Continuous Delivery and Deployment of EPICS IOCs at FRIB

Enrique Bernal
Controls Engineer

MICHIGAN STATE UNIVERSITY

U.S. DEPARTMENT OF ENERGY | Office of Science

This material is based upon work supported by the U.S. Department of Energy Office of Science under Cooperative Agreement DE-SC0000661, the State of Michigan and Michigan State University. Michigan State University designs and establishes FRIB as a DOE Office of Science National User Facility in support of the mission of the Office of Nuclear Physics.

Outline

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

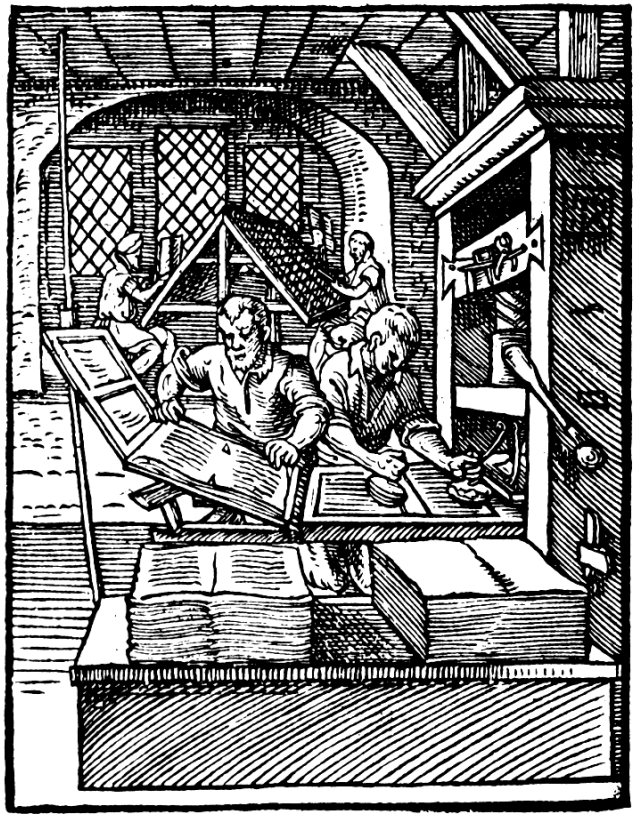
FRIB Controls Environment

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

- 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

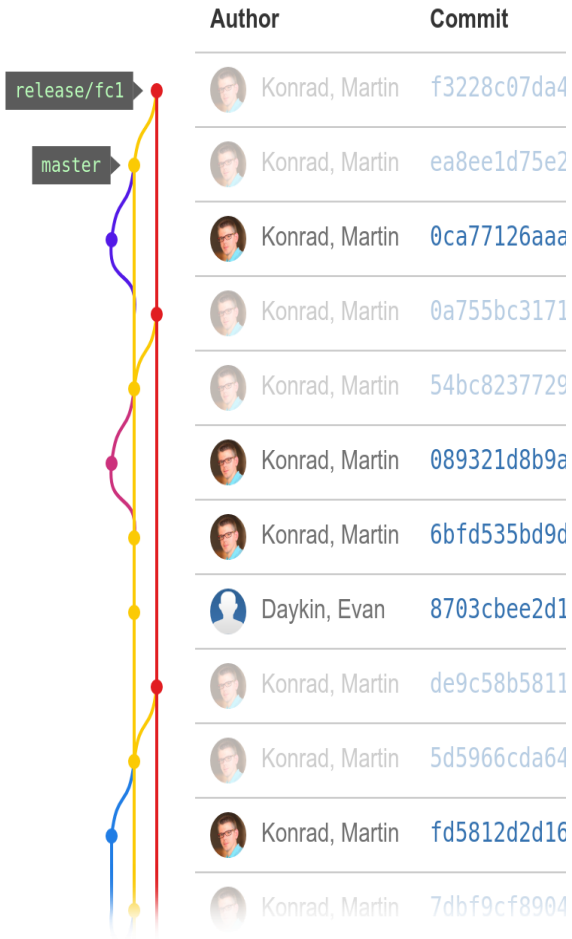
- Maintain a code repository
- Automate the build
- Make the build self-testing
- Merge changes into a shared mainline several times a day
- Every commit to mainline should build
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone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

Continuous Delivery vs Continuous Deployment

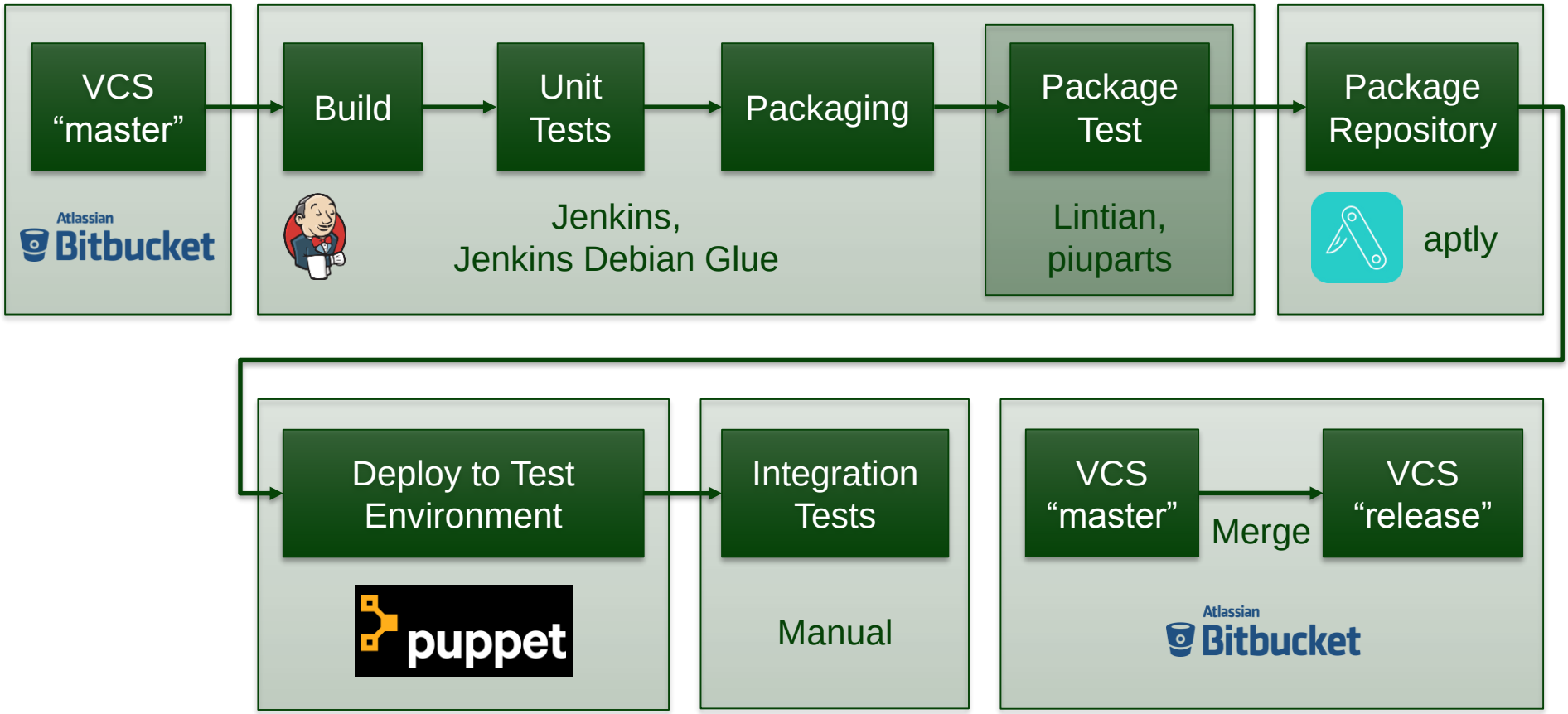
- Continuous *Deployment* (used with FRIB test environment)
  - Continuous Integration
  - Automatic deploy after each change on desired VM.
- Continuous *Delivery* (used with FRIB production environment)
  - Continuous Integration
  - Automatic build of a candidate after each change that could *potentially* be deployed
  - Deployment process is automated but requires approval (e. g. one-click deployment or merge into a release branch to deploy)

Version Control System at FRIB

- 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

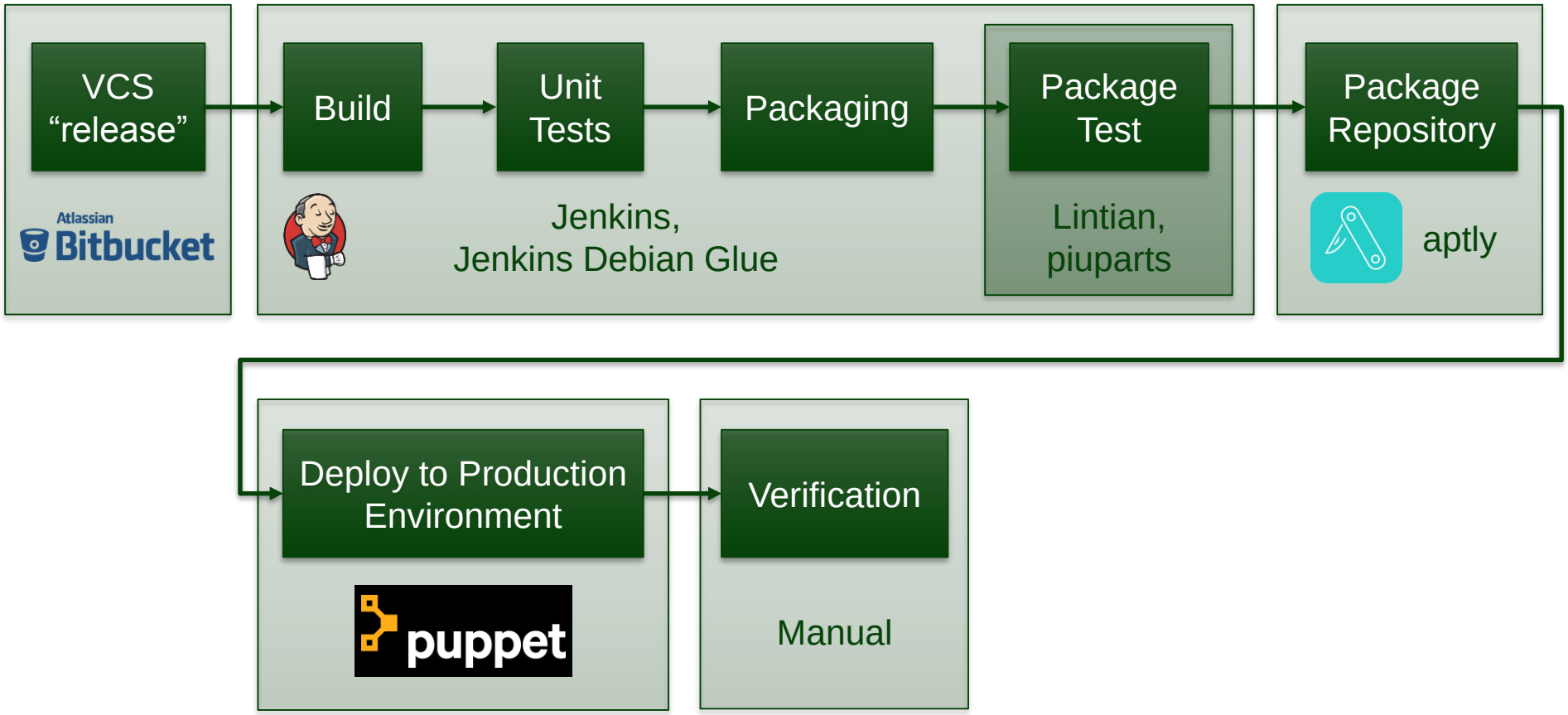


Continuous Deployment Pipeline for Test Environment



- Merge to "release" branch initiates deployment to production system

Continuous Delivery Pipeline for Production Environment



- 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 { 'drvctbox-dev':
    ensure  => latest,
  }

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

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

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

 epics_softioc::ioc { "ctboxioc":
   bootdir            => "iocBoot/iocctbox",
   consolePort        => 4051,
   auto_restart_ioc   => true,
   ca_sec_file        => "${lookup('default_ca_sec_dir')}/ctbox.acf",
   require            => [
     Package['drvctbox-dev'],
     Package['epics-stream-dev'],
     Package['epics-asyn-dev'],
   ],
   subscribe          => [
     Vcsrepo["${::profile::frib_softioc::iocbase}/ctboxioc"],
     Package['drvctbox-dev'],
     Package['epics-stream-dev'],
     Package['epics-asyn-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)

r10k

- 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.

```
##############################
# Local FRIB specific repos
##############################

mod 'profile',
  :git => 'git@git.nscl.msu.edu:Controls/puppet-profile-CTS.git',
  :ref => 'production'

mod 'role',
  :git => 'git@git.nscl.msu.edu:Controls/puppet-role-CTS.git',
  :ref => 'production'

mod 'frib',
  :git => 'git@git.nscl.msu.edu:Controls/puppet-frib.git',
  :ref => 'production'

mod 'ioc',
  :git => 'git@git.nscl.msu.edu:Controls/puppet-ioc.git',
  :ref => 'add-ctboxioc'
```

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.

Summary

- 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