The Large Hadron Collider (LHC) at CERN produces more than 50 TB of diagnostic data every year, shared between normal running periods as well as commissioning periods. The data is collected in different systems, such as the LHC Post Mortem System (PM), the LHC Logging Database and different file catalogs. To analyze and correlate data from these systems it is necessary to extract data to a local workspace and to use scripts to obtain and correlate the required information. Since the amount of data can be huge (depending on the task to be achieved) this approach can be very inefficient. To cope with this problem, a new project was launched to bring the analysis closer to the data itself. This paper describes the concepts and the implementation of the first prototype of an extensible framework, which will allow integrating all the existing data sources as well as future extensions, like hadoop clusters or other parallelization frameworks.

### Motivation

The LHC produces about 50 TB of diagnostics data per year, mainly stored in two systems:

- **LHC Post Mortem System (PM):** Stores data at a high resolution over short time ranges.
- **Common Accelerator Logging Service (CALS):** Provides continuous logging of equipment signals all around the LHC.

### Problems:

- Hard to correlate data
- Slow Data extraction
- High code duplication

### Requirements

#### Basic Requirements:
- Calculations close to the data
- Horizontal Scalability

#### Should also handle:
- Data incompleteness
- Data invalidity
- Mathematical operations
- Physical units
- Error propagation

### Parallelization

Prototype using Akka [4]:

- Gained factor of 20 (parallel data extraction) on one node [5].
- Next steps:
  - Parallelization of Dispatcher
  - Prototype on real cluster

### eDSL and Execution

**Very flexible layer of execution:**
- Start with a very simple solution
- Later optimize execution without any impact on the language level

**An Example:**

```
assertThat([I_MEAS]).isLessThan(55.0, AMPERE).
```

**Produced Expression Tree [3]:**

- Square brackets ([]): labels for the nodes
- Blue: resolved (known values)
- Green: unresolved (to be calculated)

### References