Review: HSF Reference Implementation Conditions Database for BelleII
3. Migration timeline and milestones

Ruslan Mashinistov, John S. De Stefano Jr, Michel Villanueva

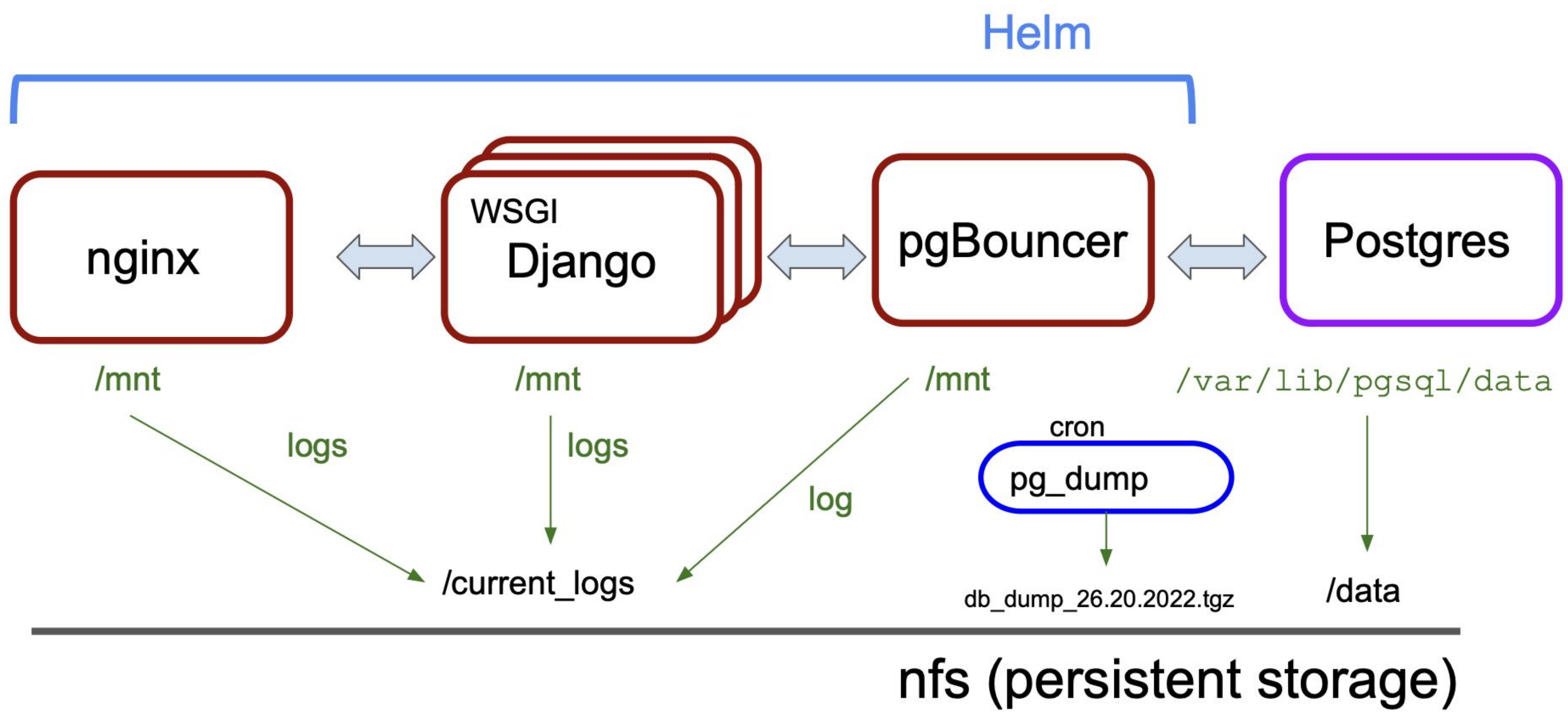
11 July 2024

@BrookhavenLab

Required development efforts

- Migration of Belle II to the HSF solution requires:
  - Infrastructure and Deployment update
  - Server Development work
  - Client Development work

Deployment on OKD (OpenShift)



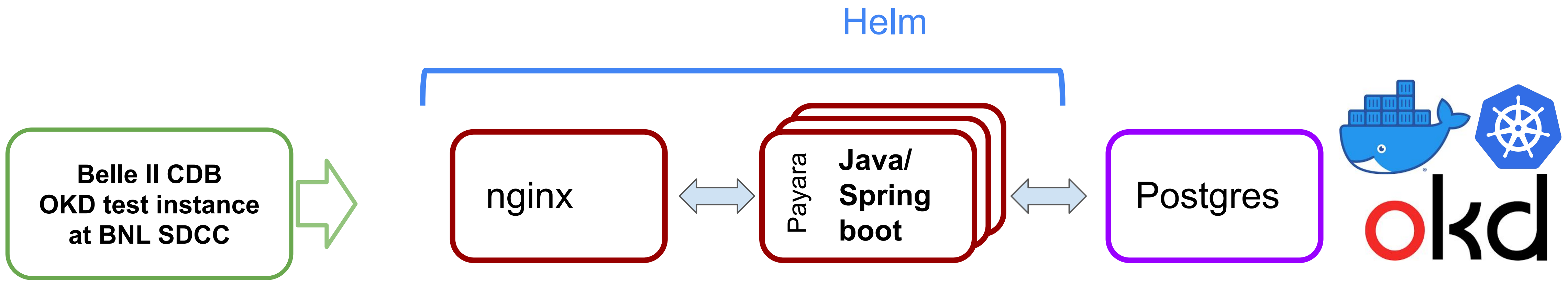
- Helm streamlines the deployment of Kubernetes clusters
- Classic deployment at VMs also possible and has been tested
- all-in-a-single-container image available

- Automated deployment on OKD (OpenShift) using Helm chart
- Horizontally scalable
- Open Source only

Easily adoptable for various HEP experiments

Belle II migration to OKD/OpenShift

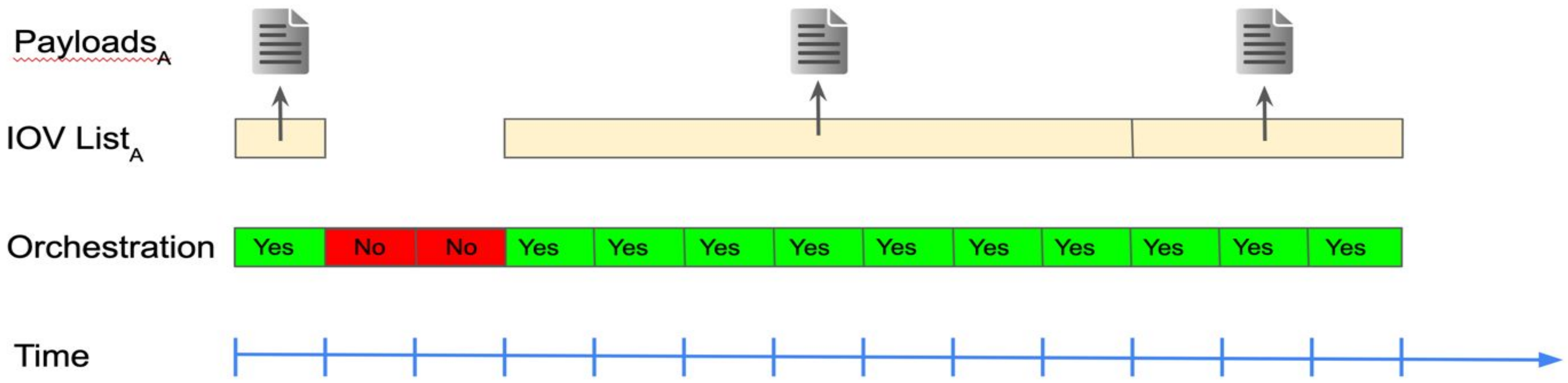
- Due to issues with our existing Kubernetes infrastructure, we have initiated a migration to OKD/OpenShift
  - We are adapting the HSF Helm deployment configuration to support our current Java application
    - We're also planning to include advanced DB pooler (pgBouncer) in the deployment
  - We have already successfully conducted a series of functional tests
  - This progress will significantly streamline the complete future migration



Conditions Data – Use Cases

- HSF Conditions Database meeting: **use cases** https://indico.cern.ch/event/1280790/
- Most can be realised w/ HSF Recomm.
- Most demanding use-case is **Fast-Processing**. Goal:
  - Publish data for analysis fast
  - Maximize physics performance

Use case	Example
Online	• High Level Trigger
Reprocessing	• Run reco w/ improved calib.
Analysis	• High level physics analysis
Development	• Test new calib. within existing GT
★ Fast-processing	• Process data w/ just-in-time calib.



All development work on a single slide

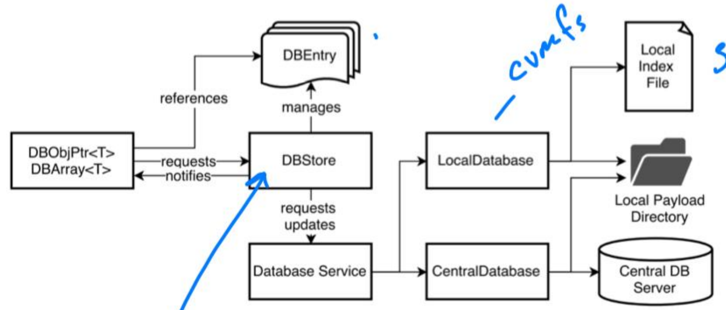


Figure 3. Relationships for the Conditions Database Interface. The user only interacts with the DBObjPtr and DBArray classes, everything else is handled transparently and can be configured independently.

Status ?

Versions ?

REST calls

CentralMetadataProvider.h

- updatePayload()

getEntry()

DBStore ← Database.h ← MetadataProvider.h

Datastore

- find()

Payload provider.h

(Enable remote calls)

- getLocalFile()

- getRemoteFile()

Search for files with https:

Downloader.cc

Will use the same ?

Configuration.cc

- Config for CDB conexion

Just a fragment of the outcome notes from the technical discussion with Belle II experts

Migrating data to the new DB schema

- **Data Migration**

We plan to migrate the existing data to a new schema, ensuring that all critical information is transferred without loss.

- **JWT Authentication/Authorization**

We will migrate the same JWT authentication scheme we currently use.

- The Django application possesses a custom method to verify JWTs
- We will create an additional, configurable, Belle II-specific function to implement the current authentication logic

The JWT for each authenticated user includes customized claims that define their permissions:

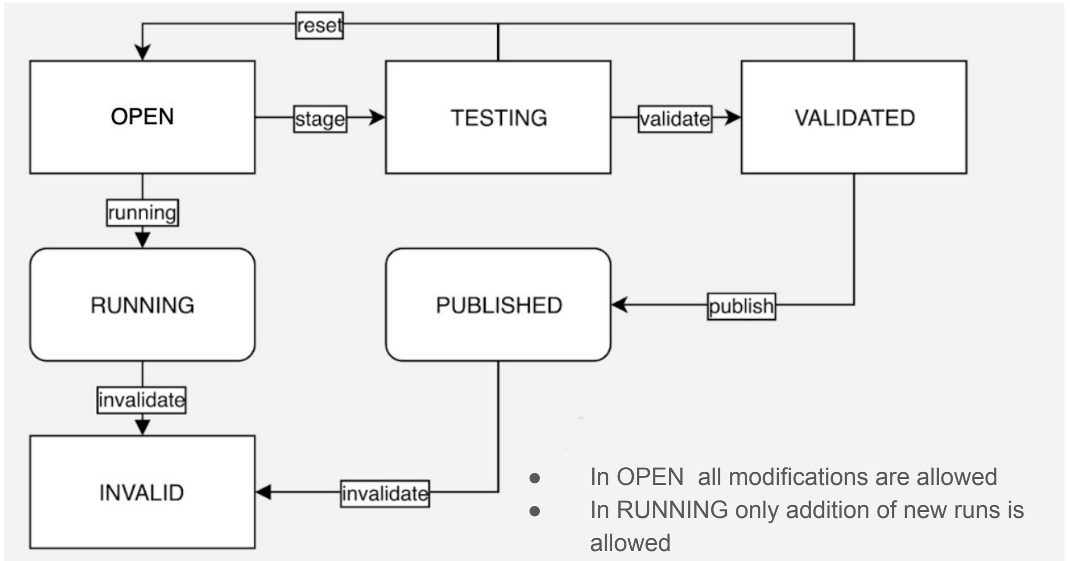
- `b2cdb:admin` for administrative roles on global tags
- `b2cdb:createiov` for manager roles on global tags
- `b2cdb:createpayload` for permissions related to payloads

Each of these claims is associated with a list of regular expressions:

- The first two claims pertain to global tag names
- The last claim pertains to payload names

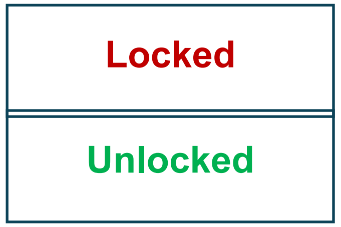
JWT only used for writing. Read APIs don't require JWT

Current CDB



- In OPEN all modifications are allowed
- In RUNNING only addition of new runs is allowed
- For all others no modifications are allowed
- Only RUNNING and PUBLISHED will be usable for users

HSF CDB

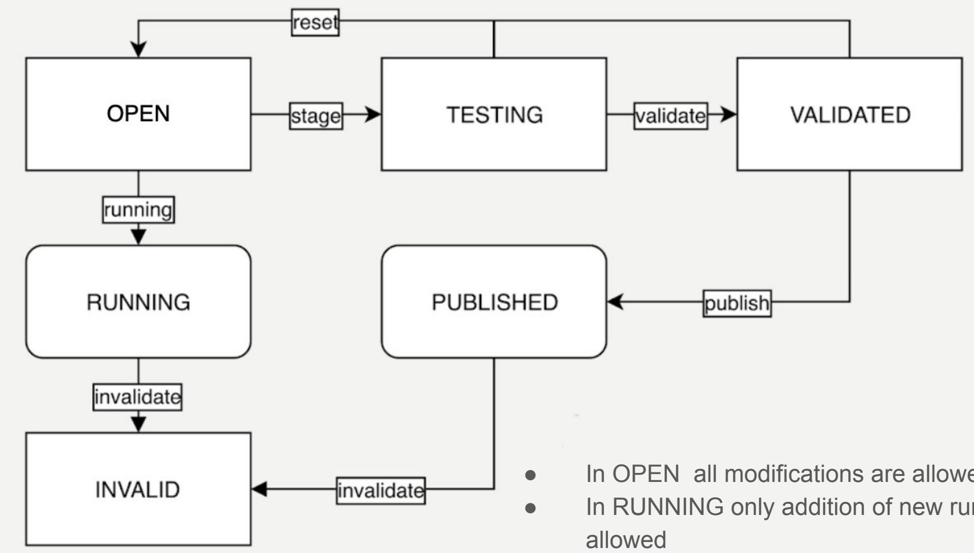


- In Unlocked all modifications are allowed
- In Locked only addition of new runs is allowed

Migrating data to the new DB schema

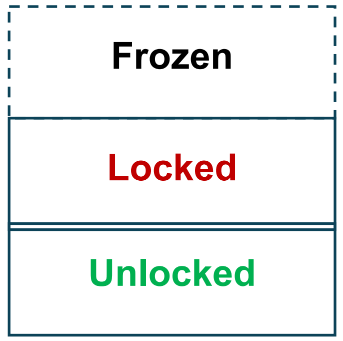
- A review of the current set of GT statuses with the software team confirmed that the current workflow is unnecessarily and overly complicated.
  - We are considering simplifying it by dropping some unneeded statuses. For example, the VALIDATED status is currently not in use
  - After consulting with the software team, we have decided to include an additional status:
    - **Frozen**: No modifications to the GlobalTag and its content are allowed.
- We will provide a custom Belle II-specific API to manage status changes, controlling the allowed transitions between statuses
  - This function will override the default experiment-agnostic one and can be activated in the configuration.
  - To store information about allowed transitions, this custom function will use the "Description" field in the GlobalTagStatus table, or a new optional field will be added.

Current CDB



- In OPEN all modifications are allowed
- In RUNNING only addition of new runs is allowed
- For all others no modifications are allowed
- Only RUNNING and PUBLISHED will be usable for users

HSF CDB



- In **Unlocked** all modifications are allowed
- In **Locked** only addition of new runs is allowed
- In **Frozen** no modifications are allowed

Client / Basf2 migration to HSF CDB

- Belle II uses Python client tools and C++ modules

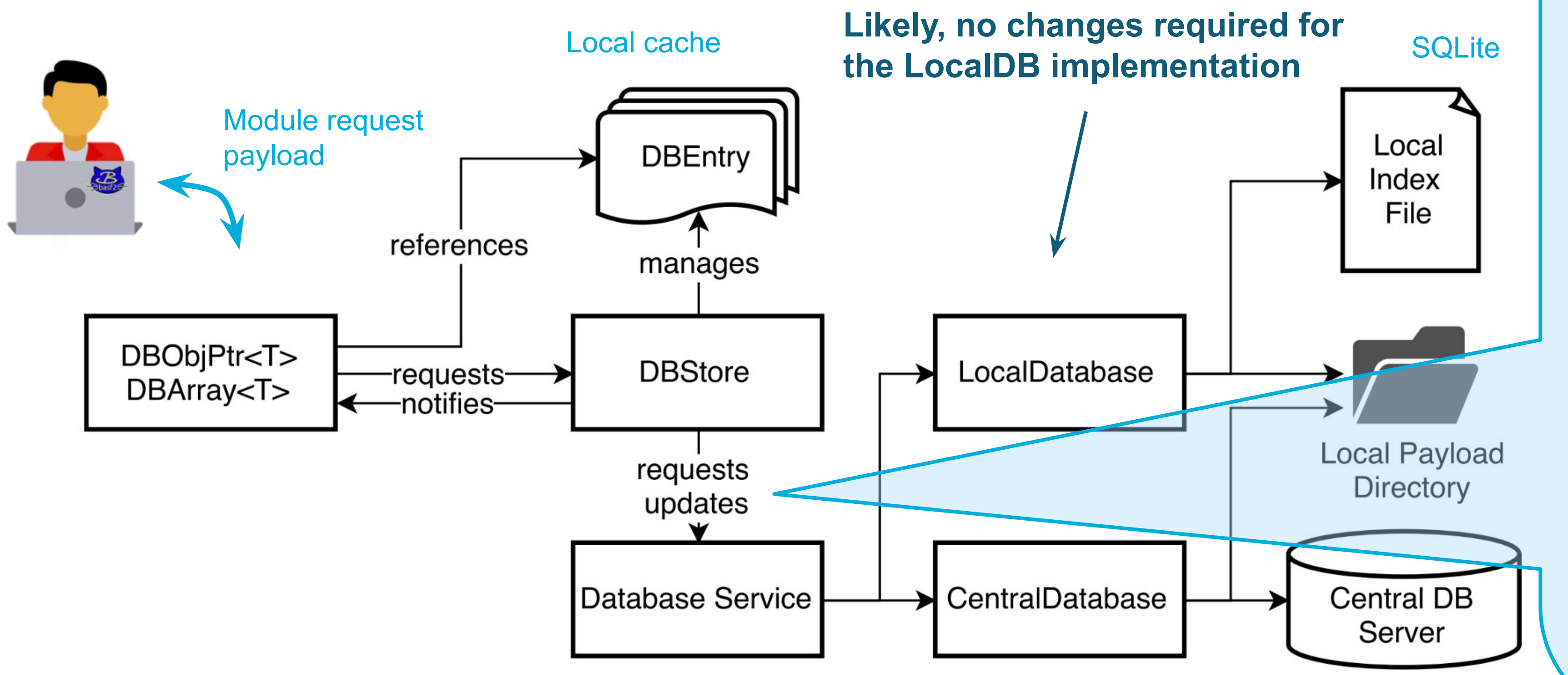


Figure: DOI 10.1088/1742-6596/1085/3/032032

1. **Database Service**:
   - Outcome of the discussion with DP and SW experts was to prepare a third metadata provider in basf2
   - We will have for a while **three providers inside basf2 in parallel**
     - CentralMetadata - current CDB
     - **NewCentralMetadata - HSF CDB**
     - LocalMetadata - SQLite on CVFMS
   - The new one will work for a while in read-only mode
   - Once the data taking stops in summer, we will switch to NewCentralMetadata for writing
2. **DBStore**: This component will likely need modifications to accommodate the changes in GT statuses when handling payload information.

Other development efforts

- **Versioning**

  The current CDB implementation includes the concept of a Payload version. Occasionally, Payload files are updated, and the existing implementation permits overlapping IOVs. In such cases, the client software resolves conflicts by taking the latest version.

  While the new HSF implementation prevents overlapping IOVs, we still want to track updates to the Payload files. To enhance human control, we have decided to use the "Description" field or add a new optional field in the PayloadIOV table to store version information.

- **Bulk update running global tag**

  HSF CDB supports the concept of a running GT. In this scenario, the IOV is open-ended, meaning the current payload is valid from the beginning of its IOV indefinitely. Even if the GT is locked and no modifications are allowed to already appended PayloadIOVs, users are still permitted to append new PayloadIOVs if the new IOV starts after the beginning of the last IOV and new is also open-ended.

  We have already provided an API to append a single PayloadIOV to the GT. However, we have agreed to implement a new bulk API, which will allow users to append a consecutive, seamless group of PayloadIOVs, provided the group starts after the beginning of the last appended IOV and the last IOV in the group is open-ended.

- **Python interface**

  A standalone, command-line client tool developed in Python facilitates various CDB tasks, such as uploading payloads and downloading GTs along with their corresponding PayloadIOVs. These tools interact with the CDB through its REST interface. Additionally, an auxiliary mechanism is provided for setting and retrieving CDB authentication tokens.

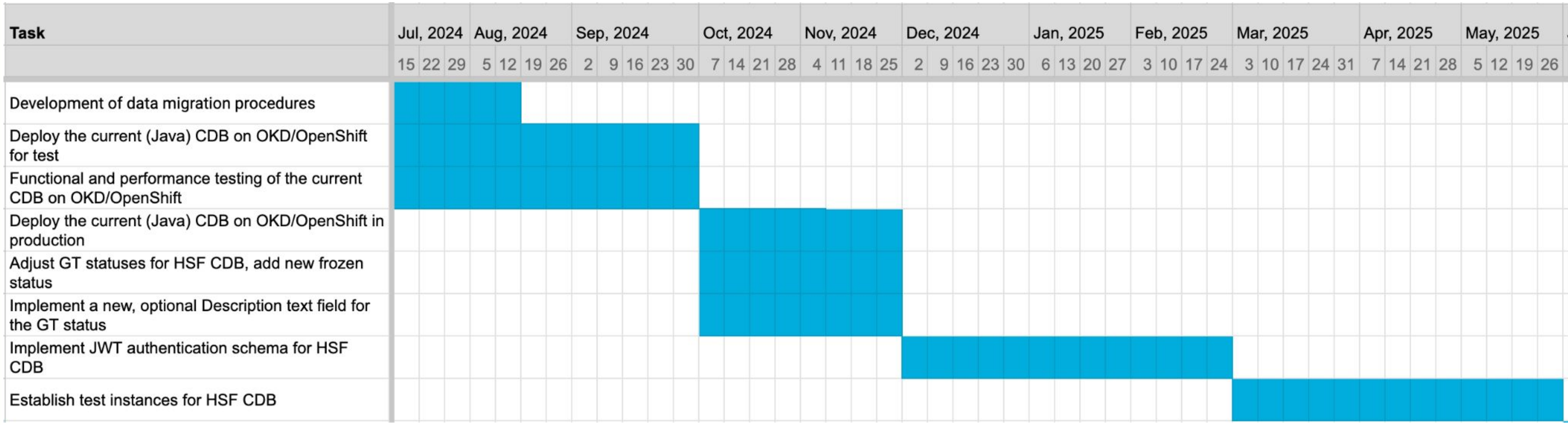
  Migrating to the HSF CDB will demand adapting the Python interface to handle the updated REST API queries and new endpoints. The authentication mechanisms will remain unchanged.

Timeline

	2024						2025												2026					
Task	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun
Development of data migration procedures	■	■	■	■	■	■																		
Deploy the current (Java) CDB on OKD/OpenShift for test	■	■	■																					
Functional and performance testing of the current CDB on OKD/OpenShift	■	■	■																					
Deploy the current (Java) CDB on OKD/OpenShift in production				■	■																			
Adjust GT statuses for HSF CDB, add new frozen status				■	■																			
Implement a new, optional Description text field for the GT status				■	■																			
Implement JWT authentication schema for HSF CDB						■	■	■																
Establish test instances for HSF CDB									■	■	■													
Development on basf2 client												■	■	■	■									
Combined testing of client with the new server												■	■	■	■									
Complete migration procedures and scripting																■	■							
Estimated completion of basf2 client adjustments																■	■							
Conduct comprehensive functional and performance testing																		■	■					
Adjust/reimplement GT admin tools																				■	■			
Open instance for validation testing by critical Belle II computing groups																				■	■			
Achieve expected production readiness and final migration																						■	■	■
Deploy the new HSF CDB on OKD/OpenShift in production																						■	■	■

Backup

Timelines



Task	Jul, 2024			Aug, 2024				Sep, 2024					Oct, 2024				Nov, 2024				Dec, 2024					Jan, 2025				Feb, 2025				Mar, 2025					Apr, 2025				May, 2025			
	15	22	29	5	12	19	26	2	9	16	23	30	7	14	21	28	4	11	18	25	2	9	16	23	30	6	13	20	27	3	10	17	24	3	10	17	24	31	7	14	21	28	5	12	19	26
Development of data migration procedures	■	■	■	■	■																																									
Deploy the current (Java) CDB on OKD/OpenShift for test	■	■	■	■	■	■	■	■	■	■	■	■																																		
Functional and performance testing of the current CDB on OKD/OpenShift	■	■	■	■	■	■	■	■	■	■	■	■																																		
Deploy the current (Java) CDB on OKD/OpenShift in production													■	■	■	■	■	■	■	■																										
Adjust GT statuses for HSF CDB, add new frozen status													■	■	■	■	■	■	■	■																										
Implement a new, optional Description text field for the GT status													■	■	■	■	■	■	■	■																										
Implement JWT authentication schema for HSF CDB																					■	■	■	■	■	■	■	■	■	■	■	■	■													
Establish test instances for HSF CDB																																		■	■	■	■	■	■	■	■	■	■	■	■	■

Timelines

Task	Jun, 2025					Jul, 2025				Aug, 2025				Sep, 2025					Oct, 2025				Nov, 2025			
	2	9	16	23	30	7	14	21	28	4	11	18	25	1	8	15	22	29	6	13	20	27	3	10	17	24
Development on basf2 client	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■								
Combined testing of client with the new server	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■								
Complete migration procedures and scripting																			■	■	■	■	■	■	■	■
Estimated completion of basf2 client adjustments																			■	■	■	■	■	■	■	■
Conduct comprehensive functional and performance testing																										
Adjust/reimplement GT admin tools																										
Open instance for validation testing by critical Belle II computing groups																										
Achieve expected production readiness and final migration																										
Deploy the new HSF CDB on OKD/OpenShift in production																										

Task	Dec, 2025					Jan, 2026				Feb, 2026				Mar, 2026					Apr, 2026				May, 2026			
	1	8	15	22	29	5	12	19	26	2	9	16	23	2	9	16	23	30	6	13	20	27	4	11	18	2
Development on basf2 client																										
Combined testing of client with the new server																										
Complete migration procedures and scripting																										
Estimated completion of basf2 client adjustments																										
Conduct comprehensive functional and performance testing	■	■	■	■	■	■	■	■	■																	
Adjust/reimplement GT admin tools										■	■	■	■	■	■	■	■	■								
Open instance for validation testing by critical Belle II computing groups										■	■	■	■	■	■	■	■	■								
Achieve expected production readiness and final migration																			■	■	■	■	■	■	■	■
Deploy the new HSF CDB on OKD/OpenShift in production																			■	■	■	■	■	■	■	■