ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA183

STRUCTURE AND DEVELOPMENT OF SESAME'S CONTROL SYSTEM CLIENT

A. Al-Dalleh, A. Ismail, I. Saleh, SESAME, Allan, Jordan

Abstract

SESAME is a third-generation 2.5 GeV synchrotron-light source based in Allan, Jordan. The Pre-injector (Microtron) and Injector (Booster Ring) have been commissioned while the commission of the storage ring began in January 2017 and we expect machine operation in late 2017. The current components of the control systems software side are IOCs developed using EPICS software tools, Operator Interfaces (OPI) designed using Control System Studio (CSS) software tools, process variables archiving using CSS BEAUTY toolkit, alarm handling using CSS BEAST toolkit and tools to help in automation and reporting. This paper will present the current design of the client system which includes what was needed for the active commissioning period as well as upgrades that are under research including EPICS Qt framework as a client replacement for CSS and the pros and cons of this replacement and upgrading the archiver engine to a scalable and higher performance engine.

INTRODUCTION

SESAME consists of a 22 MeV Microtron, an 800 MeV Booster Synchrotron and a 2.5 GeV Storage Ring.

Control System Implementation uses (EPICS) base R3.14.12. Servers are implemented as EPICS Input/output Controllers (IOCs). Clients are implemented using a custom build of Control System Studio (CSS) based on V.3.16. CSS version 4.5 is under testing. Siemens S7 PLC controllers are used for the machine interlocks. An Allen Bradley PLC controller is used for the Personal Safety System (PSS). VME hardware is used for the timing system. Development and administration platforms use Scientific Linux 7.3 while maintaining version 6.4 for legacy support. A Git version control is used to track development. All clients, servers, and controllers are connected to an isolated machine network. There are twelve virtual servers are reserved to run the IOCs, archive system, alarm system and Git repositories.

The control systems have been implemented for the entire machine from Microtron all the way to the Storage Ring. Both the Booster and Storage Ring's control system is divided into seven subsystems: vacuum, power, RF, diagnostics, cooling, timing and Personal Safety System (PSS). Each control subsystem consists of one or more clients, servers, and controllers [1].

CLIENT SYSTEM STRUCTURE

The control system client at SESAME is divided based on the machine stages as a first level, then each stage is divided based on the stage's subsystems. Examples of the subsystem's divisions are

1. Power supplies.
2. Vacuum.
3. Diagnostics.
4. Radio-Frequency.

Figure 1 shows the main interface OPI of the control systems.

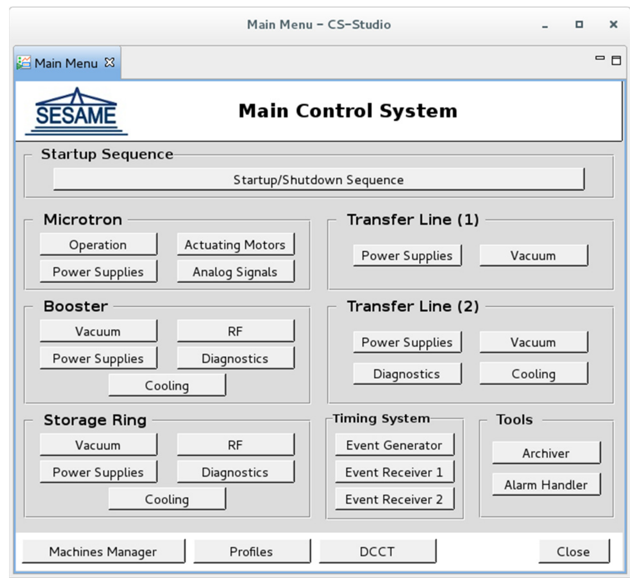

Figure 1: Main interface for the control system client.

Storage-Ring DC Power Supplies

The Storage-ring power supplies system consists of the following:

1. One power supply for bending magnet.
2. 64 power supplies for quadrupole magnets.
3. 4 power supplies for sextupoles magnets.
4. 64 power supplies for corrector magnets.
5. 8 power supplies for skew quad magnets.

With a total of 141 power supplies making it the largest GUI in the client system. Figure 2 shows the main OPI of the power supplies system.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA183

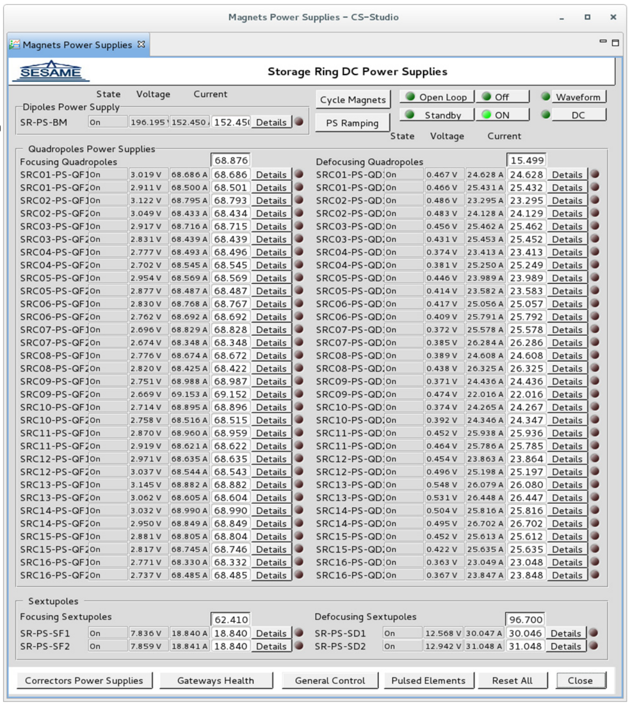

Figure 2: Storage-ring main power supplies OPI.

There are functions provided in the power supplies OPI which can be done per power supply or as a general control:

1. Power supply mode.
2. Waveform settings.
3. Monitor faults, readings and tuning adjusting.
4. Firmware upgrade.

Storage-Ring RF Control

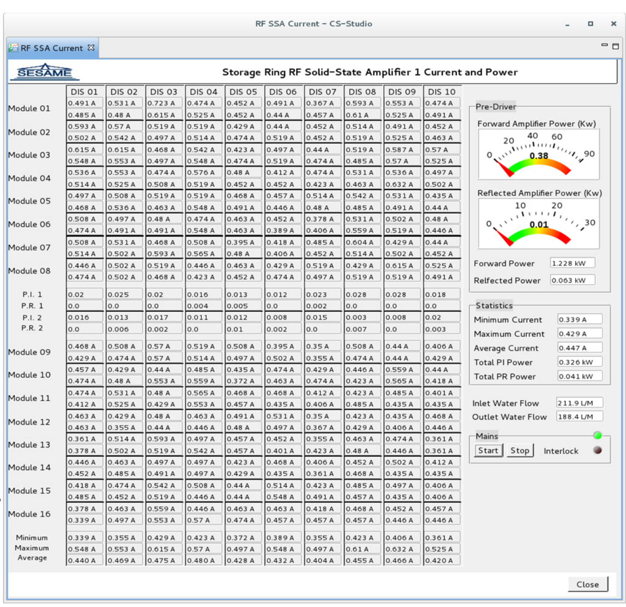

Figure 3: Storage-ring RF amplifier OPI.

In the storage ring we have 4 RF cavities, each cavity is connected to a Solid State Amplifier (SSA). Each SSA consists of 10 dissipaters reporting forward and reflected power, each one containing 16 modules, with 5 for the supervising dissipater, which report temperature and current information to the controlling dissipater. The dissipaters are being controlled through SNMP protocol. Figure 3 shows the storage ring RF amplifier OPI.

Storage-Ring Vacuum Control

The storage-ring vacuum system is divided into what is called "vacuum cell" which groups every two magnet cells together. The vacuum equipment for the storage-ring consists of 106 ion-pumps and 27 gauges. Figure 4 shows the storage ring vacuum OPI.

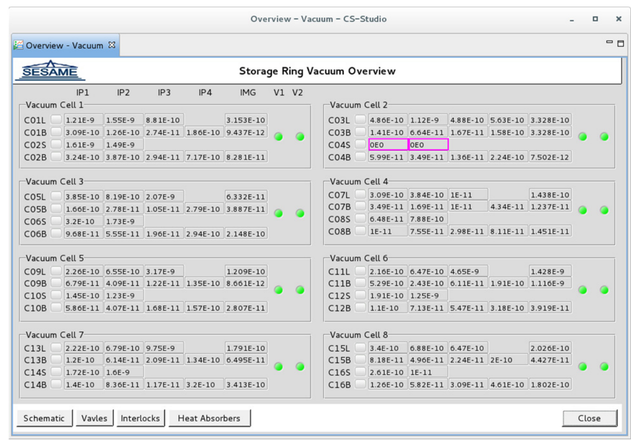

Figure 4: Storage-ring vacuum OPI.

DB-CHECK TOOL

The db-check tool is a command line tool built at SESAME that automatically analyses, validates and aids in applying continually evolving in-house rules for EPICS records databases. It was primarily built to help in reviewing, unifying and maintaining the numerous EPICS databases present, besides that it creates XML files to be imported to the archiver and alarm handler systems. All of the tool's functions are done using an input file, consisting of a custom structure, passed to the program and then the resulting output is to be used for further processing.

The tool has been re-designed from the ground up using native C++. There has been some issues installing support for the D-language along some problems in parsing parts of input files which involves applying regular expressions. In order to make it easy deploy and use the tool we decided re-implementing it again using the C++ language with the powerful C++11 standard. The new version accepts the same files format and outputs the same data format as the previous one. Instead of relying on a separate module for parsing (Pegged dictionary)[2], we used C++11 regular expression library to parse record databases and extract information using regular expressions designed for parsing records to simple data structures to be used later for processing.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA183

QT FRAMEWORK AND EPICS QT

The EPICS Qt Framework is a high-level framework built on top of EPICS to provide channel access functionality within the Qt framework which allows for faster prototyping of control systems clients [3].

Due to its popularity and very wide ecosystem and community, we decided to take a look and experiment with EPICS Qt as a possible replacement for Control System Studio. It also provides an excellent and well accepted world-wide development experience.

The decision for testing and developing with the Qt Framework and EPICS Qt as a possible replacement for the CSS came after careful consideration, testing and comparison between both CSS and Qt. The main advantage of CSS is it is easy to deploy, develop GUIs and using them. The main disadvantage of CSS is its structure which led to poor performance from time to time and not allowing for custom versions through its open source code. On the other hand the advantage of EPICS Qt is it is lightweight, easy-to-use source code, widely used development experience and better GUI looks.

Currently for developing control systems GUI with Qt, we use the latest Qt 5.9.1 along with EPICS Qt's latest release from the maintainers' official git repository hosted on github [4]. Figure 5 shows an example of the microtron Qt GUI. Figure 6 shows an example of booster's vacuum Qt GUI.

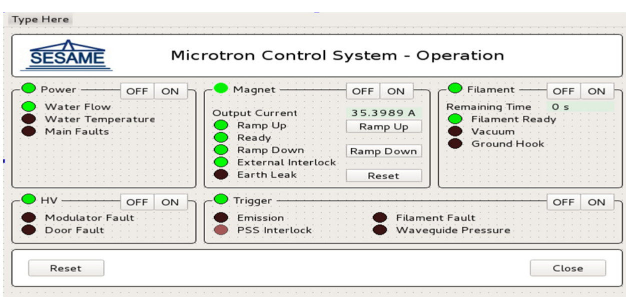

Figure 5: Microtron control system in Qt framework.

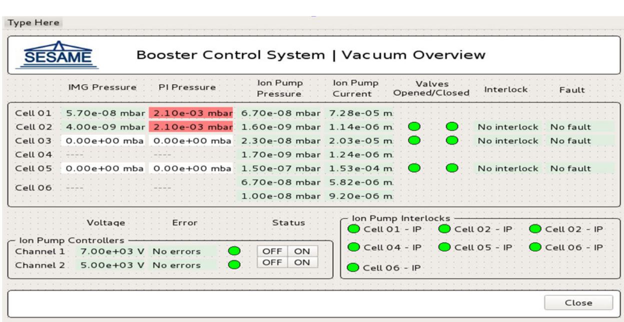

Figure 6: Booster vacuum control in Qt framework.

THE ARCHIVER APPLIANCE

The archiving system that is being used right now at SESAME is the standard CSS RDB Archiver. Initially, in the first stages of SESAME, there has not been much demand for the archiver due to low EPICS process variables implemented. Now with storage-ring is done and finalizing and operating the machine, the demand for archiving data has grown on a much larger scale than before.

The CSS RDB Archiver we use stores data in PostgreSQL database. During booster operation we had a total process variables count of around 1800 which was very reasonable to an RDB database. Now with the storage-ring in operation we have a total of more than 1200 process variables with near 1200 in archiving having weekly data rate of 23 to 50 GB a week of storage (both data and indexes). This resulted in a week performance of the CSS data browser due to caching of the data on the server all the time by the RDB Archiver itself.

The choice to such a problem is to use a clustered, scalable data storage solution. We went for the EPICS Archiver Appliance due to its better EPICS community support and easy installation and deployment. Right now we are testing a proof-of-concept installation with one server in the cluster, the testing includes installation, data insertion and retrieval, performance testing, and client deployment.

The archiver appliance aims for archiving millions of PVs with large focus on data retrieval performance. It uses 4 Tomcat webapps: Engine for data archiving, ETL for storage management, Mgmt for Web interface and retrieval for client data retrieval like CSS. Figure 7 shows the architecture of a single appliance [5].

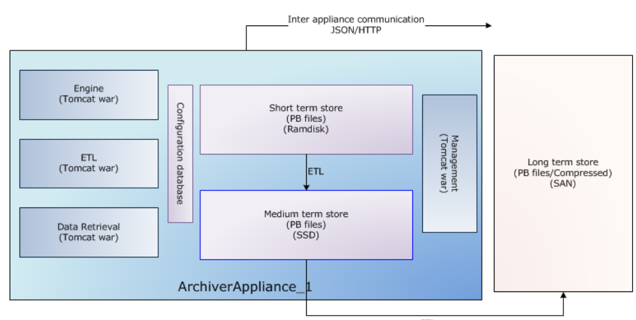

Figure 7: Architecture of a single appliance.

CONCLUSION

The control system of SESAME is based on EPICS and CSS. Development of new tools and upgrading existing systems at SESAME is important to make the control systems up to date and more consistent. Standards are used for both EPICS databases and CSS client screens.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA183

REFERENCES

[1] A. Ismail, I. Saleh, Y. Dabain, “Clients Development of Sesame's Control System Based on CSS”, *Proceedings of PCaPAC2014*, Karlsruhe, Germany, 2014.
[2] PhilippeSigaud/Pegged, https://github.com/PhilippeSigaud/Pegged/
[3] EPICS Qt Framework, https://sourceforge.net/projects/epicsqt/
[4] qtepics/qeframework, https://github.com/qtepics/qeframework/
[5] The EPICS Archiver Appliance, https://slacmshankar.github.io/epicsarchiver_docs/