EPICS

IOC Application Development

Andrew Johnson
APS

EPICS

Outline

- IOC development environment review
- Applications as components
- Support vs. IOC Applications
- Managing enhancements
- Revision Control – CVS
- The APS Configuration Management Procedures

EPICS

IOC Development Environment Review

- <top>
  - configure
    - RELEASE
    - CONFIG
    - Other build-system files
  - xxxApp
    - src
      - C sources, .dbd files
    - Db
      - .db files, templates & substitutions
  - iocBoot
    - iocxxxx
      - st.cmd
  - <installation directories>
    - dbd, db, include, bin/<arch>, lib/<arch>

Applications as Components

- The <top> structure and Makefile rules are designed to encourage modularity
  - An IOC is built up out of many components
    - Channel Access, database access, scanning software, other core libraries
    - Record, device & driver support
    - Databases
    - Sequence programs
    - etc.
  - These components do not have to be defined in the same <top> as the IOC itself
    - Most of the IOC software comes from Base
    - Other <top> areas can provide additional components
    - Other <top> areas can override (replace) components from Base
- The configure/RELEASE file determines what other <top> areas will be searched for required components, and in what order
  - Only the installation directories of other <top> areas are searched

EPICS

Support Applications vs. IOC Applications

- Most sites distinguish between <top> applications that provide commonly-used components, and those that build IOCs
  - Record, Device and driver support are usually shared by many IOCs
  - Having multiple copies of source files is a recipe for disaster
  - Different engineers maintain device support than IOCs
  - The lifecycles of the two are usually very different
- Applications that provide components are ‘Support apps’
  - `/usr/local/iocapps/R3.14.1/support/...`
- Applications that build IOCs are ‘IOC apps’
  - `/usr/local/iocapps/R3.14.1/ioc/...`
- IOC apps use entries in configure/RELEASE to indicate which support apps they will use components from
- IOC apps may contain local record/device/driver support, but only for hardware that is *only* found in that IOC subsystem

EPICS

Managing Change: Support Applications

- Support applications generally only change when a bug is fixed, or some new functionality is added
  - It should be a deliberate decision of the engineer responsible for an IOC subsystem to use a new version of a support application
    - Bug fixes can introduce new bugs
    - New functionality might include changes that break old applications
  - Therefore: once installed in use by an operational IOC application, a support application's <top> area should never be changed
  - New versions of the support app should be installed alongside the old one
    - The engineer responsible for an IOC subsystem can switch to the new version when s/he is ready for it by changing the IOC <top> configure/RELEASE file to point to the new version
    - The old version is not deleted until all agree it will never be required again
      - Disk space is cheap
      - It's easy to revert to a previous version if problems are found
  - Need some way of documenting what changed in each version of a support app, and notifying IOC engineers that a new version is available

EPICS

Managing Change: IOC Applications

- IOC applications change very frequently in small ways
  - Updating alarm limits, revised sequences, adding new I/O points etc.
- It should be relatively easy to modify the files that configure an IOC (databases, subroutines, sequence programs etc.)
  - Requiring elaborate version control procedures makes it harder to respond to change requests from the scientists, so less change will happen
- However it is important to retain the history of what changes were made when and by whom
  - Being able to quickly back out of recent modifications can be essential to recovering from incorrect changes
- This is very different to the requirements of managing a support application

Source Code Revision Control – CVS

- Every site should be using a source code revision control system to record the history of the control system's source files
- At APS we use CVS for both EPICS and IOC development
  - CVS is free, flexible, easy to use, and very reliable
  - Subversion is a new system being designed to replace CVS
  - Some commercial systems require a trained administrator
- Recommended reading:
  - Book on CVS by Karl Fogel, published by Red-bean
    - The CVS-specific chapters of this book are available for free online
  - "The Cederqvist" is the CVS reference manual, and is installed along with the software. Type '`info cvs`' to read it.

APS Configuration Management Procedures

- Chapter 7 of the "IOC Software Configuration Management Guide" concentrated the requirements for the two kinds of <top> into a series of procedures developed for use at APS
- A "Quick Reference" guide for these procedures gives engineers all the CVS commands they need in to use most situations
- Other labs have also adopted our procedures (SNS)
- We discourage hiding the CVS commands inside a set of scripts
  - This would prevent engineers from understanding how we use CVS and thus from being able to do unusual things
  - The command set we use is relatively simple, thus written procedures and a quick-reference card have proved to be sufficient

EPICS

APS: Support Applications

- All paths depends on the relevent EPICS base release version

  ```
  /usr/local/iocapps/R3.13.6/support
  ```

- Within the support directory, base is just another application

  ```
  zeus% ls /usr/local/iocapps/R3.13.6/support
  allenBradley/ base/     bitBus/          directNetBug/
  directNetMpf/ ipac/     motorTransform/ mpf/
  mpfGpib/      mpfIp330/ mpfSerial/       share/
  ```

- Support subdirectorise often contain multiple revisions of a support application

  ```
  zeus% ls /usr/local/iocapps/R3.13.6/support/mpfSerial
  R1-3/ R1-4/ R1-4-asd1/ R1-5/
  ```

- Version numbers used are those from the official maintainer, with a local modifier added where necessary (-asd*n*) due to:
  - Locally created bug fixes
  - Repeat builds of the same source version with an updated dependency
    - Above, mpfSerial R1-4 was built against two different versions of mpf

EPICS

APS: Support Apps & CVS

- All support applications have their source kept in CVS
  - A different repository to that used for EPICS development
- We use the CVS "vendor-tracking" feature to import new source versions of externally maintained packages

```
zeus% cvs import ipac v2-3 support/ipac
```

- After doing an import like the above, it is very important to perform the merge step that follows

```
zeus% cvs checkout -jipacV2-2 -jipacV2-3 support/ipac
zeus% cd support/ipac
```

fix any conflicts reported by the checkout line above, and test

```
zeus% cvs checkin -m "Merged in ipacV2-3"
```

- Omitting this would leave old files un-deleted, and omit changes in files that have also been edited locally
- All versions that are to be installed for operations must be tagged

```
zeus% cvs tar R2-3
```

EPICS

APS: Building Operational Support Apps

- Support applications are retrieved from CVS and built by staff in the Operations group, on request from a controls group member

```
helios% cd /usr/local/iocapps/R3.13.6/support/ipac
helios% cvs -r export -d R2-3 -r R2-3 support/ipac
```

  - The ‘cvs -r export’ command does not create CVS management directories, and makes all files Read-Only.
  - The application version directory name matches the CVS tag
- All build output messages are logged

```
helios% cd R2-3; gnumake |& tee build.lst
```

- The support application is now available for use

EPICS

APS: IOC Applications

- All paths depends on the relevent EPICS base release version

```
 /usr/local/iocapps/R3.13.6/ioc
```

- IOCs are grouped by functionality into different <top> areas

```
zeus% ls /usr/local/iocapps/R3.13.6/ioc
booster/ fb/ fe/    id/     linac/ mcr/ par/ rf/
s35misc/ sr/ srbpm/ srtune/ video/
```

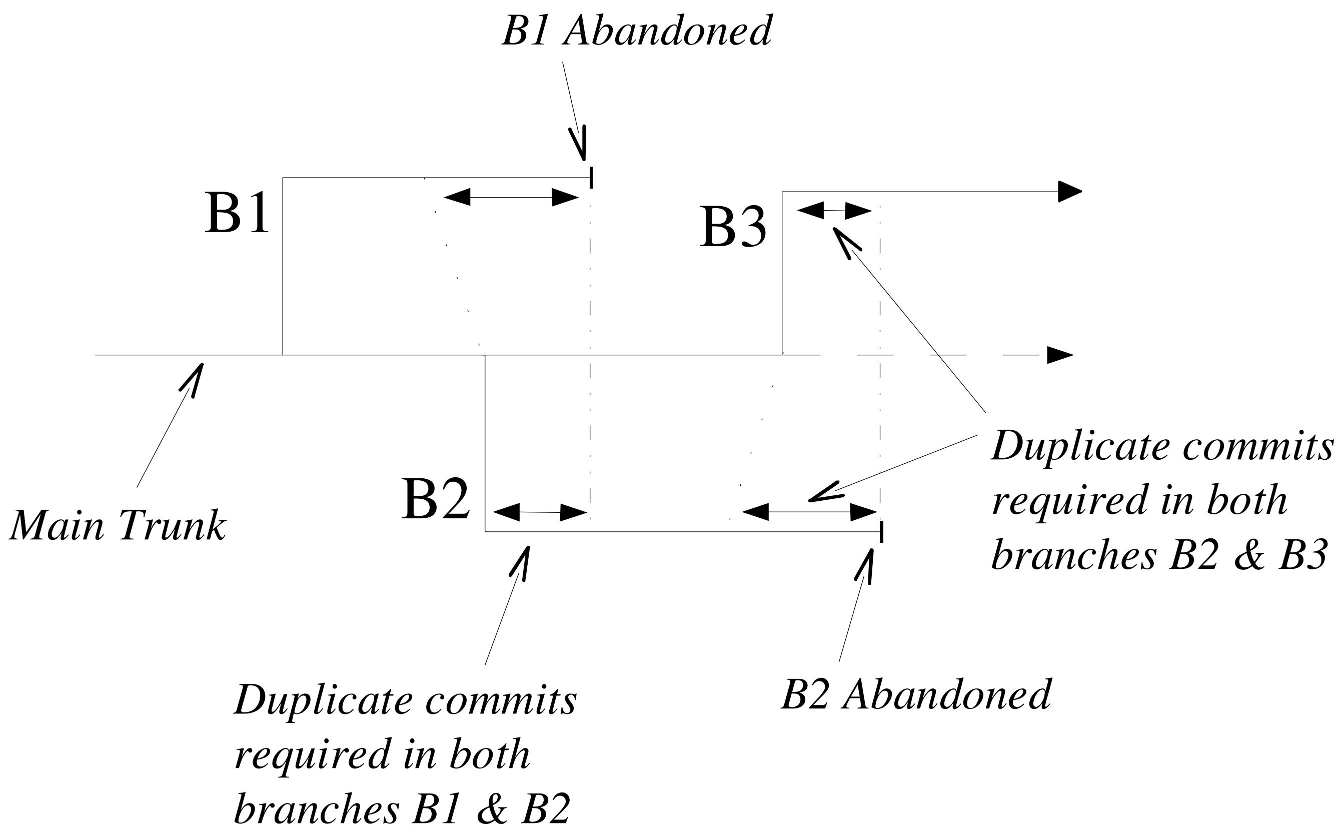
- Each IOC area comprises a line of development of a single <top>
  - IOC development is managed along CVS branches

    ```
    zeus% ls /usr/local/iocapps/R3.13.6/ioc/sr
    B2/ B3/
    ```

  - These 'Bn' directories are CVS working directories
  - Major application modifications are made on a new branch
  - A new branch is also created if an update cannot be used for all of the IOCs built within a single <top> area at the same time

EPICS

IOC Branch Management


B1 Abandoned
B1
B3
B2
Main Trunk
Duplicate commits required in both branches B2 & B3
Duplicate commits required in both branches B1 & B2
B2 Abandoned

EPICS

APS: IOC Applications & CVS

- Developers check out the relevent branch of an IOC <top> that they need to work on from CVS

```
zeus% cd ~/iocapps/R3.13.6/ioc
zeus% cvs checkout -d sr -r B2 ioc/sr
```

  This creates the directory tree `sr` *with the latest files from branch B2*

```
zeus% cd sr
```

  Make, build and test application changes as required

- IOCs can be booted from the developers' area if desired for tests
- The changes are checked in, and the operations group informed

```
zeus% cvs commit -m 'fixed vacuum alarm levels'
```

  - CVS tags are not required for application changes along a branch, but may be used if desired to mark the start or end of particular phases of work
- See the documentation for instructions on creating new branches

EPICS

APS: Building Operational IOC Apps

- Operations staff perform a CVS update in the operations area

```
helios% cd /usr/local/iocapps/R3.13.6/ioc/sr/B2
helios% cvs update
```

Check that the files modified correspond to those the developer said they changed

```
helios% gnumake rebuild |& tee build.lst
```

- The updated IOC area is now ready for reboot & commissioning from the operations tree
- To revert to a previous version, cvs update can be given a date/time to update to, or use tags if applied by the developers