Concept and First Evaluation of the Archiving System for FAIR

V.Rapp[1], V. Čuček[2]

[1]GSI (Darmstadt, Germany), [2]XLAB (Ljubljana, Slovenia)

Abstract

Since the beginning of computer era the storing and analyzing the data was one of the main focuses of IT Systems. Therefore it is no wonder that the users and operators of the coming FAIR complex have expressed a strong requirement to collect the data coming from different accelerator components and store it for the future analysis of the accelerator performance and its proper function. This task will be performed by the Archiving System, a component, which will be developed by FAIRs CSCO team in cooperation with XLAB d.o.o, Slovenia. With more than 2000 devices, over 50000 parameters and around 30 MB of data per second to store the Archiving System will face serious challenges in terms of performance and scalability. Besides of the actual storage complexity the system will also need to provide the mechanisms to access the data in an efficient matter. Fortunately, there are open source products available on the market, which may be utilized to perform the given tasks. This paper presents the first conceptual design of the coming system, the challenges and choices met as well as the integration in the coming FAIR system landscape.

Reasons

- Collecting and storing the data of all relevant devices existing in the FAIR facility
- Correlation of the actual and historic data
- Analyzing the accelerator performance and its proper function by comparison of current and historical settings

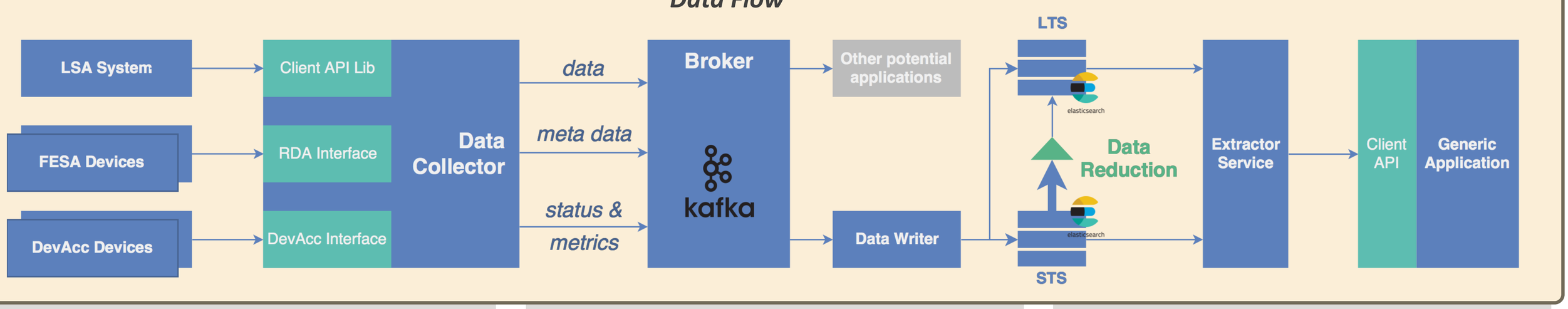
Requirements

- Support of currently active middleware protocols: RDA3 and DevAcc
- Collection of more than 50000 parameters
- 300 TB of data in short term (*STS*) and about 10 TB (per year) of data in long term storage (*LTS*)
- Communicate with other systems to receive supplementary information

Architecture

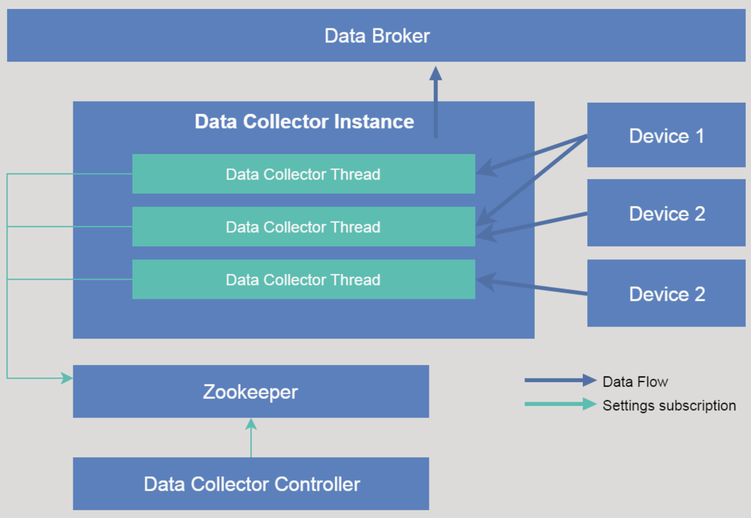
- Consists of several independent multi node components along the data flow
- Strong separation of responsibilities
- Elasticsearch database used for storage
- Apache Kafka broker to distribute data
- Apache Zookeeper for configuration and coordination of instances

Data Flow



Data Collector

- Multiple instance configuration - each consist of multiple threads
- All running instances are coordinated by Controller component, which writes the configuration to the Apache Zookeeper
- Support subscription for multiple middleware protocols using the JAPC API
- Collected information is written to the Apache Kafka broker



Data Collector components

Data Writer

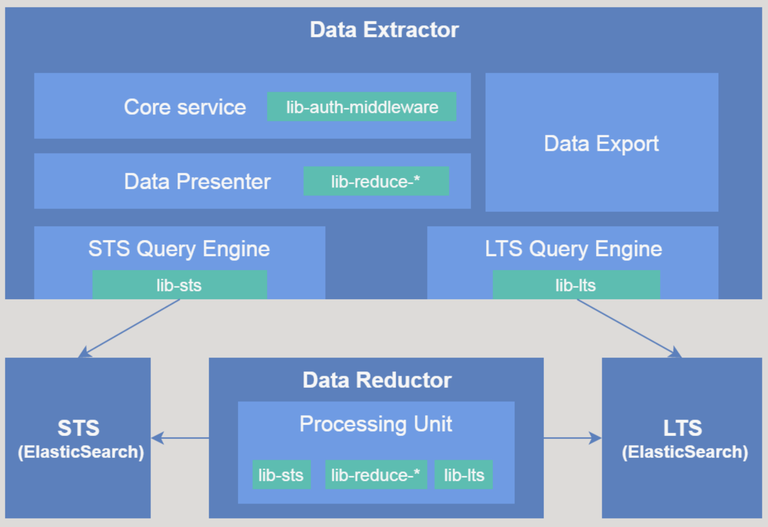
- Writes the data collected from the broker to the short and long term storage
- Encapsulates database calls
- Controlled by a separate controller component

Storage

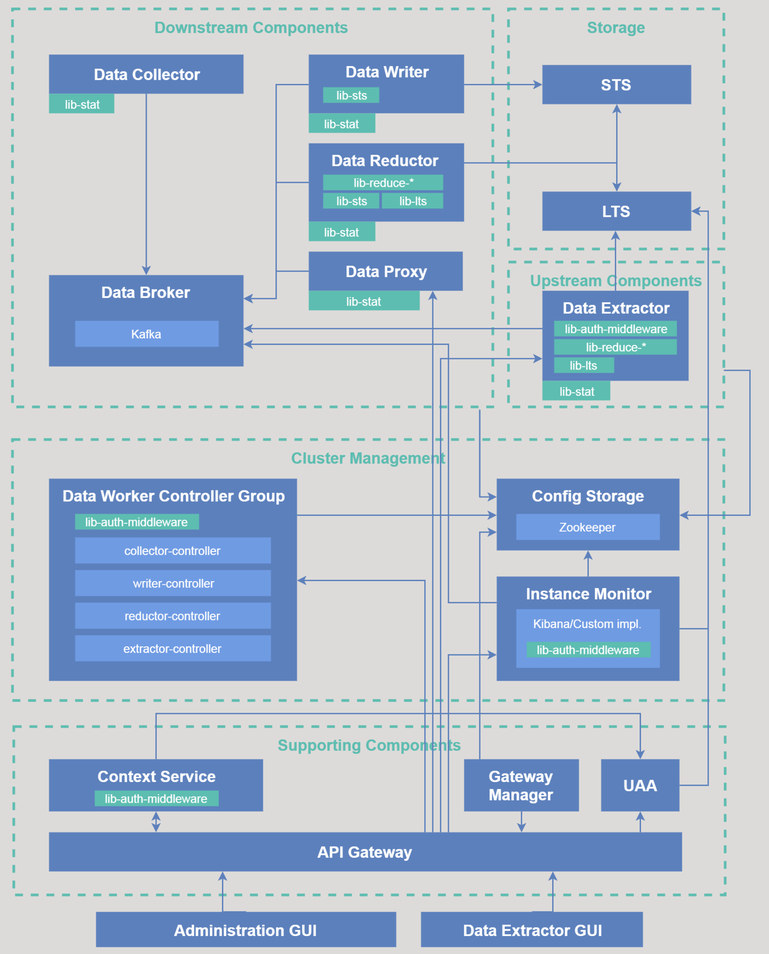
- Based on Elasticsearch database
- Two storage types:
  - Short Term Storage (STS) holds fine granular data for a time period about 3 month
  - Long Term Storage (LTS) persist compressed data, reduced in size and granularity
- Reduction process is performed by a separate Data Reductor component
- Different reduction algorithms will be supported

Data Extractor

- Hides the STS and LTS implementation details from the client
- Provides RESTfull interface for data access
- Additional client library for Java and C++ to simplify and aggregate calls



Data Extractor components



Component overview

Current Status and Outlook

- First prototype with limited functionality was deployed at GSI and is currently under test
- Remaining functions and components are under development by XLAB and GSI
- Performance of the Elasticsearch database as well as Kafka broker will be the focus of the coming evaluations
- Aim to use the extended prototype for evaluation during the 2018 beam time period