A Million Frames A Week

(while not going crazy)

Building an astronomical data archive
for CIT's TοφCAM (“ToffeeCAM”)

Ronan Cunniffe
UCD 18/08/2008

Stating the problem (1 of 3)

Cork Inst. of Tech. are building ToφCAM:

- 2-channel photometer - 2 x Andor 1Kx1K EMCCDs
- Designed for high time-resolution blazar monitoring
- For 0716+714, means ~5fps on a 1.5m telescope
- 5fps for a whole night ..... >1TB; 360,000 frames

360,000 frames is a bigger problem than 1TB.

I need to provide a system to store, manage, search and retrieve (and delete!) data arriving at this rate.

Stating the problem (2 of 3)

CIT already have ~300GB (~1M images)

- CAHA2.2 (CIT Andor/CIT software)
- CAHA1.23 (CIT Andor/CIT software)
- Kryoneri1.5 (CIT Andor/CIT software)
- BOOTES-IR0.6 (CIT Andor/RTS2)
- BOOTES-2 (IAA Andor/RTS2)
- WHT/ULTRACAM
- JKT
- Abastumani data (of some kind...)

I need to provide a search facility among this data - effectively a derived "common dialect"

Stating the problem (3 of 3)

This must be a fairly common situation:

- Mix of data sources with different FITS dialects
- Very large volume (more than a single PC)
- Have developed in-house pipelines to cope with volume....
- ...but that don't easily handle different instruments
- ...are not integrated into any kind of system.

I should provide a platform that pipelines can be built on/interface to.

Stating the problem

Official project goals:
- Scale to 10TB or more
- Make millions of images searchable
- Hide differences between FITS header dialects
- Make it easy to feed query results to processing scripts.

Unofficial (but obviously relevant) goals:
- Use commodity protocols/technologies.
- Make it easy to add new dialects
- Make it easy to add more storage!
- Support for update-by-sneakernet
- Facilities for backup, mirroring, rebuild-from-scratch

Decisions (Part 1)....

There will be a DB.... Will it contain the image data? ***No.***

So where are the images? ***Separate external filestore(s).***

How do filestores work? ***Each one is a http server, and the DB contains the URLs.***

If I observed 0716+714 for 14,509 exposures (@1sec each) yesterday, and for 4,143 exposures (@ 2 secs each) today, how many URLs would the DB return if I asked about observations of 0716 in the last 2 days? “Results 1-10 of approximately 18,000” or “2”?

Storing and locating

Example: (the 3 kinds of unit)

/0716+714/AndorLaMayora/20080319/ might contain:

20080319-210423-004-RA.fits *// single file*

20080319-210424/20080319-210424-114-RA.fits *// group of FITS*
20080319-210424/20080319-210426-130-RA.fits
20080319-210424/20080319-210428-252-RA.fits

20080319-210429/fits_header.txt *// Andor spool*
20080319-210429/timing.txt
20080319-210429/00001.dat
...
...
20080319-210429/00600.dat

Decisions (Part 2)....

How do we search the DB? ***Via a web page. The search terms are "common dialect" FITS keywords.***

So.... what *exactly* is (in DB terms) a dialect? ***A fingerprint, and a keyword translation table.***

A fingerprint is? ***A combination of keywords: present in all FITS files from a given source, and unique to them.***

And what do we do if there isn't enough uniqueness in the FITS headers for a fingerprint? ***Not sure... hunt down the person responsible, to begin with.***

How are these things created, anyway?

Ah....

Decisions (Part 3....)

So the dialect of a system is...? ***The dialect of the first data uploaded to it, then extended by any genuinely new keywords used in data imported subsequently.***

We don't start with a kind of core keyword set? ***Possibly RA and DEC, otherwise ... probably no. We don't need to.***

So all systems will be different? ***Yes.***

Won't that cause problems? ***Pipelines will need some keyword search-and-replace before they'll work at a new site.***

What about sharing data? ***We never modify the original FITS files. Two separate systems generally won't ever see each other's local dialects.***

Decisions (Technology)

LAMP (But also porting to Windows...)
- CIT teaches/uses Windows (almost exclusively?).
- Want ability to maintain code without Linux skills.

However...
- Unix scripting is far more powerful than any Windows shell - and far more easily debugged and improved by others.
- Unix shell-scripts (e.g. wrapped around IRAF... are pretty-much the natural language for doing astronomy processing).

The archive meets RTS2...

RTS2 writes images to /images/<complicated>.... *and signals to the archive upload mechanism where this is.*

If RTS2's /images is available to the DB directly (and available externally via http), it can refer to that hierarchy without ever moving the files (zero-copy).

Need to implement a connector to get all of the target DB data associated with the images into the archive.

(Note: the archive is not aiming for real-time behaviour, there is no problem with many RTS2 instruments talking to a single archive - no more ssh'ing to find the target id!)

The archive as processing platform

The obvious: a program can be launched from outside, using the DB search interface (via web, maybe RPC-XML) to find work to do.

The automatic: a script can be attached to a dialect profile in the DB (i.e. a particular data source). Incoming data matching that dialect is imported and stored as normal, then the URL is passed to the script.

The possibly unworkable..... using GUIDs as a form of citation...

The citation scheme

Niall Smith wants to be able to look at a light curve, identify something odd, and immediately go and retrieve the source frame whose photometry generated the oddness. Right now, this is hard (for ~14,509 reasons).

This is Very Important for CIT, and so Very Unfortunate, that I can't do it.... it requires smart tools, not just a smart archive.

The citation scheme 2

Suppose we make 30 flats on a particular night, stored as UUID <X>:1-30, and reject #20 (cosmic ray hit).

So we create our master flat (UUID <Y>), and note in it that it was formed from UUID X:1-19,21-30.

When we reduce our data, we note that the flat was UUID X. If we later discover a problem with that flat, we can automatically identify, those data products that used it, and recompute again.

Scalability

Capacity:

- The “structure” of the system is the LAN - we can build in as many hard-disks as will fit in all the PCs we can buy.
- The DB doesn't store per-image headers for most images (only per-group). Particularly for the Cork group (long time series), this is a big win. Cork's 300GB has 30MB of metadata (~x2 in DB form).

Performance:

- DB server is lightly loaded, only hit for queries. Image serving is distributed, load borne by network.

Status

In development....

The big pieces:

- The help-user-to-build-new-dialect mechanism. Status: I hate JKT. Was mostly working (under an earlier design), then I read the JKT keyword comments.

- The search-engine interface. Status: Most of the pieces work, but my web-design skills suck... slow going.

- Archive management tools (diff, sync, check-and-delete, add/remove storage) are mostly buggy and incomplete.