

## **CICADA – Configurable Instrument Control and Data Acquisition**

Peter J. Young, William H. Roberts, Kim M. Sebo

*Mount Stromlo and Siding Spring Observatories, Private Bag, Weston Creek, ACT, 2611, AUSTRALIA*

**Abstract.** CICADA (Young et al. 1997) is a multi-process, distributed application for the control of astronomical data acquisition systems. It comprises elements that control the operation of, and data flow from CCD camera systems; and the operation of telescope instrument control systems. CICADA can be used to dynamically configure support for astronomical instruments that can be made up of multiple cameras and multiple instrument controllers. Each camera is described by a hierarchy of parts that are each individually configured and linked together. Most of CICADA is written in C++ and much of the configurability of CICADA comes from the use of inheritance and polymorphism. An example of a multiple part instrument configuration – a wide field imager (WFI) – is described here. WFI, presently under construction, is made up of eight  $2k \times 4k$  CCDs with dual SDSU II controllers and will be used at Siding Spring’s ANU 40in and AAO 3.9m telescopes.

### **1. Introduction**

Early releases of CICADA<sup>1</sup> had many built-in site-specific sections of code which meant that it was very difficult to “export” to other observatories. One of the aims of CICADA v2 was to replace these sections of the code with “self-configuring” code based around simple ASCII tables. Another aim of release 2 of CICADA was to provide support for multi-controller, multi-CCD instruments. This requirement also impacted on the design of the Configuration Tables. The table design now used by CICADA is a linked list of table items and each item is a hash table of fields which describe the item. There are tables describing Instruments, Cameras, Controllers, CCDs, Temperature Sensors, Focalplanes, Dewars, etc.

### **2. Building a Configuration – the Configuration GUI**

Figure 1 shows the CICADA Hardware Configuration windows for the Controller Table. It is by using these and other configuration windows that the tables are built which are eventually used to dynamically configure the CICADA software.

---

<sup>1</sup><http://msowww.anu.edu.au/computing/cicada>

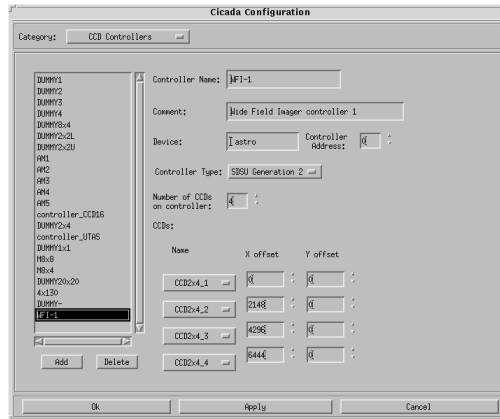


Figure 1. The Controller Configuration GUI.

### 3. Dynamic Implementation of a Configuration

The Cicada\_Master process is the communication and operations control centre for CICADA, functioning as the request interpreter, the activity status monitor and the traffic director. It is started at the beginning of a CICADA run and is responsible for interpreting the instrument configuration.

This is done by instantiating a Cicada Instrument object that represents the hierarchical instrument description by using a set of individual Instrument\_Part objects. Each of these Instrument\_Part objects starts a set of instrument control processes on configured instrument hosts and handles activity and status requests passed to it by the Cicada\_Master process. Each activity request runs as a separate POSIX thread, so simultaneous control of each part is maintained. In addition separate threads are started to fetch status information from the instrument parts. Only one activity can be running on an individual part at one time, attempts to start new activities are blocked.

The use of threads allows for full control of activities running on an instrument part, including monitoring, pausing/resuming and aborting.

The Cicada\_Master process also starts a TCL interpreter and handles the execution of embedded CICADA commands in TCL scripts. These commands are passed to the Instrument\_Part objects in the same way as the interactive command stream which could be from a GUI or from a command line interface.

### 4. Handling Arbitrary Multi-Amp Detectors

An important component of the configurability of CICADA is the ability to handle the geometry of CCD mosaic layouts in a generalised way.

At the GUI level (see Fig. 2, left), the astronomer selects one (or several) regions of interest, spanning some part, or all of the mosaic and possibly including overscan and/or pre-scan pixels. However, at the readout level, a simple rectangular region in mosaic coordinates may actually span multiple CCDs and amplifiers of the mosaic.

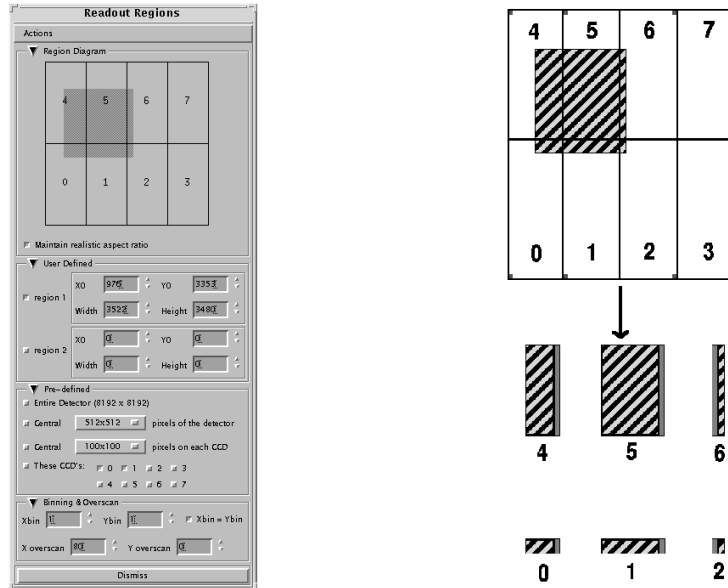


Figure 2. The regions GUI (*left*) and output from the CICADA Regions Class (*right*).

The CICADA Regions class is a general solution to the problem of converting the specified rectangular readout subregion (in mosaic coordinates) into the actual pixels clocked out by the CCD controller(s) (in CCD/amplifier coordinates).

The geometry of the mosaic camera is specified in the CICADA hardware configuration window, where the sizes, positions and readout directions of the CCDs in the mosaic are enumerated.

The CICADA Regions class is capable of dealing with almost any practical mosaic geometry. It handles any (reasonable) number of CCDs of any size, with arbitrary multiple amplifier positions and directions, including support for overscan rows and columns for each amplifier, and independent pixel binning in either axis. Up to two possibly overlapping (mosaic-space) rectangular regions are supported, however this limit is arbitrary and could be increased without difficulty. The only assumption made is that the size of the full readout of each amplifier must be the same for all amplifiers. This constraint is likely to be true for all conventional CCD mosaic systems.

In general, a single mosaic sub-region spanning multiple amplifiers is converted to multiple smaller readout regions, one region per readout amplifier. Each of these smaller regions are ultimately written into separate FITS extensions of a multi-extension FITS file. The FITS extension headers contain sufficient geometry information so that the original mosaic-level geometry can be reconstructed.

The returned information from the Regions class is used by CICADA to construct the output FITS file from the interleaved stream of pixels coming from the controller(s), and also to display the resultant image reassembled into the same rectangular mosaic-space region selected by the astronomer.

Figure 2 at right shows at the top an example readout region selected by the astronomer in mosaic coordinates, spanning multiple CCDs. Below are shown the multiple sub-regions for each amplifier, returned to CICADA by the Regions class including overscan from each amplifier.

## 5. Site-specific Plug-Ins

Currently CICADA can be extended to support telescopes and filter controllers for sites other than MSSSO by user-written “plug-in” code. This is done by writing code that conforms to a CICADA defined API, building a shared library using a supplied Makefile and then replacing the default shared library. A base level CICADA C++ class implements the API as pure virtual methods that need overriding by the user derived class. Template telescope and filter control definitions are provided for a user to easily follow the required steps.

The GUI for telescope and filter control is then dynamically built based on descriptions provided in the CICADA configuration tables. This method has been used successfully at the UNSW APT telescope, the University of Tasmania and work is now progressing toward code to control the Anglo Australian Telescope.

## 6. Current Applications

CICADA is in use at MSSSO, controlling instruments at several telescopes. The same software is used at each telescope, only the configuration files differ to cater for different controller types and CCD configurations. Currently, CICADA supports the SDSU Generation I and II controllers as well as the Astromed 3200 controller. MSSSO is currently building a Wide Field Imager (WFI)<sup>2</sup> in collaboration with the Anglo-Australian Observatory and the University of Melbourne. This new Instrument is an 8 CCD mosaic using dual SDSU Generation II controllers and will make full use of the configurability of CICADA as well as its multi-process, distributed design. The next major project for CICADA is control of MSSSOs existing Double Beam Spectrograph which is a dual-camera configuration (i.e., two shutters, two controllers, two CCDs, two image displays).

## References

- Young, P. J., Brooks, M., Meatheringham, S. J., & Roberts, W. H. 1997, in ASP Conf. Ser., Vol. 125, *Astronomical Data Analysis Software and Systems VI*, ed. G. Hunt & H. E. Payne (San Francisco: ASP), 385

---

<sup>2</sup><http://msowww.anu.edu.au/observing/wfi/>