CLIENTS DEVELOPMENT OF SESAME'S CONTROL SYSTEM BASED ON CSS

A. Ismail, I. Saleh, M. Mansouri, Y. Dabain, SESAME, Allan, Jordan

Abstract

SESAME is a third generation synchrotron light source under construction near Amman (Jordan). It is expected to begin operation in 2016. SESAME's injector (Microtron) and pre-injector (Booster Ring) have been commissioned. Commissioning of the storage ring is expected in 2016. The control system at SESAME is based on EPICS. EPICS IOC's are used for the servers. Control System Studio (CSS) is used for the clients. CSS BEAST alarm handler is used to identify all the critical alarms of the machine including configuration and visualization. This paper presents the architecture and design of the CSS BOY graphical user interfaces (GUIs) and CSS BEAST alarm handler for the different subsystems. It presents the standards followed in the development of SESAME's clients. SESAME will use an archiving tool based on CSS to access process variable history.

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. 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 6.4. A Git version control is used to track development and documentation. 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 Microtron, Transfer Line 1 (TL1) and Booster. The Booster'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. This paper will focus on the design and implementation of SESAME's clients based on CSS.

CUSTOMISED CSS

Control System Studio (CSS) is a combined effort of several parties, including DESY (Hamburg, Germany) and SNS (Oak Ridge, TN). It provides a collection of control system tools in a common environment, based on Eclipse [1]. Control System Studio (CSS) is designed to serve as an integration platform for engineering and operation of today's process controls as well as machine controls systems. CSS is a complete environment for the control room covering alarm management, archived data displays, diagnostic tools and last but not least operator interfaces [2].

A custom build of Control System Studio (CSS) based on V.3.16 has been implemented in SESAME (Fig. 1). Archive system and alarm handler plugins have been integrated to the CSS custom build as they were not included in the basic build. A thumbwheel function has been added to the input box widget, which provides a compact fine-tunable control to the set values and it replaces the old thumbwheel widget (Fig. 2). A digit of a set value is marked and changed by using the arrow keys of the keyboard. The custom build also includes a new feature for showing the CSS screens name on the title bar of each screen. Another feature has been added when multiple CSS windows are opened, only one instance of identical windows is allowed at a time.

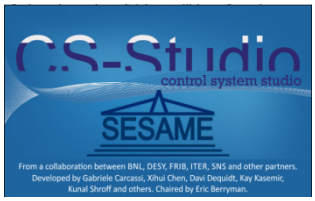

Figure 1: CSS Start Screen.

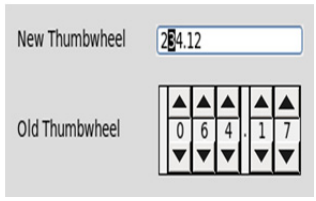

Figure 2: New Thumbwheel.

OPERATOR INTERFACES

SESAME uses CSS BOY (Best OPI, Yet). It is an OPerator Interface (OPI) development and runtime environment which enables monitoring and controlling of an EPICS system. It has a modern graphical editor and a modern web browser style runtime. It is dynamic via rules or scripts and it has comprehensive types of widgets. SESAME has two environments of CSS; one is used for development in which OPIs can be edited and modified, and the other is used for deployment which disables modification and hides all the development windows of eclipse. SESAME's client development is maintained and tracked using a version control system called Git. Git is a distributed revision control and source code management system [3]. Clients' repository contains CSS core build and CSS workspaces that contains all the developed OPIs held in main folders. SESAME OPIs are arranged and classified depending on the machine main parts; there are three main folders which contain all the required OPIs for the Microtron, TL1and Booster control systems. The booster control system is divided into seven subsystems; each one has its own sub-folder and OPIs for the control

system. The Microtron and the TL1 also have their subsystems.

SESAME's main OPI (Fig. 3) represents a launcher which can access all of the subsystems of the machine each on a separate OPI window. Each subsystem can be monitored and controlled using its specific OPI. The main OPI has different action buttons which are connected to different OPIs specified for the subsystems.

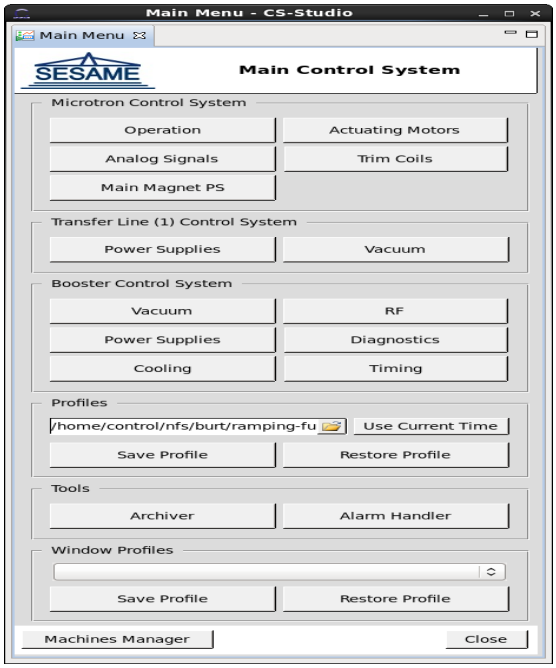

Figure 3: Main OPI.

The main OPI includes other action buttons connected to different functions such as:

- Profiles Save/Restore: Based on BURT (Back Up and Restore Tool) which is a tool for saving the current state of a list of Process Variables (PVs) at a given moment and restoring them at a later date [4]. It is regularly used by the operators.
- Window Profiles Manager: It is a python script, developed by SESAME, running under CSS which can open, move and close windows according to a previously saved windows workspace profile. A window profile contains the list of open windows and their OPI file paths, macros used in these windows and their screen positions.
- Machines Manager: It is an EPICS IOC developed by SESAME to manage different machines and IOCs. It can enable/disable IOCs, synchronize IOCs with the repository and monitor the uptime, free RAM, free space and the average load of the IOCs.
- Archive system and alarm handler can also be accessed from the main OPI.

SESAME OPI STANDARDS

SESAME started using standards for the design and the layout of different OPIs. These standards will give the operators simpler and clearer view to the OPIs. Standards are used for windows, widgets and colors. Figure 4 shows an example how these standards are applied.

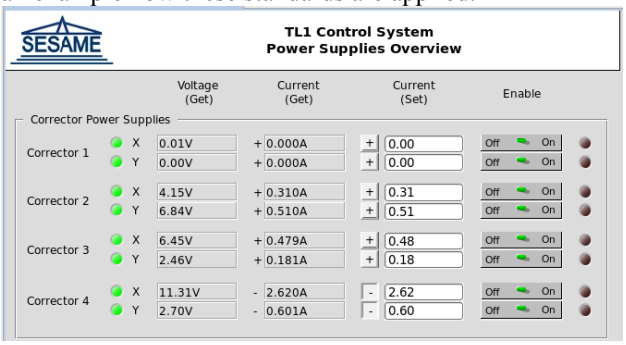

Figure 4: OPI for the TL1 Power Supplies.

Windows

- Name property must be set (which is used as a window's title).
- Same background color specified in color.def should be used.
- Header should be white color. It should contain SESAME's logo and a window's title.

Widgets

- Text Update: This widget is usually used for monitoring getter values. It should have a transparent background. Its border style should be Etched Style.
- Text Input: This widget is mainly used for setting PV values. It should have the system's default background color. Its style should be Native. Thumb Wheel Mode should be set if the value is numeric.
- LED: This widget is used to show devices on/off state, interlocks, limit switches and possibly other Boolean signals. Its color should convey the type of signal shown according to the color standards. Its size should be 20x20 pixels.
- Switch (custom widget): It is used mainly to turn devices on and off, although it can also be used for other stuff (e.g. valve open/close command).

Colors

- Color definitions are present in the file color.def within the workspace. This file contains mappings between names and colors, defined as rgb values.
- Current listing in color.def is shown in Table 1.

Table 1: SESAME Color Definitions

color.def listing				
Name	**red**	**green**	**blue**	**color**
Background	220	220	220	
Readonly Background	220	220	220	
Group Title	0	0	0	
Error Off	60	20	20	
Error On	240	5	5	
Start Off	20	60	20	
Start On	5	240	5	
Limit Switch Off	60	60	20	
Limit Switch On	240	240	5	

ALARM SYSTEM

SESAME uses BEAST (Best Ever Alarm System Toolkit). The alarm engine is installed on a virtual machine on the machine network. It monitors PVs for alarms and sends update messages to the connected clients using an Apache ActiveMQ server. The client shows alarms divided by systems and subsystems in a hierarchical way where the state of the division is equal to the state of the most severe alarm underneath it. A notable feature is that alarms stay latched onscreen even if the fault is no longer present, until they are acknowledged. In this way the operator is made aware of both transient faults and steady faults.

We are facing an issue about the choice of the alarms that we want to show on the client. If a large number of alarms are shown, the operators will likely miss the important ones. On the other hand, some low importance alarms may be the key to figuring out a problem in the operation. A balance is still to be found, however, one of the ideas to solve this issue is to show and hide different alarms at different states of the operation. In other words, when the machine is off, for a simple example, some alarms may be normal and hidden.

ARCHIVING SYSTEM

The archiving system is an important part of the control system in any accelerator. SESAME has integrated the BEAUTY (Best Ever Archive Toolset Yet) archiving system. The system is divided into two parts: the archive engine and the data browser. The archive engine is installed on a virtual machine and setup to log the changes of approximately all PVs to a local PostgreSQL database. The value, alarm state and severity are recorded on each change. The data browser, which is the client side, is integrated into CSS as a plugin. It can be used to search for PVs and to plot them on a graph in any specific time interval. Figure 5 shows the architecture of the archiving system with the data browser showing some vacuum readings.

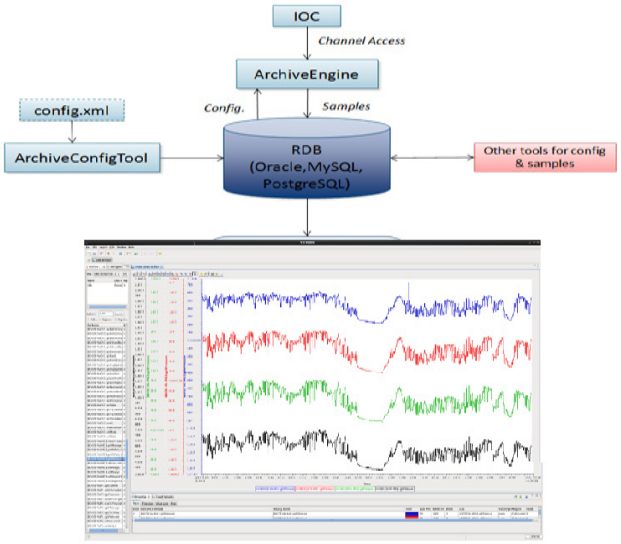

Figure 5: Archiving System Architecture.

CAMERA DRIVER AND CLIENT

One of the in-house developments on the client side is a camera epics client. Currently, SESAME has two Basler acA1300-30gm GigE cameras installed in the booster ring. A driver was developed to interface the Basler cameras with EPICS. The camera client uses the EPICS channel access libraries to monitor and control the various records provided by the driver. A waveform record contains the grayscale image data and various analog input/output records provide access to camera's gain, exposure, ROI and trigger source. A feature that was requested by the operators to ease visual inspection of the beam is the ability to switch between different color maps. The camera client provides the normal grayscale color map and a hot-cold colour map. The client was developed using C, EPICS CA libraries for connecting with EPICS, OpenGL for image rendering, SDL for window creation and input handling and AntTweakBar for controlling the camera's various parameters.

CONCLUSION

SESAME uses a custom build of Control System Studio (CSS) based on V.3.16. Archive system and alarm handler plugins have been integrated to the CSS. CSS adds a complete environment for the control room covering alarm management, archived data displays and operator interfaces. Work will continue to provide compact simple OPI screens for the operators.

ACKNOWLEDGMENT

We would like to thank Pascale Betinelli (SOLEIL), Mark Heron (DLS) and Igor Kriznar (ANKA) for their support. We would like to thank also our technical director Erhard Huttel (SESAME) for his support.

REFERENCES

[1] DESY: http://css.desy.de/
[2] M. Clausen, J. Hatje, J. Penning, “THE CSS STORY”, Proceedings of PCaPAC2012, Kolkata, India, 2012.
[3] http://en.wikipedia.org/wiki/Git_(software)
[4] http://aps.anl.gov/epics/extensions/burt/index.php