ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-THAPP04

EPICS TOOLS FOR SMALL EXPERIMENT BASED ON PLC

K. Saintin†, P. Lotrus††, Q. Bertrand, G.A. Durand, F. Gohier, T.J. Joannem, N. Solenne
IRFU, CEA, Université Paris-Saclay, Gif-sur-Yvette, France

Abstract

Irfu [1] control software team is involved from feasibility studies to equipment deployment into many different experiments in their size and running time. For many years, Irfu is using PLC solution for controlling part of the experiment, and two different SCADA:

- MUSCADE®, an in-house SCADA dedicated to small experiments.
- EPICS [2] for bigger facilities.

With MUSCADE®, Irfu has developed a set of tools that gives an easy and a fast way for PLC developers to configure the SCADA. As EPICS projects are growing in our department, we are working now on adapting those tools to EPICS:

- PLCPARSER generates an EPICS database for PLC communication (S7PLC, Modbus).
- CAFEJava (Channel Access For Epics Java) API runs a simulated EPICS IOC to test EPICS synoptic, and provides EPICS process variables access for any Java application.
- DXF2OPI converts AutoCAD DXF files into OPI files for CSS [3] software.
- MOONARCH (Memory Optimizer ON ARCHiver Appliance) reduces EPICS Archiver Appliance [4] data files storage.

INTRODUCTION

Irfu is working on different kind of experiments from particle accelerator installed in big facilities to gas station installed in the middle of the fields. Irfu participates regularly to the construction of accelerator around the world and its control system team has developed EPICS skills for more than 20 years. EPICS and associated software are mainly designed for big facilities, which is not convenient for some of our experiment; especially small ones based on PLC control (see Fig. 1). Therefore, we have developed a set of software that will help our PLC developers to work with EPICS as SCADA, and on adapting EPICS tools to tiny structure.

To help PLC developers, we are working on generating EPICS communication from TIA Portal data blocks description, on a multi-platform IOC (Input Output Controller) for testing the communication, and on automating GUI (Graphical User Interface) generation from AutoCAD drawing. To adapt EPICS for tiny experiments that are using only one computer, we have developed a tool that will decimate or remove archiving data. The idea is to provide tools that gives enough autonomy to non-EPICS specialist for configuring and deploying an EPICS SCADA on one PC.

† katy.saintin@cea.fr
†† paul.lotrus@cea.fr

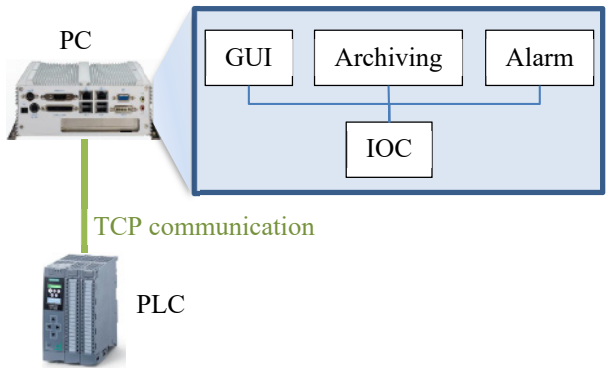

Figure 1: Small experiment architecture based on PLC.

PLCPARSER

At Irfu, a typical hardware configuration is a Siemens PLC and a PC with Linux and EPICS as SCADA. A recurrent development task is the communication between the two devices. There are several ways to automate this development; we have chosen to start from the PLC development code. The main reason is that PLC developers are implementing quickly the physicist requests and therefore PLC code is the core of the machine behavior, so it is natural to generate the communication from this reference. The principle is to read the PLC's memory spaces involve in the communication with EPICS, and then generate the IOC code. To do that, we are using an export feature of TIA Portal, the PLC developer can export each memory block in a text file format (Fig. 2). For Modbus/TCP or S7PLC (PSI) communication, we are exporting data-block in awl file format, which gives few information like the memory position, the name, the type and a description.

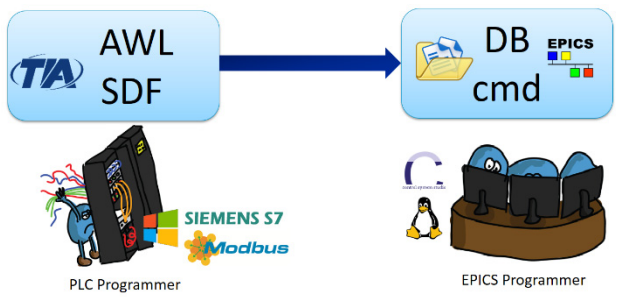

Figure 2: PLCPARSER tool principle.

The next step is to use PLCPARSER application, which parses all the awl files contained in a folder. Then, the user configures the communication parameters like IP address and adds information like alarm, physical units, etc... Once PLCPARSER is generating IOC files, the user can copy those files into the dedicated EPICS top folder (See Fig. 3).

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-THAPP04

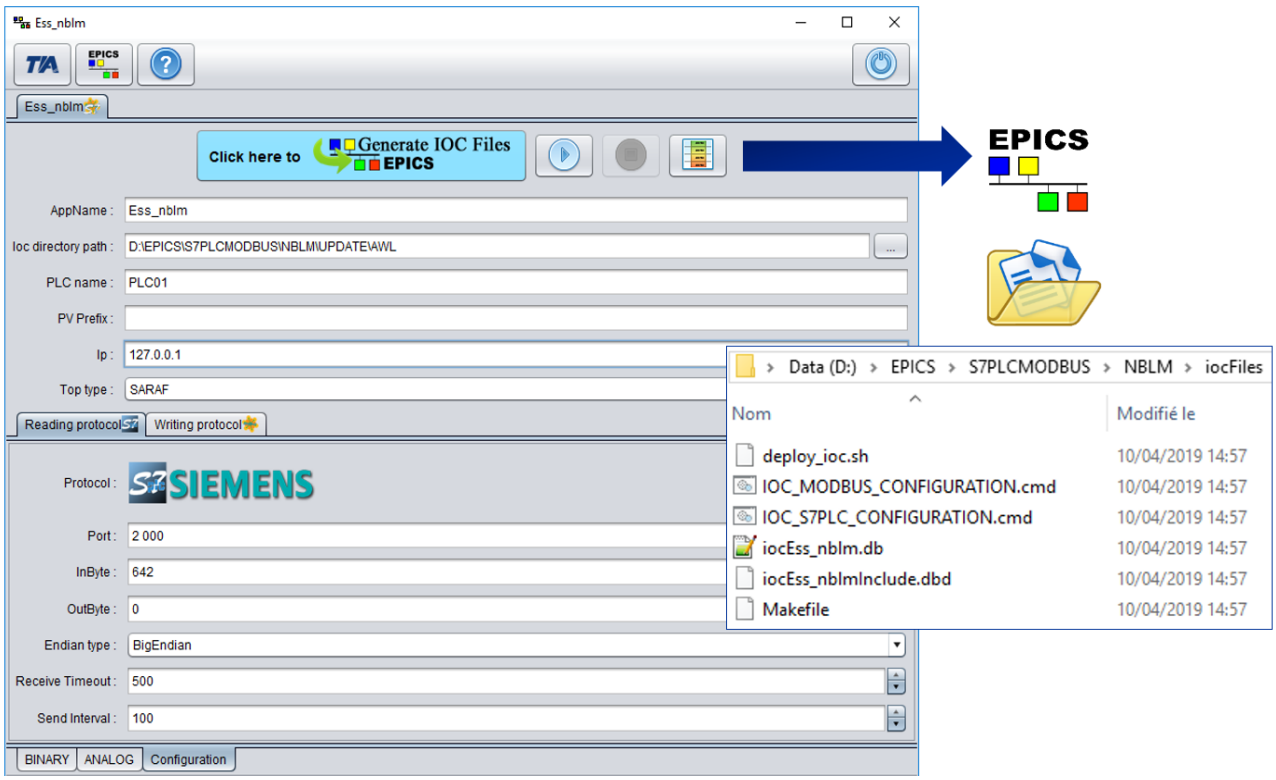

Figure 3: PLCPARSER tool configuration panel.

Since programming is a recursive process, the PLC developer will have to modify the communication. The application is able to manage merging of new data, deleted or modified. Data are displayed in two different tabbed panels, binary panel and analog panel. There are important EPICS fields that are created by default (SCAN, OSV) by the application, but it is also possible to add or remove fields.

This software is fully developed in Java and Swing library, so it can run on many OS. Because Eclipse SWT frame can embed Swing library, PLCPARSER tool is provided as a standalone tool (see Fig. 4) or as a CSS Eclipse RCP plugin (see Fig. 5). Finally, the tool has an interface to Phoebus [5] application (see Fig. 6), because Swing can also be embedded as a JavaFX frame.

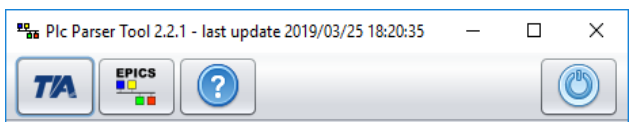

Figure 4: PLCPARSER tool as a standalone tool.

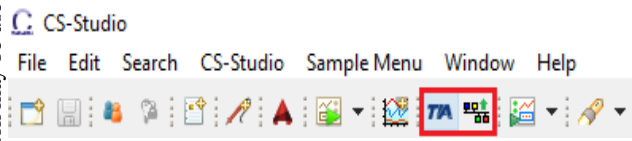

Figure 5: PLCPARSER tool as a CSS plugin.

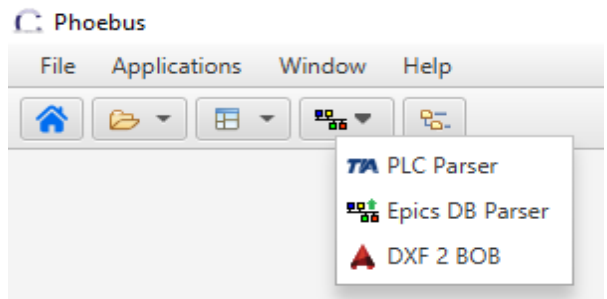

Figure 6: PLCPARSER Tool as a Phoebus module.

DXF2OPI

At Irfu, part of the team is using AutoCAD for 2D drawings, especially for cryogenic sequential function chart (SFC or GRAFCET), which can be tricky (see Fig. 7). AutoCAD is a vector drawing solution, which is convenient for precise drawing, and so for complex synoptic. It is also a popular software, especially in the industry, by architects, project managers, engineers, graphic designers, city planners and other professionals.

Furthermore, Irfu PLC programmers are used to using the custom SCADA Muscade® [6], developed in Java. This software is able to import AutoCAD file for generating synoptic views, and is easy to use.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-THAPP04

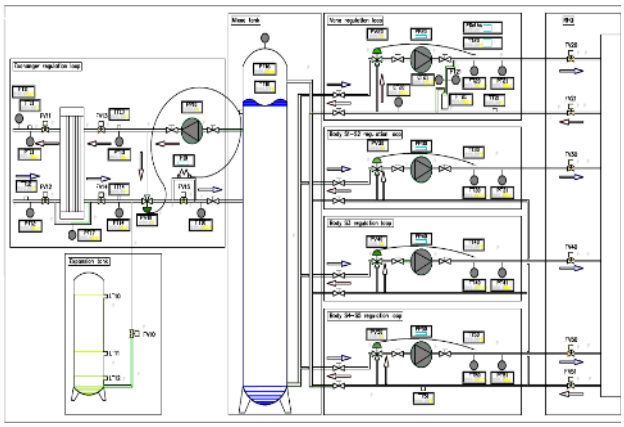

Figure 7: Synoptic design with AutoCAD.

To avoid to hand crafted twice a GUI in AutoCAD and in CSS software; we have developed a software, which is parsing AutoCAD file and generating a view in CSS application. Like PLCPARSER software, we are using a functionality of AutoCAD to export a drawing to a text file format (with the extension dxf). DXF2OPI is reading and parsing the dxf file, and it generates a CSS opi file (see Fig. 8).

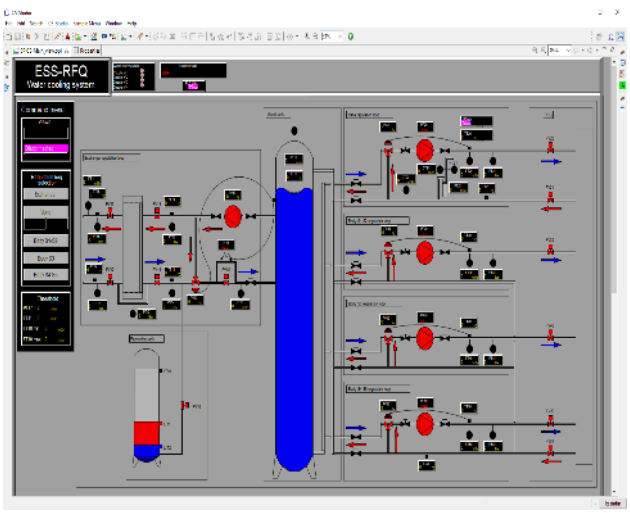

Figure 8: Opi CSS synoptic generated from AutoCAD.

To be able to associate several drawing elements with an EPICS PV (Process Variable), we are linking one or several drawing elements with the same AutoCAD "color" to an EPICS PV by its name (see Fig. 9). For that, we have created an AutoCAD block called PALET that we insert into the AutoCAD drawing. The PALET block has several properties. The SYMBOLE property value is the PV name that we want to associate, and the COULEUR property value is the color value (integer) of the drawing elements that we want to associate to the PV.

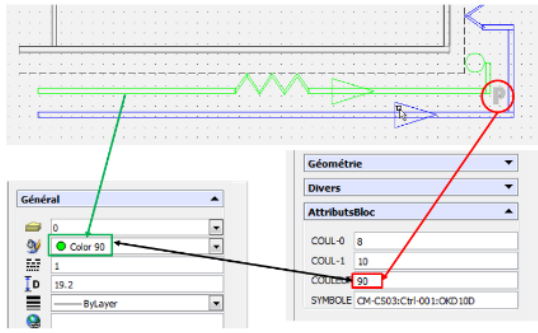

Figure 9: AutoCAD blocks association.

For Boolean PV, we need to change the color according to the value of the PV. In PALET block, the two other fields for a Boolean PV are COUL-0 and COUL-1. If the Boolean PV value is at 0, all the drawing elements associated to the PV will have the color defined by COUL-0. The color is between 0 and 16 (0=black, 1=blue, etc.).

Many widgets have been implemented, and we have successfully used this solution for several projects already. The software is fully developed in Java and Swing library, so it can run on many OS.

Finally, a new Phoebus application module DXF2BOB is in progress, it will be possible to generate a Phoebus BOB file starting from an AutoCAD file.

CAFE JAVA

With PLCPARSER, we are able to generate IOC files of the communication between a PLC and an EPICS PC and with DXF2OPI, we are able to generate an OPI synoptic starting from an AutoCAD file. The two next steps to facilitate the EPICS IOC development by a PLC programmer are:

- To test a communication quickly without any EPICS clients tools (CSS OPI or Channel Access command client) available on the local computer,
- To test a GUI behavior without a running IOC connected to a real PLC.

Using those two features enables to fix bugs and GUI behavior before deployment. The final solution is more reliable and only technical problems linked to the other parts of the system like network and instrumentation needs to been solved.

To make it simple for the user and have a solution that is working in several OS, we have developed a Java library based on CAJ [7] library (Channel Access for Java) called CAFE Java **C**hannel **A**ccess **F**or **E**PICS that provided those main functionalities:

- Reading and writing an EPICS process variable
- Running a simulated IOC server that contains the process variables defined in EPICS database file.

CAFE library is integrated into PLCPARSER tools for:

- Providing a convenient control panel, generated dynamically by loading an EPICS database file (see Fig. 10).
- Running a simulated IOC in order to test a GUI without the hardware (see Fig. 11).

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-THAPP04

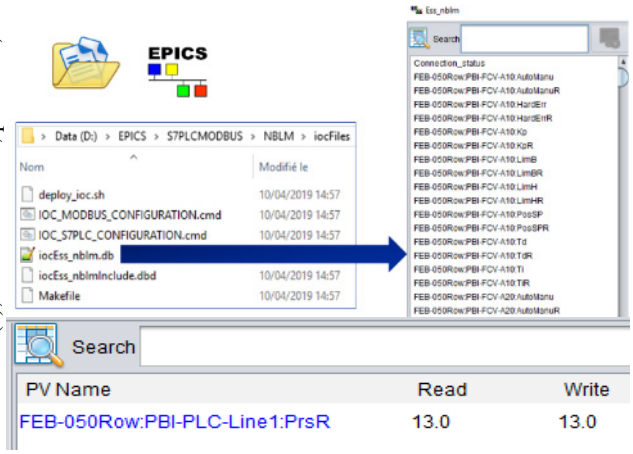

Figure 10: PV list control panel.

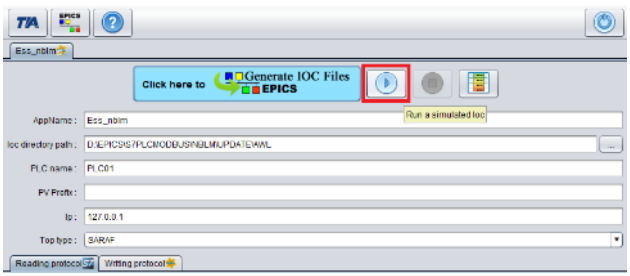

Figure 11: Run a simulated Java IOC.

MOONARCH

For small experiments, we are using PC that could run 10 years with low maintenance. Since mobile mechanical parts of a PC are the weak point, we have successfully been using industrial fanless PCs for many years but this solution also implies low resources and performance. Therefore, we need to tune the EPICS installation, especially the archiving to fit it into the limitation in term of RAM and disk space. Currently, we are using Archive Appliance as archiving software, which is a great solution but design for large space disk. To answer our need, we have developed a Java application that can reduce the number of data archived.

We are using two optimization methods for limiting the archived data size. The first one is to remove data that are not interesting. In many experiments with PLC, data collected by the PLC does not need a fast acquisition, but the programmer of the PLC is interested in data analysis when a problem occurs for understanding how the process implemented in the PLC reacted. In that case, he needs the maximum of details; otherwise, a low frequency is enough. Based on this principle, MOONARCH (Memory Optimizer ON ARCHiver appliance) application will read new data of the last day, check when alarms occurred, defined temporal windows around them, keep all data in those windows, and decimate data that are under a defined period like 10 seconds outside those windows (see Fig. 12).

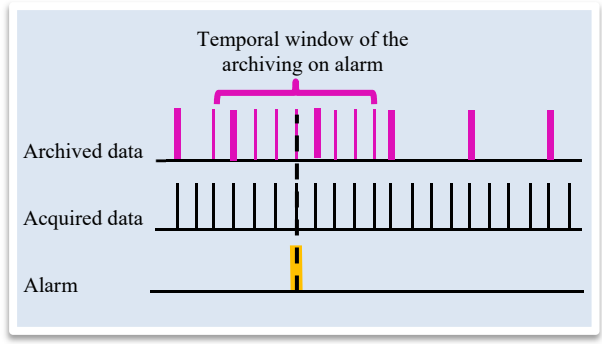

Figure 12: MOONARCH remove data principle.

The second optimization is to define a maximum size of archiving data on the disk or a duration (for LTS data). Typically, MOONARCH will check the total archiving files space, and if the size is above a limit configured by the user, it will remove the oldest data of each PV archiving file.

MOONARCH directly read the archiving files generated by Archiving Appliance (Google protobuff pb files) stored in long term memory storage (LTS). Therefore, it can be installed on any computer that can access to the LTS folder, and it does not require EPICS to work. It is developed in Java and it is deployed as a servlet in a Tomcat container following the installation logic of Archiving Appliance (see Fig. 13).

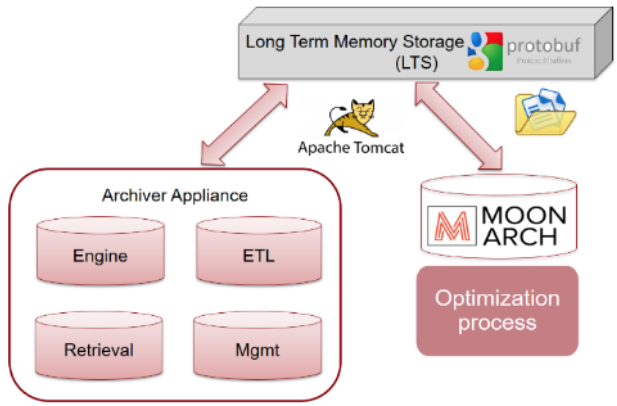

Figure 13: MOONARCH software architecture.

Finally, MOONARCH provides a GUI through a web interface for an easy access to the configuration (see Fig. 14).

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-THAPP04

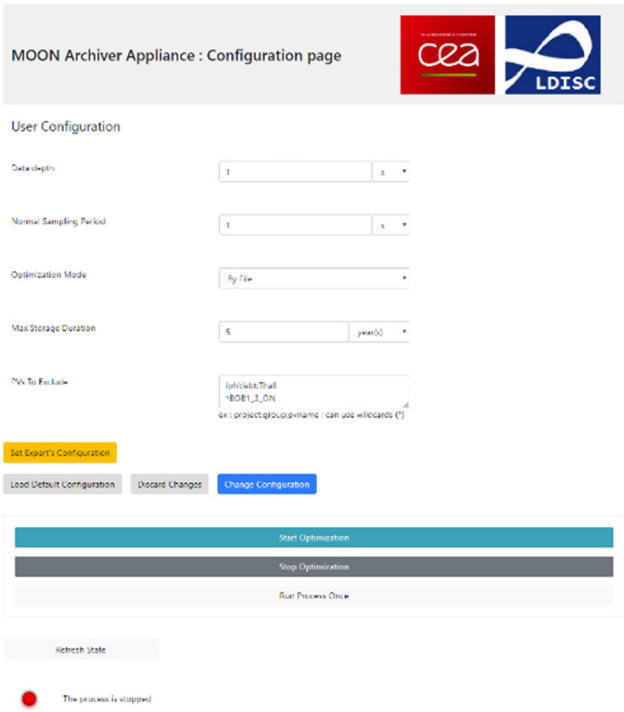

Figure 14: MOONARCH configuration GUI.

In the present day, the module has been tested with GBAR experiment (**G**ravitational **B**ehaviour of **A**ntihydrogen at **R**est) archiving data copy. Last year, GBAR experiment generated about one Giga bytes of data. After the optimization processed by MOONARCH, the storage took 800 Megabytes of data, that represents a reduction of 40 % of the memory size. It is planned to deploy the solution on experiment that are limited in memory size next year.

CONCLUSION

We have now a set of tools for helping non-EPICS user to create communication with Siemens PLC, create views in CSS and control size of the archived data. We are currently using those tools on accelerator projects, and the next step is to deploy them on a small experiment with only one PC and one PLC. This has been done in the laboratory during tests this year, and we expect to deliver this solution in 2020 as part of a deployed project.

ACKNOWLEDGEMENTS

The authors would like to especially thank Gilles Durand and Christian Walter for their help and expertise.

REFERENCES

[1] Irfu, htt://irfu.cea.fr

[2] EPICS, ttp://epics-conrtol.org

[3] CSS, htp://controlsystemstudio.org

[4] M. V. Sankar, L. F. Li, M. A. Davidsaver, and M. G. Konrad, "The EPICS Archiver Appliance", in *Proc. 15th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'15)*, Melbourne, Australia, Oct. 2015, pp. 761-764. doi:10.18429/JACoW-ICALEPCS2015-WEPGF030

[5] C. Rosati, K.-U. Kasemir, and K. Shroff, "JavaFX and CS-Studio: Benefits and Disadvantages in Developing the Next Generation of Control System Software", in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'17)*, Barcelona, Spain, Oct. 2017, pp. 770-774. doi:10.18429/JACoW-ICALEPCS2017-TUPHA154

[6] P. Lotrus and G. A. Durand, "Saclay GBAR Command Control", in *Proc. 14th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'13)*, San Francisco, CA, USA, Oct. 2013, paper TUPPC040, pp. 650-653.

[7] M.Sekoranja, "Native Java implementation of Channel Access for EPICS", in *Proc. 10th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'05)*, Geneva, Switzerland, Oct. 2005, paper PO2.089-5.