Beamline Setup Demo

Introduction

This is a demo for how to setup a beamline experiment control system from scratch. The example setup can be downloaded from
http://ics-web.sns.ornl.gov/share/xihui/ihepTraining/BLDemoSetup.zip
Some files in the example setup are needed during the setup. In the example setup, all the path are started with /home/5hz/work/ihep because I set up the demo under that directory. You should remove this part or replace it with your start path.
Annotations:
= comment
$ = command prompt
* = note

Java

install OpenJDK java1.7
 sudo yum install java-1.7.0-openjdk

Directory Layout

$mkdir /home/controls
$cd /home/controls
$mkdir css epics share var bldemo download
$cd share
$mkdir opi scan
$cd bldemo
$mkdir applications css opi scan

*bldemo is our demo beamline name. In reality, it should be your beamline name
*note: move all download files to /home/controls/download

EPICS Base

download epics base
unzip to epics/R3.14.12.2
$cd R3.14.12.2
$ vi Makefile

```
# Build EPICS base, support modules in correct order, ...
# kasemirk@ornl.gov

DIRS=base extensions support

.PHONY: dirs $(DIRS)

dirs: $(DIRS)

$(DIRS):
	$(MAKE) -C $@
```

$ vi setup.profile

```
#local site setup for epics environment

export EPICS=/home/controls/epics/R3.14.12.2
export EPICS_BASE=${EPICS}/base
export EPICS_SUPPORT=${EPICS}/support
export EPICS_EXTENSIONS=${EPICS}/extensions
export EPICS_HOST_ARCH=`"${EPICS_BASE}"/startup/EpicsHostArch.pl`
```

```
export
PATH="${EPICS_EXTENSIONS}/bin/${EPICS_HOST_ARCH}:${EPICS_EXTENSIONS}/bin:
EPICS_BASE}/bin/${EPICS_HOST_ARCH}:${PATH}"

export EPICS_CA_AUTO_ADDR_LIST=NO
export EPICS_CA_MAX_ARRAY_BYTES=100000000

#java
export JAVA_HOME=/usr/lib/jvm/jre-1.7.0-openjdk.x86_64
export PATH="${JAVA_HOME}/bin:$PATH"

# New files should be user and group writable
umask 0002

# Niceties
alias ..='cd ..'
alias la='ls -la'
```

$source setup.profile
$cd base
$make
$cd ..

EPICS Extensions

Download extensionsTop: http://www.aps.anl.gov/epics/extensions/configure/index.php
unzip it to R3.14.12.2
$cd extensions
$make
Download http://sourceforge.net/projects/procserv/
unzip it to extensions/src/
$mv procServ-2.6.0 procServ
$vi versions.txt

```
procServ 2.6.0
```

$cd procServ
$./configure
$make
$sudo make install
$cd ..

$copy home/controls/epics/R3.14.12.2/extensions/src/iocs to src/
$cd iocs
$make

EPICS Support

$cd ../../../support
Download Sequencer http://www-csr.bessy.de/control/SoftDist/sequencer/Installation.html
Download Asyn http://www.aps.anl.gov/epics/modules/soft/asyn/
Download Busy http://www.aps.anl.gov/bcda/synApps/busy/busy.html
Download motor record http://www.aps.anl.gov/bcda/synApps/motor/
Download StreamDevice http://epics.web.psi.ch/software/streamdevice/
$tar -xvzf seq...
$tar -xvzf busy...
$tar -xzvf motor...
$tar -xzvf asyn…
$tar -xzvf streamdevice..
$mv seq-2.1.11 seq
$mv asyn4-21 asyn
$mv motor-R6-7-1 motor
$mv busy-1-4 busy
$vi versions.txt
Write down version numbers for every modules.

$cd seq
$vi configure/RELEASE
Change this line:
EPICS_BASE=/home/controls/epics/R3.14.12.2/base

$make

$cd asyn
$vi configure/RELEASE

```
#change these lines
SUPPORT=/home/controls/epics/R3.14.12.2/support
#comment out IPAC and SNCSEQ since we don't need it so far
#IPAC=$(SUPPORT)/ipac-2-11
SNCSEQ=$(SUPPORT)/seq
EPICS_BASE=/home/controls/epics/R3.14.12.2/base
```

$make

$cd ..
$cd busy
$vi configure/RELEASE
$make

$cd ..
$cd motor
$vi configure/RELEASE

```
SUPPORT=/home/5hz/work/ihep/home/controls/epics/R3.14.12.2/support
-include $(TOP)/../configure/SUPPORT.$(EPICS_HOST_ARCH)

# If any motor controller communication mechanism besides the VME backplane is
# required, then ASYN must be defined.
# Recommended ASYN release: R4-18
ASYN=$(SUPPORT)/asyn

# Need the sequencer and the bust record for the MM4005 and XPS trajectory scanning
# Recommended SNCSEQ release: R2-1-13
SNCSEQ=$(SUPPORT)/seq
BUSY=$(SUPPORT)/busy

# Recommended EPICS release: R3.14.12.1
EPICS_BASE=/home/5hz/work/ihep/home/controls/epics/R3.14.12.2/base
-include $(TOP)/../configure/EPICS_BASE.$(EPICS_HOST_ARCH)

# The following must be defined for the MXmotor device driver.
#!MX=$(SUPPORT)/mx/mx

# The following support modules are required for the Hytec driver and for the examples in
# <motor>/motorExApp.  To build examples, the top Makefile,
# <motor>/Makefile must also be edited.

# Recommended IPAC release: R2-11
#IPAC=$(SUPPORT)/ipac/R2-11

# The following is only needed for the motor examples in iocBoot.
MOTOR=$(TOP)
```

$vi Makefile
Uncomment the motor examples lines

$cd ..

$mkdir streamdevice
$makeBaseApp.pl -t support
(hit enter for application name)

$cp ../StreamDevice-2-6 ./
$vi configure/REALEASE
Add path to ASYN and below lines

```
PCRE_INCLUDE=/usr/include/pcre
PCRE_LIB=/usr/lib
```

$cd ..
Create a top Makefile in support so it can build from here
$vi Makefile
*Note that using TAB instead of 8 spaces:

```
# Build EPICS support modules in correct order
# kasemirk@ornl.gov

DIRS=seq asyn busy motor streamdevice/StreamDevice-2-6 streamdevice

.PHONY: dirs $(DIRS)

dirs: $(DIRS)

define cleanDir
$(1)clean:
	$(MAKE) -C $(1) clean
endef

$(foreach dir, $(DIRS), $(eval $(call cleanDir, $(dir))))

clean: $(foreach dir, $(DIRS), $(dir)clean)


$(DIRS):
	$(MAKE) -C $@
```

$cd /home/controls/epics/R3.14.12.2
Create a top Makefile so it can build base, extension and support in one make command from here:
$vi Makefile

```
# Build EPICS base, support modules in correct order, ...
# kasemirk@ornl.gov
```

```
DIRS=base extensions support

.PHONY: dirs $(DIRS)

dirs: $(DIRS)

$(DIRS):
	$(MAKE) -C $@
```

Create Demo Beamline IOC Applications

Create an empty IOC to host applications

$cd /home/controls/bldemo/applications

#create main ioc app from ioc template
$makeBaseApp.pl -t ioc main

#create iocBoot directory from ioc template, when it ask application name, just hit enter
$makeBaseApp.pl -i -t ioc main

#make to make sure everything is fine
$make

Add motor IOC

$cd /home/controls/bldemo/applications

Add dependencies to mainApp

$vi configure/RELEASE
Add following lines after TEMPLATE_TOP... so these paths will be searched for dbd and libraries.

```
# Support modules
SUPPORT=/home/controls/epics/R3.14.12.2/support
ASYN=$(SUPPORT)/asyn
MOTOR=$(SUPPORT)/motor
SNCSEQ=$(SUPPORT)/seq
BUSY=$(SUPPORT)/busy
```

$vi mainApp/src/Makefile

```
#add these lines after main_DBD += xxx.dbd
main_DBD += asyn.dbd
```

```
main_DBD += motorSupport.dbd
main_DBD += motorSimSupport.dbd

#add these lines after #main_LIBS += xxx
main_LIBS += motorSimSupport
main_LIBS += motor
main_LIBS += asyn
```

*To learn which dbd or libs are needed for your ioc, check the example IOC, for example: /home/controls/epics/R3.14.12.2/support/motor/motorApp/MotorSimSrc/Makefile

#make to make sure everything is OK so far
$make

Create motorApp

$mkdir motorApp
$cd motorApp
$cp ../mainApp/Makefile ./
$mkdir Db
$cp ../mainApp/Db/Makefile Db/
$cd Db/
$cp $EPICS_SUPPORT/motor/iocBoot/iocSim/motor.substitutions simMotor.substitutions
$vi simMotor.substitutions
Edit the first line to
file “/home/controls/epics/R3.14.12.2/support/motor/db/basic_asyn_motor.db”
$vi Makefile
add following line under #DB += xxx.db
DB += simMotor.substitutions
$make

Create st.cmd

$cd /home/controls/bldemo/applications/iocBoot
$mkdir iocSimMotor
$cd iocSimMotor
$cp ../iocmain/Makefile ./
$cp $EPICS_SUPPORT/motor/iocBoot/iocSim/st.cmd.unix st.cmd
$make
$vi st.cmd

```
#!../../bin/linux-x86_64/main

# Allow invocation from anywhere, including procServ
cd /home/5hz/work/ihep/home/controls/bldemo/applications/iocBoot/iocSimMotor
```

```
# The is the ASYN example for communication to 4 simulated motors
# "#!" marks lines that can be uncommented.

< envPaths
cd ${TOP}

dbLoadDatabase("dbd/main.dbd")
main_registerRecordDeviceDriver(pdbbase)
dbLoadTemplate("db/simMotor.substitutions")

#keep rest of lines unchanged
# Create simulated motors:
….
```

$chmod +x st.cmd
#test if IOC can successfully run
$./st.cmd
#exit IOC
>exit

Add neutron beam simulation IOC

$cd /home/controls/bldemo/applications
$mkdir simBeamApp
$cd simBeamApp
$cp ../mainApp/Makefile ./
$mkdir Db/
$cd Db
copy .../bldemo/applicatoins/simBeamApp/Db/simulation.db from example setup to here
$cp ../../mainApp/Db/Makefile ./
$vi Makefile
Add this line:
DB += simulation.db
$cd ..
$make

$cd ../iocBoot/
$mkdir iocSimBeam
$cd iocSimBeam
$cp ../iocmain/Makefile ./
$vi st.cmd

```
#!/bin/sh
```

```
softIoc -d /home/controls/bldemo/applications/db/simulation.db
```

$chmod +x st.cmd
#test if ioc run
$./st.cmd
exit ioc

Create standalone StreamDevice IOC

$mkdir NIM8304IOC
$cd NIM8304IOC
$create main ioc app from ioc template
$makeBaseApp.pl -t ioc main

#create iocBoot directory from ioc template, when it ask application name, just hit enter
$makeBaseApp.pl -i -t ioc main

#make to make sure everything is fine
$make

$vi configure/RELEASE
#add below line befor EPICS_BASE

```
….
SUPPORT=/home/controls/epics/R3.14.12.2/support
ASYN=$(SUPPORT)/asyn
#SNCSEQ=$(SUPPORT)/seq
STREAMDEVICE=$(SUPPORT)/streamdevice
```

$vi mainApp/src/Makefile
#Add below lines

```
main_DBD += base.dbd
main_DBD += stream.dbd
main_DBD += drvAsynIPPort.dbd
….
main_LIBS += $(EPICS_BASE_IOC_LIBS)
main_LIBS += stream asyn
```

$mkdir protocol
$vi protocol/nim8304.proto

```
Terminator   = CR LF;
readchan { out "$CMD:MON,CH:1,PAR:VMON"; in "#CMD:OK,VAL:%f"; }
```

$cd mainApp/Db
$vi nim8304.db

```
record (ai, "vmon")
{
    field (DESC, "Read current of PS1")
    field (DTYP, "stream")
    field (INP,  "@nim8304.proto readchan nim8304")
    field (EGU,  "v")
    field (PREC, "2")
    field (SCAN, "1 second")
}
```

$cd ../../
$make
$cd iocBoot/iocmain/
$vi st.cmd

```
#!../../bin/linux-x86_64/main
# Allow invocation from anywhere, including procServ
cd /home/controls/bldemo/applications/NIM8304IOC/iocBoot/iocmain

< envPaths

cd ${TOP}
epicsEnvSet "STREAM_PROTOCOL_PATH" "$(TOP)/protocol"


## Register all support components
dbLoadDatabase "dbd/main.dbd"
main_registerRecordDeviceDriver pdbbase

drvAsynIPPortConfigure("nim8304","192.168.37.50:8100",0,0,0)

## Load record instances
#dbLoadRecords("db/xxx.db","user=chenxhHost")

dbLoadRecords("db/nim8304.db")

cd ${TOP}/iocBoot/${IOC}
iocInit

## Start any sequence programs
#seq sncxxx,"user=chenxhHost"
```

$chmod +x st.cmd

$./st.cmd

Configure Beamline Demo

$cd /home/controls/bldemo
$vi setup.profile

```
#beamline specific environment settings
source /home/controls/epics/R3.14.12.2/setup.profile

# Channel access networking
#
# Need broadcast address on 'local' subnet to access all CA servers
#
# Must serve on specific interface in case there are multiple
export EPICS_CA_ADDR_LIST="192.168.2.255"
#export EPICS_CAS_INTF_ADDR_LIST="192.168.2.1"
#export EPICS_CAS_BEACON_ADDR_LIST="192.168.2.1"
```

$source setup.profile

copy beamline.xml from example setup to here
$vi beamline.xml
Change name, path etc, corresponding to your environment

#iocs.py should always run in the same directory with beamline.xml
$iocs.py -h
$iocs.py start
#check if iocs are running
$iocs.py status
#test if you can telnet the ioc console
$iocs.py console simMotor
***use Ctrl+] to exit ioc console!**
*use q to exit telnet
#test if you can access PVs on all IOCs
$caget motor_x
$caget IOC:m1
#if you can only get one, try to disable iptables and try again
$sudo service iptables stop

CSS

download css from http://ics-web.sns.ornl.gov/css/products.html
download Scan Server from http://ics-web.sns.ornl.gov/css/updates/apps/
$cd /home/controls/css

$unzip css...zip
$unzip scanserver
#make the directory for workspaces
$mkdir workspaces
copy css/settings.ini from example to here so you can easily get started
#edit common css settings for your site
$vi settings.ini

$cd /home/controls/bldemo/css
#create beamline specific settings, start from copying it from example setup
$vi settings.ini

copy bldemo/css/scan_server from example setup to here
#make corresponding changes
$vi scan_server

#create css startup script or copy it from example setup
$vi css

```
#!/bin/sh

CSS=/home/controls/css/CSS_3.1.4

mkdir -p $HOME/CSS
LOG=/tmp/css_`date +%Y-%m-%d-%H-%M-%S`.log
#WS=$HOME/CSS/Default
WS=/home/controls/css/workspaces/$USER

# Create combined settings
INI=/tmp/css$$.ini
cat /home/controls/css/settings.ini /home/controls/bldemo/css/settings.ini >$INI

$CSS/css -consoleLog -pluginCustomization $INI -workspace_prompt $WS -share_link
/home/controls/share=/share,/home/controls/cg1d=/cg1d,/home/controls/bl99=/bl99 "$@"
>$LOG 2>&1  &
```

$chmod +x css

#make css command global available
$sudo ln -s /home/controls/bldemo/css/css /usr/local/bin

Global Settings

Initialization for all users

Add this to $HOME/.bash_profile for all users, maybe to global /etc/bashrc:

```
EPICS=/home/controls/bldemo
if [ -f $EPICS/setup.profile ]
then
  source $EPICS/setup.profile
fi
```

Install iocs.py as service

# Install /home/controls/bldemo/css/iocs into /etc/rc.d/.. to control soft IOCs as Linux service
sudo cp /home/controls/bldemo/css/iocs /etc/rc.d/init.d
sudo chkconfig --add iocs
sudo chkconfig iocs on
sudo chkconfig --list iocs

Install scan server as service

#following commands will start san_server and make it automatically start on computer reboot
#You can use service scan_server start/stop/status
# Check scan_server script: Version of Java, version of ScanServer, settings.ini, then:
sudo cp /home/controls/bldemo/css/scan_server /etc/rc.d/init.d
sudo chkconfig --add scan_server
sudo chkconfig scan_server on
sudo chkconfig --list scan_server

Install Scan Client UI

#start css
$css

After CSS started, install via Help/Install these features from
http://ics-web.sns.ornl.gov/css/updates
1) Scan UI Feature

On restart, configure PyDev:
1) Menu Edit/Preferences/PyDev/Interpreter Python:
Use auto-config.
Then add /home/controls/share/scan to the interpreter's PYTHONPATH.
2) Menu Edit/Preferences/PyDev/Interpreter Jython:
Browse to CSS plugins/org.python_{version}/jython.jar.
Respond to warnings with 'Proceed Anyways'.

Add /home/controls/share/scan to the interpreter's PYTHONPATH.

copy opi, scan files from example setup to bldemo/opi, share/opi, bldemo/scan, share/scan respectively

Play with CSS, BOY, Scan System

In CSS, Open /bldemo/opi/BLDemo_Main.opi