Camera Visualization Concept of Operations

October 20, 2016: First draft

The purpose of this document is to describe the way that camera visualization will operate. By comparing this document with a similar one from Data Management (DM) we hope to identify common issues and solutions – one goal being avoiding duplication of effort. For clarity, the relationships are divided into three categories:

- User interface
- Functionality
- Relation to other LSST software (*e.g.*, camera data acquisition and the LSST stack).

Visualization architecture

The camera visualization consists of four major components, which can run on two platforms, back end (BE) and front end (FE), although separation is not required. The BE has access to the camera images and other data repositories, possibly via a data butler, and implements functions that require direct access to the pixel values. The FE implements the user interface: command input and image displays and other output. It does not directly access any data repositories.

- Data butler. This SLAC-written code has access to the image repositories. It retrieves requested images and reformats them (if necessary) into single-extension FITS files, which (at this time) are required by Firefly.
- Firefly. This IPAC-written code provides the primary image display functionality. It has two parts, one on the BE and one on the FE. Its BE part turns the FITS file into an image that is displayable by the FE component, which may reside on a remote computer (*e.g.*, over the internet). The FE displays images and also implements some user interaction, described below.
- FE user interface. This UIUC-written code provides command entry, window management, and help.
- BE analysis code. Written by UIUC, it implements algorithms that require access to the raw image pixels, and other functionality that may require access to data repositories.

Front end user interface

The user interacts via web browser, with functionality implemented in JavaScript. The URL points to the back end server, which must already be running. Low bandwidth communication between the FE and BE allow the user and the display to be remote from the database and data processing. Commands that do not require processing of raw image data or access to other remote data (*e.g.*, region selection on a displayed image) are performed locally, either by Firefly or JavaScript. The user can create windows as needed, so that, for example, multiple images or analysis results can be simultaneously displayed. Help is available interactively.

Functionality

The set of implemented functions are determined by camera needs, so they will be described here in some detail. That will enable the identification of needs that are common to the various visualization environments.

Functions that are needed in real-time (as data is taken):

- Display latest:
  Display the most recent image that has been taken. This command puts the specified window in an auto-update mode, which will persist until cancelled. Updating is done with some latency, so that the user has time to respond appropriately. In particular, the user can pause updating in order to perform more detailed studies of an interesting image.

Functions that display or analyze a single image:

- Display (a specified image):
  The user can specify the image via URL or by specifying selection criteria that the data butler will interpret appropriately. This latter functionality remains TBD.
- Select a region of an image:
  Many functions operate on a specified region of an image. Firefly supports some region selection (points, lines, and rectangles, with circles to come). The user interface has added the selection of hardware regions (e.g., amplifier regions). When necessary, the selected region is passed to the BE as a command parameter to specify the pixels to be processed.
- Display the pixel coordinates, and the hardware region (amplifier and CCD) as the cursor moves

Functions that display or analyze multiple images:

- Display (another specified image)
  After creating another window, the user can request that another image be displayed in it.
  TBD: The second image can be specified as having a relationship (*e.g.*, DAQ conditions) to the previously displayed image.
- Blink two images.
  Firefly supports the ability to flip the display between multiple images.
- Display the difference or ratio of two images.
  This is implemented in the BE python code. A new image is created, and displayed. Note that the Firefly architecture requires that image display commands be initiated in the FE, so the URL of the new image must be passed back to the FE.

Relation to other LSST software

yyy