Controls System Tool Development Cross-training Session

Andrew Johnson

16 March 2005

Online documentation on the Controls Wiki

http://www.aps.anl.gov/asd/controls/wiki/wikka.php?wakka=Ctlsys

What is Ctlsys?

- A new development area for tools that manage and monitor operational IOCs
- Initially meant for scripts and ALH/MEDM screens used when coming out of a shutdown
- The official ctlsys area is located at
  `/usr/local/iocapps/ctlsys`
- The directory structure within ctlsys is very similar to an EPICS Extensions directory tree
- Users can create their own ctlsys areas, linked to the official area, for doing script development

Directory Structure

/usr/local/iocapps/ctlsys

- Makefile
- configure/
  - RELEASE
  - CONFIG
  - … *other files* …
  - os/
    - CONFIG_SITE.*
- src/
  - Makefile
  - ctlsys/
  - … *other source dirs* …
- templates/
  - … *template dirs* …

- adl/
  - … *MEDM screens* …
- alh/
  - … *ALH config files* …
- bin/
  - darwin-ppc/
  - linux-x86/
  - solaris-sparc/
- lib/
  - darwin-ppc/
  - linux-x86/
  - perl/
  - python/
  - solaris-sparc/
  - tcl/

Creating a Development Area

```
phoebus% mkdir ~/iocapps/ctlsys
phoebus% cd ~/iocapps/ctlsys
phoebus% /usr/local/iocapps/ctlsys/bin/arch/
  makeCtlsysDir.pl mytoolname
```

- Populates my ctlsys directory area
- Creates a source directory src/*mytoolname*

```
phoebus% cd src
phoebus% vi Makefile
```

- Add my new source directory to the DIRS list:

  DIRS += *mytoolname*

- Now make will descend into my directory

```
phoebus% cd mytoolname
```

- Work: Create files, edit Makefile etc.

```
phoebus% cd ~/iocapps/ctlsys
phoebus% gnumake
```

- Must build from the ctlsys top the first time

Ctlsys Naming Conventions

- Programs that affect operations must be named according to these rules:
  - Scripts that will dump beam must be named BEAMX_*scriptName*
  - Scripts that will not dump beam but that will have some other operational effect must be named OPX_*scriptName*
- Controls on MEDM screens that will have similar effects on operations must be clearly marked
  - We may define a color convention — Ned?

Makefile Rules

- Alarm Handler

  `ALHS += vacuum.alhConfig`

  - Files will be installed in my ctlsys/alh directory

- MEDM screens

  `ADLS += srrfCharacterization.adl`
  `ADLS += explosion.gif`

  - Files will be installed in my ctlsys/adl directory

- Shell scripts

  `SCRIPTS_HOST += BEAMX_PSCURampTest.sh`
  `SCRIPTS_HOST += BEAMX_PSCULinearize.csh`

  - Files install into my ctlsys/bin/*arch* directory

Perl and Python Rules

- Perl
  - Install scripts into my ctlsys/bin/*arch* directory:

    ```
    PERL_SCRIPTS += vpReadiness.pl
    ```
  - Install modules into my ctysls/lib/perl directory:

    ```
    PERL_MODULES += checkVacuum.pm
    PERL_MODULES += checkPSUs.pm
    ```
- Python
  - Programs install like any regular shell script:

    ```
    SCRIPTS_HOST += OPX_RFCheck
    ```
  - Packages install into ctlsys/lib/python/*package*

    ```
    PYTHON_PACKAGE = RFtools
    PYTHON_MODULES += RFtest.py
    PYTHON_MODULES += Rfdisplay.py
    ```
  - There are other rules for Python using swig, see the Extensions documentation for details

Tcl Script Rules

- Tcl scripts
  - Scripts install into my ctlsys/bin/*arch* directory:

```
TCL_SCRIPTS += OPX_IDcontrol
```

  - These files get installed into ctlsys/lib/tcl:

```
TCLLIBNAME += IDmodel.tcl
TCLLIBNAME += IDview.tcl
TCLLIBNAME += IDcontroller.tcl
```

  - To create and install an autoload index file for the library files listed above:

```
TCLINDEX = IDlibrary
```

Path Variable Expansion

- Scripts that use shared library modules have to be able to locate their installed library files
- The ctlsys build system can do text substitution of file paths before installation:
  - **@TOP@** is replaced by the path to my ctlsys area
  - **@CTLSYS@** is replaced by /usr/local/iocapps/ctlsys
  - **@EPICS_BASE@** and **@EPICS_EXTENSIONS@** are replaced with the relevent paths
  - Any variable defined in configure/RELEASE works
- Substitution can be done on almost any file type
- The source file containing the **@VARIABLES@** must have an @ suffix appended to its name

Path Variable Example

- The makeCtlsysDir.pl@ source file contains:

```
use lib '@TOP@/lib/perl';
use Ctlsys::Getopts;
use Ctlsys::Utils;
```

- The first line expands to this when installed:

```
use lib '/usr/local/iocapps/ctlsys/lib/perl';
```

- This tells Perl it must search the ctlsys/lib/perl directory when looking for library modules
  - Without it, the Ctlsys::Getopts and Ctlsys::Utils libraries would not be found
- The ctlsys/src/ctlsys/Makefile contains

```
EXPAND += makeCtlsysDir.pl@
PERL_SCRIPTS += makeCtlsysDir.pl
PERL_MODULES += Ctlsys/Utils.pm
PERL_MODULES += Ctlsys/Getopts.pm
```

Adding Variables

- Additional variables can be defined for each file to be expanded
- For example:
  - To expand a **@DEF_TMPL@** variable used in my makeCtlsysDir.pl@ file, add this to the Makefile:

```
makeCtlsysDir.pl_EXPANDFLAGS = -D DEF_TMPL=default
```

  - The new variable is only defined during the expansion of the named file