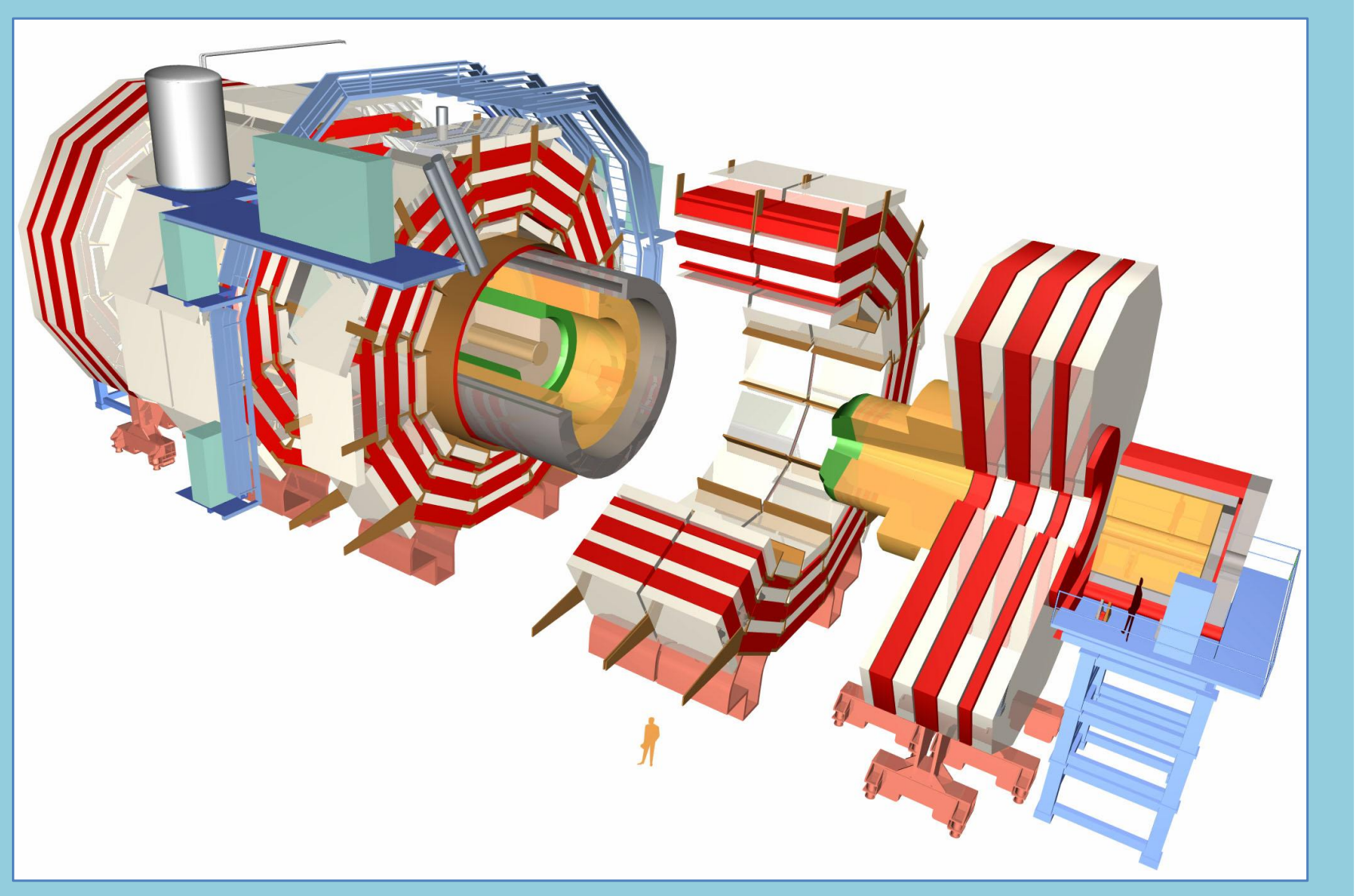
STUDY OF NOSQL TECHNOLOGIES FOR THE CMS CONDITIONS DATA

Roland Sipos, for the CMS Collaboration
E-mail: Roland.Sipos@cern.ch

Introduction

CMS (Compact Muon Solenoid) is an experiment at the LHC at CERN, with the goal to answer the most fundamental questions about the universe, with the essential help of several fields of computer science. The CMS Physics, Software & Computing Group, Computing Operations section (**CMG-CO**) plays a main role in the development and maintance of the complex infrastructure for managing the detector's Alignment and Calibration constants („conditions"). This data is essential for the analysis and reconstruction of the recorded data.

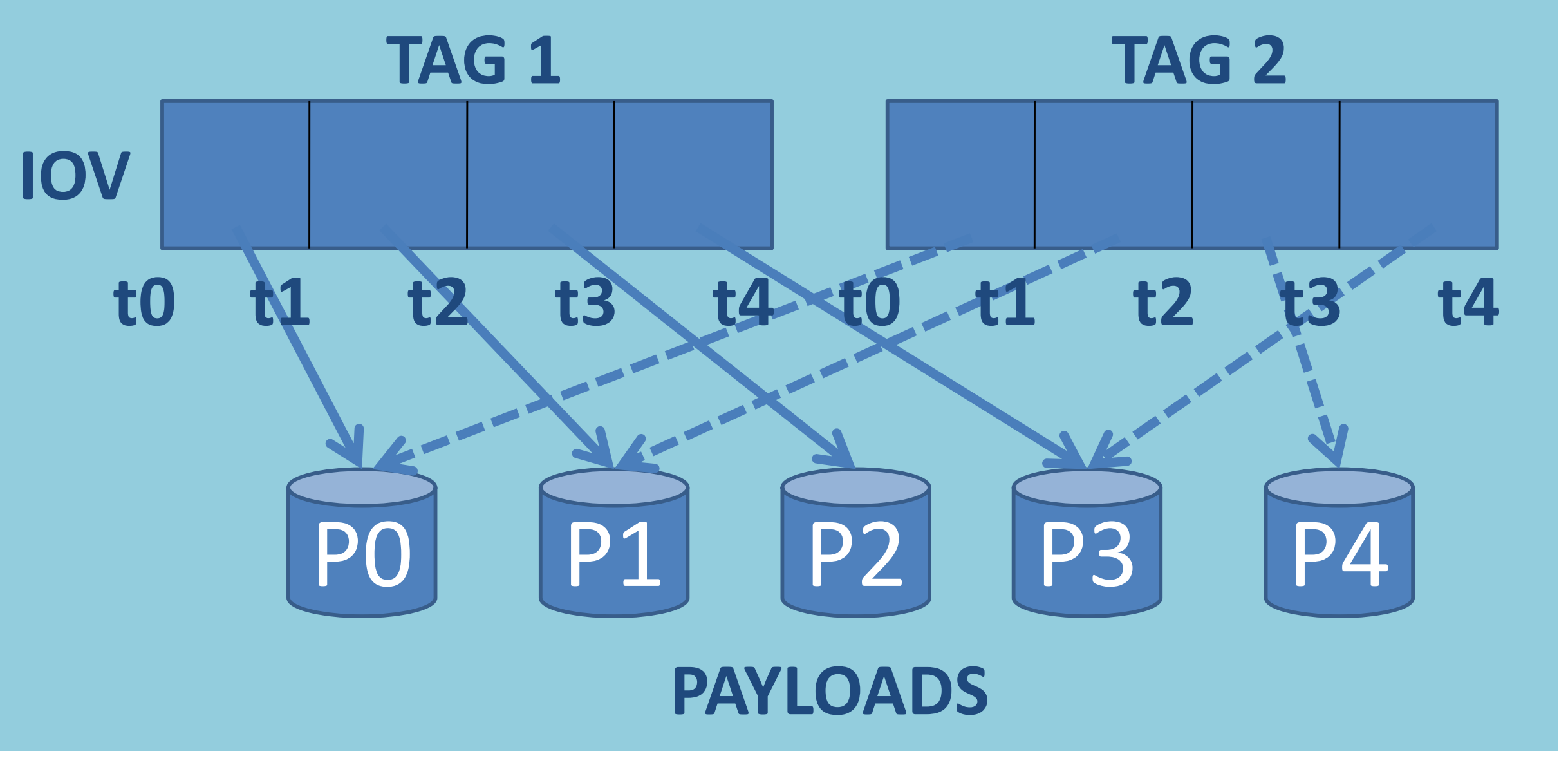

The schematic of the CMS Detector

Conditions Data

"**Conditions**"[1] are non-event data, varying with time. These are critical for the dataflow and need to be properly re-synchronized during the data processing. For this, the **Conditions Database** is a dedicated infrastructure that includes applications and services for both online and offline environments.

The data model consists of the following elements:
- **Payload** – data structure for a given task, detector
- **IOV** (Interval of Validity) – array of intervals
- **Tag** – a label for categorizing a specific IOV
- **Global Tag** – a consistent set of Tags, dedicated for a given workflow.

Conditions are:
- written once,
- never deleted.



NoSQL - One size does not fit all

Currently the Conditions Databases uses a central, CERN provided database service that uses ORACLE servers. As the CMS Conditions' Data Model is relatively simple, and transactions are not necessary, the strict relational model not necessary for such application.

NoSQL in a nutshell:
- different models,
- CAP theorem,
- ACID vs BASE,
- sharding, etc.

It means:

Alternative options!

MySQL Cluster, riak, CouchDB, Cassandra, HBASE, HYPERTABLE

Performance evaluation

The empirical evaluation of proposed NoSQL database candidates is necessary in order to check if a given prototype meets the performance criteria of the original CMS Conditions Database. For this I used a professional tool, called JMeter, thanks to it's following attributes:
- extendable (most important),
- mature implementation,
- remote test support,
- complex workflows,
- open source, etc.

A JMeter extension (called **CustomSamplers**) was implemented with CMS specific needs, in order to measure different scenarios of the Condition Database. It has support for a wide range of relational and non-relational databases. For each database the extension has:
- **Deployers** – build up the data model,
- **QueryHandlers** – simulate the CMS workflow,
- **ConfigElements** – configure persistency objects,
- **Samplers** – report to the testplan listeners.

With this, and basic JMeter features, very complex CMS related workflows and test scenarios can be made that helps the deep performance evaluation of each candidate as a proposed alternative of the current databases.

Deployment

The administration of databases for each test scenario is a complex and time consuming task if it's not automated. In order to make the deployment environment independent, Puppet was used.

With CMS specific, and already existing scripts, the most complicated configurations are set up in minutes on the deployment machines, that are provided by the CERN Virtual Machine and OpenStack virtualization services.

Outlook

Further work in the project involves:
- proper documentation and evaluation of test results,
- prototype creation for the promising NoSQL candidates,
- evaluate necessary code changes in the Conditions software.

References
[1] CMS experience with online and offline databases
A. Pfeiffer et al., J.Phys.Conf.Ser. 396 (2012) 052059

2014