Upgrade of Hardware Controls for the STAR Experiment at RHIC

Jaroslav Adam\textsuperscript{a,1,}\textsuperscript{*}, Michael G. Cherney\textsuperscript{a}, Joey D’Alesio\textsuperscript{a}, Emma Dufresne\textsuperscript{a}, Lukáš Holub\textsuperscript{b}, Janet E. Seger\textsuperscript{a}, David Thustý\textsuperscript{a}

\textsuperscript{a}Creighton University, Omaha, United States
\textsuperscript{b}Czech Technical University in Prague, FNSPE, Prague, Czech Republic

Abstract

The STAR experiment has been delivering significant physics results for more than 20 years. Stable operation of the experiment was achieved by using a robust controls system based on the Experimental Physics and Industrial Control System (EPICS). Now an object-oriented approach with Python libraries, adapted for EPICS software, is going to replace the procedural-based EPICS C libraries previously used at STAR. Advantages of the new approach include stability of operation, code reduction and straightforward project documentation. The first two sections of this paper introduce the STAR experiment, give an overview of the EPICS architecture, and present the use of Python for controls software. Specific examples, as well as upgrades of user interfaces, are outlined in the following sections.

Keywords: EPICS, Python, PyEpics, PythonSoftIoc

1. Introduction

The Solenoidal Tracker at RHIC (STAR) experiment \cite{1} at the Relativistic Heavy Ion Collider (RHIC) was designed to study strongly interacting matter at the highest energy densities. It is equipped for precise tracking and identification of charged particles, electromagnetic calorimetry at central angles,
measurements of charged particle yields at forward and backward angles, and neutron calorimetry at very forward and backward directions \[2\].

Fig. 1 shows the components of the STAR experiment. Central tracking, particle identification, and electromagnetic calorimetry are provided by a large gaseous Time Projection Chamber (TPC), Time-Of-Flight measurement (TOF) and by the Barrel Electromagnetic Calorimeter (BEMC). Forward charged particles are detected by several scintillator detectors: the Event Plane Detector (EPD), the Beam-Beam Counter (BBC) and the Vertex Position Detector (VPD). Very forward neutrons are measured by the Zero Degree Calorimeters (ZDC).

As the experiment is located in a high radiation area, remote control for all the hardware was developed along with design of the experiment \[3, 4, 5\]. It is necessary to control and monitor over 50000 parameters of the running experiment. These include voltages and currents on detection and readout elements, temperatures, power supply parameters, magnetic fields and environmental pa-
The detector control system is based on the Experimental Physics and Industrial Control System (EPICS) [6]. EPICS provides the means to control and monitor the detector parameters via Process Variable (PV) objects that represent the physical detector parameters in question. The PV objects are managed by the Input-Output Controllers (IOC), which act as servers in the distributed EPICS system. The PVs can be accessed by a large variety of client applications. Communication between the IOCs and clients is performed over a local ethernet network using the Channel Access (CA) protocol. All interactions with physical hardware are provided solely by the IOCs. Fig. 2 gives an overview of the EPICS architecture: several IOCs may be connected via the CA protocol to various applications such as graphical user interfaces, archiving databases or alarm handlers.

At STAR, the original IOCs were designed using EPICS 3.12, running on Motorola MVME147 and MVME167 single-board processors with the VxWorks 5.2 operating system. During the experiment’s evolution, some of the VxWorks based IOCs were replaced by EPICS 3.14 software IOCs, running on standard PCs under Scientific Linux. The software IOCs were built upon procedural-based EPICS libraries specific to incorporate communication with hardware and to manage the process variables.

---

**Figure 2:** Overview of EPICS architecture. The Channel Access protocol provides communication between servers (IOC) and client applications.
2. Python based controls software

EPICS has recently developed tools that are used to create and manage software IOCs entirely in Python [7, 8]. The main advantage is a purely object-oriented approach to IOC design which, together with the relative simplicity of the Python syntax, results in a well-structured and easily maintainable code. Python supports all standard communication protocols with hardware, which reduces the need for a specific hardware support. The use of such standard libraries contributes to further code reduction. Fig. 3 gives an outline of how the Python environment can interface with both the EPICS control system and several hardware control libraries.

As the IOCs now consist of a set of object-oriented Python programs only, it is straightforward to keep and document them by means of the Git version control system [9]. The STAR experiment has created an official Github repository [10] for detector controls, maintaining all the Python-based IOCs.

3. IOC implementations with Python

3.1. High voltage for trigger detectors

The BBC, VPD and ZDC detectors require high voltage for the Photomultiplier tubes (PMTs) that convert optical scintillation responses to electrical
signals. The voltages to individual PMTs are provided by a LeCroy 1440 high voltage system. The communication is done using the RS232 serial protocol.

The original VxWorks IOC code at STAR had a design flaw, which required the experiment operating crew to restart that IOC on a daily basis. Therefore, it was decided to build a new Python IOC \cite{11} using the pyserial module \cite{12} for RS232 communication. The new IOC mimics all the capabilities of the old one, namely the ability to load all demand voltages from a set of text files. In addition, it allows the user to set several voltage profiles for the detector to ensure a straightforward means of calibration for the detectors. Robust operation of the new system was achieved during the 2019 data taking period.

3.2. Gas monitoring

The central gaseous detectors, the TPC, TOF and Muon Telescope Detector (MTD), require the monitoring of pressure, flow and temperature of the working gases and their mixing parameters. The measuring hardware transports the data over an isolated network to one of the detector control PCs where it is written to a set of local ASCII files.

The original EPICS 3.14 IOC used a set of elementary C functions to access these data. However, it was not able to recognize a connection loss on the side of the gas monitoring hardware. The problem was overcome by employing a Python based IOC \cite{13}, which uses the watchdog.observers module \cite{14} to automatically recognize when new data arrive. In the case that no new data arrive during a predefined time interval, the appropriate alarm levels are set to gas-related PVs, thus notifying the operating crews of the problem.

3.3. EPD integration in the alarm system

The Event Plane Detector (EPD) was installed to precisely measure the geometrical plane in which each heavy ion collision occurs. The EPD group chose the MQTT protocol \cite{15} for control. During commissioning, the request was made to include the EPD in the standard STAR alarm system.
The set of parameters monitored for conditions outside of the normal operating range are voltages, currents and temperatures on individual detector elements. In order to make these available to the alarm system, it was necessary to introduce a one-to-one mapping from the MQTT variables to the EPICS PV objects. The task was solved by developing a Python IOC [16], holding the PVs according to the EPD naming convention, and retrieving the MQTT variables via the paho.mqtt.client module [17]. Effectively, the IOC acts as a MQTT to Channel Access bridge.

3.4. Grid leak wall for iTPC upgrade

The inner sectors of the TPC (iTPC) were upgraded for a more precise readout, allowing the detection of tracks at previously inaccessible kinematic intervals.

As a result of electron avalanches near the TPC sense wires, large numbers of positive ions are created. The ions are kept from entering the TPC drift volume by a set of dedicated grid wires. When introducing the iTPC upgrade, it was necessary to also block the ions from moving between adjacent readout sectors. This was achieved by placing electrodes at a negative high voltage between the TPC inner and outer sectors. The setup is commonly referred to as Grid Leak Wall Suppression. The voltages are provided by the ISEG modules in a Wiener control assembly. Communication to the modules is provided by the Wiener MPOD controller via the SNMP protocol.

The task here was to develop the entire IOC. The Python based IOC was created [18], utilizing the standard SNMP command line tools: snmpwalk and snmpset. Access to the SNMP tools from Python is provided via the subprocess standard Python module. The solution was very simple to implement and proved its robustness during the experiment run in 2019.

3.5. Air conditioning for TPC and eTOF

The gaseous TPC and endcap TOF (eTOF) detectors are held at pressures slightly higher than atmospheric pressure to avoid air leaking into the detectors.
In addition, constant air temperature and humidity and steady air flow are necessary for the stable operation of both the TPC and the eTOF. Two new high volume air conditioning units were put in place to provide stable air flow around the TPC and eTOF detectors. The units communicate the air flow, temperature and humidity values using the ModbusRTU protocol [19]. A TCP to Modbus converter is attached to each unit as a convenient way to provide remote access to the internal RS485 line.

A Python-based IOC is set to read the air parameters and report them via their PVs. Communication to the units is done by the pyModbusTCP module [20]. The IOC is capable of also turning the units on or off by issuing Modbus signals to the internal power relay.

3.6. eTOF Low Voltage Power Supplies Monitoring

A group from the CBM collaboration [21] installed the eTOF system at STAR with already existing EPICS PVs for monitoring currents and voltages of the custom made front-end electronics boards. However, the values of these PVs contain raw ADC values only. Moreover, as currents are measured on these boards by Hall probes, the STAR magnetic field affects these values. The magnetic fields from relays on these boards can also affect the measured current. Therefore, new PVs have been defined using a Python-based IOC and the numpy package [22]. These new PVs are assigned with currents and voltages calculated from the raw ADC values of the pre-defined CBM PVs. Additionally, the known errors in the current measurements caused by magnetic fields can be corrected in these new PVs. The new PVs are updated based on callback functions on the value change which guarantees the efficient use of computing resources.

3.7. STAR upgrade plan

An ambitious upgrade is planned for STAR to enhance tracking and calorimetry in the forward region [23]. Tracking in the forward region will be provided by the small-strip Thin Gap Chamber (sTGC), the Forward Silicon Tracker, and electromagnetic and hadron calorimetry. A prototype for the sTGC was
installed during the 2019 run period. Readout of the sTGC requires a positive high voltage for charge multiplication in a gas (as does the TPC). The voltage to the prototype was provided by a set of unused channels in the CAEN SY1527 used for the TOF. The IOC was written in Python \[24\], employing the ctypes.cdll Python module to communicate with the CAEN via the vendor specific .so library.

4. User Interfaces

4.1. Slow Controls Monitoring Webpage

The slow controls monitoring webpage, shown in Fig. 4, allows detector experts and run coordinators outside the control room to monitor information about the detectors during the run. For a quick glance at the major experiment operations, this webpage provides information on the run performance and can alert users in case of an operational disruption. For example, information about water and gas alarms, the operating status of the sub-detectors, and environmental conditions can be monitored. Weather values presented in Fig. 4 are neither red nor green, as their values do not deter operational validity. They are still important to monitor for potential hazards to the experiment’s mechanics. New code was written using PyEpics to gather the useful information and fill the webpage. With this update, outdated values were removed and the use of a legacy unstable software was eliminated. With an object-oriented approach in PyEpics, calls to the Channel Access are minimal, allowing for a quick response time.

4.2. Upgrade of Graphical User Interface

The Graphical User Interface (GUI) in EPICS is a Channel Access application, aimed at visualizing and/or editing values of the PVs.

The Motif Editor and Display Manager (MEDM) has been in use at STAR since the beginning of its operation. Recently, the support for the motif graphical libraries, upon which the MEDM is built, was discontinued. The situation
called for replacement of the MEDM. caQtDM [25] was found to be a suitable replacement for MEDM. It is based upon the well-supported Qt graphical libraries, giving the GUIs a fresh modern look. An example of the GUI created with caQtDM, for the case of the air conditioning for the TPC and eTOF (Section 3.5), is shown in Fig. 5.

An issue might arise when a caQtDM GUI is accessed on a remote machine with X11 forwarding via ssh. The higher resolution in caQtDM, compared to simpler shapes in MEDM, requires a larger amount of data to be transmit-

![Figure 5: An example of a GUI created with caQtDM. The GUI shows the operation of the air conditioning for the TPC and eTOF.](image-url)
Another approach to controls for experts is based on a text interface rather than a GUI. The npyscreen python module \[26\] makes it possible to create text mode monitoring and controls objects. When coupled together with PyEpics to manage the Channel Access, it is straightforward to create an interactive text mode semi-graphical interface, having all the capabilities of a standard graphical GUI. An example is shown in Fig. 6. The user can monitor and/or set several alarm levels related to the operation of the EPD detector (Section 3.3).

5. Conclusions

The STAR experiment has been run for 20 years. Further operation is planned, including an ambitious detector upgrade to provide additional tracking and calorimetry in the forward region. This would not be possible without a robust control system that is continuously maintained. Use of an object-oriented Python approach for EPICS software has proven to be a convenient solution to challenges in the experiment control. The new approach has been successfully deployed to enhance the existing control components, and is widely accepted at STAR for all intended detector upgrades.
CRediT authorship contribution statement

**J. Adam:** Conception and design of the work, Software development, Writing - original draft. **M. Cherney:** Writing - review & editing, Conception. **J. D’Alesio:** Contribution to software, Writing - original draft. **E. Dufresne:** Contribution to software, Writing - original draft. **L. Holub:** Contribution to software, Writing - original draft. **J. Seger:** Supervision, Writing - review & editing, Conception. **D. Tlustý:** Contribution to software, Writing - original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors thank Jiro Fujita for useful discussions and help.

This work was supported in part by the Office of Nuclear Physics within the US DOE Office of Science.

References

References

[1] S. K. Radhakrishnan, Highlights from the STAR experiment, PoS Hard- Probes2018 (2019) 034. doi:10.22323/1.345.0034

[2] K. H. Ackermann, et al., STAR detector overview, Nucl. Instrum. Meth. A499 (2003) 624–632. doi:10.1016/S0168-9002(02)01960-5

[3] D. Reichhold, et al., Hardware controls for the STAR experiment at RHIC, Nucl. Instrum. Meth. A499 (2003) 792–801. doi:10.1016/S0168-9002(02)01976-9
[4] J. Lin, et al., Hardware controls for the STAR experiment at RHIC, IEEE Trans. Nucl. Sci. 47 (2000) 210–213. doi:10.1109/23.846148

[5] J. Gross, T. S. McShane, M. Cherney, A unified control system for the STAR experiment, IEEE Trans. Nucl. Sci. 41 (1994) 184–187. doi:10.1109/23.281486

[6] The EPICS collaboration, Experimental Physics and Industrial Control System (2021).
URL https://epics-controls.org/

[7] EPICS IOC with Python Interpreter Linked in (2021).
URL http://controls.diamond.ac.uk/downloads/python/pythonSoftIoc/

[8] pyepics: Python interface to Epics Channel Access (2021).
URL https://github.com/pyepics/pyepics

[9] S. Chacon, B. Straub, Pro Git (2014).
URL https://git-scm.com/book/en/v2

[10] STAR Slow Controls repository (2021).
URL https://github.com/star-controls

[11] STAR Lecroy 1440 control system (2021).
URL https://github.com/star-controls/BBClecroy

[12] Python Serial Port Extension (2021).
URL https://pypi.org/project/pyserial/

[13] STAR Gas monitoring system (2021).
URL https://github.com/star-controls/gasread

[14] Filesystem events monitoring (2021).
URL https://pypi.org/project/watchdog/
[15] MQTT: The Standard for IoT Messaging (2021).
URL http://mqtt.org/

[16] STAR Alarm monitoring for EPD (2021).
URL https://github.com/star-controls/epdAlarm

[17] Python Client for MQTT (2021).
URL https://www.eclipse.org/paho/clients/python/

[18] STAR iTPC Grid Leak system (2021).
URL https://github.com/star-controls/tpc-grid-leak

[19] Modbus-IDA, Modbus application protocol specification v1. 1a, North Grafton, Massachusetts (www.modbus.org/specs.php) (2004).

[20] A simple Modbus/TCP library for Python (2021).
URL https://github.com/sourceperl/pyModbusTCP

[21] P. Senger, Status of the Compressed Baryonic Matter experiment at FAIR, Int. J. Mod. Phys. E 29 (2020) 2030001. arXiv:2004.11762, doi:10.1142/S0218301320300015.

[22] Scientific computing with Python (2021).
URL https://numpy.org/

[23] Q. Yang, The STAR BES-II and Forward Rapidity Physics and Upgrades, Nucl. Phys. A982 (2019) 951–954. doi:10.1016/j.nuclphysa.2018.10.029.

[24] STAR sTGC high voltage system (2021).
URL https://github.com/star-controls/sTGC_HV

[25] A. C. Mezger, caQtDM, an EPICS Display Manager Based on Qt, Proc. ICALEPCS’13 San Francisco, CA, USA, Oct. 2013, paper TUPPC121 (2013) pp. 864–866.

[26] A framework for developing console applications (2021).
URL https://pypi.org/project/npyscreen/