Continuous Delivery and Deployment of EPICS IOCs at FRIB

Enrique Bernal
Controls Engineer
Outline

- Overview of Continuous Delivery/Deployment at FRIB
- Deploying EPICS IOCs using Puppet
- Experience
FRIB Controls Environment

<table>
<thead>
<tr>
<th>Device</th>
<th>Interface to IOC</th>
<th>IOC Runs On</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Supplies, RF Amplifiers, Vacuum Gauges/Pumps,…</td>
<td>Ethernet (TCP with text protocol)</td>
<td>Virtual Machine</td>
<td>Thousands</td>
</tr>
<tr>
<td>LLRF Controllers</td>
<td>Ethernet (UDP)</td>
<td>Virtual Machine</td>
<td>~350</td>
</tr>
<tr>
<td>MPS Controllers</td>
<td>Ethernet (UDP)</td>
<td>Virtual Machine</td>
<td>~50</td>
</tr>
<tr>
<td>MTCA.4 Systems</td>
<td>PCIe</td>
<td>MTCA CPU (Intel)</td>
<td>~25</td>
</tr>
<tr>
<td>PLCs</td>
<td>Ethernet</td>
<td>Virtual Machine</td>
<td>~20 processors</td>
</tr>
<tr>
<td>Timing Master/Receiver</td>
<td>PCI</td>
<td>cPCI CPU (Intel)</td>
<td>2</td>
</tr>
</tbody>
</table>

- Almost all IOCs run on virtual machines in the data center
  - Improves availability
  - Reduces hardware cost and maintenance burden
  - Resources can be assigned flexibly

- All IOC machines run Debian GNU/Linux 9

- Development, Test and Production environments
How should a SW update solution look like?

- Standardized and automated
- Update all machines and IOCs fast but safe
- Flexible enough to allow multiple configurations in different environments
- Helps to catch issues before code is deployed to production system
- Full traceability
- No risk of breaking anything (you can always roll back)
- Facilitates team work
Continuous Integration Principles

- Maintain a code repository
- Automate the build
- Make the build self-testing
- Merge changes into a shared mainline several times a day
- Every commit to mainline should build
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build

Automated deployment
Continuous Delivery vs Continuous Deployment

- **Continuous Deployment** (used with FRIB test environment)
  - Continuous Integration
  - Automatic deploy after each change on desired VM.

- **Continuous Delivery** (used with FRIB production environment)
  - Continuous Integration
  - Automatic build of a candidate after each change that could potentially be deployed
  - Deployment process is automated but requires approval (e.g. one-click deployment or merge into a release branch to deploy)
- **Central Git version control system**
- **Engineers follow next Gitflow approach**
  - Feature branches for development
  - Master branch (deployed to Test environment)
  - Release branch (deployed to Production environment)
- **Branch permissions prevent accidental push to “release” branch**
  » Pull requests are enforced
Continuous Deployment Pipeline for Test Environment

- VCS "master"
- Build
- Unit Tests
- Packaging
- Package Test
- Package Repository

- Jenkins, Jenkins Debian Glue
- Lintian, piuparts
- aptly

- Deploy to Test Environment
- Integration Tests
- Manual

- VCS "master"
- Merge
- VCS "release"

- Merge to "release" branch initiates deployment to production system
Continuous Delivery Pipeline for Production Environment

- VCS “release”
- Build
- Unit Tests
- Packaging
- Package Test
- Package Repository

- Jenkins, Jenkins Debian Glue
- Lintian, piuparts
- aptly

- Deploy to Production Environment
- Verification

- Requires manual decision to deploy, but fully automatic from there
Deploying IOCs with Puppet: Motivation

- The FRIB approach
  - Deploy EPICS base and support modules as Debian packages
  - Build IOCs on the target machine
    » Allows tweaking of IOC database in the production environment

- Challenges
  - Hundreds of IOCs, maintained by multiple engineers
    » Consistency is important
  - Wide variety of IOCs require flexible deployment solution
  - Steps for setting up an FRIB IOC evolve over the years
  - Typical problems include
    » New revision of IOC database gets pulled from Git repo but IOC maintainer forgets to restart IOC
    » New version of support module gets deployed, but IOC doesn’t get rebuild
    » Out of disk space due to missing logrotate configuration for procServ log files
EPICS softIOC Puppet Module

- Features
  - Supports multiple IOCs on the same machine
  - Automatically builds and restarts IOC if something has changed
  - Runs IOCs as a daemon with systemd
  - Provides access to IOC shell via procServ
  - IOC directory can come from any source
  - By default runs IOC process with limited user privileges
  - Rotates procServ log files
  - Lots of configuration options including
    - Setting environment variables like EPICS_CA_MAX_ARRAY_BYTES
    - Managing autosave directories
    - CA security configuration
Example of Puppet manifest file

class ioc::bernalru {
    # required by ctbox IOC
    package { 'drvtbox-dev':
        ensure => latest,
    }

    # required by ctbox IOC
    package { 'epics-stream-dev':
        ensure => latest,
    }

    # required by ctbox IOC
    package { 'epics-async-dev':
        ensure => latest,
    }

    # config environment vars for all iocs on this node
    Epics_SoftIOC::Loc {
        ensure  => running,
        enable  => true,
    }

    epics_SoftIOC::iop { "ctboxIOC":
            bootdir  => "iomet/ctbox",
            consolePort => 5051,
            auto_restart_loc => true,
            ca_sec_file => "$(lookup('default_ca_sec_dir'))/ctboxacf",
            require => [
                Package['drvtbox-dev'],
                Package['epics-stream-dev'],
                Package['epics-async-dev'],
            ],
            subscribe => [
                Vcsrepo("$(::profile::frrib::iopbase)/ctboxioc"),
                Package['drvtbox-dev'],
                Package['epics-stream-dev'],
                Package['epics-async-dev'],
            ],
    }
}

- Install support packages
- Ensure EPICS Base, procServ etc. are installed
- Configure IOC process (use multiple of these sections to run multiple IOCs on the same machine)
- Create different puppet environments according to different needs.
- Gives engineers all flexibility they need for testing
- Each VM and IOC can be tested using any SW version of any desired package
- Easy to configure by non-experience Puppet users.
Experience

- Works very smoothly
- Saves quite some time when upgrading many IOCs at the same time
- For most use cases we rebuild and restart IOCs automatically after upgrading database files or support modules
  - Thus we always know that we are running the latest version
    » Avoids surprises when an IOC needs to be restarted later
  - It took a while until all engineers were comfortable with this behavior
- Very useful solution to solve unexpected issues. Almost every engineer know which code is running and which version, on every VM or IOC.
FRIB uses
- Continuous Deployment with test environment
- Continuous Delivery with production environment (needs approval)

Libraries are being build as Debian packages on CI server

IOCs are being build on the target machine

EPICS Soft-IOC Puppet module automates deployment of IOCs
- It’s generic (no FRIB-specific functionality)
- It’s free software
  - https://forge.puppet.com/mark0n/epics_softioc
  - https://github.com/frib-high-level-controls/mark0n-epics_softioc

Very flexible and easy to use by non-experienced users