Fermilab EPICS Deployment

Pierrick Hanlet, Mariana Gonzalez

26 October 2021

A Partnership of:
US/DOE
India/DAE
Italy/INFN
UK/UKRI-STFC
France/CEA, CNRS/IN2P3
Poland/WUST
Fermilab Control System Content

- Who are we
- Motivation
- Introduction
- Continuous Integration/Continuous Deployment (CI/CD)
- Testing
- IOC Deployment and Monitoring
- Security
- Present Work
- Summary
Who are we

• Pierrick Hanlet, Ph.D
  - At Fermilab since 1989 as visiting scientist; joined staff in 2018
  - 15+ years of experience using EPICS
  - Working in Accelerator Division’s Front End controls team
  - Leading development and implementation of EPICS infrastructure at Fermilab

• Mariana Gonzalez
  - Software Engineer with 7 years experience at Industry
  - Experience with DevOps (CI/CD)
  - Working in Accelerator Division’s System Services team
**Motivation**

- PIP-II will use EPICS as its control system framework
- Treating EPICS deployment as a green field to simplify deployment for non-experts
- Small controls team, therefore we:
  - require robust build of infrastructure
  - require automated build procedures
  - require extensive testing
  - require minimal functionality to automate monitoring of IOCs
- Developed a standard EPICS infrastructure to simplify developing IOCs for new developers
  - “base” and “Support” software are built (all supported platforms) and available on controls network
  - developers start from template IOC
  - template IOC has minimal basic functionality required of all FNAL IOCs
- Standard deployment for ease of Controls team debugging IOCs
- Automated build for ease and robustness
- Standard deployment for ease in maintaining software
- Options – e.g. containers – kept open for supporting legacy hardware
- Implement modern computing practices Continuous Integration/Continuous Deployment (CI/CD)
Introduction – FNAL EPICS code structure

- Structure follows conventional EPICS implementations
- The Fermilab “standard deployment” of EPICS code assumes a 3-tier build:
  - EPICS base – main core of EPICS, comprising the build system and tools, common and OS-specific interface libraries, Channel Access and PV Access client and server libraries, static and run-time database access routines, the database processing code, and standard record, device, and driver support. Production code resides in `/usr/local/epics/base`
  - EPICS Support – contains modules which are analogous to drivers one might add to the kernel for a computer to run specific functions and/or hardware drivers. We presently support ~50 support modules and expect this to grow. Production code resides in `/usr/local/epics/Support`
  - EPICS IOCs (Input/Output Controller) – specific front end servers for controls and monitoring; these are built by pulling in Support modules and adding application specific code. A template IOC is provided to developers and already has minimal Fermilab required functionality. Production code resides in `/usr/local/epics/iocTops`
- EPICS base, Support, & iocTops are built for different architectures/platforms
- EPICS base, Support, & iocTops are hosted by NFS server
- Code base is built and tested in Continuous Integration/Continuous Deployment processes
- Goal is to have a robust EPICS code base to simplify novice EPICS-developer deployment and to help experts in debugging
- Builds for: linux-x86_64, arm/Cyclone-V, arm/Arria-10, arm/RasPi2, arm/RasPi3 arm/RasPi4, mvme8100
CI/CD – Deployment of New IOCs

• Implementation being developed by Mariana González.
• Will use github for code management, documentation and issue tracking.
• Using Jenkins tool to automate building and testing
• Test of full chain is immanent (tests scheduled in early November)
  • Migration of code and documentation to Github – [Pending]
  • Migration to SCD Jenkins (name of the division) - [Pending]
  • Automated build of the 3-tier EPICS in all supported archs - [Done]
  • Automated unit testing – [Done for host arch, pending for others]
• Automated functional testing:
  • Test infrastructure – [Done ?]
  • Pipeline readiness – [Done]
  • Test plan – [Work in progress]
  • Scrip test content – [Pending]
  • Automated deploy to Chablis – [Pending]

• Will use CD Jenkins infrastructure
• Next slide for examples
CI/CD – Deployment of New IOCs

- Controls team maintains EPICS base, Support, and other IOCs
- New IOCs can be added by granting permission to write to github repository
- It is assumed that new IOC is built from template and tested
- New or modified IOC push to github triggers Jenkins job
- Testing of new IOC includes:
  - Basic functional tests (based on templateIOC tests)
  - Check for duplicate PV names
  - Passing tests allows for new IOC to be pushed to NFS host
- New IOC owner will take responsibility for starting IOC
CI/CD – Software Path to Deployment

Build Artifacts

- epics-base-R7.0.5_2021-10-13.tar.gz 443.82 MB

Started by user mariana

Revision: 7f142e03f5cef74b5516e76494f29c3a0f143f4f
Repository: https://github.com/epics-base/epics-base.git

  • tags/R7.0.5

Revision: d3cb9b09181169eb314784b2c52bf1d936f4096a43
Repository: https://cdcvs.fnal.gov/projects/base-configuration

  • refs/remotes/origin/master

Revision: 7e7f3597eb4154cf0a2eeb75e4f6e9167ef54b8d
Repository: https://cdcvs.fnal.gov/projects/epics-environment

  • refs/remotes/origin/fix-path

TAP Extended Test Results

This build contains 183 TAP test set(s), and 0 parse error(s).

---

Check out from version control

- https://cdcvs.fnal.gov/projects/base-configuration -- Git

```bash
make -s test-results
```

```bash
+ make -s test-results

/usr/local/epics/base-R7.0.5_2021-10-13/test/tools/0_linux-x86_64
Snippets.tap .. ok
All tests successful.
Files=1, Tests=29, 0 wallclock secs ( 0.04 usr + 0.00 sys = 0.04 CPU)
Result: PASS

/usr/local/epics/base-R7.0.5_2021-10-13/modules/libcom/test/0_linux-x86_64
epicsUnitTestTest.tap ............ ok
epicsTypesTest.tap ................. ok
epicsInlineTest.tap ............... ok
epicsCalcTest.tap ................. ok
epicsAlgorithmTest.tap ........... ok
```

Pull epics-base on test env
Scope of Testing

In order to deliver a robust framework for the controls system, the tests we intend to implement are:

- Unit tests
- Functional tests
- Stress tests:
  - Check connection loss of NFS host
  - Scan rate
  - Data transfer bandwidth
- Operational:
  - Register IOCs when coming online
  - Monitor IOC for monitoring registered IOCs
By default, EPICS IOCs, once deployed on the network, make all their PVs accessible to any other EPICS process. We want to limit this capability by restricting access using currently implemented kerberos authentication. To accomplish this:

- Use existing EPICS capability – “access security”
- Restrict users to single restricted account name
- Restrict access to IOCs via DPMs
- Access to DPMs require kerberos authentication from client side
Documentation and Versioning

- Presently documented on redmine
- Will move to github
- Versioning in redmine/github
- Branch protection for main/master
- Will version code/configurations in github
  - All versions of code/configurations
  - Limited recent versions of executables
  - Policy for determining number of executable versions
Work in Progress

- Build EPICS infrastructure for SoM/SoC architectures/platforms
- FPGA programmer does the hard work of programming FPGA and providing API
- It is not the job of the EPICS developer to define FPGA requirements
- Define requirements for API:
  - Initialization function
  - All large waveforms read directly from shared memory of FPGA and loaded to EPICS waveforms
  - Set value functions
  - Read value functions
  - This functionality is already being used with some BPMs, though not on SoC/SoM
- EPICS IOC will then call these functions from FPGA programmer (provided in a share-object library) and populate or set process variables (PVs in EPICS-speak)
- IOC will run on arm processor on SoM/SoC
  - IOCs are lightweight; i.e. they use little system resources (templateIOCs use ~1-2% CPU)
  - Don’t require additional transport layer to communicate with the code (done in API)
- Presently have EPICS builds for: TerasicSoc (Cyclone V) and Achilles (Arria-10)
- Will add Xilinx based SoMs: MiniZed (Zynq 7Z007S), ZedBoard (Zynq-7000), TE0803 (Zynq UltraScale)
Work in Progress

- HRMs (Hotlink Rack Monitors) are used extensively around Fermilab complex
- 2U chassis and has 64-16-bit ADCs, 8 bytes of digital I/O, 8-16-bit DACs and 8 TCLK timer channels
- Old hardware and serial field bus read out using old VME hardware
- Replacement under development is the PiRM with RaspberryPi controller built in
- These will be used in PIP-II and will replace HRMs in existing complex
- EPICS support module and template IOC under development
Summary

- Plans for deploying EPICS IOCs at Fermilab is well developed
- Much of the infrastructure is already in place and tested
- Software is built for several architectures/boards
- Expect the deployment to be robust, maintainable, and appropriately versioned
- CI/CD path is ready for immanent testing
- Template IOC developed to:
  - help new EPICS developers
  - ensure required basic functionality of all IOCs
- Development in progress on simplified IOCs to interface with FPGAs on SoC boards and with new PiRMs