iris hep
Institute for Research & Innovation in Software for High Energy Physics

HSF
HEP Software Foundation

U.S. DEPARTMENT OF ENERGY

Brookhaven™ National Laboratory

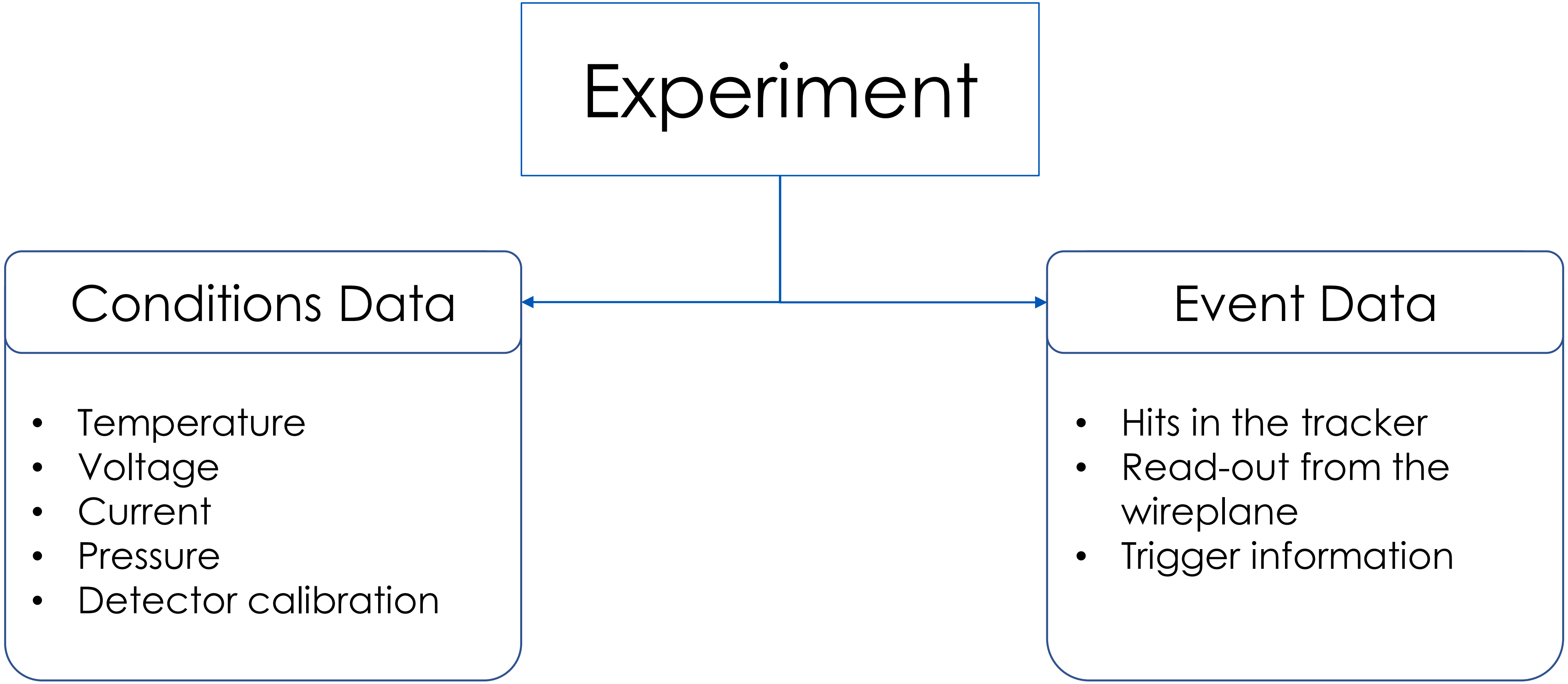
The HSF Conditions Database: Intelligent Caching

Candidate: Ernest Sorokun

Mentor: Lino Gerlach

Conditions Data

“Conditions data is any additional data needed to process event data”



Experiment

Conditions Data

- Temperature
- Voltage
- Current
- Pressure
- Detector calibration

Event Data

- Hits in the tracker
- Read-out from the wireplane
- Trigger information

Conditions Data: Challenges

Changes over time	Heterogenous data	High access rates
• Repeat detector calibration with larger cosmic dataset • Improve calibration algorithms	• Granularity varies (time indexed, run-indexed, constant) • Structure of payload varies (3D map, time-indexed values, single number, …)	• Distributed computing jobs access same conditions data simultaneously • Access rates up to ~kHz
Versioning & configuration	Payload agnostic by design	Fast DB queries & effective caching

Similar challenges for various HEP experiments

HSF Conditions Database

- Discussions across various experiments
- Creation of a uniform conditions database system for various HEP experiments
- Key recommendations for conditions data handling
  - Loose coupling between client and server using RESTful interfaces
  - The ability to cache queries as well as payloads
  - Separation of payload queries from metadata queries

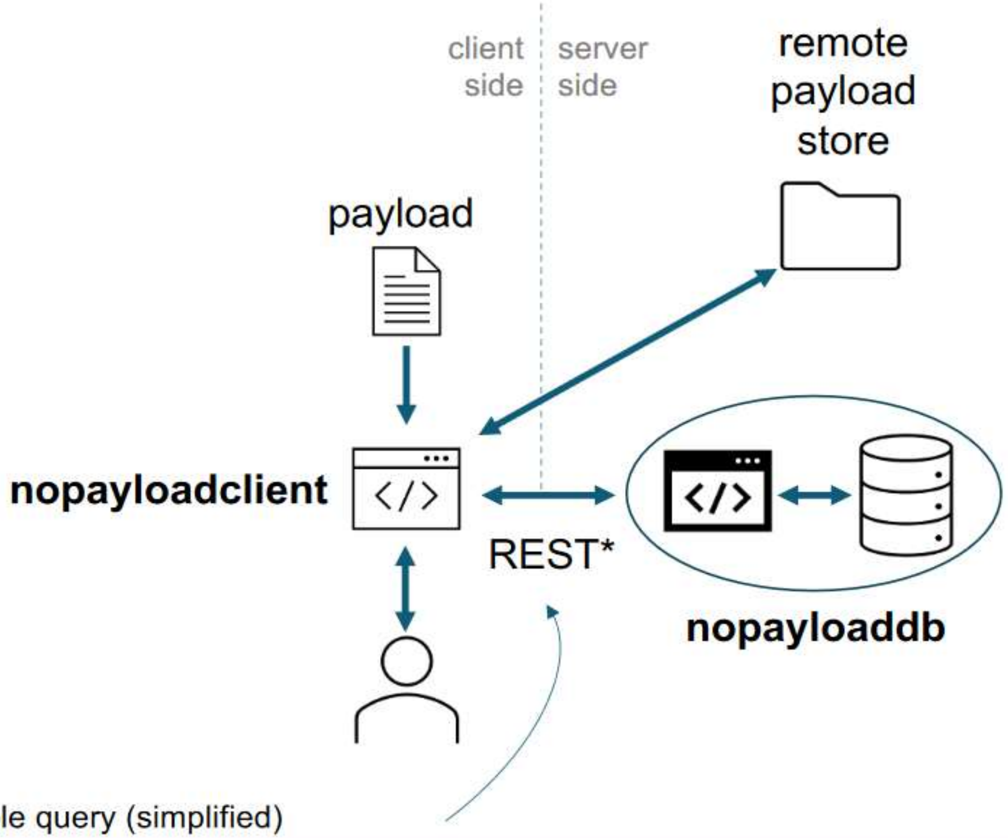
*HSF Conditions Databases activity: https://hepsoftwarefoundation.org/activities/conditionsdb.html

Implementation – Overview

nopayloadclient:

- Client-side stand-alone C++ tool
- Communicates with **nopayloaddb** (server)
- Local caching
- Handling of payloads

- Database stores <u>only</u> file urls
- Use existing solution to store payloads
    - For example /cvmfs/



*Example query (simplified)

```
curl http://<host>/api/payloadiovs/?gtName=test_gt&iovNum=42
-> {type_1: url_1, type_2: url_2, …}
```

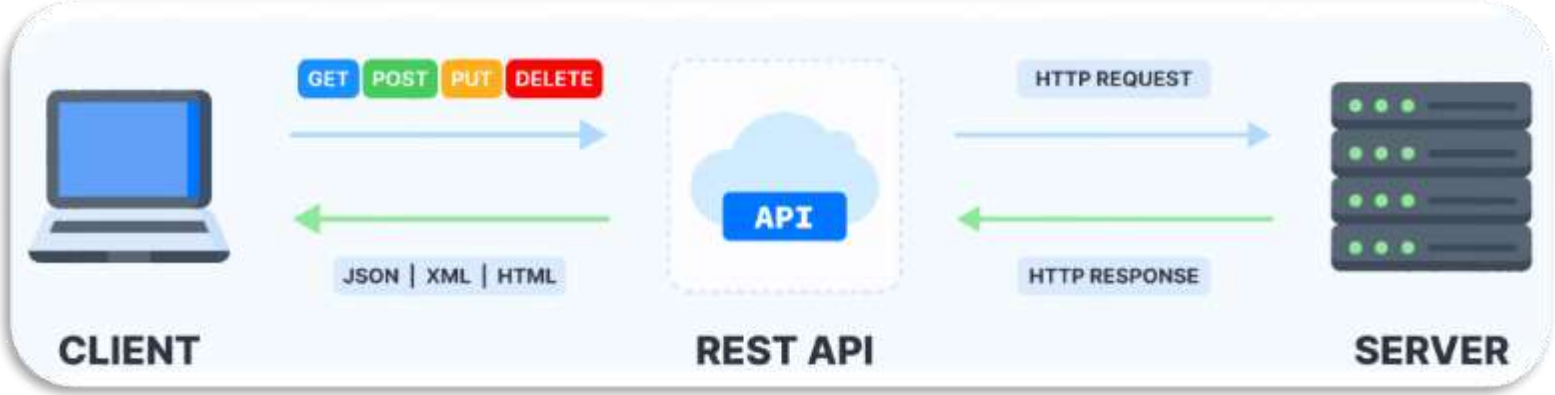
REST API

Representational State Transfer (REST)

Application Programming Interfaces (API)

REST API Rules

- Uniform Interface
- Client – Server
- Stateless
- Cacheable



Request format

Example*

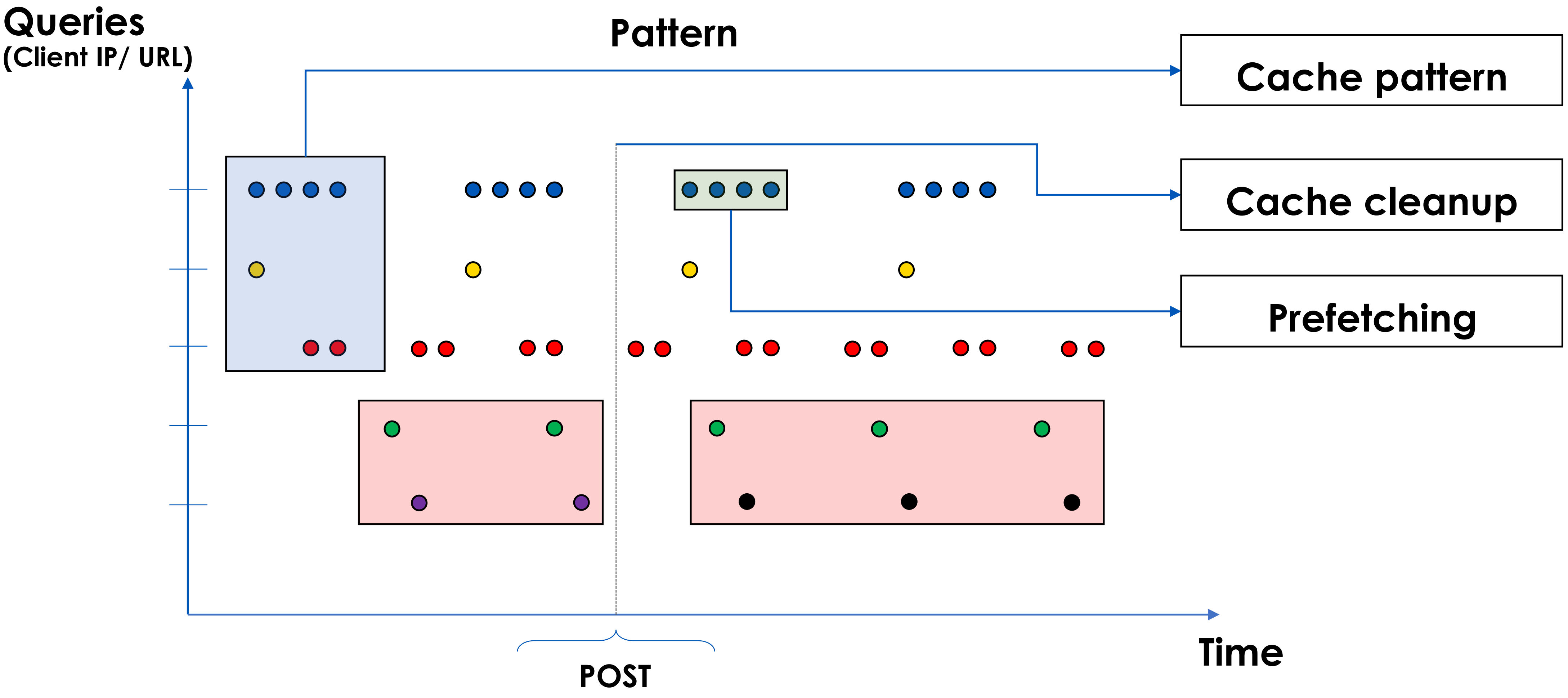
Nginx log

```
10.130.2.1 - - [16/Jun/2023:10:00:32 +0000]
"GET / api / cdb_rest/ payloadiovs/ ?gtName=ExampleGT & majorIOV=0&minorIOV=1000000000
HTTP/1.1" 200 2 "-" "curl/7.29.0" "130.199.148.194“
```

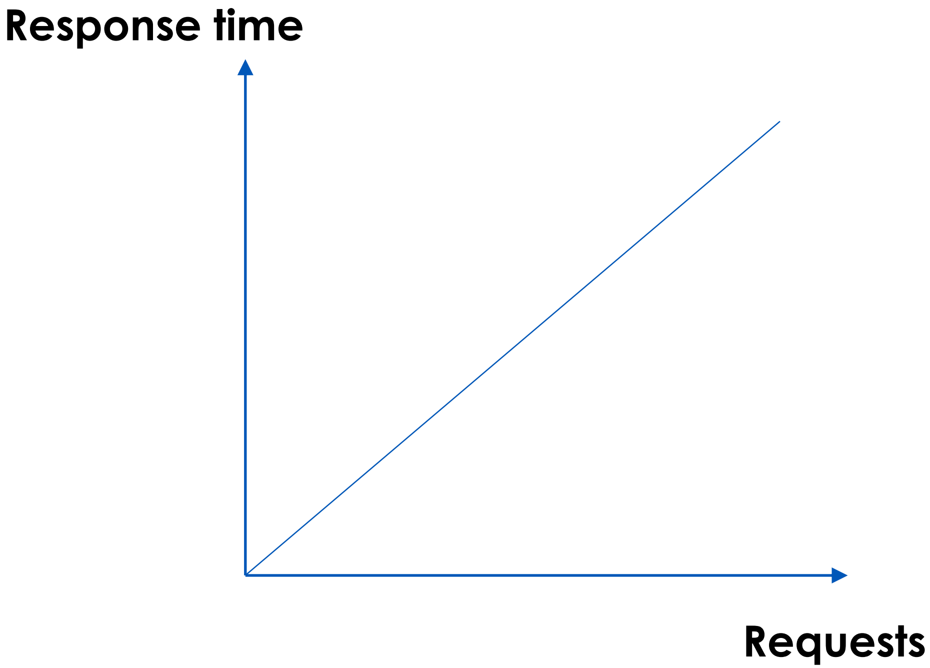
- Real Data Usage
  - Log files of sPHENIX
  - Production usage
- Multi-Dimensional analysis
- Cache and Prefetching Patterns
- Consideration of Request Types

<Client IP> - - **[Local time]**
<Type of request> <URL address> <gtName> <majorIOV> <minorIOV>
<HTTP protocol> <status> <body bytes sent> <additional info>

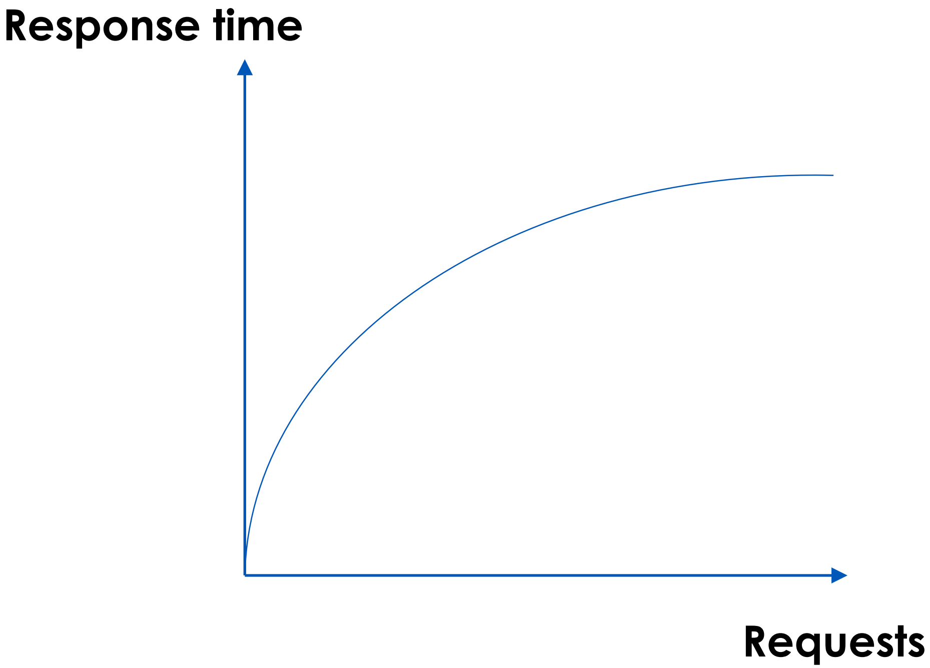
Cache patterns



Response time result



No Cache



Cache

Timeline

Week 1-2

Understand basic principles of Databases, caches, webservers and REST APIs, read literature (HSF conditions data white paper), look at source code of the implementation at hand.

Week 2-3

Understand the data format of the log files that document the DB access and extract time-series data from it for further analysis. Make first plots.

Week 4-5

Understand different access patterns qualitatively and decide on an optimized caching strategy.

Week 6-8

Develop an algorithm that automatically and in real time identifies the current access pattern and suggests an optimized caching strategy.

Week 8-10

Write documentation, summarize findings on slides. Present results in IRIS-HEP Fellowship meeting. Fine-tune and deploy automated.

Thank you for attention

sPHENIX

Radical makeover of Pioneering High Energy Nuclear Interaction eXperiment

- The first experiment that deployed the HSF Conditions Database
- Using a superconducting solenoid on 4° K
- The giant 3D digital camera detector captures 15,000 particle collisions per second, 3 times faster than PHENIX.