Getting started with Eclipse CDT for IOC development

The Eclipse CDT (C/C++ Development Tool) is well suited to IOC development, and easily customized for editing .db/.template/.substitutions files. Even if you never touch the C code, you can do most of your IOC development with this tool. It includes more generally useful features like search and replace in a directory tree for file names or contents (Ctrl-H), regex search/replace, CVS access, make, gdb with local and remote targets, and launching external tools like edm and vdct. Feel free to try the installation at /afs/slac/u/re/gwbrown/afsbin/eclipse/eclipse. This is not maintained as a site installation. It's easy to download and install your own configuration, where you can try out plugins etc. and untar a clean uninstall if it causes problems. Make sure you've requested a big afs quota.

Download and Install Eclipse CDT

Download CDT from http://www.eclipse.org/downloads (one of many flavors of eclipse from eclipse.org)
Unzip/ tar -xvf filename in the directory you want to run it from, e.g. ~/bin/
"cd eclipse" (to get to the newly created installation) and run "./eclipse".

EPICS friendly C/C++ build settings

If you're mostly using this workspace for EPICS work, you can change global default settings by going to Window -> Preferences -> C/C++ -> New C/C++ Project Wizard -> Makefile Project. On the Behaviour tab, delete "all" after "Build (Incremental build)", and change "clean" to "distclean".

Or change the settings on individual project settings by right clicking the project name -> Properties.
Click C/C++ Build. Make sure "Generate Makefiles automatically" is unchecked, and Build directory doesn't have any path past your project name. On the Behaviour tab, delete "all" after "Build (Incremental build)", and change "clean" to "distclean".

If you want Eclipse to know about base, modules, other things, expand C/C++ General, select Paths and Symbols, and Add... include paths.

Editor colorization

For reasonable colorization, I associate most EPICS files with the makefile editor
To transfer file associations, copy
oldWorkspace/.metadata/.plugins/org.eclipse.core.runtime/.settings/org.eclipse.core.runtime.prefs (or use example attached)
into newWorkspace/.metadata/.plugins/org.eclipse.core.runtime/.settings/

Other settings can be transferred similarly, but copying the entire .metadata directory tree will cause problems
Or
Go to Window -> Preferences, expand General->Content Types, expand Text -> Makefile, click Add...
and add everything in configure directory, *.db, *.substitutions, *.template, st.cmd

EDM

You can have Eclipse launch external tools and editors. If you find a way of passing command line options, update this. It runs an executable with the selected file name as the only argument. As a workaround, instead of trying to pass command line options to edm, create a launch script that has command line options. Here's an example, ~gwbrown/scripts/eclipseEdm:

```
#! /bin/bash
# Launch edm from eclipse with a few options, since eclipse won't let you add flags to "external tools"
edm -eolc $1&
```

Then in Eclipse, set that script as the editor for *.edl files.

Window -> Preferences -> General -> Editors -> File Associations, Add... File type *.edl, to that Add... Associated editor of your launch script as n External program.

VDCT launcher

Both Eclipse and VDCT are good about picking up on changes made by other programs to the files you have open, so you can use both at once. You can have Eclipse launch VDCT as an external tools. Navigate from the Run menu -> External Tools -> External Tools Configurations... . Click the rectangle with a + icon ("New launch configuration"), name it something meaningful like vdct, for Location: use /afs/slac/lcls/tools/script/vdct, for Working Directory: use ${project_loc}, and click Apply.

New workspace:

When you start Eclipse, it asks for a workspace. A workspace is just a directory. Give it a directory name, and it will create it if it isn't there already. It will put hidden metadata files and possibly project directories in the workspace directory.

New Project:

An Eclipse workspace contains projects. Generally a project is tied to the top directory of a buildable unit, like one IOC or module. Projects are of a certain type, which tells Eclipse what type of content to expect, like a C++ makefile project which can be compiled and indexed, or a general project which is just a place for files. Though Eclipse will let you create a project pointing to any directory, e.g. /afs/slac, it's generally a good idea to keep them small. Not only will that speed up searching, compiling, and indexing, but some plugins such as CVS have caused big problems when they find files they're associated with in an existing subdirectory. ***Never*** create a project that has someone else's cvs controlled directory under it!

Eclipse has lots of project wizards and other ways of creating a new project. Here are the ones that are most often useful in IOC development:

Use a directory created outside Eclipse (e.g. with eco or git):

A directory where you can run "make" can be brought into Eclipse as a project. You can create a subdirectory of the workspace directory, and run makeBaseApp.pl there, use eco, use something you already have outside the workspace directory, copy or check out a directory... The important thing is the you start with a directory where you can run "make" (e.g. top of IOC or module).

Start Eclipse. Click File -> New -> Makefile Project with existing code. Pick a name and directory path. Do not select GNU Autotools Toolchain (other than that, does the toolchain setting matter?)

CVS:

Eclipse supports CVS, Subversion, GIT, and various others through plugins. They tend to share a common look and functionality.

Configure CVS from an existing workspace

To copy CVS and C/C++ preferences, in old workspace go to File->Export, in dialog select General->Preferences, create the preferences file (or use example attached)
In new workspace, go to File->Import, in dialog select General->Preferences, select the file from old workspace

Or Set up LCLS CVS repository from scratch

Go to Window -> Open Perspective -> Other..., select CVS Repository Exploring. Under the CVS Repositories tab, right
click -> New -> Repository Location
Host: mcclogin
Repository Path: /afs/slac/g/lcls/cvs
User: yourusername
Connection type: ext

Check something out, and it becomes a project

If you're not already there, open the CVS Repository Exploring perspective, Window -> Open Perspective -> Other..., select CVS Repository Exploring

Browse directory tree to the code you want, right click -> Check Out

Or Check Out As... to specify other options, like a project name different from the cvs directory name, or making it a C/C++ project

General vs. C/C++ projects

Saves resources and some headaches if it's not configured as a C/C++ project, but limits features.
Making it a C/C++ project gives you: building in Eclipse, smart code editing features, debugging

Note: converting base or a directory containing many IOCs or modules to a C/C++ project can bog down your system or crash Eclipse. It's best to be more selective.

Converting to C/C++ project:

A project you created as a general project can be converted.

Click project name, right click -> New -> Convert to a C/C++ Project

AVOID CLICKING ON ANYTHING THAT SAYS AUTOTOOLS. IT DOESN'T WORK WITH EPICS, AND IT'S LIKE A VIRUS IN YOUR PROJECT.
Select Project type "Makefile project", then Toolchains Cross GCC and Linux GCC. Click "Finish".

Building the IOC:

Click the hammer icon to run "make"
Check the console tab for errors. The "Problems" tab isn't very reliable.

Note: depending on how this project was created, you may have to change the build default behavior or it will "make all". Click on the project name, right click, select properties. Click C/C++ Build, select the Bahaviour tab. In the text box after "Build (Incremental build)" delete what's there, it should be empty. In the text box after "Clean" put "distclean". Click OK and try building again.

Click the project name, right click -> Clean Project to run "make distclean"
Check the console tab for errors. The "Problems" tab isn't very reliable.

Debugging the IOC:

General documentation is at http://help.eclipse.org/luna/index.jsp?topic=%2Forg.eclipse.cdt.doc.user%2Ftasks%2Fcdt_o_run.htm&cp=10_3_5 and http://wiki.eclipse.org/CDT/User/FAQ

For RTEMS specific steps, see this subpage.

A similar page for remote debugging LinuxRT targets is in the works. Some key points:

Where your app is started on the target, command line or startup script, start it with gdbserver. For example, the script could contain the line "/usr/bin /gdbserver :10000 ./bin/evrLab iocStartup.cmd".

When setting up the debug configuration in Eclipse, use the gdb corresponding to the compiler. For example, if your LinuxRT app gets compiled with

/afs/slac/package/linuxRT/buildroot-2014.08/host/linux-x86/usr/bin/i686-linux-g++

then you want to select for your gdb

/afs/slac/package/linuxRT/buildroot-2014.08/host/linux-x86/usr/bin/i686-linux-gdb

Other Installations at SLAC:

There are a few Eclipse and CSS configurations available on SLAC file systems. They are generally not maintained much, and aren't set up for IOC development, but may be useful for other purposes.

Eclipse for RCP and RAP Developers, Indigo Service Release 2, with Python plugin:

```
/afs/slac/package/eclipse/i386_linux26/3.7.2/eclipse
```

Eclipse Java EE IDE for Web Developers, Indigo Service Release 2

from lcls-builder at:

```
/usr/local/bin/eclipse
```

from dev network at:

```
/afs/slac/g/lcls/bin/eclipse
```

2004 version at

```
/afs/slac/g/glast/applications/eclipse/eclipse-SDK-3.0.1-linux-gtk/eclipse/eclipse
```

[list more that are around?]