National Synchrotron Light Source II

Brookhaven National Laboratory

U.S. DEPARTMENT OF ENERGY

Updates on EPICS Deployment at NSLS-II

Anton A. Derbenev

NSLS-II Data Science and Systems Integration Program

EPICS Collaboration Meeting

April 25, 2023

NSLS-II

- Brookhaven National Lab is located on Long Island, New York
- NSLS-II is a state-of-the-art facility, medium-energy 3GeV electron storage ring
- Hosts accelerator + 28 beamlines with distinct control system needs
- Standards are RHEL, EPICS, with many hundreds of IOCs running, hundreds of thousands of PVs

Outline

Updates in the following areas:

- Building, packaging, distributing EPICS base and modules
- Managing source code for EPICS apps (e.g., IOCs)
- Deploying EPICS apps and services

Building EPICS base and modules

- Custom build tool (Python): https://github.com/NSLS-II/installSynApps
- Downloads, patches, builds, restructures, creates EPICS base and modules installation (resolving dependencies)
- Config-driven: started with base 7.0.5 + most popular modules, added AD
- Now moving to base 7.0.7, more modules

```
#MODULE_NAME    MODULE_VERSION          MODULE_PATH                         MODULE_REPO
#-------------------------------------------------------------------------------------------

GIT_URL=https://github.com/epics-base/

EPICS_BASE      R7.0.5              $(INSTALL)/base                     epics-base

GIT_URL=https://github.com/EPICS-synApps/

SUPPORT         R6-2                $(INSTALL)/support                  support
CONFIGURE       R6-2                $(SUPPORT)/configure                configure
UTILS           R6-2                $(SUPPORT)/utils                    utils

WGET_URL=https://www-csr.bessy.de/control/SoftDist/sequencer/releases/

SNCSEQ          2.2.8               $(SUPPORT)/seq                      seq-2.2.8.tar.gz

GIT_URL=https://github.com/epics-modules/

IPAC            2.16                $(SUPPORT)/ipac                     ipac
ASYN            R4-41               $(SUPPORT)/asyn                     asyn
AUTOSAVE        R5-10-2             $(SUPPORT)/autosave                 autosave
BUSY            R1-7-3              $(SUPPORT)/busy                     busy
CALC            R3-7-3              $(SUPPORT)/calc                     calc
DEVIOCSTATS     master              $(SUPPORT)/iocStats                 iocStats
```

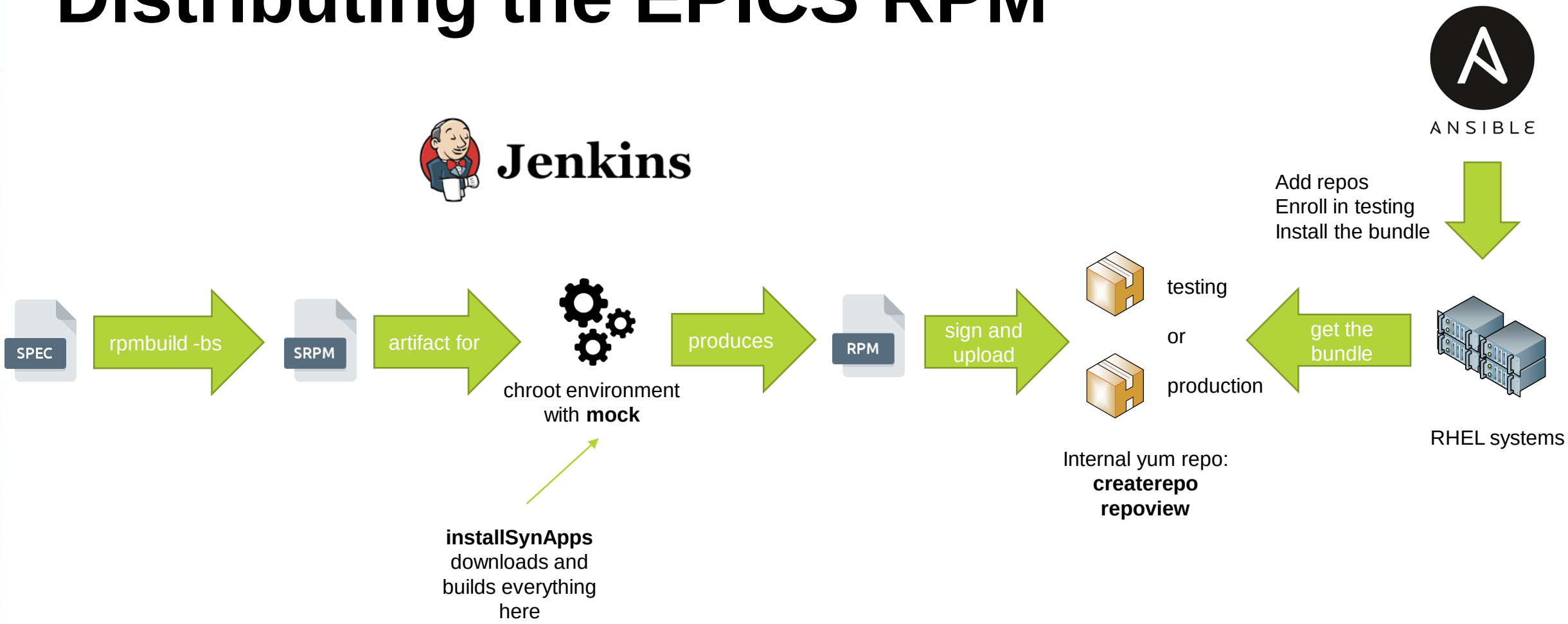
Packaging EPICS for RHEL

- **epics-bundle.spec** wraps the build tool
- All goes in **/usr/lib64/epics**
  - A single stop in **configure/RELEASE** for most IOCs
- Considered, and opted against package “splitting”

1. The EPICS base version, followed by an underscore.
2. A three number string, each separated by periods, with the following meanings:
   - Addition/Removal of a module (potentially breaking change).
   - Change (i.e. version bump) of module with dependants.
   - Change to module with no dependants.
3. A minor release number, meant to signify non-invasive changes to specfile or build process.

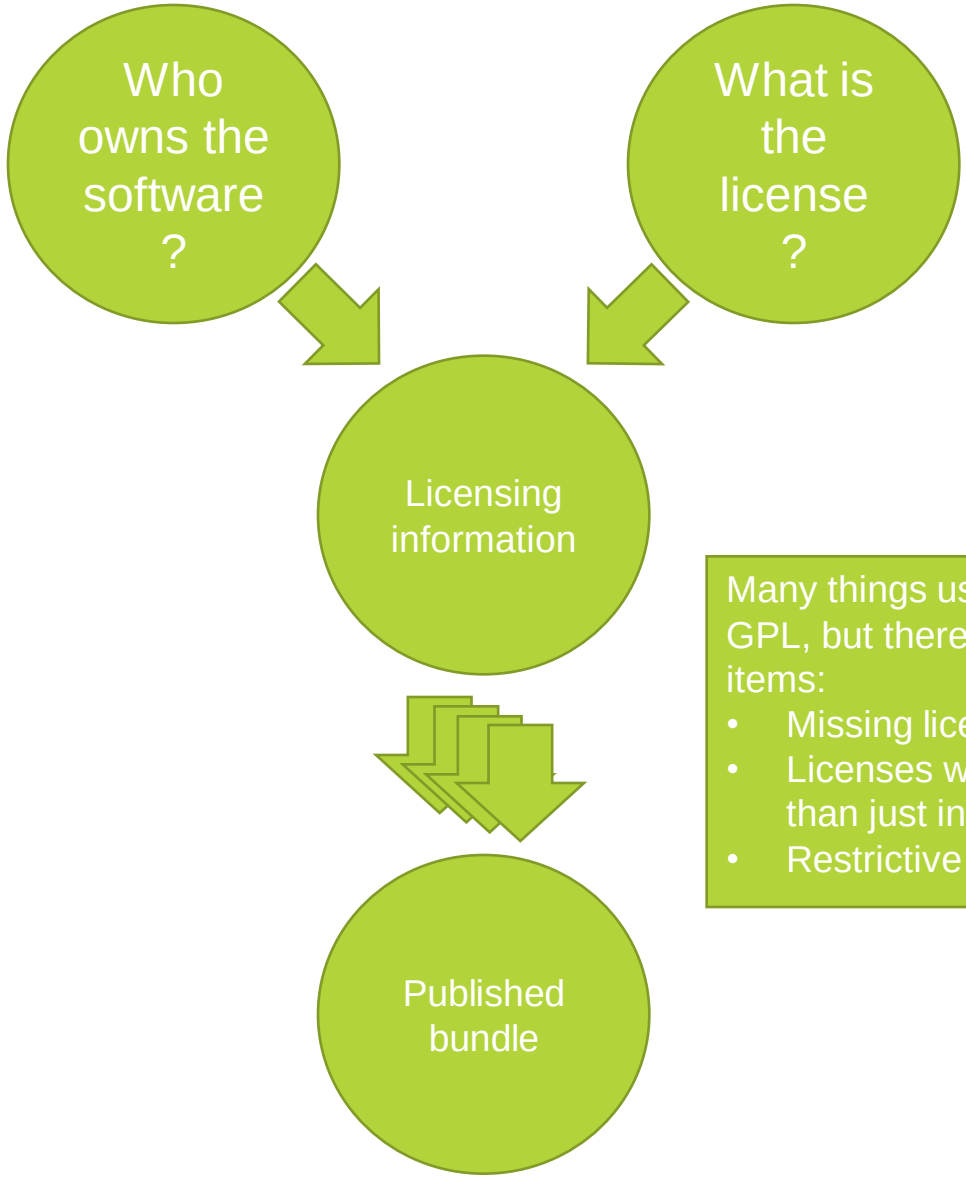
```
Name:           epics-bundle
Version:        7.0.5_0.0.0
Release:        2%{?dist}
Summary:        EPICS Base and Modules bundle

License:        BSD-3-Clause
URL:            https://github.com/NSLS2/rhel8-epics-config
Source0:        %{name}-%{version}.tar.gz

BuildRequires:  python3 boost-devel cmake gcc gcc-c++ giflib-devel git
BuildRequires:  libraw1394 libtirpc-devel libusb-devel libusbx-devel
BuildRequires:  libXext-devel libxml2-devel libXt-devel libXtst-devel
BuildRequires:  make motif-devel net-snmp-devel pcre-devel perl-devel
BuildRequires:  pkgconf re2c readline-devel rpcgen tar wget zeromq-devel
BuildRequires:  git-rpm-tools
Requires:       bash boost giflib libraw1394 libtirpc
Requires:       libusb libusbx libXext libxml2 libXt libXtst
Requires:       motif net-snmp-libs pcre perl re2c readline rpcgen zeromq
```

Distributing the EPICS RPM



Bundle licensing

- The build tool and packaging files are BSD 3-Clause
- They do not include actual base or modules code
- But… software included in the bundle comes with its own licensing
- Not something trivial to resolve for sharing/publishing



Many things use EPICS license or GPL, but there are problematic items:

- Missing licenses
- Licenses which require more than just including a license
- Restrictive licenses

Version control for EPICS apps code

- GitHub Enterprise for all our repositories
- All beamline OPIs in the same repo
- Using “monorepo” approach for IOCs – code for different apps is stored in the same repository
- Monorepo structure defines where app code is supposed to go

Top level directories (areas)

- `acc/` - Accelerator applications area
- `xf/` - Beamline applications area

Subdirectories - (sites)

Accelerator subdirectories correspond to various systems on the accelerator side:

- `bms/` - Utilities / BMS
- `diag/` - Diagnostics
- `eps/` - EPS
- `hla/` - High-level applications including physics and operation

Beamline subdirectories correspond to individual beamlines identified by a three-letter acronym (TLA):

- `amx/` - xf17id1
- `bmm/` - xf06bm
- `chx/` - xf11id
- `cms/` - xf11bm

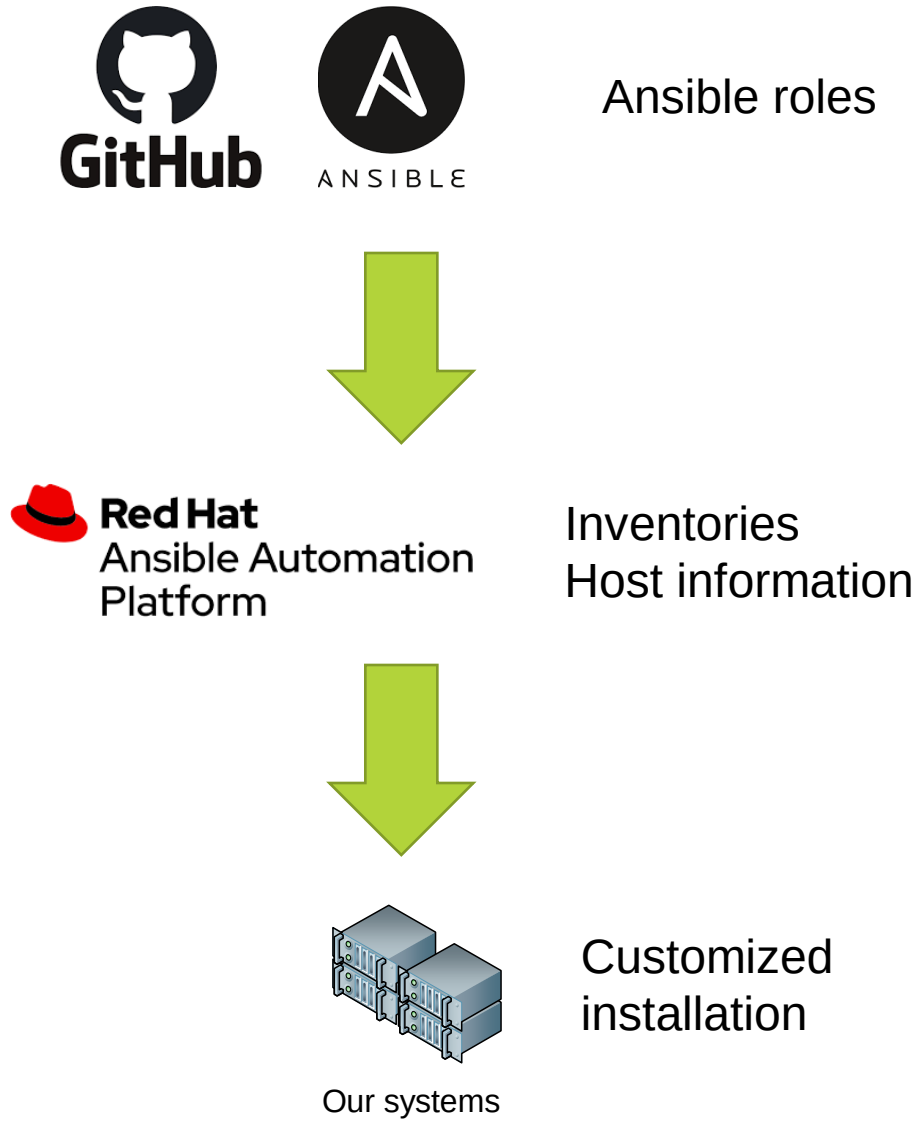
Monorepo tooling

- Convenience is wanted when working with a monorepo
- Plenty of tools can be found, tailored to specific needs
- IOC specifics is that we have many hundreds of them
  - https://github.com/NSLS-II/app-deploy-tools
- **git-mrt** CLI provides the familiar clone/pull/push/status capability

```
14:28:52-aderbenev@dbox:~/src$ git-mrt clone xf/srx/mc01
[INFO] Using monorepo link https://github.com/NSLS2/app-deploy-epics.git
[INFO] Using monorepo home /nsls2/users/aderbenev/.monorepo
[INFO] Using monorepo dir name app-deploy-epics
[INFO] Using local monorepo path /nsls2/users/aderbenev/.monorepo/app-deploy-epics
[INFO] Using monorepo git link git@github.com:NSLS2/app-deploy-epics.git
[INFO] Preparing to extract monorepo subdir as local subrepo
[INFO] Using the specified location as the monorepo subdirectory
[INFO] The monorepo subdirectory is validated to xf/srx/mc01
[INFO] Creating and validating local monorepo
[INFO] Cleaning up the local monorepo
[INFO] Performing monorepo sparse checkout
[INFO] Checking for monorepo updates
[INFO] Initializing a subrepo
[INFO] Extracting the subdirectory into the temporary monorepo branch 'filter-repo'
Switched to a new branch 'filter-repo'
Parsed 10567 commits
New history written in 1.00 seconds...
HEAD is now at 5b802aef0 Repo clean-up and commit latest changes before monorepo migration
Completely finished after 1.10 seconds.
[INFO] Removing 'Merge remote-tracking branch ...' to get a clean history for the subrepo
[INFO] Pulling changes from the branch 'filter-repo' into the subrepo
remote: Enumerating objects: 684, done.
remote: Counting objects: 100% (684/684), done.
remote: Compressing objects: 100% (314/314), done.
remote: Total 684 (delta 422), reused 587 (delta 357), pack-reused 0
Receiving objects: 100% (684/684), 252.30 KiB | 4.59 MiB/s, done.
Resolving deltas: 100% (422/422), done.
From /nsls2/users/aderbenev/.monorepo/app-deploy-epics
 * branch            filter-repo -> FETCH_HEAD
[INFO] Writing subrepo metadata
[INFO] Local subrepo clone complete
14:28:58-aderbenev@dbox:~/src$
```

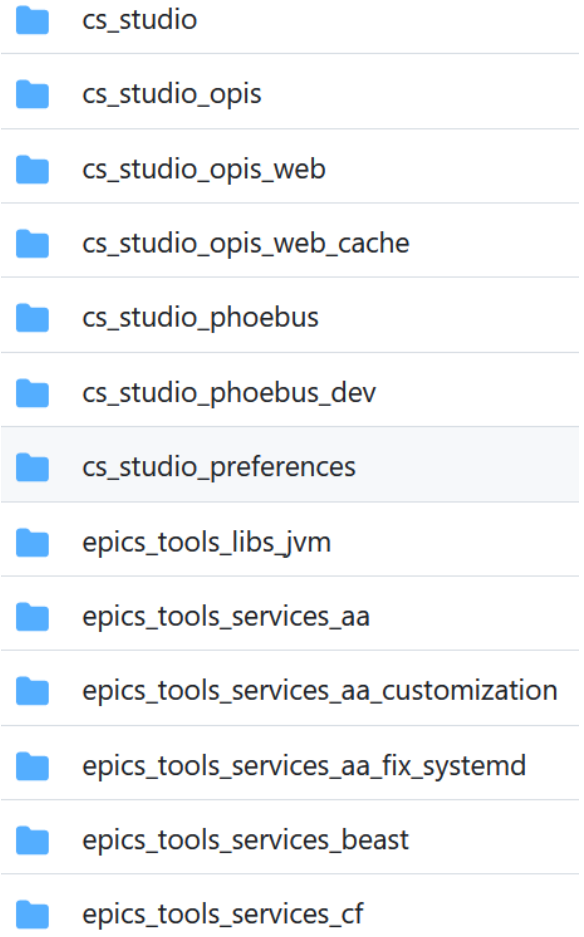
Deployment for EPICS

- Configuration-driven, version-controlled, Ansible-based
- Dedicated Ansible roles for standard apps (like services), OPIs and preferences (for Phoebus, CS-Studio)
- Generic role for templated and unique apps (like CAGWs, IOCs)
- RedHat Automation Platform to run jobs on managed hosts



Deployment: apps, OPIs, preferences

- Accelerator + 28 beamlines have their own configuration, preferences, settings
- Roles are parametrized to deploy anywhere
- Periodic deployment for OPIs to keep them up to date



Deployment: CAGWs, IOCs

- One Ansible role for all
- Dedicated vault identity, encrypted deploy key
- Specify host and app name for deployment, everything else comes from the configuration
- CA gateway deployment is fully templated (no code cloned)
  - Handles 120+ production instances
- IOC deployment can pull, build, install from any repo, with the aim to use our monorepo on GitHub as a primary code location
  - Harder to standardize as IOCs are often customized beyond just configuration

Summary: lessons learned

- “Bundled” distribution is [too] easy to manage
- It’s also [very] easy to get stuck with a stable version
- Licensing matters are not trivial – so no public repo/RPM
- Version control trickiness is proportional to fanciness
- Automation tools allow powerful solutions (after initial investment)
- Standards are best set early and followed consistently