing Job
- 7 : Image File Notification
- 8 : Run Processing Workflow
- 8a: Process Raw Image Data
- 8b: Write Processed File
- 8c: Request Upload of Processed File
- 9 : Transfer Processed File to APS Storage
- 10: Catalog Processed File Metadata
- 11: Processed File Notification



Alternative Setup 1: AD to AD



Alternative Setup 2: AD to Image Service



Use Case: Batch Processing Workflow @ 8-ID-I

- 8-ID-I uses X-ray Photon Correlation Spectroscopy technique (XPCS) for the studies of equilibrium fluctuations and fluctuations about the evolution to equilibrium in condensed matter in the Small-Angle X-ray Scattering (SAXS) geometry
- SPEC software is used for instrument control and data acquisition
- For every raw data file SPEC scripts can start DM processing job based on one of the implemented workflows
- Batch processing workflow:
 - 1) Run a custom shell script to prepare processing environment.
 - 2) Copy raw data file to APS HPC cluster using GridFTP (the *globus-url-copy* command).
 - 3) Append XPCS metadata to the data file by running a custom 8-ID-I utility.
 - 4) Submit a processing job to the SGE batch scheduler via the *qsub* command. This job runs a custom 8-ID-I processing executable.
 - 5) Monitor batch job by running a shell script that interacts with SGE via the *qacct* command.
 - 6) Copy resulting output file into designated beamline storage area using GridFTP (the *globus-url-copy* command).

- Jobs are monitored via static web pages generated by a cron job running DM utilities

Output timestamp: 2018/09/26 17:57:21 CDT

Number of processing jobs: 144

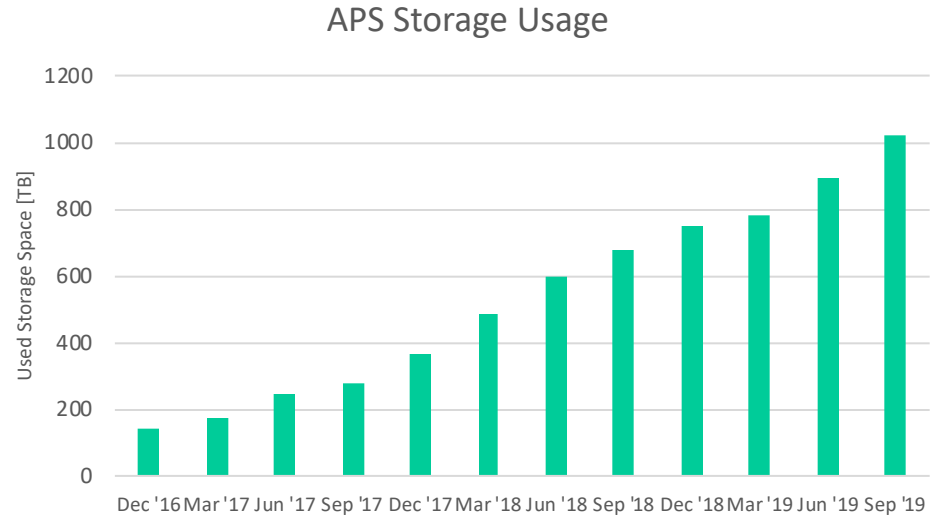
startTimestamp	endTimestamp	sgeJobName	sgeJobId	status	stage	runTime id
2018/09/26 17:56:06 CDT		B137 PI2 hetero 1 step2 10mm Strain010 att0 Lq0 056	19275543	running	5-Monitor	69.58 57023b6
2018/09/26 17:22:45 CDT	2018/09/26 17:24:07 CDT	B137 PI2 hetero 1 step2 10mm Strain010 att0 Lq0 055	19275521	done	6-CopyBack	82.02 1637058
2018/09/26 16:54:41 CDT	2018/09/26 16:55:21 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 054	19275503	done	6-CopyBack	39.42 5678860
2018/09/26 16:52:26 CDT	2018/09/26 16:53:11 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 053	19275498	done	6-CopyBack	45.59 bec3a6a
2018/09/26 16:48:58 CDT	2018/09/26 16:50:07 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 052	19275481	failed	2-XPCS	11.56 598563a
2018/09/26 16:42:35 CDT	2018/09/26 16:43:08 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 051	19275466	done	6-CopyBack	33.48 a7ef69fa
2018/09/26 16:37:19 CDT	2018/09/26 16:37:58 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 050	19275437	done	6-CopyBack	39.20 18e16a4
2018/09/26 16:33:59 CDT	2018/09/26 16:34:39 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 049	19275428	done	6-CopyBack	39.41 ffcde18
2018/09/26 16:30:15 CDT	2018/09/26 16:31:09 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 048	19275416	done	6-CopyBack	53.32 087a7c6
2018/09/26 16:11:45 CDT	2018/09/26 16:12:50 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 047	19275378	done	6-CopyBack	64.78 d132f21
2018/09/26 12:22:28 CDT	2018/09/26 12:23:44 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 046	19275345	done	6-CopyBack	75.74 80f892c
2018/09/26 12:10:59 CDT	2018/09/26 12:12:15 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 045	19275321	done	6-CopyBack	75.59 99fec5b
2018/09/26 11:48:26 CDT	2018/09/26 11:49:45 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 044	19275307	done	6-CopyBack	78.84 395c9ae
2018/09/24 15:56:08 CDT	2018/09/24 15:57:19 CDT	A021 sample8 Gi Ion45deg 2kev f3 1p0EN4 Op26deg 043	19275295	done	6-CopyBack	71.11 4856023

Output timestamp: 2018/09/26 17:57:21 CDT



Summary

- The DM system has grown significantly over the last couple of years, in terms of both its usage and capabilities
 - December 2016: 5 beamline deployments, about 150 TB used storage
 - September 2019: more than 30 beamline deployments, about 1 PB of used storage space, over 2100 experiments in DM DB
- The software can be customized and extended to serve individual beamline data management needs
- *DM workflows can be used to fully automate data acquisition and processing pipelines on APS beamlines*
- Future Plans:
 - Conda-based installation (split git repository), easier deployment outside APS
 - Add workflow and processing job management capabilities to the DM Station GUI, Web Portal
 - Add support for data archiving
 - System monitoring enhancements (provide easy real-time access to system usage information)
 - Expand system usage at APS beamlines, as well as APSU usage
 - Improve documentation (User Guide)



Acknowledgements

- John Hammonds for organizing and managing this meeting
- APS IT group (R. Sersted, D. Wallis, D. Leibfritz, and T. Lutes) for their work on building and maintaining the APS On-site Storage, networking, and virtual machines used to host beamline DM services
- APS IS group (F. Lacap and Y. Huang) for their help and cooperation with accessing APS databases, and their recent development work on ESAF DB API (Y. Huang)
- APS SDM group for many useful discussions (B. Frosik, A. Glowacki, and F. Khan), and help with the system support and development (D.P. Jarosz, J. Hammonds)
- APS 1-ID (J.-S. Park and P. Kenesei), 6-ID-D (D. Robinson), and 8-ID-I (S. Narayanan) beamlines for their patience, support and help with system testing and troubleshooting since the early prototype versions.
- All of those who are using, have used, or attempted to use the system in any way

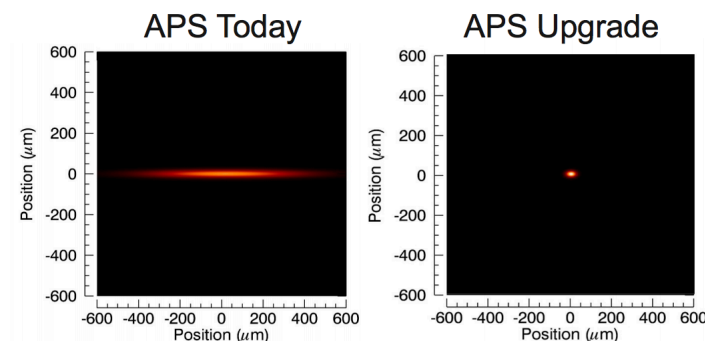


Additional Slides



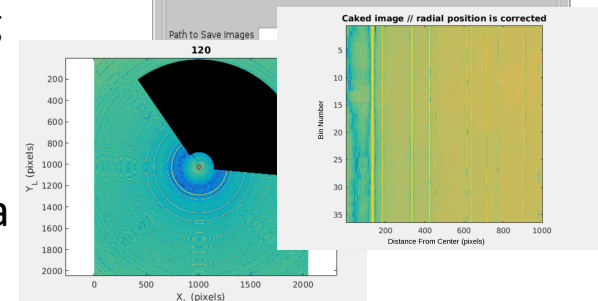
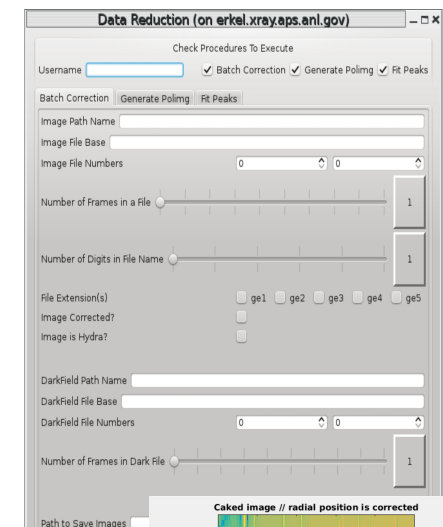
Growing Beamline Data Needs

- X-ray detector capabilities are constantly improving: bigger frames, higher frame rates => more raw data
- APS Upgrade: Higher brightness => more x-rays can be focused onto a smaller area => more raw data in greater detail and less time
- APS 1-ID:
 - Today: 4x Hydra GE detector, 8MB frame, 8 fps => 256MB/s data rate (1TB/hr)
 - Production Data Rates (including overhead, on a good week): 10TB/day, 60TB/week
 - Near future (1-3 years): 2x 2923 Dexela (or similar), 23MB frame, 26 fps => 1.2GB/s data rate
 - Near future: Pilatus 2M (or similar/larger), 9.5MB frame, 250 fps => 2.3GB/s data rate
- APS 8-ID-I:
 - 2010-2016: ANL/LBL FCCD, 2MB frame, 100fps, compressed in real-time with 10% non-zero pixels on the average => about 200MB/s data rate
 - 2016-Today: Lambda 750K, 1.5MB frame, about 10% non-zero pixels, 2000 fps => about 300 MB/s data rate
 - Production Data Rates: 8-10 TB/week (max), 1-2TB/week (average)
 - Future: research on VIPIC (>1MHz frame rate) and UFXC (50 kHz frame rate)

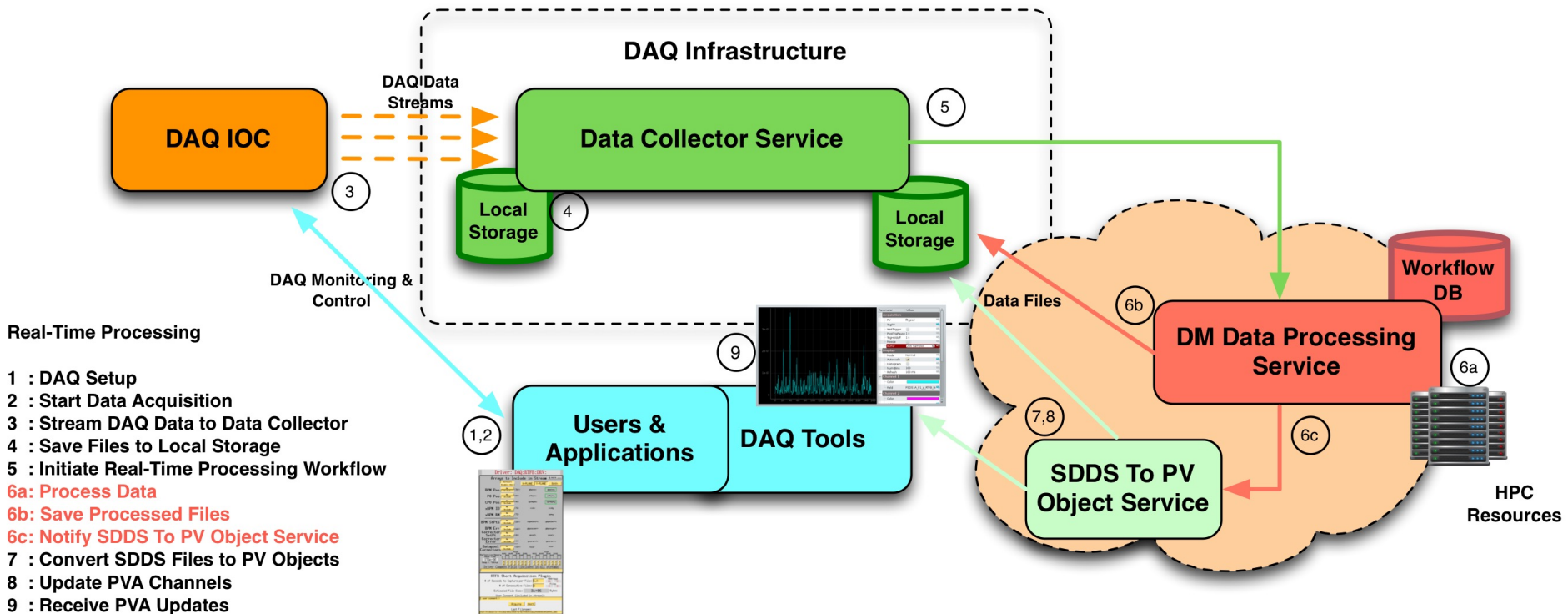


DM Workflow Use Case: Data Reduction @ 1-ID

- APS 1-ID Beamline provides high energy monochromatic X-rays to support several experimental techniques for investigating a wide range of polycrystalline materials: micro-computed tomography (μ -CT), wide angle X-ray scattering (WAXS), small angle X-ray scattering (SAXS), near- and far-field high energy diffraction microscopy (NF- and FF-HEDM)
- Data Reduction Process:
 - 1) Custom GUI is used to prepare data reduction and processing (batch correction, generating polar images, etc.)
 - 2) GUI submits processing job based on the data reduction workflow to DM; the workflow incorporates 1ID Matlab scripts that:
 - a) Extract individual frames from detector data
 - b) Subtract dark field, remove bad pixels
 - c) Integrate to produce intensity vs lattice spacing
 - d) Fit peaks of interest
 - 3) DM runs the job generating processed files/images; processed files keep track of data provenance



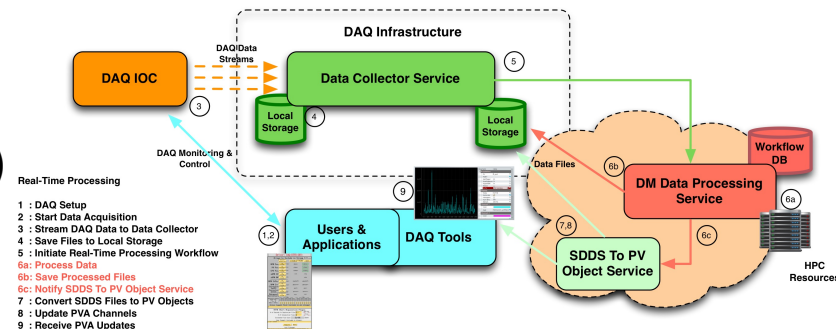
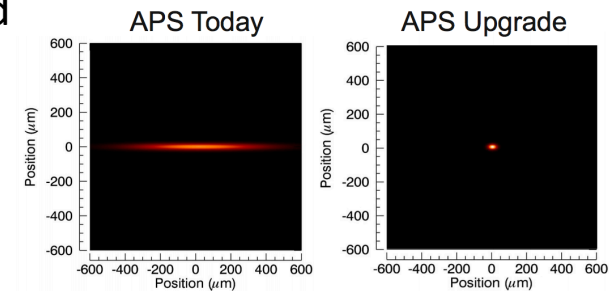
DM Workflow Use Case: Automated Real-Time Processing for APSU



DM Workflow Use Case: Automated Real-Time Processing for APSU

- APS Upgrade will replace existing accelerator with Multi-Bend Achromat Lattice in order to dramatically increase x-ray brightness
- APS Controls group is working on modernizing software that will be used for commissioning, monitoring, diagnostics, troubleshooting, and early fault detection of the new machine
- First use case: real-time Power Spectral Density (PSD) calculation using APSU Real-Time Feedback (RTFB) DAQ data
- User scripts used to do the following:

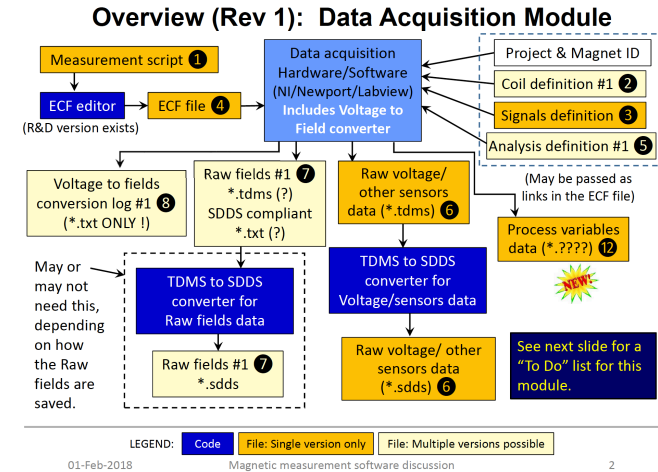
- 1) Monitor “last saved file” PV, keep track of RTFB files
- 2) Prepare PSD calculation (copy RTFB files)
- 3) Run PSD calculation on APS HPC cluster via SSH
- 4) Wait for calculation to finish (look for “done” file)
- 5) Copy resulting files back to user’s workstation
- 6) Cleanup



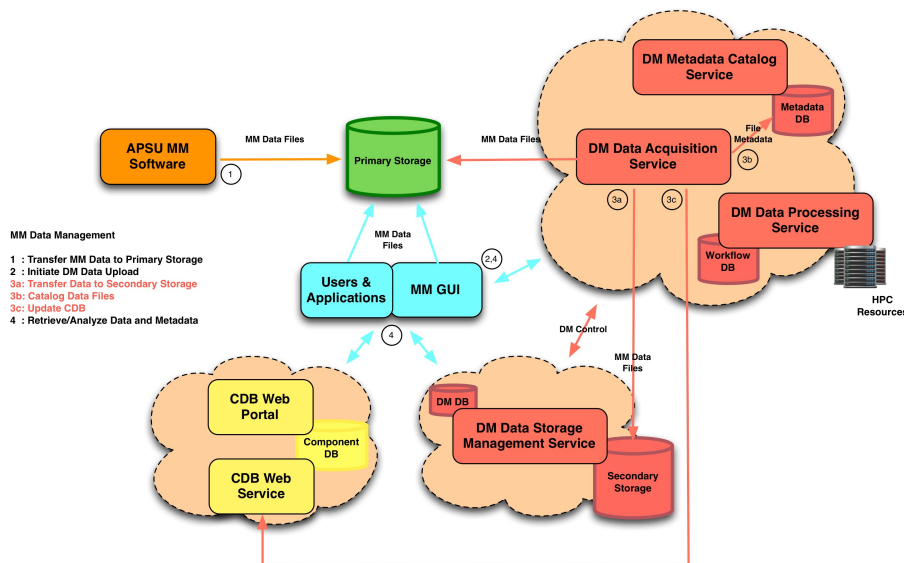
- With the corresponding DM workflow user needs to worry only about details of the actual calculation (#3); number of lines of user code reduced to about 1/3 of the original

System Usage: APSU MM Data (A. Jain)

- APSU uses Component Database (CDB) for identifying and tracking components needed to build the new facility
- APSU Magnetic Measurements (MM) software works with and creates many different types of files: various definition/configuration files, measurement and analysis files, scripts, raw measurement data, PV data, processed data, log files,...
- Over 1K magnets, each will require multiple measurements => MM software will generate large amounts of data and numerous data files



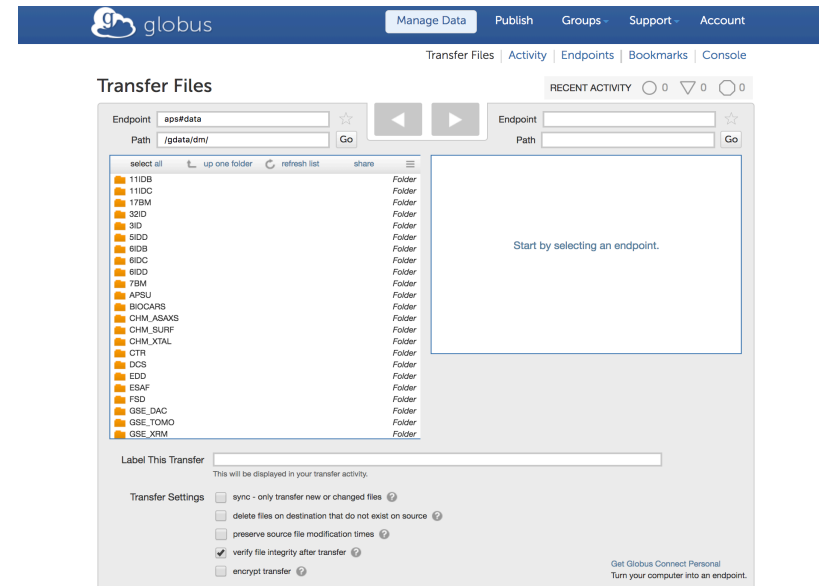
APSU MM Data Management



- DM data upload tools will associate MM experiment with corresponding magnet QR ID in CDB
- During the DM data upload, each MM data file will be processed as follows:
 - Metadata plugin stores file metadata in the APSU DM metadata database
 - CDB plugin adds file item to CDB, and also adds file metadata as item's property
- After the DM data upload, MM experiment item views on the CDB web portal contain links to corresponding magnet, allow downloading experiment files, etc. (CDB work: D. Jarosz)

How Does Globus Fit In?

- Globus Connect Server: provides remote data access to APS on-site storage (via the aps#data endpoint)
- Globus MyProxy OAuth Server: handles aps#data endpoint authentication
- GridFTP servers and clients: used by DM software for internal data transfers between beamline (local) storage and APS on-site storage, for transfers between local storage and HPC resources, etc.
 - Issue: Support for Open Source Globus Toolkit ended in early 2018



Software Installation

- Each beamline at APS has its own Data Management software installation visible to all beamline machines
- Deployment area contains DM software, support software packages, beamline databases, configuration files, various runtime and log files
- All beamlines are fully independent of each other, which works well in the APS environment (beamline machines typically have different maintenance cycles)
- The DM software has fully scripted installation, upgrade, and deployment testing processes, which reduces to a minimum maintenance overhead due to independent beamline software installations
- The DM services typically run on a designated beamline server machine, which can be either virtual or physical (VMs are preferred)
- Services are controlled via a standard set of control scripts suitable for the RHEL operating system used at APS
- Typically, beamline staff has no involvement with DM software installation and maintenance

