EPICS 7 and CS-Studio Introduction

Overview

This will be an interactive introduction to EPICS 7, i.e. the combination of EPICS V4 with EPICS base, and CS-Studio.

Review of EPICS IOC and Channel Access

Check examples/CAApp and examples/iocBoot/iocCA that were created by
    makeBaseApp.pl -t example

Start the IOC in a terminal window:
    # Alternatively, you can use the shell alias “start_iocCA”
    cd ~/epics-train/examples/iocBoot/iocCA
    ./st.cmd

In another terminal window, run some basic Channel Access commands:

cainfo training:calcExample	*(connection and type info)*
caget training:calcExample	*(get the value)*
caget -a training:calcExample	*(same, with full output)*
camonitor training:calcExample	*(subscribe to changes)*

With Channel Access, channels can be accessed via a predefined list of DBR_* types:
    caget -h
    caget -d CTRL_DOUBLE training:calcExample

Records and their Fields vs. Channels and their Properties

- The Record “training:calcExample” has Fields VAL, TIME, EGU, STAT, SEVR, ..
- The Channel “training:calcExample”, read as a CTRL_DOUBLE, has Properties for the value, timestamp, units, alarm status, severity, ..
  Q: How are the fields of the record mapped to the properties of the channel?
  Q: What are the properties of the channel “training:calcExample”, read as a STRING?
  Q: What are the properties of the channel “training:calcExample”, read as a CTRL_ENUM?

There is no Channel to capture the complete set of fields for each record, but one can create a channel for each individual field of a record.

- The channel “training:calcExample.EGU”, read as a CTRL_STRING, has properties for the value, timestamp, units, alarm status, severity, ..
  Q: How are the fields of the record mapped to the properties of this channel?

pvAccess

examples/PVAApp was manually created as a minimal IOC with just one database, but adding pvaSrv, so all channels can be accessed via the new pvAccess protocol in addition to Channel Access.
Check examples/PVAApp/src/Makefile and examples/iocBoot/iocPVA/st.cmd, comments indicate the additions for the PVAccess server.

Start the IOC in yet another terminal window:

```
# Alternatively, you can use the shell alias “start_iocPVA”
cd ~/epics-train/examples/iocBoot/iocPVA
./st.cmd
```

Do the following steps in the client terminal window again.
While pvAccess is in general similar to Channel Access, it allows you to list the available servers, or list the available channels in a specified server:

```
pvlist
pvlist 0x1232134234  # Using the number shown by previous command
```

You can read respectively monitor PVs via pvAccess:

```
cainfo training:ramp
pvinfo training:ramp
caget training:ramp
pvget training:ramp
pvget -m training:ramp
```

Pvget can actually fall back to Channel Access as a provider.

```
pvget -p ca training:ramp training:calcExample
pvget -m -p ca training:ramp training:calcExample
```

Examples for pvRequest specifier -r “field()”.

```
pvget -r “field()” training:ramp
pvget -r “field(value)” training:ramp
pvget -m -r “field(value, timeStamp, display.units)” training:ramp
pvget -t -m -r “field(value, timeStamp, display.units)” training:ramp
```

- The pvData types we’ve seen so far are “Normative Types”, standard containers representing legacy DBR_* types.
- Custom Data Types: You can define your own data types for specialized servers and clients. We’ll see an example later.

For more details, see the “Introduction to EPICS V4” slides.

CS-Studio and pvAccess

CS-Studio currently defaults to using Channel Access.
Enter PV names with “pva://” prefix to select pvAccess.

- Try ‘Probe’ with “pva://training:ramp”
    - Behaves just like the Channel Access case
- Try ‘EPICS PV Tree’ with “pva://training:ramp”
    - At this time, you only see the value, not the database tree.

Elements of the pvData structure can be read via pva://channel/structure/element.

- Try ‘EPICS PV Tree’ with the following:
    - “pva://training:ramp/display”
    - “pva://training:ramp/display/limitLow”

Images

Run example pva image server:

```
# May also use alias start_imagedemo
cd ~/epics-train/tools/EPICSV4Sandbox/ntndarrayServer/bin/linux-x86_64
./ntndarrayServerMain IMAGE
```

The ntndarrayServer source code is an example for a custom PVA server implemented in C++. Details are beyond this seminar. If interested, please refer to EPICS V4 documentation, http://epics-pvdata.sourceforge.net .

Basic components of a PVA server:

- pvData ‘fields’: Description of a data structure, as e.g. seen by `pvinfo`.
- pvData ‘pv’: Data structure with values, as read by e.g. `pvget`.
- pvAccess ‘channel’: Named network channel that serves pvData to clients.
- pvDatabase: Helper for defining ‘records’ based on ‘pvs’. Handles serving changes to the data of the record via pvAccess. Your application code needs to update the data in the record.

This PVA server serves a data structure of type NTNDArray, a normative type based on the data handled by the Area Detector and commonly used for images.

Explore image data:

```
pvinfo IMAGE
pvget IMAGE
pvget -r “field(dimension)” IMAGE
```

Create CS-Studio display builder panel for image:
Add an Image widget (listed in the Plots category), set its PV name to “pva://IMAGE”.

Custom data structures

Start SNS beam line demo data server:

```
cd ~/epics-train/tools/EPICSV4Sandbox/neutronsDemoServer/iocsBoot/neutrons
./st.cmd
```

This starts an IOC with some plain records and a pvAccess server that serves a custom data type to simulate SNS neutron data.

Alternatively, the custom PVA server can also be started as a standalone program, which offers command line options to configure the simulated data:

```
cd ~/epics-train/tools/EPICSV4Sandbox/neutronsDemoServer/bin/linux-x86_64
./neutronServerMain -h
./neutronServerMain -d 0.5 -m -e 1000 -r
```

Explore neutron data:

```
pvinfo neutrons
pvget -m neutrons
```

This type of data is not meant to be displayed “as is”. It is instead read by custom client software which then for example creates a histogram. During initial evaluation of pvAccess for SNS neutron data, the neutronDemoClient program was used to test if the received series of monitors was complete, without gaps, for varying sizes of the data.

While there is no CS-Studio display builder widget for displaying a custom data structure, you can still create a panel for the elements of the custom structure, for example pva://neutrons/proton_charge and pva://neutrons/pixel.

Conclusion

- IOCs remain fundamentally the same
- The new pvAccess protocol offers advantages over Channel Access
- Clients like CS-Studio will allow a smooth transition from CA to PVA

We hope you enjoyed this introduction to EPICS 7 and CS-Studio!