Data Acquisition System Deployment Using Docker Containers for the SMuRF Project

J. Vasquez
SLAC National Accelerator Laboratory, Menlo Park, California, USA

Abstract:

The SLAC Microresonator Radio Frequency (SMuRF) system is being developed as a readout system for next generation Cosmic Microwave Background (CMB) cameras. It is based on a FPGA board where the real-time digital processing algorithms are implemented, and high-level applications running in an industrial PC. The software for this project is based on C++ and Python and it is in active development. The software follows the client-server model where the server implements the low-level communication with the FGPA while high-level applications and data processing algorithms run on the client. SMuRF systems are being deployed in several institutions and in order to facilitate the management of the software application releases, dockers containers are being used. Docker images, for both servers and clients, contain all the software packages and configurations needed for their use. The images are tested, tagged, and published in one place. They can then be deployed in all other institutions in minutes with no extra dependencies. This paper describes how the docker images are designed and build, and how continuous integration tools are used in their release cycle for this project.

The SMuRF Project

The next generation of cryogenic CMB (Cosmic Microwave Background) cameras require densely instrumented sensor arrays. These arrays have large number of sensors, in the order of 10,000 to 100,000 per camera. The readout of this large number of sensors is a big challenge that requires substantial improvements in highly-multiplexed readout techniques.

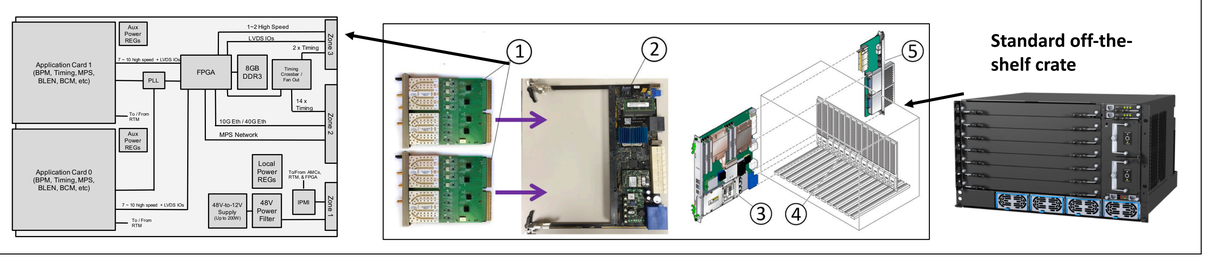
The SMuRF system is being developed as a readout system for this next generation CMB cameras. It aims to read 4000 microwave channels between 4 and 8 GHz, in a compact form factor. The system reads out changes in flux in resonators by monitoring the change in transmitted amplitude and frequency of RF tones produced at each resonator's fundamental frequency.

The SMuRF system is unique in its ability to track each tone while minimizing the total RF power required to read out each resonator, thereby significantly reducing the linearity requirements of the system.

The SLAC Common Platform

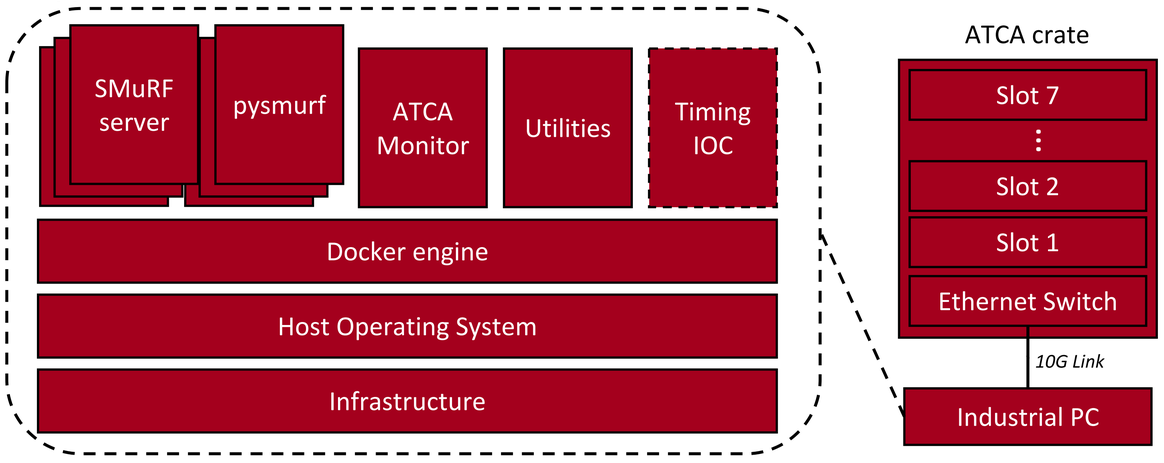
The SLAC Common Platform is a set of Hardware devices, firmware and software libraries, used for several projects at SLAC.

- **Hardware:** it is based on the ATCA (Advanced Telecommunication Computing Architecture) standard.
- **Firmware:** it is a set of VHDL libraries which contain protocols, device access and commonly used modules for all applications that use the SLAC Common Platform hardware.
- **Software:** it is based on Rogue, a C++ library with Python bindings. It is used as a framework to write the low-level software application that communicates directly with the FPGA



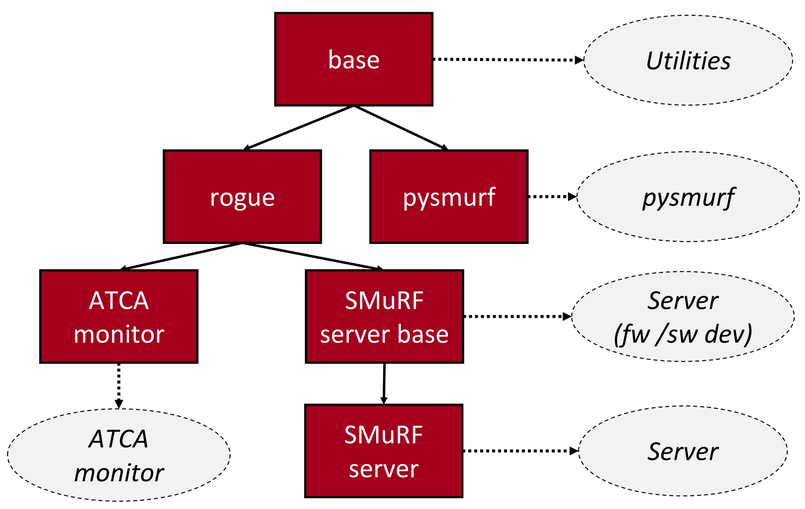
Docker containers for the SMuRF System

Each application used in the SMuRF system runs in an independent docker container.



- **SMuRF Server:** C++/Python application that provides access to the firmware registers and handles the reception and processing of asynchronous streams of data originated at the FPGA application (2240-byte packets at a 10kHz).
- **pysmurf:** Python application that provides a set of high-level routines that allow uses to configure and use a SMuRF system, including data taking and analysis.
- **ATCA Monitor:** it is used to monitor the status of the ATCA crate, as well as all the cards installed in it via IPMI.
- **Utilities:** A set of heterogeneous debugging tools.
- **Timing IOC:** EPICS IOC to control a timing master generator, used in some deployments.

Docker containers Layered Structure



Base:
- Ubuntu:18.04
- + Common packages/tools

Rogue:
- + Rogue framework

pysmurf:
- + pysmurf application

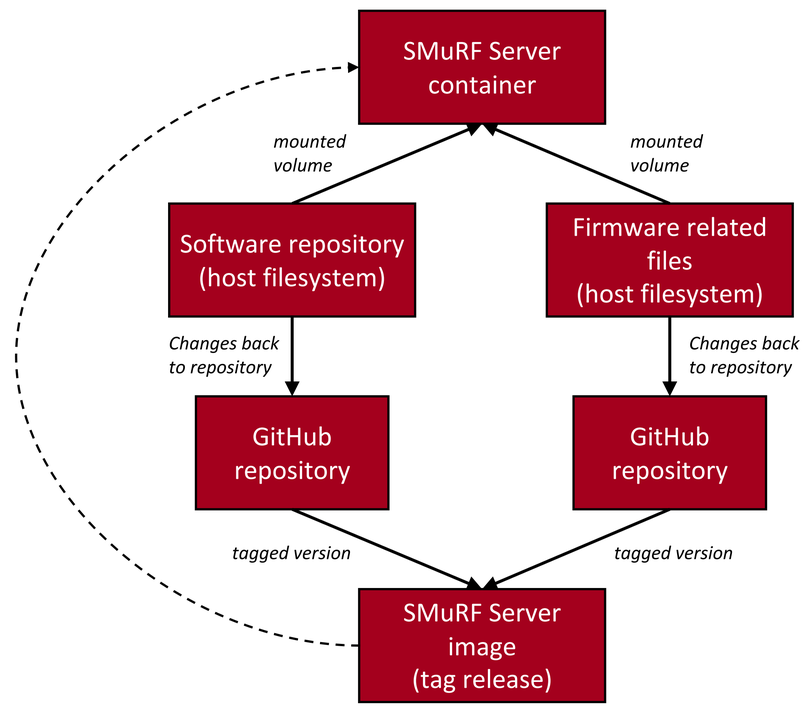
ATCA monitor:
- + ATCA monitor application

SMuRF server base:
- + server application

SMuRF server:
- + firmware files

Docker Container for Development

Beside deployment, docker containers are used for software and firmware development. They provide a uniform development environment, which is also consistent with the final release environment. It is based on the concept of volumes, which allows to mount host's directories inside the container environment. This allows the developer to make persistent changes to the code, while still compiling it and running it inside the docker container environment.



Docker Image Release Cycle

The use of Continuous Integration (CI) tools is heavily used for the automation of the docker image release cycle. The code for each application is hosted in GitHub repositories; Travis jobs are triggered when new tags are pushed to those repositories. The Travis jobs are setup to build the docker images, as well as pushing the resulting image to Dockerhub. Once there, the image can be pulled to any server in the SMuRF project.

Conclusions

The SMuRF project is under continuous development in several institutions across the US. Software and firmware applications are not just under rapid development and evolution, but they also are tools needed for other types of developments (hardware, sensors, etc.).

Docker containers are used as a deployment method for these applications. The use of containers facilitates both the release of new version from part of the developers as well as the deployment of new versions from part of the users. Once a docker image is built, it can be run as a container in minutes in any of the SMuRF institutes. Integration with modern CI tools has made the release process automatic for the developer.

Docker containers are also used for development purposes. They provide a uniform development environment, which matches exactly with the final release environment.

17th Biennial International Conference on Accelerator and Large Experimental Physics Control Systems
Hosted by Brookhaven National Laboratory
October 5 - 11, 2019
New York, NY

Jesus A. Vasquez Stanescu
jvasquez@slac.stanford.edu