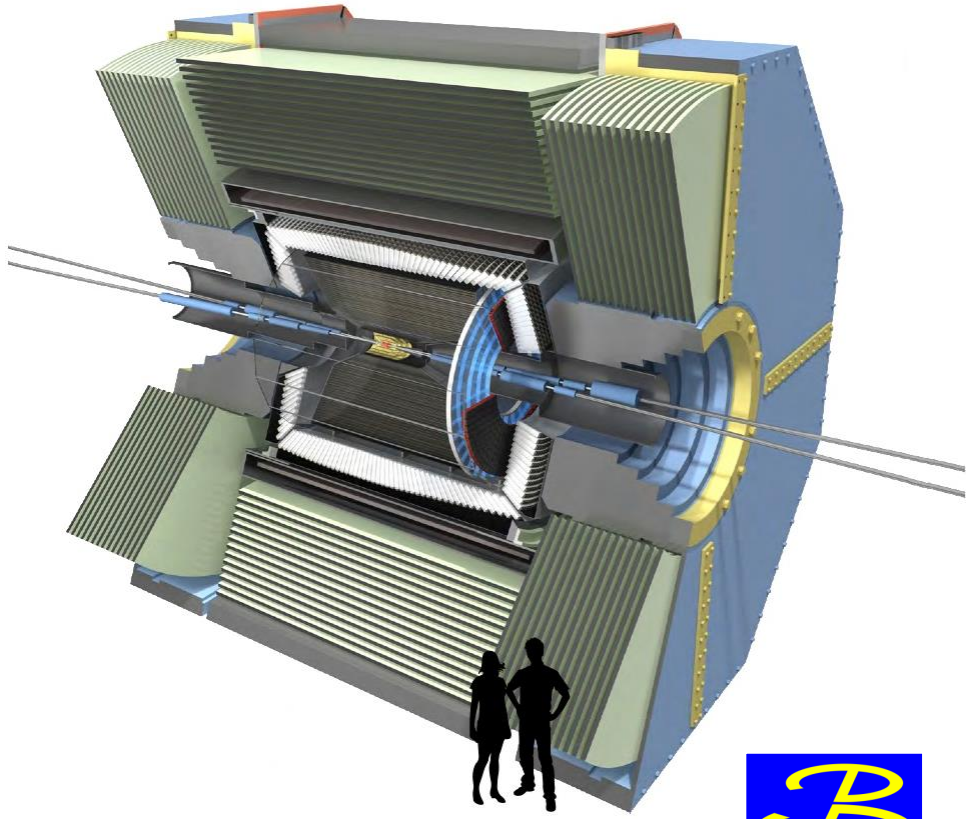
Performance of the Belle II Conditions Database

TODD ELSETHAGEN, KEVIN FOX, LYNN WOOD
(PACIFIC NORTHWEST NATIONAL LABORATORY)

MARKO BRACKO (INSTITUT JOŽEF STEFAN)

THOMAS KUHR, MARTIN RITTER
(LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN)

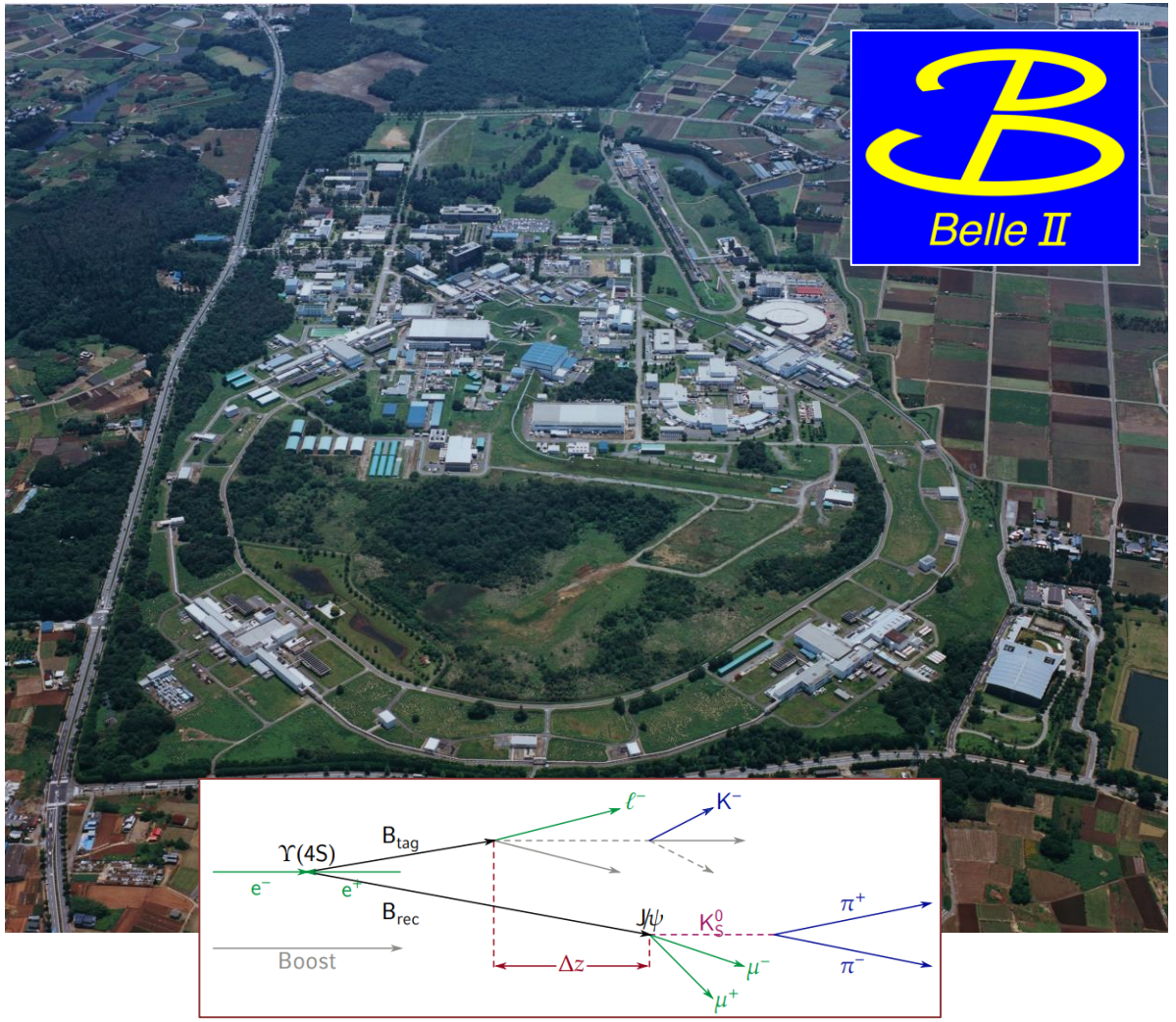
CARLOS FERNANDO GAMBOA (BROOKHAVEN NATIONAL LABORATORY)



23rd International Conference on Computing in High Energy and Nuclear Physics – CHEP 2018

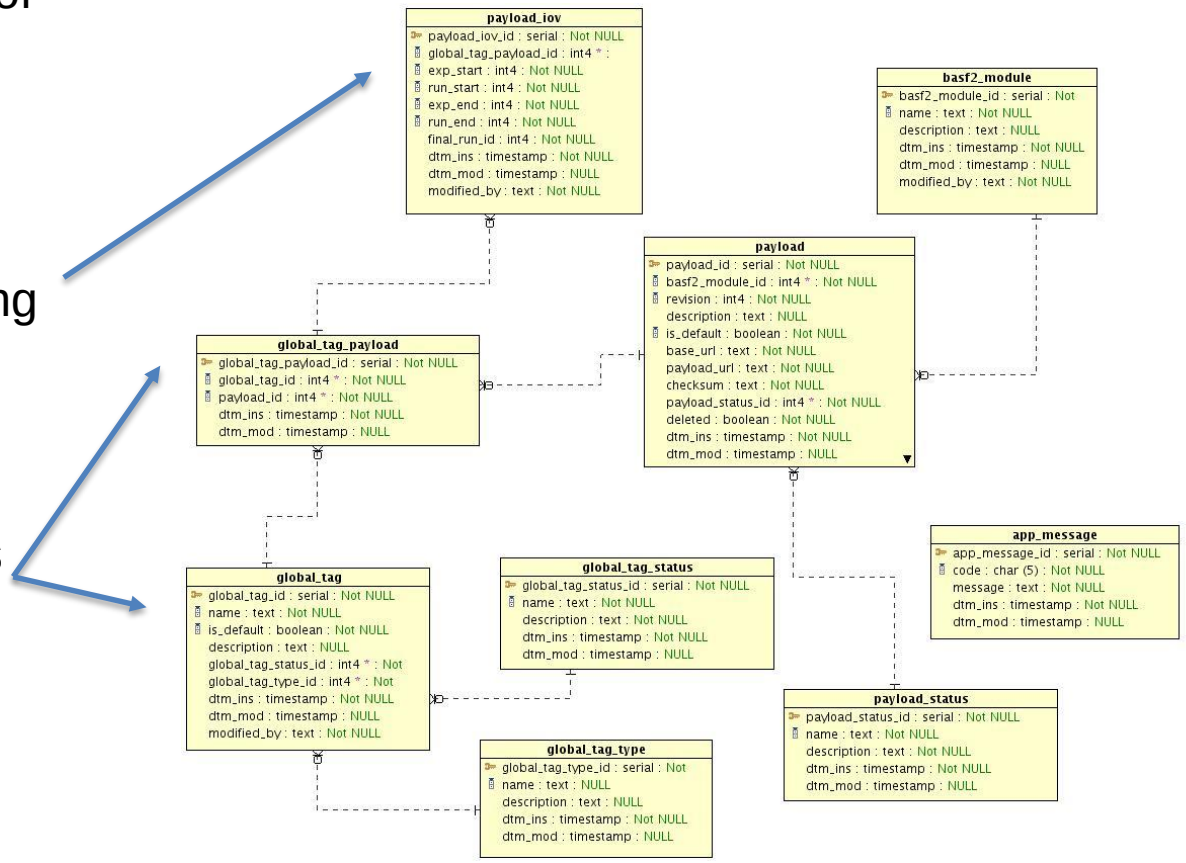
The Belle II Experiment at KEK Tsukuba, Japan

- The Belle II experiment is part of a broad-based search for **new physics** in the intensity frontier
  - precisely measuring and comparing with theory branching fractions, angular distributions, CP asymmetries, forward-backward asymmetries, and a host of other observables
- The SuperKEKB accelerator upgrade will provide 40x the luminosity of KEKB and 50x the data taken with Belle
- "Phase 2" data-taking started in **May 2018**, concurrent with beam-tuning efforts
- "Phase 3" full running starts in **2019**



Database Architecture – Concepts

- Conditions database holds **time-dependent status** of detectors for data processing/reprocessing
  - Constants or time-varying parameters: calibration, geometry/alignment, etc.
- **Intervals of Validity** (IOV) specify starting and ending experiments and runs for a given payload for that global tag
- **Global tags** contain list of IOV-payload relationships and are used to select a complete set of conditions for a given reprocessing effort
  - Different types (dev, release)
  - Multiple status types (new, published, invalid)

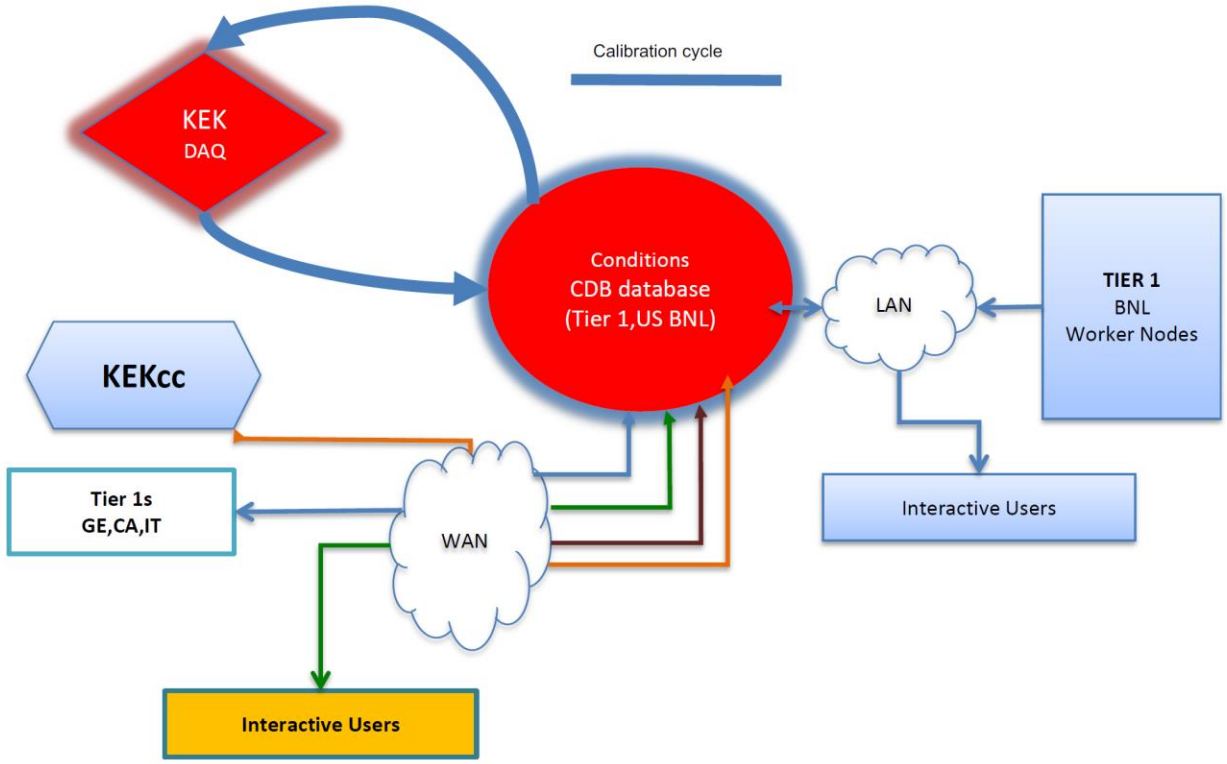


Database Architecture – Access

- Design separated into two components: database (metadata) and payload files
  - Each has different requirements for workflow and scalability

- Database has **REST API** – HTTP accesses
  - Ex: “iovPayloads?gtName=production&
    expNumber=3&runNumber=345”
  - Restricts access to allowed operations
  - **Easy to scale with industry standard (HTTP) tools**

- Payloads are kept as **external files** outside of the database
  - Database request returns partial URL of payload file
  - Avoids database bottleneck for large payload files
  - Allows for different scenarios for file storage and transfer



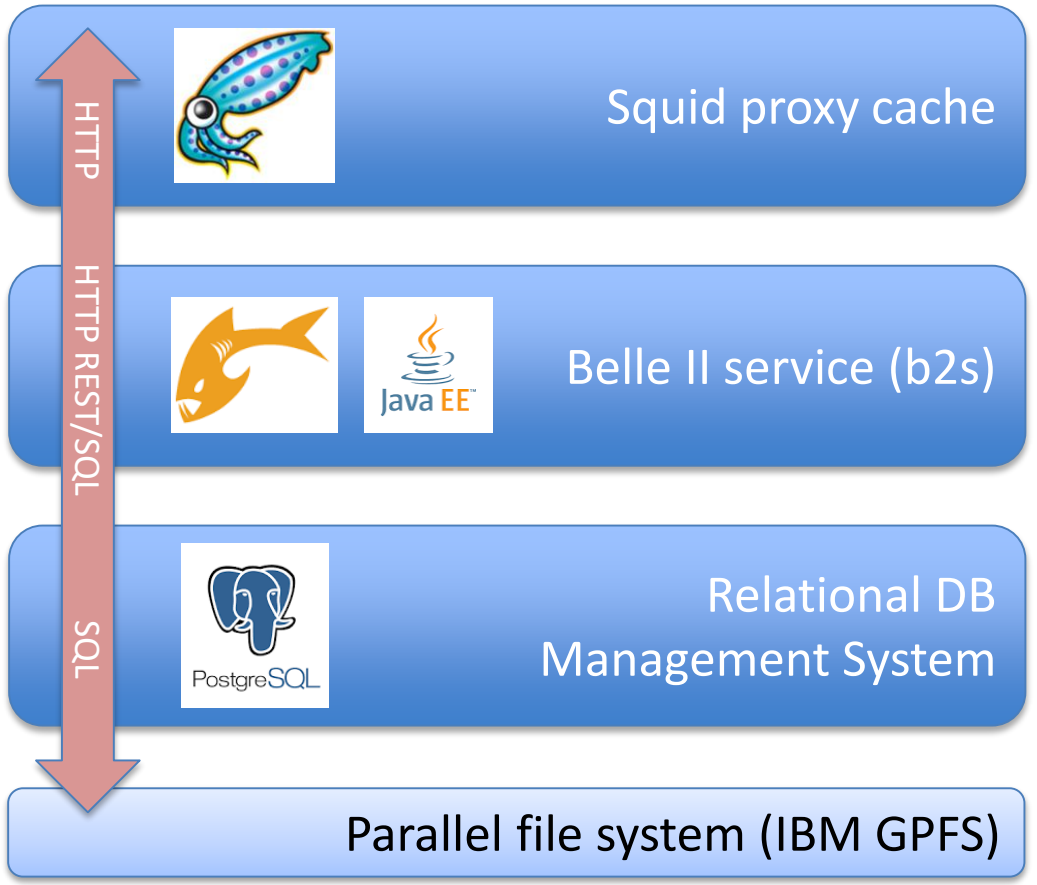
Database Architecture – Multiple Layers

- Required performance attained at multiple levels by relying on commercial tools designed for scalability

  - **Squid HTTP cache**
    - Configured as reverse proxy – many clients, few servers
    - Supports multiple requests for the same query
    - Caches the most common global tags in Belle II

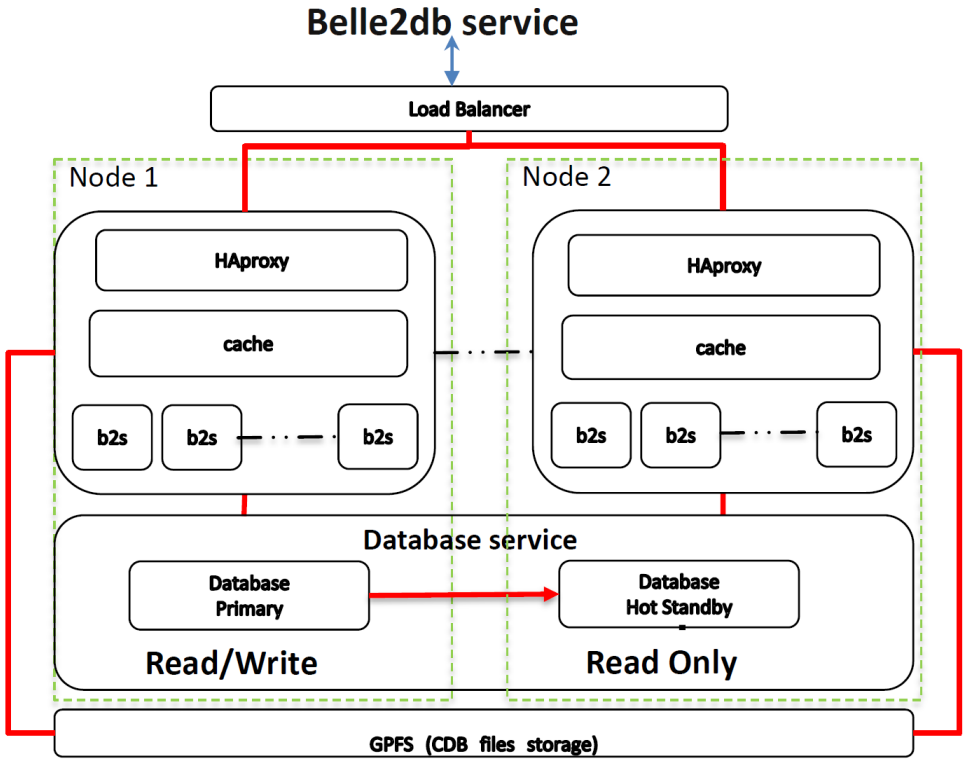
  - **“b2s” Belle II service layer**
    - **Payara**-based (JavaEE) server to translate REST requests into SQL queries
    - Caching via **Hazelcast** In-Memory Grid service
    - REST API built using **Swagger** tools

  - **PostgreSQL database**

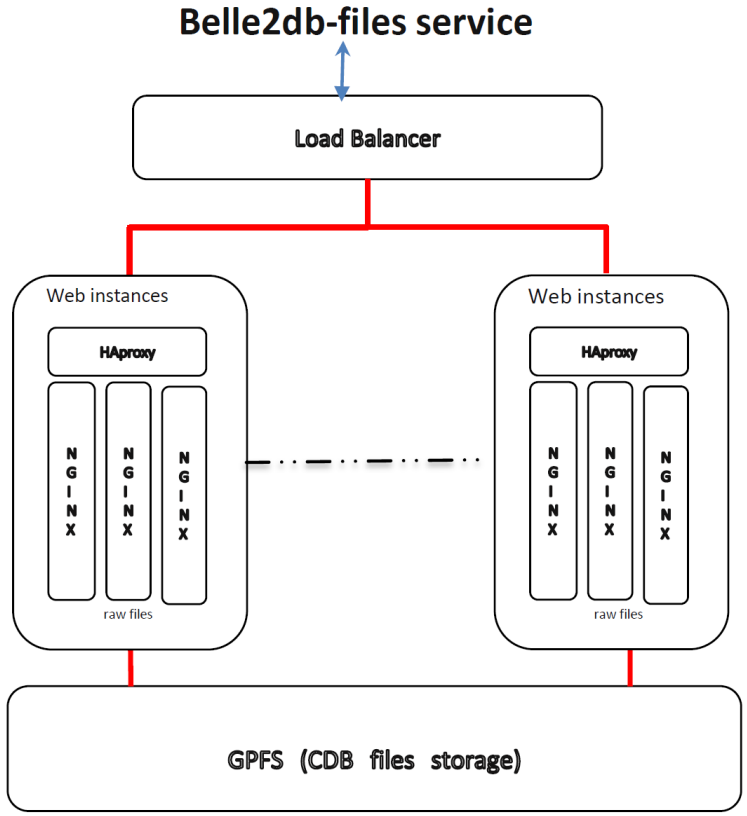


Database Architecture – Separate DB and Payload Servers

- Database server built with multiple instances of all components
  - Major components run as **docker** containers
  - Greater reliability and performance



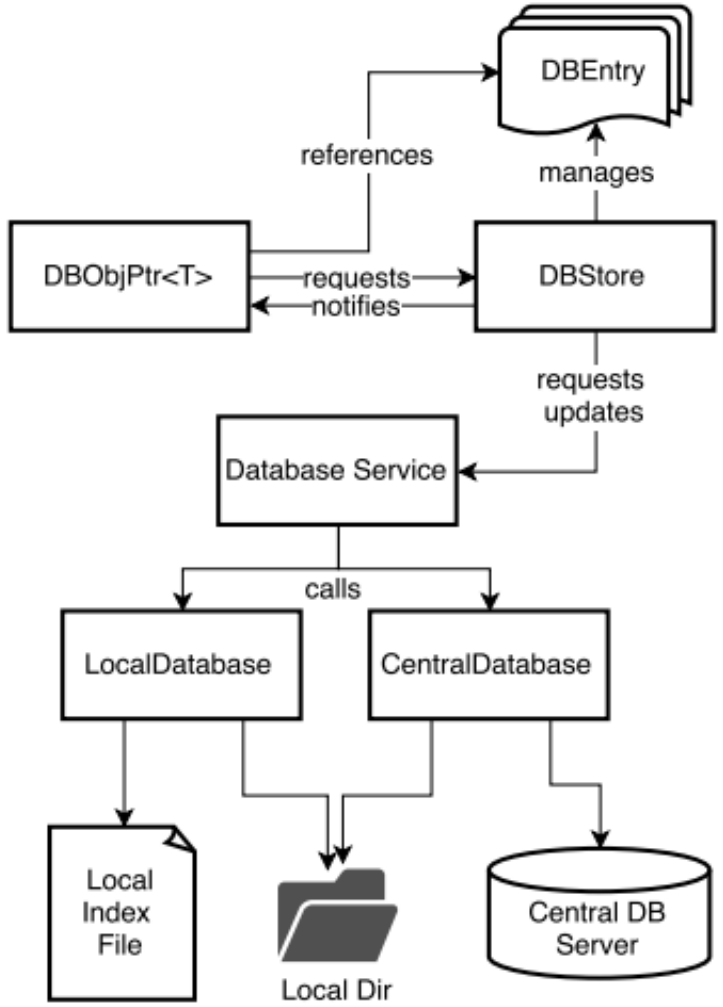
- Payload server consists of multiple nodes of high-performance NGINX HTTP servers



Client Interfaces

Basf2 Software Framework

- Run granularity; finer granularity handled on client side
- Payloads assumed to be ROOT objects (but DB doesn't care)
- "DBStore" manages storage and automatic updates of payloads
  - Template classes that always provide pointer to the correct current payload
- Transparent access to remote database or local files
- Configured via Python steering file
  - Can generate "chain" of locations to search (both remote and local)

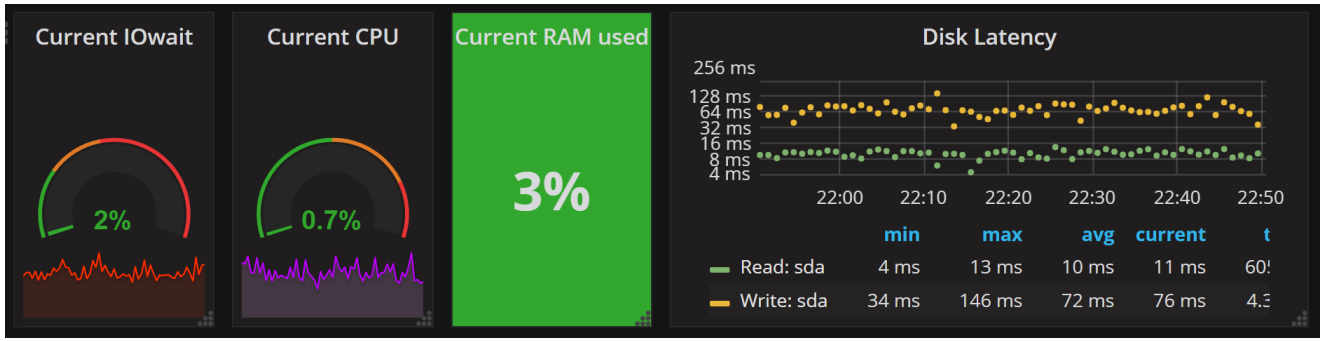


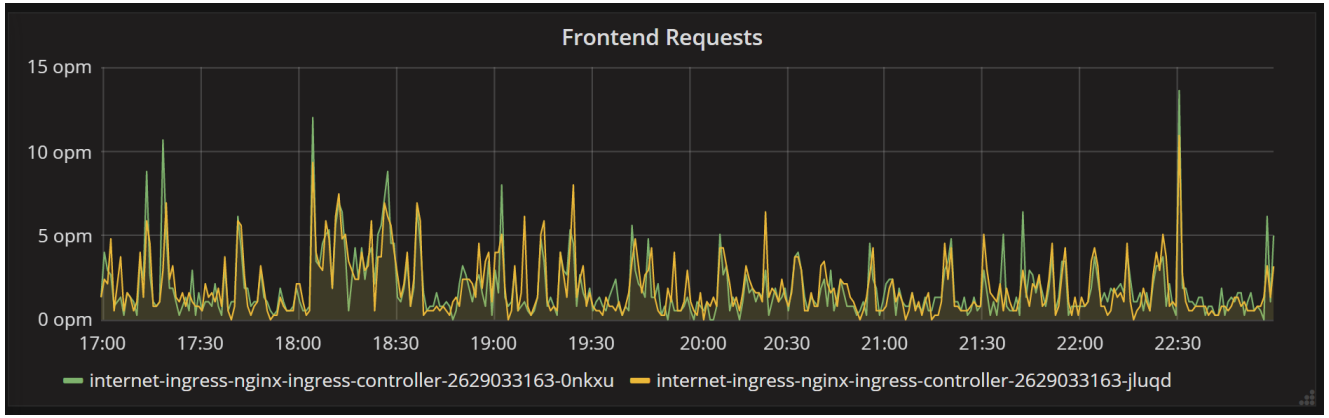
Other Interfaces

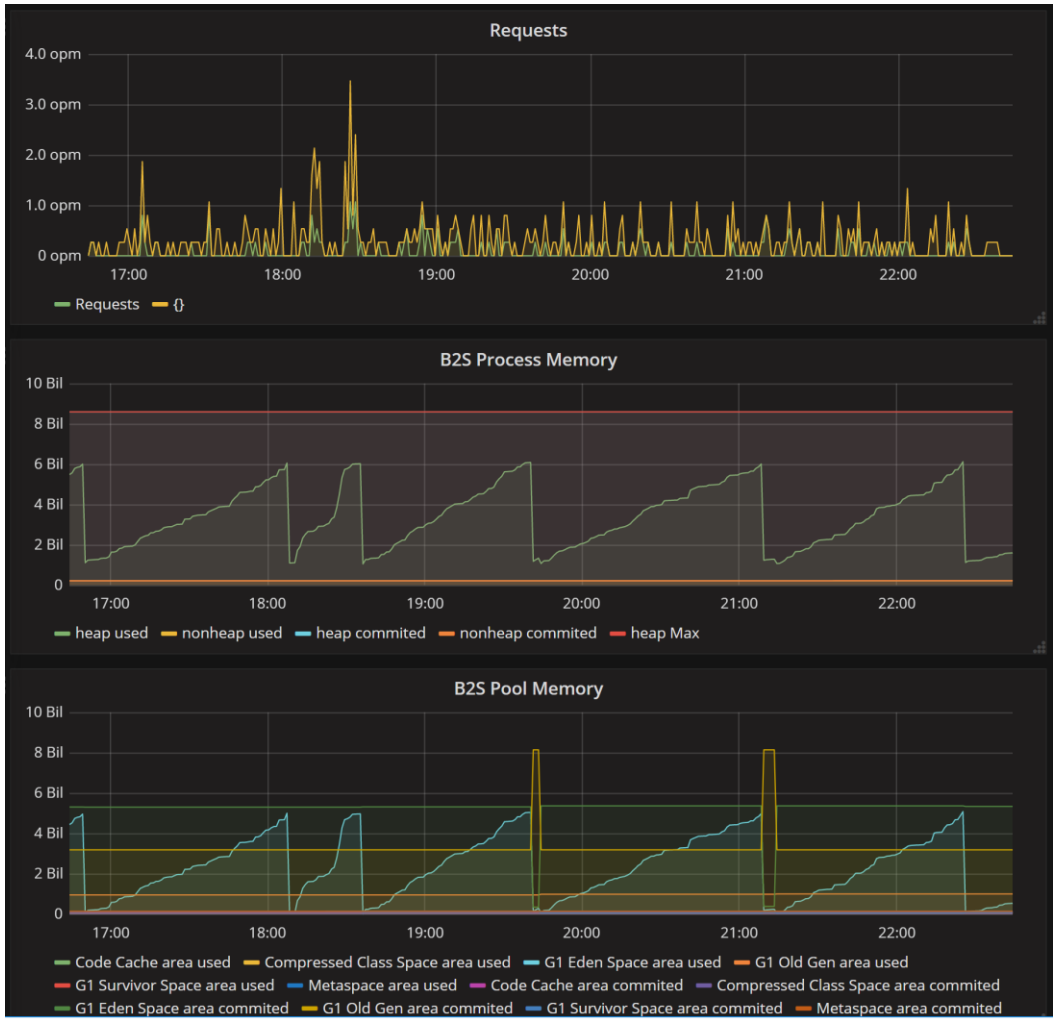
- Command-line client
- Swagger API web-based interface

Database Monitoring

- Extensive monitoring of all layers (HW, database, server, cache) are in place
  - Subset provided to all collaborators via web interface

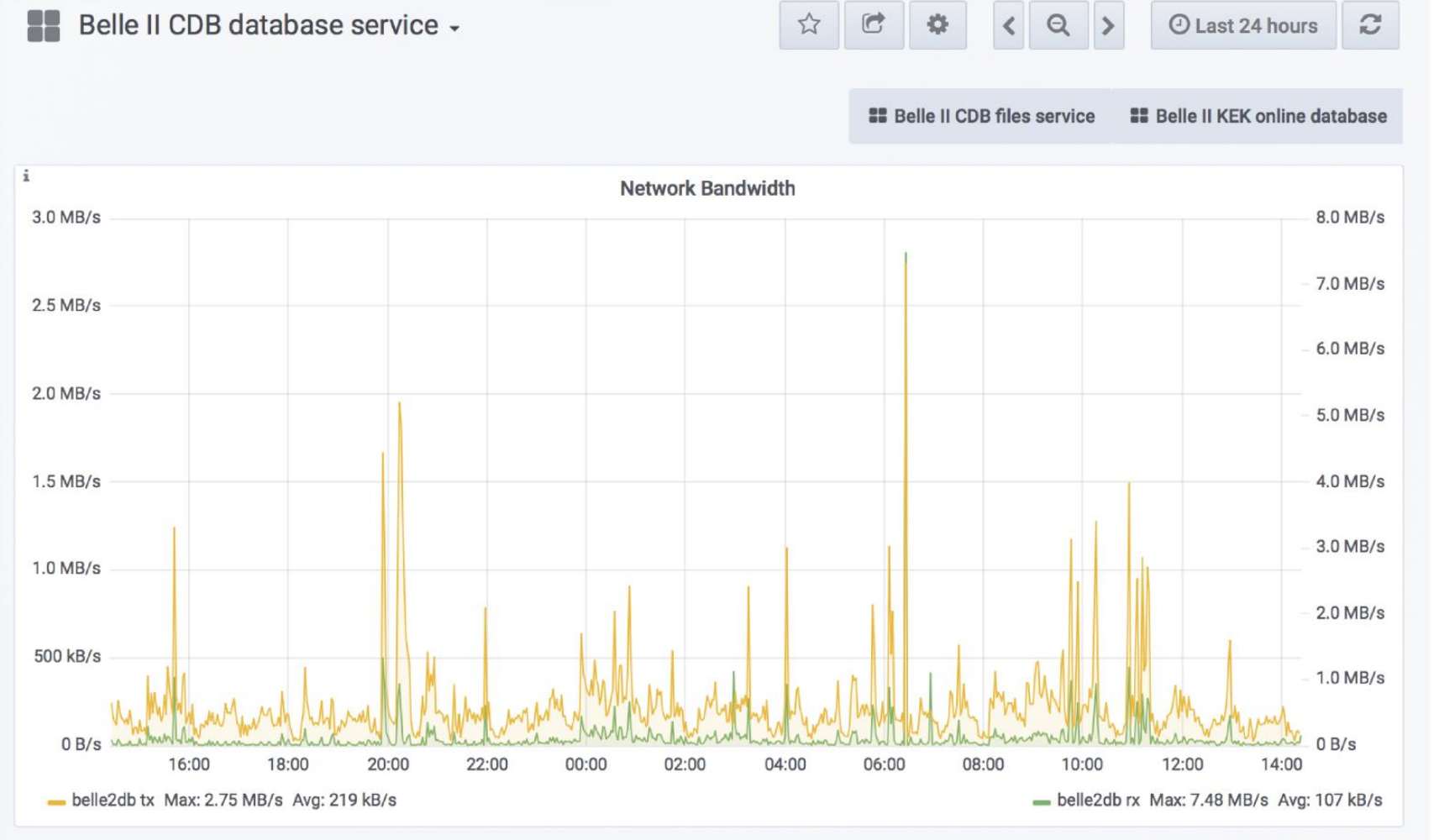






Database Performance – Monte Carlo and Data Reprocessing

- Current MC and Phase 2 data reprocessing do not put a significant load on the DB and payload file servers
- Current biggest stressor: users submitting large number of non-grid jobs
- Directed tests necessary to confirm that performance is sufficient for full running



Database Performance – Directed Tests

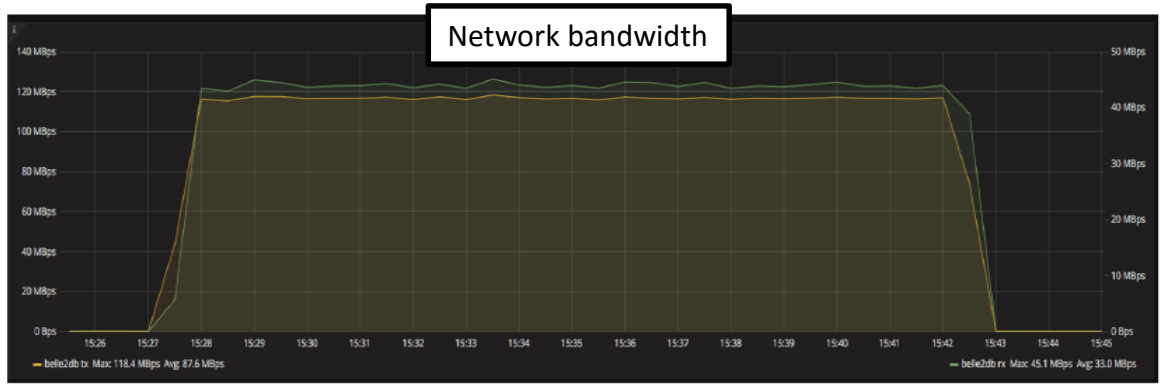
Tests of production workflow

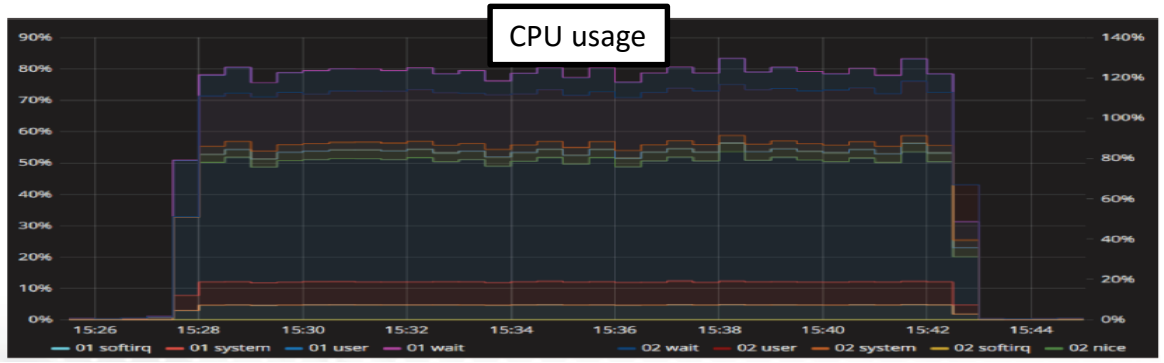
- Verify proper operation for expected production jobs
- Estimated rates for full production:
  ~100 req/sec continuous, higher bursts
- Both directed tests and MC job submission
  - Verify software releases
  - Specific job scripts for “problem” workflows
  - Confirm proper operation and database access
- Directed tests using Gatling HTTP performance testing tool

- No issues or increased workload seen

Performance during directed tests

- Example: Gatling configured to issue 400 req/second over a period of 15 minutes
  - Input fed from list of 21k different requests



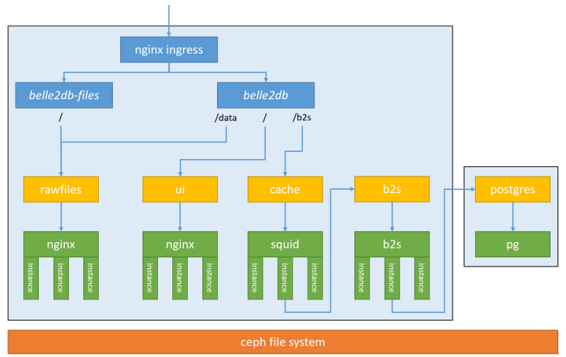


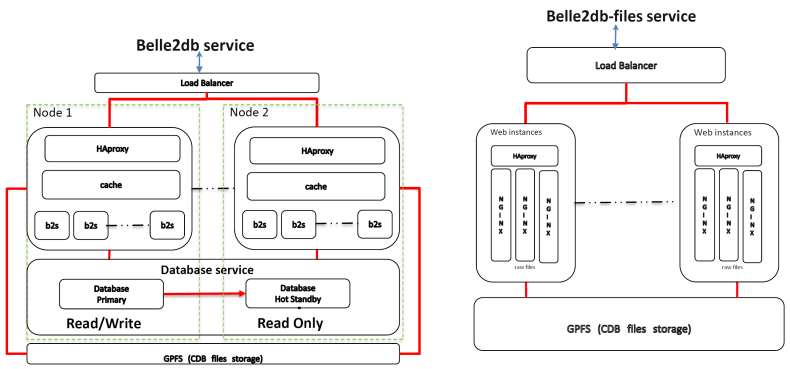
Database Migration from PNNL to BNL

- U.S. Department of Energy's Office of Science is aligning unique laboratory capabilities with project needs
  → **Belle II computing moving to BNL in 2018**

- This required **significant preparation** – not only did operation of the various services need to be transferred, but also **training** of staff for development efforts on distributed data management (DDM) and conditions DB
  - PNNL was primary developer for Belle II in both instances!

- Compute nodes moved in fall of 2017, DDM services and the conditions database in June 2018
  - Similar (but not identical) configurations at both sites
  - Careful planning performed to minimize down time
  - Payload files in cvmfs during transition
  - **Effectively no interruption to users!**





Upcoming Database Development

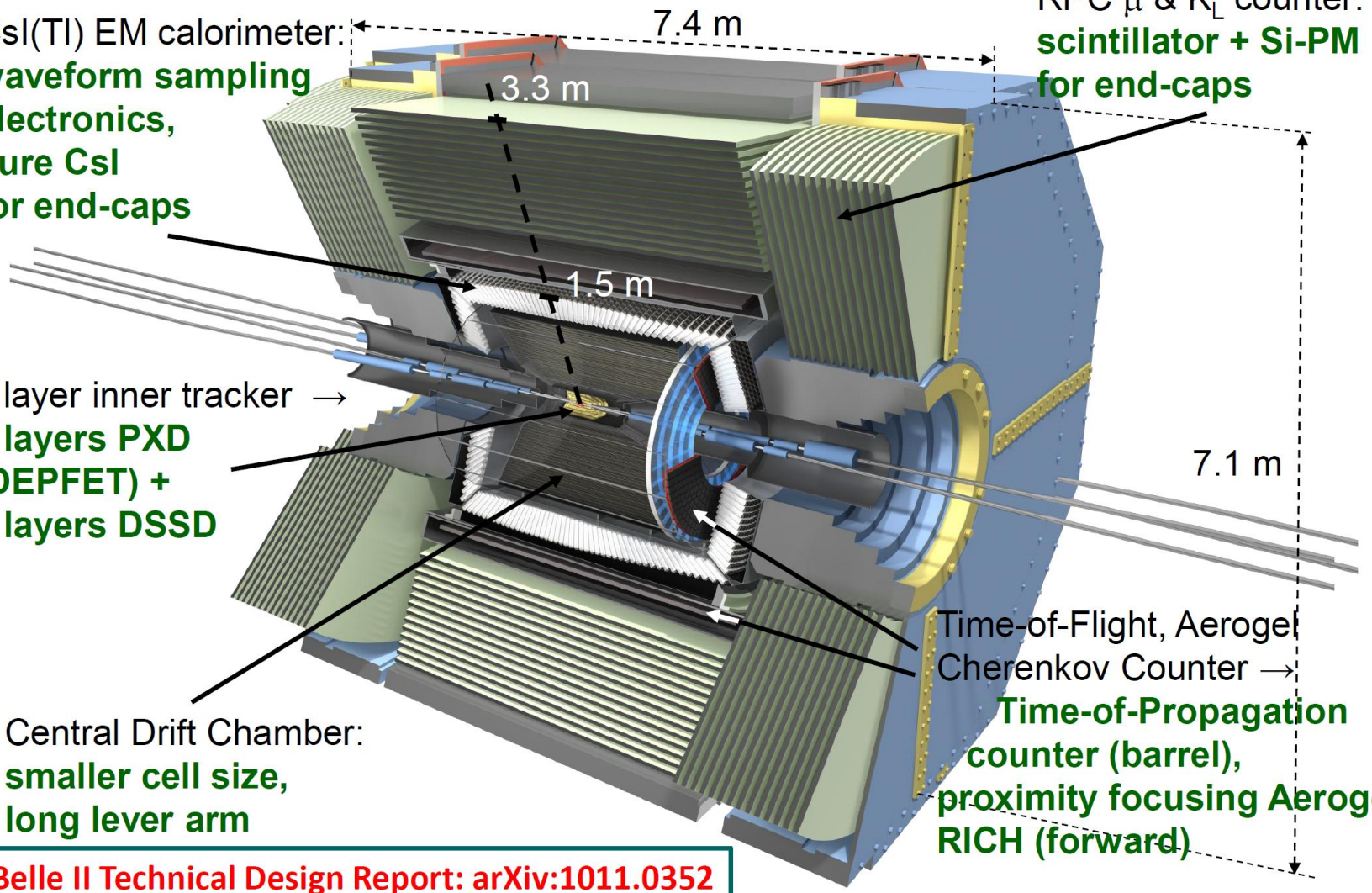
- **Authentication and Authorization**
  - Currently in development
  - Expectation is that reads will be open, but write access needs to be authenticated (who are you?) and authorized (you can't do that)
- Proposal: leverage existing **X.509** certificate authentication used for Belle II grid processing
  - Can add "roles" to X.509 certificates, ex.:
    - User (read-only)
    - Developer (can add data to open global tags)
    - Coordinator (can modify global tag status)
- **Site Replication**
  - Current performance is sufficient for Belle II, but single site is less reliable
  - Current backup of payloads on cvmfs – ***temporary!***
- PostgreSQL replication (streaming, log-based)
  - Built-in functionality
  - Issues: latency, hardware requirements
- More advanced tools?
  - OpenStack Trove, CockroachDB
  - Would require significant tool migration
- **Persistent remote squid caching**
  - Caches at remote sites w. long invalidation times

Conclusions

- **The Belle II Conditions Database was designed for ease of operation and support**
  - REST interface
  - Industry-standard tools for scalability and ease of operation
- **The database was successfully transitioned from PNNL to BNL in May 2018**
  - Careful planning resulted in no issues for users
  - Effectively zero downtime due to availability of backup payloads in cvmfs
- **Demonstrated performance well above current MC and Phase 2 data-processing needs; expected to handle full Belle II running as well**
  - Able to scale cleanly if higher performance needed

Extra Slides

The Belle II Detector



Belle II Technical Design Report: arXiv:1011.0352

Payara is an application server for production Java EE applications

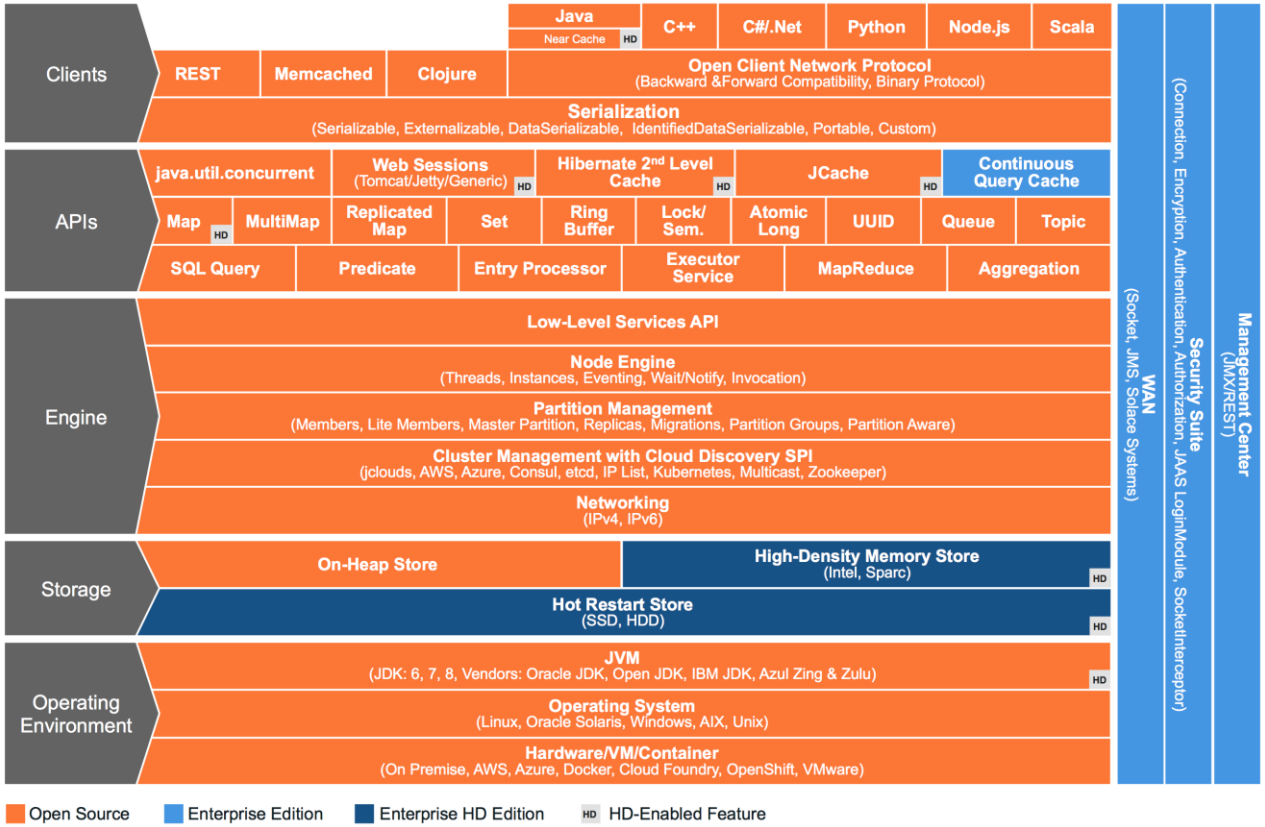
Payara Server is a drop in replacement for GlassFish Server Open Source Edition, with the peace of mind of quarterly releases containing enhancements, bug fixes and patches.

There is also Payara Micro, which enables you to run war files from the command line without any application server installation. It is small, < 70MB in size and incredibly simple to use. With its automatic and elastic clustering, Payara Micro is designed for running Java EE applications in a modern containerized/virtualized infrastructure, using automated provisioning tools like Chef, Ansible or Puppet.

Feature	GlassFish 4.x	Payara Server
License	Open Source	Open Source
Release frequency	Irregular	Quarterly
Releases in 2016	0	4
Patch releases	For versions < 4.0 only	• Monthly for support customers • Quarterly for community
Security Fixes	Infrequent	• Instant emergency & backported fixes for support customers • As soon as possible for community
Production Support	✗	✓
Developer Support	✗	✓
Component Upgrades (e.g. Tyrus, Mojarra)	Irregular	Quarterly
Supported IDEs	• Eclipse • Netbeans • Intellij IDEA	• Eclipse • Netbeans • Intellij IDEA
Caching tools	Shoal	• Shoal & JCache (open source) • Payara Scales (enterprise)
Automatic Clustering	✗	✓ via Hazelcast
Asadmin command recorder	✗	✓
Slow SQL Logging	✗	✓
Healthcheck Service	✗	✓
Request Tracing	✗	✓
Monitoring Logging	✗	✓
Microservices distribution	✗	✓ Payara Micro
Docker support	✓ Community provided	✓ Official images
IBM JDK Release	✗	✓ Payara Blue

The **Hazelcast** operational in-memory computing platform helps leading companies worldwide manage their data and distribute processing using in-memory storage and parallel execution for breakthrough application speed and scale.

Hazelcast is easy to work with and brings a highly resilient and elastic memory resource to all of your applications. At its core, Hazelcast is one of the most widely adopted open source solutions with tens of thousands of installed clusters and over 16 million server starts per month. On top of this popular open source platform, Hazelcast Enterprise HD and Hazelcast Enterprise offer licensed features for large scale deployments. Now you can free your data from slow, expensive, and hard to scale relational databases. With Hazelcast, your database remains the system of record, but bottlenecks disappear.



SWAGGER

Pacific Northwest NATIONAL LABORATORY

What is Swagger?

The goal of Swagger™ is to define a standard, language-agnostic interface to REST APIs which allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection. When properly defined via Swagger, a consumer can understand and interact with the remote service with a minimal amount of implementation logic. Similar to what interfaces have done for lower-level programming, Swagger removes the guesswork in calling the service.

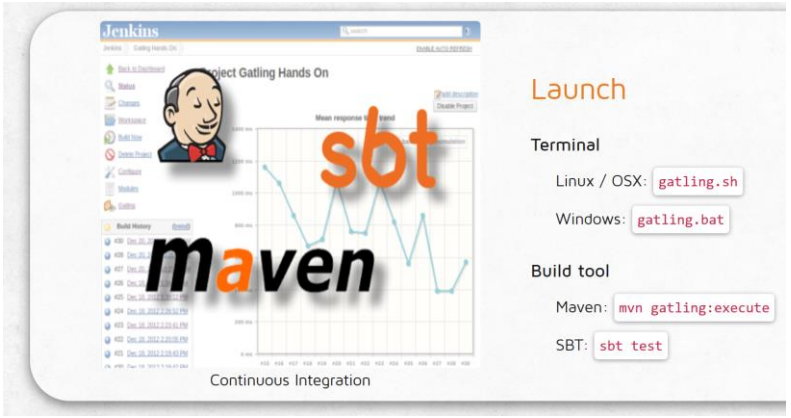
Technically speaking - Swagger is a formal specification surrounded by a large ecosystem of tools, which includes everything from front-end user interfaces, low-level code libraries and commercial API management solutions.

Gatling

Pacific Northwest NATIONAL LABORATORY

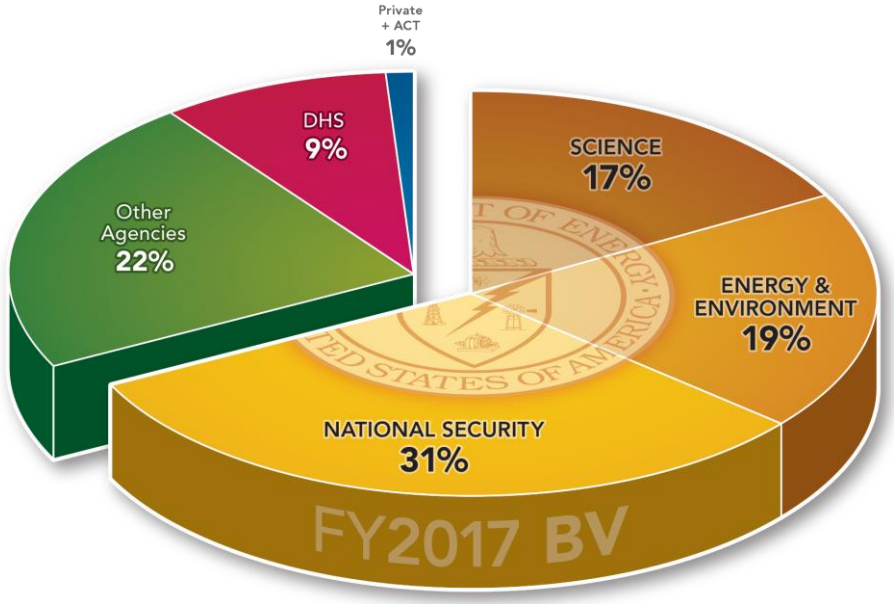
Open-source load testing framework based on Scala, Akka, and Netty

- High performance
- Ready-to-present HTML reports
- Scenario recorder and developer-friendly DSL



PNNL – FY2017 at a Glance

- $987M in R&D expenditures
- 4,486 scientists, engineers and non-technical staff
- 64 U.S. and foreign patents
- 7 R&D 100 Awards, 2 FLC Awards
- 1,127 peer-reviewed publications



- Mission-driven collaborations with government, academia and industry
- Among DOE's top-performing labs; a premier chemistry, environmental sciences and data analytics laboratory