



DPPS Milestones and Tasks

Doc. no.: -

2024-02-12

	First/Last Name, Organization, Role	Digital signature
--	-------------------------------------	-------------------

Table of Contents

0.1	Notes:	3
0.1.1	BDMS issues	3
0.1.2	Release 0.x	4
0.1.3	How we will deploy in the future	4
0.1.4	Deliverables and Releases	4
0.1.5	How to make a release	5
0.1.6	Blocking Points and risks:	6
0.1.7	Major issue:	6
0.2	ACTIONS	6
0.2.1	TODO Ask DESY for CI runner or VM that allows docker-in-docker (Frederic)	6
0.2.2	TODO Look into deploying singularity images on CVMFS (there is a docker plugin)	6
0.2.3	TODO Start Ops ReqSpec (Nektar)	6
0.2.4	TODO Consider common software core for Benchmarking/Release Verification/Quality Monitoring	6
0.2.5	TODO Ask Roberta about new milestone: DPPS provides public performance plots	6
0.2.6	TODO (karl) Update R0.1 UC MR	6
0.2.7	TODO create R0.0 UC	6
0.2.8	TODO (Frederic) Generate minutes from the PIC demo to help with merging BDMS	7
0.2.9	TODO determine DPPS database architecture	7
0.3	DPPS AIV Use Cases/Releases/Milestones	7
0.3.1	R0.0: Systems Integration Test on Test Cluster BDMS:WORKLOAD	7
0.3.2	BDMS Ingest and Retrieve functionality integrated with WMSBDMS:WORKLOAD	8
0.3.3	R0.1 Integrate Pipelines with WorkloadBDMS:WORKLOAD:DATAPIPE:CALIBPIPE:SIMPIPE	8
0.3.4	Run Workflows on Test cluster	9
0.3.5	Test replication rules (within test cluster)	9
0.3.6	Run a pipeline workflow WORKLOAD:BDMS:DATAPIPE:CALIBPIPE	9
0.3.7	PDR ALL:MANAGEMENT	9
0.3.8	Ingest legacy simulation data into BDMS (prod6 only) BDMS	9
0.3.9	Integrate DPPS with Off-site ICT (1+ DCs) ALL	9
0.3.10	Run Workflows on Off-site ICT	9

0.3.11	R1.0: IRF Maker (analyze Prod6)	BDMS:WORKLOAD:DATAPIPE	9
0.3.12	SM: Produce or replicate Public IRFs	ALL	9
0.3.13	Test user rights management	BDMS:AAI	9
0.3.14	Finalize subsystem architecture	ALL	10
0.3.15	Integrate LIDAR Input	DATAPIPE:CALIBPIPE	10
0.3.16	Integrate FRAM Input	DATAPIPE:CALIBPIPE	10
0.3.17	Process and apply Pointing Corrections	DATAPIPE:CALIBPIPE	10
0.3.18	Integrate DPPS with On-site ICT and	ACADA:BDMS:WORKLOAD:OPS:ACADA	10
0.3.19	Store LST-1+ACADA data	BDMS	10
0.3.20	Run Workflows on On-site ICT		10
0.3.21	R2.0: Mono-Telescope Processing	WORKLOAD:BDMS	10
0.3.22	CDR	ALL	10
0.3.23	DL1 Calibration (LST)	DATAPIPE:CALIBPIPE	11
0.3.24	SM: LST-1 Commissioning		11
0.3.25	SM: Open Science Data Challenge		11
0.3.26	DL1 Calibration (MST)	DATAPIPE:CALIBPIPE	11
0.3.27	Generate Data Quality Reports	DATAPIPE:CALIBPIPE:QUALPIPE	11
0.3.28	Preserve data (full replication policy)	BDMS	11
0.3.29	Use Separate temporary and archive storage policies	BDMS	11
0.3.30	R3.0: Stereo-Telescope Processing	ALL	11
0.3.31	Implement DPPS-SUSS interface (DL3)	BDMS:DATAPIPE	11
0.3.32	SM: Northern 2-3 Telescope Commissioning and public data		11
0.3.33	Ensure upgrade of BDMS/WMS maintains file catalog and data	BDMS:WMS:OPS	11
0.3.34	Generate Simulation Prod (e.g. Prod7)	BDMS:WORKLOAD:SIMPPIPE	11
0.3.35	Upgrade DPPS in-place at all data centers	BDMS:WORKLOAD:OPS	11
0.3.36	SM: PROD7		11
0.3.37	Generate tailored IRFs for an observation	SIMPPIPE:DATAPIPE:CALIBPIPE:DATAPIPE	11
0.3.38	DL1 Calibration (SST)		11
0.3.39	Generate Cat-C data products		11

0.1 Notes:

0.1.1 BDMS issues

- Context Metadata validation is NOT responsibility of BDMS
 - except for metadata that is archive-specific

- except for basic checks like “duplicated entry”, “completeness”
- Metadata schema is also NOT responsibility of BDMS
 - core schema comes from Data Model (Reference)
- **DECISION:** for BDMS-Workload plugin, go for the alternate implementation where Rucio is used for both storage types (ephemeral vs archived), but with two different policies. This corresponds to the right diagram in the presentation by Karl.
 - Reduced need to have two DMS systems and databases operating at the same time
 - Same technology used for both types of storage, reduced maintenance.
 - Can to start with even use only one policy, if having two is complicated.
 - Need a working prototypes.
- Metadata Catalog
 - Current prototype uses MongoDB. Could we use PostGRESql with a JSON column instead, just to reduce number of DB types? Needs benchmarking **verification test**.

0.1.2 Release 0.x

- Goal for integration: a docker compose setup we can run locally with all of DPPS installed.
- Currently DESY’s Kubernetes runner doesn’t support Docker-in-Docker, e.g. to run the DPPS container inside a CI job. However, we could simply create our own runner for those jobs.

0.1.2.1 Rel 0.1

0.1.3 How we will deploy in the future

- want common way for all data centers, everything containerized
- central monitoring portal/dashboard (part of Ops)
- requirement to be able to stop/restart
- requirement to be able to run multiple instances for upgrade testing
- requirement to be able to fall back to old version in case of upgrade failure
- would like something that is tested in the same way as real deployment eventually.
- Blocker now is that DIRAC isn’t fully containerized, but that is coming soon. Already working mostly in the Dirac-9.0 pre-release (transition between DIRAC 8 and DIRAC-X)

0.1.4 Deliverables and Releases

- We are not responsible for actual deployment of the software, only to define the procedure. Deployment to on- and off-site ICT should be done by a SDMC staff.

0.1.5 How to make a release

- Need release manager (temporary for r0)

Plan:

- Create release milestone on GitLab
- All issues associated with that milestone must be solved, or rescope.
- Test reports from each subsystem
 - what UCs have been tested
 - what requirements have been verified
- Test reports from AIV
- Tagged software releases
- Documentation Package (if not in tagged software release)
- the AIV repo should be the central point to release DPPS docker containers. It contains a list of which versions belong to this release.
- SBOM (Software bill of materials)

Pre-release Documentation:

- DPPS Release plan (defines deliverables and milestones)
- Rx.x Detailed Release Plan
- V&V plan

Release Deliverables

- Requirements Specs
- Requirements V&V
- Set of Tested use cases
- AIV report
 - QA reports (sonarqube)
 - Use Case Test reports
- DPPS user and developer documentation package
- List of released code packages and versions (machine-readable)
- Links to tagged source code in GitLab
-

0.1.6 Blocking Points and risks:

0.1.6.1 Quality Plan from CTAO

0.1.6.2 Event type definitions / cases (if needed for SDC)

0.1.7 Major issue:

- This is a very ambitious schedule
- requires everyone to work a high amount
- our teams have not yet even signed an MoU, and are still in the formation phase in some cases
 - Unclear how much manpower we have *currently*, only on a 5-year scale
 - BDMS team in particular, but also others, even DataPipe.

0.2 ACTIONS

0.2.1 **TODO** Ask DESY for CI runner or VM that allows docker-in-docker (Frederic)

0.2.2 **TODO** Look into deploying singularity images on CVMFS (there is a docker plugin)

0.2.3 **TODO** Start Ops ReqSpec (Nektar)

based on what we are developing for AIV + whatever extra is needed

0.2.4 **TODO** Consider common software core for Benchmarking/Release Verification/Quality Monitoring

All are comparing some baseline to measurements and then putting a cut on how far away is acceptable.

- See Tomas's list of requirements
- core for plotting/histogram I/O, etc.

0.2.5 **TODO** Ask Roberta about new milestone: DPPS provides public performance plots

0.2.6 **TODO** (karl) Update R0.1 UC MR

0.2.7 **TODO** create R0.0 UC

0.2.8 **TODO** (Frederic) Generate minutes from the PIC demo to help with merging BDMS

<https://indico.cta-observatory.org/event/5383/>

0.2.9 **TODO** determine DPPS database architecture

- who runs the pipelines database servers? Should be in Ops, but delegated to data centers? Or do we put them into SOSS (Felix Stoehr's suggestion)?
- From a Computing architecture standpoint, the same question should be applied to SOSS
- What about the databases needed for underlying technologies like Rucio/DIRAC/FTS/...?
- Are there two classes of database server: ones where the tables are critical and have to be backed up, and one where failures can be tolerated?
- For our databases where we control the schemas: which technology? -> Try to use Post-GRE for all
- Which technologies are needed by the underlying services? Mysql, mongodb, ...

0.3 **DPPS AIV Use Cases/Releases/Milestones**

SM: Science Milestone RX.Y: DPPS release

0.3.1 **R0.0: Systems Integration Test on Test Cluster BDMS:WORKLOAD**

0.3.1.1 **Task: Merge prototypes (there can only be one BDMS in the release)**

- Pick features of each to incorporate into released package
- Create repos in GitLab:
 - DPPS/BDMS/bdms-client: contains the code for ingest/retrieve/etc wrappers and *rucio* + related (fts, gfal, xrootd) client setups
 - DPPS/BDMS/bdms-server: contains BDMS Dockerfiles based on Rucio and FTS
- Move any code in the prototypes (that we want to retain) into these repos.
- Upgrade client software to use python project template:
 - should be pip-installable
 - have a container built by the CI,
 - have documentation (even minimal) uploaded to gitlab pages (following **python project template**)
- This is the repo where the release will be tagged and from which the container is generated.
- In the test setup we should have storage elements using the same versions as in the real data centers (e.g. dCache and xrootd).

0.3.1.2 Task: Set up CI environment

- Set up namespaces for CI for manual and automated tests (no more “Swiss” vs “Italian”)
- Ensure any manual setup is in the YAML file and is therefore reproducible
- check required ports

0.3.1.3 Ingest data products (without metadata)

BDMS

0.3.1.4 Retrieve data products by LFN

BDMS

- check that checksum matches original file

0.3.1.5 Run a “hello world” job in DIRAC (JDL)

WORKLOAD

- no data access, no Transformation or Production. Simple JDL job
- check for job completeness
- tests configuration and execution of a single-job workflow

0.3.2 BDMS Ingest and Retrieve functionality integrated with WMSBDMS:WORKLOAD

0.3.2.1 Run a job that needs input from BDMS (by LFN) and outputs to BDMS

check that the job completes and the output is stored in BDMS correctly.

0.3.3 R0.1 Integrate Pipelines with WorkloadBDMS:WORKLOAD:DATAPIPE:CALIBPIPE:S

0.3.3.1 Install Pipeline Containers on Test Cluster

- Build Docker images and convert to Singularity/Apptainer
- Base image should have all the core dependencies
- Deploy singularity containers to CVMFS

0.3.3.2 Deploy configuration files to CVMFS

0.3.3.3 Run test cases locally (not with WMS):

- CalibPipe: run CWL workflow for fetching contemporary MDP
- DataPipe: fetch a test simulation file from BDMS and run CWL workflow on WMS to process it to DL2a
- SimPipe: test and document simulation config model.
- Integration: Validate MDP produced by CalibPipe with SimPipe

0.3.3.4	Run tests on WMS	
0.3.4	Run Workflows on Test cluster	
0.3.5	Test replication rules (within test cluster)	
0.3.6	Run a pipeline workflow	WORKLOAD:BDMS:DATAPIPE:CALIBPIPE
0.3.7	PDR	ALL:MANAGEMENT
0.3.8	Ingest legacy simulation data into BDMS (prod6 only)	BDMS
	• need metadata catalog	
0.3.9	Integrate DPPS with Off-site ICT (1+ DCs)	ALL
0.3.9.1	Monitor DPPS operation	OPS
0.3.9.2	Integrate with AAI	
0.3.9.3	Install BDMS client	
0.3.9.4	Install Workload client	
0.3.9.5	Verify ingestion/retrieval/query	
0.3.9.6	Verify replication	
0.3.9.7	Verify data processing (run workflows)	
0.3.10	Run Workflows on Off-site ICT	
0.3.11	R1.0: IRF Maker (analyze Prod6)	BDMS:WORKLOAD:DATAPIPE
0.3.11.1	Test: Process Prod6 into IRFs	
	1. on test cluster (small sample test)	
	2. or off-site ICT for full verification? Needed for public IRFs	
	3. or on existing DIRAC instance... (not a DPPS milestone)	
0.3.12	SM: Produce or replicate Public IRFs	ALL
0.3.13	Test user rights management	BDMS:AAI
0.3.13.1	Check data rights:	
	• visibility (is this needed?)	
	• read access	

- write access
- delete access
- create datasets
- modify datasets

0.3.13.2 Check rights for workflows

- define new workflows
- configure existing workflows
- remove a workflow
- execute a workflow

0.3.14	Finalize subsystem architecture	ALL
0.3.15	Integrate LIDAR Input	DATAPIPE:CALIBPIPE
0.3.16	Integrate FRAM Input	DATAPIPE:CALIBPIPE
0.3.17	Process and apply Pointing Corrections	DATAPIPE:CALIBPIPE
0.3.18	Integrate DPPS with On-site ICT and ACADA	BDMS:WORKLOAD:OPS:ACADA
0.3.18.1	Implement ACADA-DPPS interface	BDMS:DATAPIPE
0.3.18.2	Install BDMS client	
0.3.18.3	Install Workload client	
0.3.18.4	Install DPPS ACADA Interface	
0.3.18.5	Verify ingestion from ACADA	
0.3.18.6	Verify replication to off-site (if offsite already integrated)	
0.3.18.7	Verify data processing (run workflows)	
0.3.19	Store LST-1+ACADA data	BDMS
No metadata, LFN only, replicate to one offsite data center		
0.3.20	Run Workflows on On-site ICT	
0.3.21	R2.0: Mono-Telescope Processing	WORKLOAD:BDMS
0.3.22	CDR	ALL

0.3.23	DL1 Calibration (LST)	DATAPIPE:CALIBPIPE
0.3.24	SM: LST-1 Commissioning	
0.3.25	SM: Open Science Data Challenge	
0.3.26	DL1 Calibration (MST)	DATAPIPE:CALIBPIPE
0.3.27	Generate Data Quality Reports	DATAPIPE:CALIBPIPE:QUALPIPE
0.3.28	Preserve data (full replication policy)	BDMS
0.3.29	Use Separate temporary and archive storage policies	BDMS
0.3.30	R3.0: Stereo-Telescope Processing	ALL
0.3.31	Implement DPPS-SUSS interface (DL3)	BDMS:DATAPIPE
0.3.32	SM: Northern 2-3 Telescope Commissioning and public data	
	LST1 + LST4 + [MST3]	
0.3.33	Ensure upgrade of BDMS/WMS maintains file catalog and data	
	BDMS:WMS:OPS	
0.3.34	Generate Simulation Prod (e.g. Prod7)	BDMS:WORKLOAD:SIMPIPE
0.3.35	Upgrade DPPS in-place at all data centers	BDMS:WORKLOAD:OPS
0.3.35.1	Test rollback to old version	
0.3.36	SM: PROD7	
0.3.37	Generate tailored IRFs for an observation	SIMPIPE:DATAPIPE:CALIBPIPE:DATAP
0.3.37.1	Re-simulate tailored sims?	
0.3.37.2	Re-process IRFs	
0.3.37.3	Re-process Events	
0.3.38	DL1 Calibration (SST)	
0.3.39	Generate Cat-C data products	
	High Precision, tailored sims,	
