GSI Helmholtzzentrum für Schwerionenforschung GmbH

Detailed Specification of the FAIR Accelerator Control System Component „Archiving System“

Document Name
F-DS-C-11e

Date yyyy-mm-dd
2011-08-10

Abstract

This document is the Detailed Specification of the accelerator control system component "Archiving System". Its task is to collect and store pertinent accelerator data centrally to permit analysis of the accelerator performance and its proper function. This work package is part of the "Control System Software Packages" work package and covers the PSP code 2.14.10.2.9.

FAIR Division/Group/Supplier
Controls Department

Prepared by:
BEL-CCT

Table of Contents

List of Tables

List of Figures

1. Purpose and Classification of the Document

The purpose of this document is to specify the Accelerator Control System component "Archiving System" for FAIR (PSP code 2.14.10.2.9).

For this component this document is the most detailed type of document in the hierarchy of Control system specifications.

Whenever regulations and requirements are specified in the General Specifications, Technical Guidelines or Common Specifications of the Control System they are only referenced in this document. The related documents are listed in the Appendix.

No legal or contractual conditions are treated in this document. All related information is given in the General Specifications for FAIR.

1.1. Responsibilities

The responsibilities with respect to changes and modifications of the present document are entirely in the hands of the Control System Project Department of the GSI Helmholtz Centre for Heavy Ion Research GmbH (GSI) Darmstadt.

For initial information please contact the administration of the Controls Department.

Further information on the organigram, names of responsible persons and task leaders, as well as the agreed document release and approval procedure is summarized in the organizational note 'Controls Project for FAIR'.

1.2. Classifications of Requirements

The following definitions of requirement classifications are being used throughout the document:

- "**Must**" or "**shall**" or "**is required to**" is used to indicate mandatory requirements, strictly to be followed in order to conform to the standard and from which no deviation is permitted.
- "**Must not**" or "**shall not**" mean that the definition is an absolute prohibition of the specification.
- "**Should**" or "**is recommended**" is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others or that a certain course of action is preferred but not required.
- "**Should not**" or "**is not recommended**" mean that there may exist valid reasons in particular circumstances when the particular behaviour is acceptable or even useful, but the full implications should be understood and the case carefully weighted before implementing any behaviour described with this label.
- "**May**", which is equivalent to "**is permitted**", is used to indicate a course of action permissible within the limits of the standard.

2. Scope of the Technical System

2.1. System Overview

The purpose of the Archiving System is to store historical data gained and generated by the control system in a permanent location. It allows storing data on a configurable base of resolution in time, triggered by timing events or on-change. Data may be either data gathered from devices, higher level data like computed physical properties or generated abstract data (see e.g. [9]).

The Archiving System includes functionality to query, filter, correlate and display historical data. For data identification and ordering purposes the system should be able to query, receive and store data from other control systems like the accelerator's settings management system, the beam transmission monitor system or the post-mortem system.

The Archiving System includes a service that allows deleting or aggregation of historical data on a configurable base.

The Archiving System provides control room applications to display, configure, and manage archived data.

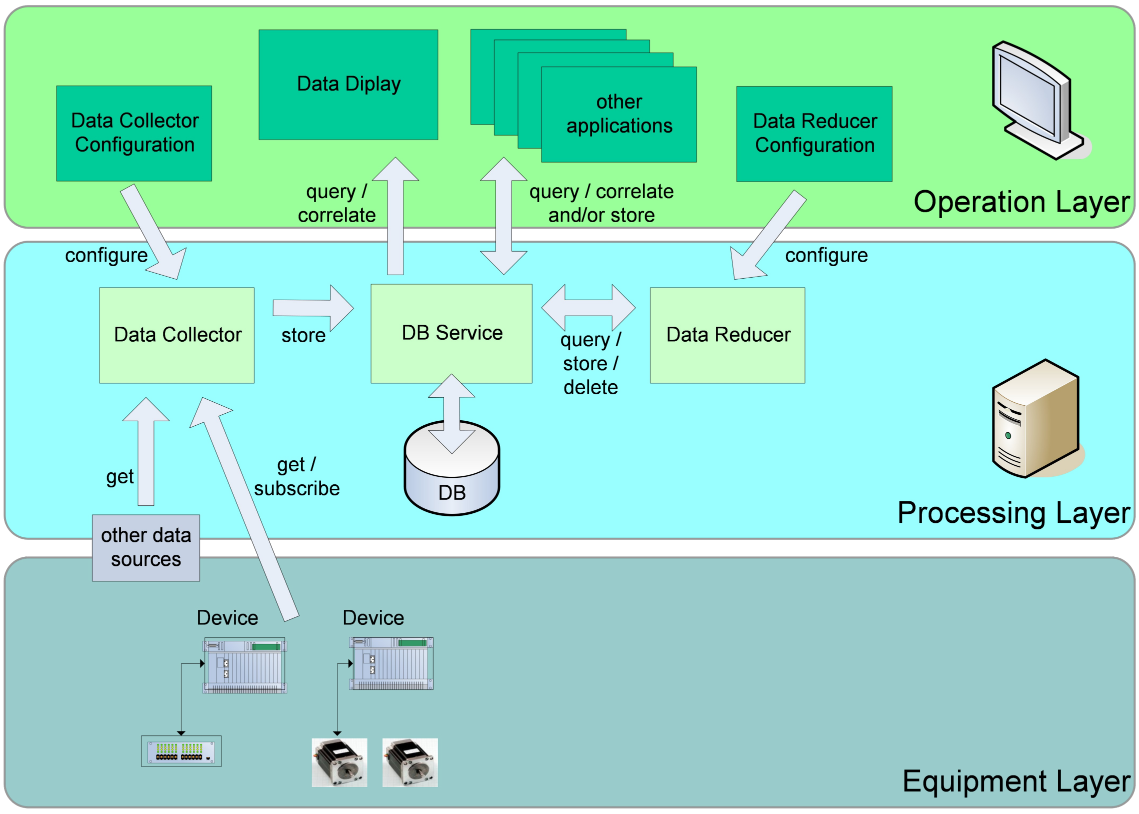

Figure 1: System Overview

2.2. Limits of the System and Environment

2.2.1. Limits

The Archiving System does not cover following

- accelerator settings data administration and storage, covered by the existing LSA settings management system [ref]
- diagnostic logging data storage, covered by the XXX [ref]

However, due of similarity of data storage concepts the design of the Archiving System should aim the possibility to provide the storage functionality for following systems:

- beam transmission monitoring system (BTM)
- post mortem system (PM)

2.2.2. Interfaces

The Archiving System shall have an interface to control system devices [6], using JAPC (Java).

In order to correlate and display archived data, control room applications linked to the Archiving System must be able to query data from the accelerator's settings management system [2], the beam transmission monitor system [4], and the post mortem system [5], as well as the UNICOS systems data storage system for short term data.

If there is a separate alarm history storage system for the alarm system [3], control room applications have an interface to the alarm system as well.

Additional the Archiving System should implemented a Database to Database transfer of the WinCC OA (UNICOS) System for long term data storage.

2.2.3. System Environment

The Archiving System runs on (dedicated) central Linux and database servers. The detailed system environment and configuration is specified by the Control System Department by the time implementation begins (defined as a project milestone).

2.3. Basis of Concept

2.3.1. Functional Requirements

Number	Description of Requirement
AR_010	The Archiving System must archive data on long term basis over the life-cycle of the accelerator facility.

AR_020	Which data is to be archived must be freely configurable.
AR_030	Archiving configurations must be storable for later re-use.
AR_040	Different archiving configurations must supported: • default configuration that is valid at the start of the Archiving System • adjustable predefined configurations that are used regularly and are available by the "push of a button" • custom configurations created by users
AR_050	The Archiving System must offer supporting functions to query and correlate archived data
AR_060	The Archiving System must offer a service to reduce or delete archived data automatically or manually
AR_070	Automatic reduction or deletion of archived data must be freely configurable
AR_080	Configurations for reduction and deletion must be storable for later re-use
AR_090	The Archiving System must offer an API • to write data into the archive database, • to read data from the archive database, and • to help correlate data read from the archive database.
AR_100	The Archiving System must provide GUI components to configure the system, to correlate and display archived data.
AR_110	The Archiving System must be able to export (correlated) archived data in a generic format (e.g. comma separated value files, csv) to support further post-processing using third-party applications and scripts.
AR_120	The Archiving System shall provide facilities to store meta data alongside to the regular data that can be used to index, preselect and filter the data to be queried and extracted from the data base. Some of the meta data include: targeted beam parameter, machine parameter, accelerator & beam modes, whether PM trigger have been issued, operational ranges (OP day, week, time between technical stop), and important operational markers (start/end operation, start/end technical stop, …),
AR_130	The architecture of the Archiving System shall allow an easy extension to propagate the collected data to the 3rd party systems. Such extension could be data proxy component, which on itself provides other systems with live data collected from the devices. It reduces the load on the front end controllers.
AR_140	Following the requirement AR_130 the proxy extension must be able to provide data to the users outside of the accelerator network.
AR_150	Load management and monitoring of the Archiving System infrastructure.
AR_160	User access management (kicking-out, limiting of piggy clients,

	bandwidth shaping/scheduling, ...

Table 1: List of Functional Requirements

2.3.2. Non-functional Requirements

The archiving system must be designed and implemented in order to be scalable in terms of performance of data acquisition. The expected maximum data traffic volume at the moment of writing of this documentation is described in following numbers. Those values however must be seen as rough estimates for minimum requirements of the system, which should be able to scale during the operation period.

Parameter	**Value**
Amount of monitorable devices	2032
Total amount of monitorable variables	~50000
Incoming data load	~10 MB/s
Required data buffer (short time storage)	~5 TB/week
Required data storage (long term storage)*	5 – 10 TB/year

**Estimated data reduction of 25-50x*

Table 2: Non-functional Constraints

Note that the provided estimations do not represent the actual storage space required on disk, since this depends strongly on diverse technical parameters used to store the data later such as: data format, database technology, replication factor etc. Hence the final estimation can only be provided after full definition of all those parameters.

~~Performance parameters will be investigated, defined, and provided by the main contractor. Parameters should be:~~

- ~~minimum transactions per second during database write~~
- ~~maximum response time for reading and correlating~~
- ~~maximum size of a data set~~
- ~~maximum size of the database~~

2.3.3. General Constraints

Any usage of means for localization of accelerator components must comply with the accelerator's control system localization guidelines.

Any user or rights management must be designed and developed in accordance with the central rights management within the control system.

All GUIs must comply with the GUI Guidelines [7].

2.3.4. Architectural Principles

The Software Architecture Guideline for the Control System [8] fully applies.

3. Technical Specifications

The Archiving System consists of various components. The main parts of the system are presented in the figure below.

There are two possible approaches how the data is collected and aggregated by the system. First is the active subscription to different measurement and status parameters of configured or known devices. Those devices would be typically FESA implemented devices and hence the subscription can be established utilizing the JAPC (Java) ~~or RDA (C++)~~ framework~~s~~. Another type of devices are UNICOS devices whose interface will be defined later in the proceeding. Another form of data aggregation is an active data supply. Therefore the Archiving System provides an API, which can be used by various clients to archive and stored their own data in the system.

The automatically collected data is verified by the Data Inspector for its correctness and afterwards is written to the database using the DB Writer.

The Database can be defined into short and long term storage. While the short term data must be available in its full granularity, but only for a given time period, the long term data should be available for the accelerators lifetime, but may be compressed.

The data reduction service reduces or deletes historical data and writes them into long term storage.

Data aggregation, checking and reduction can be configurator using an administration GUI (AdminGUI).

The configuration of data collection and data reduction is done via GUIs.

To access the stored data a Data Extractor is used. It provides an API to the 3rd party applications as well as an own User GUI, used to present the data in various formats for future analysis. Therefore the Data Extractor finds correlated records inside of long and short term DB and delivers them to the user.

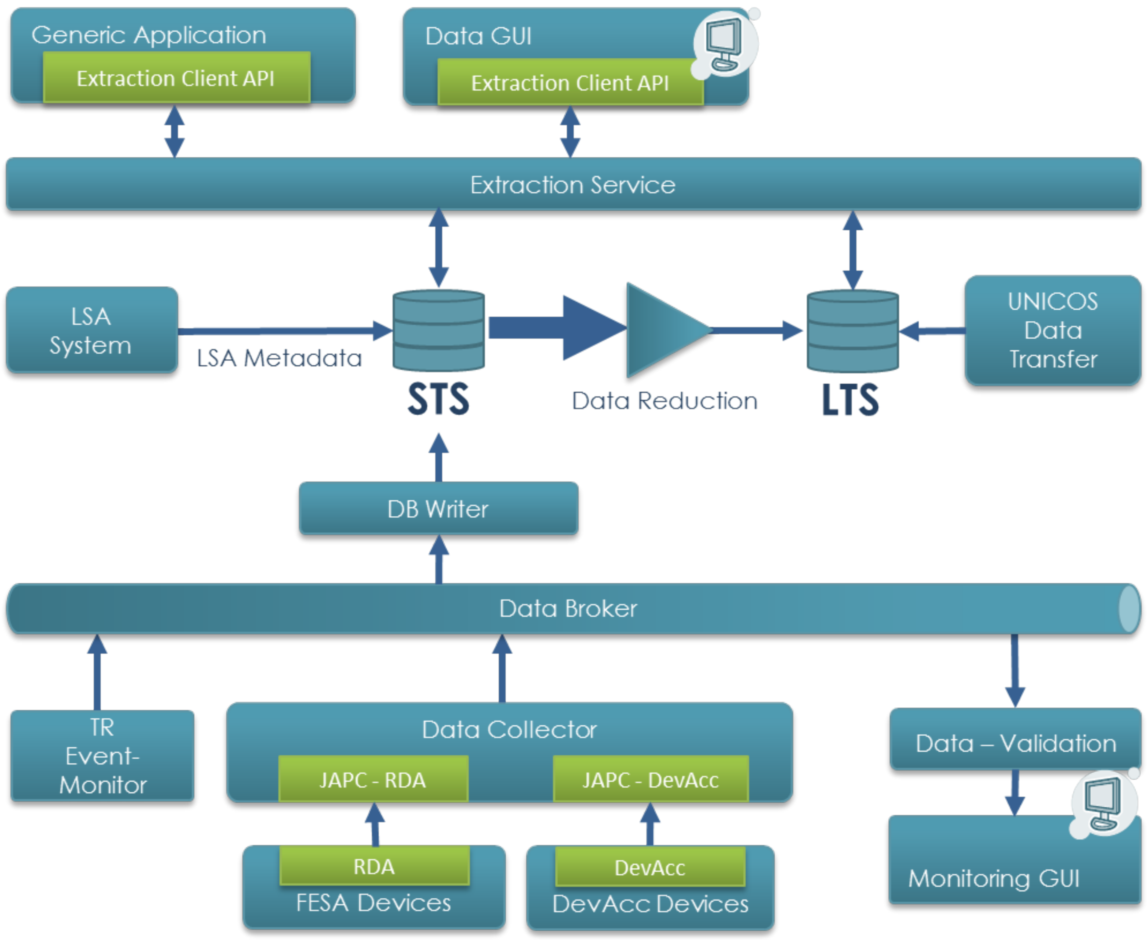

Figure 2: Components Overview

3.1. Storage and Access

To store the incoming data a Database service should be used.

The most suited database technology and/or supplier must be evaluated. The final choice must fit the GSIs infrastructural and financing constrains. The usage of alternative DB distributors or technologies shall be evaluated.

By the start of the project the Apache Cassandra (ref) is a favorited choice for the storage solution. However, the actual usage of this product should be evaluated during first releases and the prototyping phase and hence the architecture must be flexible enough to overcome the possible change of the used DB.

The final decision about the database in the final release shall be made as soon as possible after the evaluation and the first experience with Apache Cassandra.

The Oracle © Database system for storage may be considered as a main drawback option in case the NoSQL concept on base of Apache Cassandra will prove itself as not suitable for the GSI environment and needs.

3.1.1. Data Reduction

The Archiving System must be able to automatically reduce or delete historical data. This reduction process must be configurable, see chapter 3.1.2.

~~Manual data reduction by the user must be possible as well.~~

For special data reduction requirements it must be possible to easily add new algorithms via a plug-in mechanism.

At least following reduction options shall be available in the archiving system from the start:

- No Reduction: AS does not apply a data reduction
- Reduced Sampling Rate: AS stores data with an reduced sampling rate i.e. store each nth value (1kHz -> 1Hz)
- On Change: AS stores only value changes, which exceeds a certain threshold. The threshold should be adjustable for single parameters, as well as whole system
- Timed Aggregation: Aggregate and store values for multiple BPCs for a single parameter:
  - Minimum value (for each timestamp within of BPC)
  - Maximum value
  - Standard deviation (stdev)
  - Median value
  - Mean value
  - Transient values – actual values outside of a n*stdev band

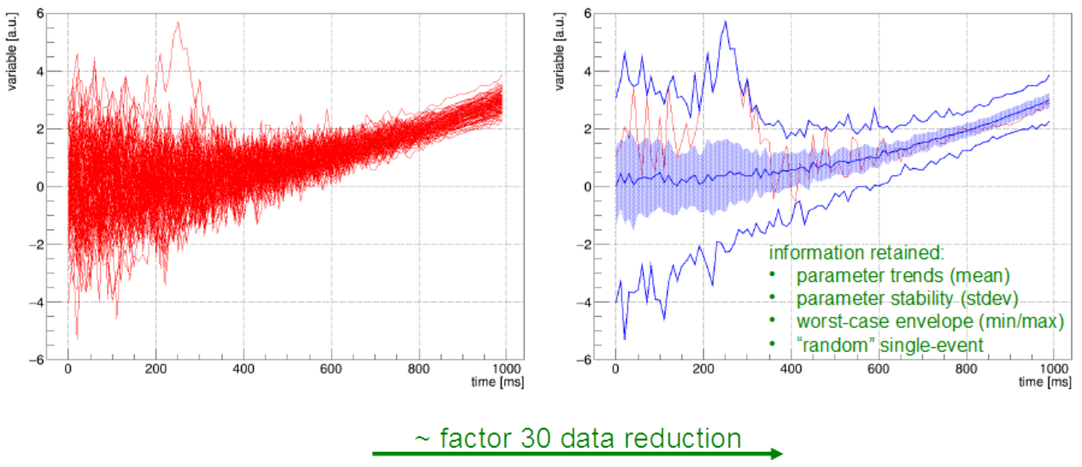

Figure 3: Timed Aggregation

It must be possible to configure the exceptions of data reduction i.e. data which should be only compressed using lossless compression (e.g. not compressed at all). Examples of such values include beam profiles at injection and extraction or post mortem data.

A compression rate of factor 20-50x must be aimed. However actual required rate can only be determined during the prototype test and evaluation phase, since it strongly depends on database technology, its storage capability and the data model behind.

3.1.2. Data Reduction Configuration

Automatic reduction or deletion of historical data must be configurable. At least following points should be adjustable:

- when or how often data should be reduced or deleted
- which algorithm should be used to reduce data
- which data should be reduced or deleted
- reduction rate

Parameters should be adjustable depending on the 'accelerator-' and 'beam mode' of the accelerator or transfer-line segment for device associated with them [ref Accelerator & Beam Mode Spec].

3.1.3. Data Extractor

The Data Extractor component is responsible for querying archive data. It consists of two parts – server (database) part and the actual client library. The server part is encapsulating the database requests, so the user should not be aware of how or where the data is stored. Another part should be implemented in form of client library, which hides the communication details between the server and the client components.

The client library must be implemented in Java and C++ symmetrically (i.e. same class, methods, interface names etc.) However the Java API has higher priority in this case. It must provide functionality to:

- query data from the archive database
- find correlated entries in the queried data

The server part should offer a clear and well documented interface to query the data. A RESTful service should be the main option here. The client component may utilize this service, but can also be based on other communication patterns (e.g. RMI, ZeroMQ).

Since the server part should directly communicate with the database its implementation depends highly on the data storage technology used. Depending on database type there are multiple tools and languages available which use should be evaluated before the actual implementation starts. Especially for Apache Cassandra the usage of Spark (ref) must be evaluated.

The extractor's server component must be easily scalable in terms of amount of requests per second. When extracting data, there should be guaranteed that the writing performance of the system remains sufficient the handle the whole write load. Clearly in this sense the writing has a higher priority over extraction and the data extraction shall not interfere with storage performance.

Extraction Use Cases

Some typical planned use cases of the Archiving System to illustrate the usage of the data extractor:

1. A user selects particular value he wants to view and defines a time period to display (given t_{start} and t_{stop}, last day, last month etc.)
2. As previously a user selects a time period but specifies additional selection parameters. Such parameters can be following:
    - Show values only for specific context
    - Show only beam production chains with beam
    - Retrieve data only between specific events (e.g injection -> extraction)
    - Select values for a given beam types (e.g. specific element beams)
3. As described before, but the system allows a selection of a particular BPC with specific parameter. Those parameters include:
    - Beam parameter
        - Ion-species
        - Beam target (experiment)
        - Actual intensity
        - Actual beam transmission
        - Targeted intensity range
    - Machine parameter
        - Ramp-rate
        - Min/max rigidity
        - Injection/extraction energy
        - Slow/fast extraction
        - Cycle/store length
    - Cycles with(-out) post-mortem events
    - Operational ranges
        - OP year
        - Between technical stops
        - OP week
        - OP day

A combination of described filter criteria should be handled as well.

3.1.4. Database Data Structures

For data querying and correlation each stored data set must at least contain the following information:

- time stamp
- accelerator context (selector)
- unique source identification, like:
    - nomenclature, property name, field name
    - host name, service name, class name
- SI unit

Data sets must at least support all JAPC (FESA) data types.

It shall be evaluated how to identify datasets in the storage. This evaluation is also related to the DB technology evaluation (see chapter 3.1) since one or another storage concept may be preferred for the underlying database. The Controls Software uses a technical device/property [/field] model to identify and access data. The user of the Archiving System however, will be interested to see the evolution of a particular physical parameter over time. Those parameters are mapped to technical identification, but this mapping may change over time. There are two options to consider. First is to store the data using its technical identification. This approach will additionally require storing the nomenclature changes during the archiving period. The second approach is to store the data using its physical identification. This premises an existence of an actual and unique (homomorph) mapping for the parameters, which need to be maintained. In addition this approach will have consequences to the generic data push API (see chapter 3.2.1).

It shall be investigated whether tables with special data types, for instance the data of the property STATUS (standard property of each device, see also [10]) or the structure of an alarm, are necessary.

Usually, it is strongly recommended to design database tables with well-defined data types. This implementation detail conflicts with the requirement to freely configure what should be archived. Therefore, it shall be investigated whether the database should accept BLOBs (binary large objects) up to certain extend. If yes, BLOB tables shall be accepted only temporarily and must be converted to well-defined data types within a certain time.

The actual data model and the storage concept (e.g storing data as JSON objects) should be evaluated.

3.1.5. UNICOS DB Data transfer

Beside of the device data coming from the JAPC interface the archiving system should also archive the information from the WinCC OA (UNICOS) devices. Those systems will use the own Database (Oracle or Raima) or to store the data. Since the date will not be stored for a long term usage, a transfer to the Archiving System is required.

3.1.6. Additional Data (Meta-information)

In order to select the stored data from devices according to the user needs the archiving system shall store some historical information coming from other systems from the GSI controls environment.

Timing Information

Timing information is required to assign the data to particular timing periods with specific properties (e.g. beam-on, beam-off). This information contains time event names with according time stamps.
A timing receiver component should therefore listen to timing event and propagate them to the database.

LSA Beam Production Chain Information

LSA system stores additional Meta information about the particular beam production chains. This information shall be transferred to the Archiving Systems storage.
The data extractor (see chapter 3.1.3) if requested must be able to read this information from database and correlate it with according archive entries.

3.2. Data Aggregation

The Archiving System should be able to accept data form following data sources.

- FESA devices via JAPC/RDA Subscriptions
- Various Sources via Push API

To collect and aggregate data a collector component shall be implemented. Its main function is to establish subscriptions to various devices and provide an API to push the data to the system. This component needs to be highly scalable and able to distribute the load over multiple instances of its own.

At least some of collected data needs to be checked against customizable rules for being consistent and correct. This validation should be performed by a separate component.

Accumulated data needs to be transferred to the database. Therefore a component called DB Writer must be implemented. It must accumulate the data coming from the collector component instances and write it to the short term data base. Additionally some parts of it shell handle the information coming from timing event monitor (see chapter 3.1.6).

All components should be configurable using the Admin GUI.

3.2.1. Data Collector

The Data Collector component is responsible for the actual data acquisition process and communicates directly with the data suppliers. Data should be polled or requested via JAPC subscription.

Since the network bandwidth as well as CPU performance of a single machine is limited the collector component must be designed to scale linearly in a distributed matter. The amount of instances/threads of collectors needs to be freely configurable.

Additionally a client side API should be provided to allow the clients actively push their own data to the Archiving System. The communication protocol between the client side API and the actual aggregator service should be designed in a generic way, allowing other clients to use it.

FESA and Device Access

Data from the FESA and DevAcc devices should be accessed using the JAPC subscriptions.

To protect the access to device properties the data collector instances should be able to utilize the Role Based Access (RBAC) library from CERN, which will be deployed at GSI.

Push API

A client-side API should be a part of the Aggregator component. This API should allow the clients to archive their own data in the system.

The Push API must be available for both C++ and Java languages.

3.2.2. Data Dissemination – Messaging

As data collected by the Data Collector component may also be used by other systems than the AS System itself the system should implement the data distribution part using a generic messaging framework. A broker based solution fits the requirements of the system at most in this point.

The Apache Kafka (ref) seems to be the most suitable product, which fulfils the GSI requirements and should be aimed to use for the system. Similar to the DB the use of the messaging broker should be evaluated and tested during the first prototype and releases. The final choice must be done as soon as possible after the first prototype has been delivered.

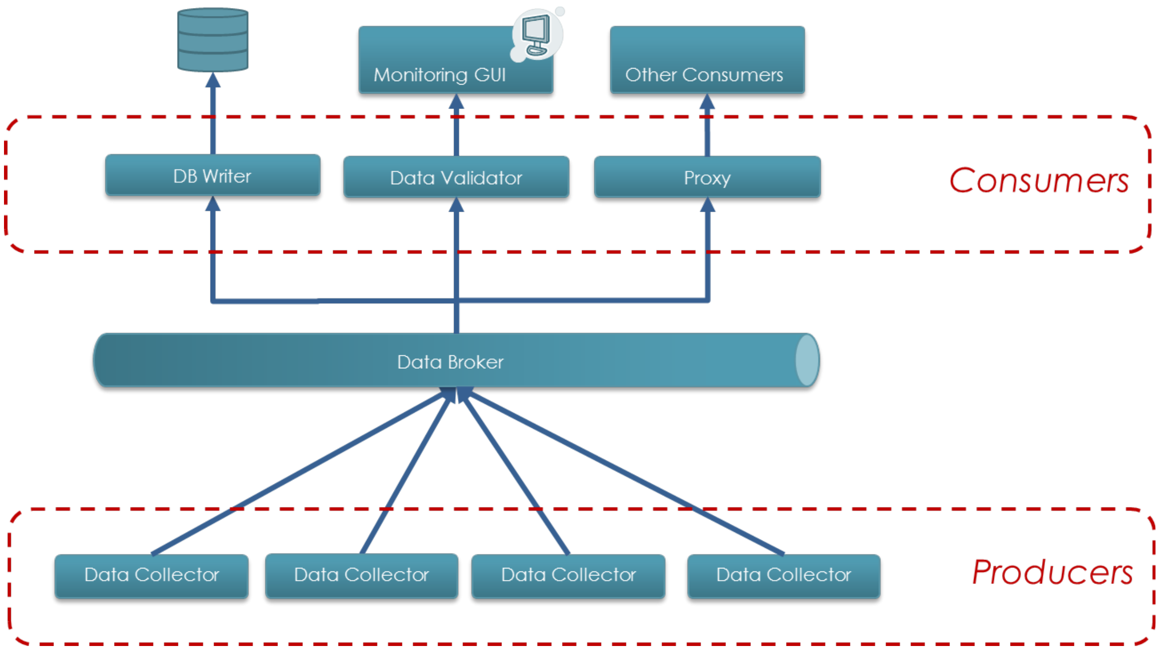

Figure 4: Message Broker Environment

3.2.3. Data Validation

Incoming data must be validated to assure its correctness and consistency hence subscriptions must be supervised to assure that no data to be archived gets lost. Checks should be configurable and are performed by Data Validator component.

The component should be extendable using plug-ins, which means that new rules may be easily added to the system during the operation time.

Due of the amount of incoming data it is an object of consideration if such validation is achievable sequentially before writing into the database. Since the focus of the system is the storing of date it should be realised as parallel process with an extra output to the GUI. In this case the errors from validation should also be logged in the archiving database.

The validator component should be connected with the underlying data collector using the messaging broker described in the previous chapter (see Figure 4)

3.2.4. DB Writer

The Information collected by multiple collector instances should be accumulated within a component called DB Writer, which purpose is to optimize and hide the actual writings into the database from the users.

An actual implementation of the writer component (especially its accumulation part) should be discussed. Currently there are some solutions on the market (e.g. Apache Kafka) which can be utilized for that purpose.

The implementation of the actual database writing depends strongly on the choice of DB technology (see chapter 3.1), which will be defined later during the specification processes.

The writer service must be able to temporarily buffer data to be archived if the archive database could not be accessed.

3.2.5. Data Storage Configuration

The Archiving Service must support a user to freely configure which, how often, and how long data should be archived.

'How long' means to specify when an archiving process should automatically terminate. The automatic termination of an archiving process should be specified using

- a period or an instant of time
- an event or
- a count of transactions or a size of data stored

At least following parameters of the system should be configurable:

- Which parameters from which devices should be archived
- Weather a push or a pull subscription should be used for particular device/parameter

- How often a particular device should send data
- Logging configuration (level and output)
- DB writing configuration (e.g. how often to commit data, how big the chunks should be)

3.3. User Interface

A graphical user interface (GUI) must be developed as a part of the delivery. The interface itself should consist of multiple independent components.

A data display component to request and present the archived data.

A monitoring component must provide monitoring functions of the Archiving System. It should visualize the results and possible data check errors from the data validation.

A configuration or admin component should allow the configuration of different component of the archiving system.

As it is not yet clear which data displays and correlations will be required in the future, an implementation shall be investigated which facilitates a generic way to easily add new data displays to the GUI.

The GUI shall be designed in accordance with the main contractor and in close cooperation with the operations group. The GUI must adhere to the GUI Guidelines [7].

3.3.1. Data GUI

The Data GUI is shall be used to query and present the data to the user. As other GUI programs this component shall be implemented is a standalone application and utilize the Client Extraction API, described before. A user should be able to select specific physical data parameters for a given period of time or for a given beam production chain. The GUI should also provide functionality to find a particular cycle with desired parameters e.g. beam on/off, cycle with particular element etc.

4. Quality Assurance, Tests and Acceptance

The system to be built must adhere to the guidelines and recommendations for software developments in the FAIR accelerator control system context, as referenced in the FAIR Common Specification F-CS-C-01e (Common Specification Accelerator Control System). The supplier of the work package must identify the relevant standards and recommendations before start of the development. Details must be fixed as part of the technical design concept in the initialization phase.

4.1. Development Methodology

The Archiving System shall be developed in an iterative and incremental methodology.

Each iteration cycle must result in a running system which can be evaluated and tested at FAIR site. The first iteration, which has to be available as early as possible, must concentrate on the most critical functionality. In successive iterations, the system is enhanced by adding features until the desired total functionality is reached.

In the initialization phase, the technical design concept and the plan for the iterations must be developed, and must be approved by the FAIR contracting body. At end of each iteration cycle the achieved status of the system will be evaluated and the iteration plan will be adjusted. Each iteration cycle must be approved by the FAIR contracting body before it can be started.

4.2. Quality Assurance System of the Supplier

Generally, the software specific measures of Quality Assurance described in Common Specification "Accelerator Control System" [1] fully apply.

4.3. FAT

The Common Specification "Accelerator Control System" [1] fully applies.

4.4. SAT

The Common Specification "Accelerator Control System" [1] fully applies.

5. Documentation

The Common Specification "Accelerator Control System" [1] fully applies.

6. Warranty

The conditions and warranty period specified in the contract applies.

7. Scope of Delivery

The following components have to be delivered:

- Database components
- Formal Datamodell description
- Data Collector service
- Data Validation service
- DB Writer component
- Data Reduction service
- Library for data correlation (helper) functions
- GUI for data correlation and display
- GUI for Configuration
- GUI for Monitoring

I. Attached Documents

List of abbreviations for controls (Abbreviations_Controls.pdf).

II. Related Documentation

[1] F-CS-C-01e, FAIR Common Specification "Accelerator Control System"

[2] F-DS-C-03e, FAIR Detailed Specification "Settings Management System"

[3] F-DS-C-09e, FAIR Detailed Specification "Alarm System"

[4] F-DS-C-12e, FAIR Detailed Specification "Beam Transmission Monitor System"

[5] F-DS-C-13e, FAIR Detailed Specification "Post Mortem System"

[6] F-DS-C-15e, FAIR Detailed Specification "FEC Device Classes (FESA)"

[7] F-DG-C-02e, "GUI Guideline"

[8] F-DG-C-03e, "Software Architecture Guideline"

[9] A Survey of Basic Storage Systems for Large Amounts of Particle Accelerator Data, Winfried Schütte, DESY MST, PCaPAC 2000

[10] F-DG-C-01e, "FESA Development Guideline"

III. Document Information

III.1. Document History

Version	Date	Description	Author	Review / Approval
0.1	06. Oct 2011	Draft	L. Hechler	
1.0	07. Oct. 2011	Draft version	CCT	CCT
1.1	17. Oct. 2011	New structure	L. Hechler	
2.0	26. Oct. 2011	Final version	CCT	CCT
2.1	17. Nov. 2011	Renaming of referenced guidelines	CCT	
3.0	03. Aug. 2012	Incorporated FAIR review comments	CCT	
4.1	31. Mar. 2016	Reworking and new Draft version	V.Rapp	
4.2	20. May 2016	Incorporated comments and feedback from the draft-review process	V.Rapp	
4.3	6. June 2016	Feedback from the FCWG Added Apache	V.Rapp	

		Cassandra and Kafka (Messaging)		