Subject: ++GreenBank - Part 1
Date: Mon, 9 Mar 92 19:23:38 EST

Brian Glendenning and Bob Sault have already described the problems we see in implementing the raw Green Bank scheme. I shall summarise these as I see them, and suggest a way forward similar that already described by Brian. This note is a distillation of my own thoughts and the results of several informal discussions with Brian and BobS, as well as James Coggins and Andrew Klein.

What's right and what's wrong with the Green Bank model

The basic framework of classes (i.e. the way in which various entities in the problem domain are represented) and their interrelations is still generally thought to be satisfactory; certainly, no objections (let alone an alternative proposal) to this framework have been presented and justified, but we should not hesitate to give a fair hearing to other proposals of substance. Bill Cotton has rightly pointed-out that there must be a detailed description of what must be done for each instrument, but I believe that this must be done within a framework which describes a common level of abstraction, if there is anything to be gained in developing a common system. Furthermore, a common abstraction is vital given that we must allow for new instruments and techniques which cannot be specified in detail at present. The Green Bank proposal provides this abstraction - at least until something better turns up.

The essence of the difficulty in implementing the Green Bank proposal is that C++ does not easily allow dynamic association (i.e. at run time) of different kinds of objects using pointers or references from one to another. This is because such pointers or references are statically-typed; their types are fixed at compile time. It is likely that any system having the degree of flexibility we require will run into this problem, and therefore we must address it immediately. One of the reasons (at least on my part) for adopting C++ was the hope that it would allow us to express our models of the problems and solutions of radio astronomical data reduction in a more meaningful way. If we find that we cannot do this, then we should not merely examine our models; we would also have to reevaluate the advantages and disadvantages of using C++ before proceeding further.

A related problem arises in coping with pointers/references in objects which are to be persistent. Clearly, we cannot store pointers or references, since these are only meaningful in the address space of the process which created them. The problem of converting pointers/references to and from a persistent form is well know in database circles ("pointer swizzling"), and could be alleviated by using an object-oriented database - someone else solves the problem for us; however, it is unclear whether we can rely on importing such a database, or have to devise our own.

A possible solution

I believe our problems lie in our lack of familiarity with C++, and thus the ways of implementing a model given the richness of the language, rather than being due to any inherent unsuitability of C++ for our purposes (but again, I am prepared to be proven wrong). The solution I propose first became apparent during a conversation with James Coggins on March 4th, and has subsequently become more clearly useful, particularly as a result of discussion with Brian and BobS. Coggins found the pointer/reference relationships between different objects to be objectionable, primarily because of the persistence problems described above. We had already recognised the problems of maintaining dynamically-typed associations between objects.

Coggins suggested that the associations should be maintained outside the objects themselves, at which point it became clear that we might describe the associations by a row in a table; thus the association between YegSets, Telescopes and the various ImagingModel and TelescopeModel objects would look something like:

UvAssoc	YegSet	Telescope	TelModel	ImagingModel	...
1	Y1	T1	TM1	IM1	...
2	Y2	T1	TM1	IM1	
3					
	:	:	:	:	
	:	:	:	:	

Here, two YegSets, Y1 and Y2 are associated with the same Telescope, TelModel and ImagingModel. Note that this preserves the relationships described in the Green Bank model, but is much more flexible.

Brian has something similar in mind which he calls an "Associator" object, and I suggested that such a thing would be a form of hierarchical database. I believe the table-like representation is more general and flexible, and could accomodate the hierarchical structure required by Brian, thus, if the table above is UvAssoc then we have another association ImAssoc:

ImAssoc	YegSet	DirtyImage	Dirty Beam	ImageTool	DeconvolvedImage
1	Y1	DI1	DB1	CLEAN1	CLEANIM1
2	Y1	DI1	DB1	CLEAN2	CLEANIM2
3	Y1	DI1	DB1	MEM1	MEMIM1

We can enquire of Y1 which objects it is associated with in any of these associations. We might even wish to form associations of associations. Furthermore, whilst many of these associations would be formed by the software system itself (e.g. a YegSet and Telescope association would be formed at the time a FITS tape is read), it would be possible for the astronomer/user to access the association database, and this might be integrated with the Project system, as suggested by Brian.

Such a system is reminiscent of the Dyna-FITS concept proposed by Don Wells, in which case our association table is might be considered to be the Dyna-FITS index. In our case, the objects stored in the table would be "light-weight handles"; i.e. they would not contain any data, but would contain (or refer to) a descriptor of the data and the location of the data. It is likely that we would derive all handles from a common base class, but this common ancestry would not have to be inflicted on all objects in the system.

The flexibility of this scheme is clear, and it is likely that we could kill a number of birds with one stone, providing a common mechanism for object association for users and internally within applications. In many respects, the associator performs the same role as the catalogue in AIPS, and this might be perceived as a disadvantage by many, for all the reasons some users dislike the AIPS catalogue system. I think this kind of objection could be answered in a number of ways:

* When used for persistent objects, the aips++ association database should allow the user to see the "real files" if the user so wishes. Meaningful names and readable headers would also help.

* It might be possible to work without modifying the association database if the associations were hard-wired in at the start, in which case it would be totally hidden from the user, but would also make the system somewhat inflexible. It would be most beneficial to make use of the association database mandatory, but allow the component files to be valid, useable entities in their own right, something like FITS tables in Dyna-FITS.

* The AIPS catalogue has often failed because it is used in circumstances for which it was never designed to cope (e.g. networked systems with multiple accessors); this does not mean the catalogue system is bad, rather it needs to be more robust for such circumstances - this should be

possible (e.g. file system directories exist and work!). However, if the component files are largely independent of each other, then such failures could become relatively unimportant.

* The previous argument might lead some to suggest that we should not be "building an operating system". In fact, we're not building anything we don't need, and the clean break between objects and the association mechanism should give us a more robust system than one in which the association between objects is scattered amongst the objects themselves, particularly if it becomes necessary to rebuild such associations.

In another note I will shortly consider the relationship of Yegs of various kinds with real data.

Dave Shone

Subject: Re: ++GreenBank - Part 1
Date: Tue, 10 Mar 92 10:33:43 EST

> The basic framework of classes (i.e. the way in which various entities
> in the problem domain are represented) and their interrelations is
> still generally thought to be satisfactory; certainly, no objections
> (let alone an alternative proposal) to this framework have been
> presented and justified...

This statement is untrue. I have repeatedly objected to the introduction of overly general and artificial objects such as imaging models and the emphasis given to such objects over actual data that must be dealt with by astronomers. The fact that members of the aips++ group have not been willing to listen to fundamental criticism of the Green Bank model does not mean that there are no objections.

Chris Flatters

Subject: Re: ++GreenBank - Part 1
Date: Tue, 10 Mar 92 13:30:43 EST

Chris Flatters writes (in reply to my original note on this subject):

DS>> The basic framework of classes (i.e. the way in which various entities
DS>> in the problem domain are represented) and their interrelations is
DS>> still generally thought to be satisfactory; certainly, no objections
DS>> (let alone an alternative proposal) to this framework have been
DS>> presented and justified...
CF>
CF> This statement is untrue. I have repeatedly objected to the introduction
CF> of overly general and artificial objects such as imaging models and the
CF> emphasis given to such objects over actual data that must be dealt with
CF> by astronomers. The fact that members of the aips++ group have not been
CF> willing to listen to fundamental criticism of the Green Bank model does
CF> not mean that there are no objections.

I suspect my long sentences cause some people to fall asleep before they have a chance to read the whole thing; the full clause with qualification reads thus:
"...certainly, no objections (let alone an alternative proposal) to this framework have been presented and justified, but we should not hesitate to give a fair hearing to other proposals of substance."

Members of the group usually seem willing to listen to calm, reasoned arguments, and fa
regurgitated proceedings of the last OOPSLA conference are not particularly helpful unless some explanation of their relevance is given, preferably by example, so that we can understand our inadequacies.

Chris' comments on "artificial objects" such as the imaging model deserve a reply, so let me express my own view, which seemed to be shared by many others who attended the Green Bank meething, but they may wish to correct me.

I think that the fact that, in the past, the astronomer has not been able to express the (quite important) relationships between data and desired results (such as an image or spectrum) in a rigorous way does not imply that we should always do things in such a way. Many (most?) astronomers, particularly those who are new to a particular observing technique, tend to think of the data analysis in terms of fairly fundamental relationships (such as the Van-Cittert Zernike theorem for synthesis imaging) and any approximations which may be involved. Of course the data are important - but they are meaningless without models for their interpretation. In this respect the imaging/measurement model is certainly not artificial.

It is easy for most of us to understand what is happening in a system with which we are already familiar, and I can understand that the imaging model does initially appear to be an artificial concept from such a standpoint. However, if we are to attempt to meet the requirements which have been specified, we should not resort to shoehorning everything into a model for processing observation data dominated by the way we handle interferometer data now. This would be disastrous for non-interferometric applications, and would provide little (if any) benefit to the astronomer interested purely in conventional interferometry.

Dave Shone