From: Brian Glendenning
Images, coordinates, confusion
Date: Wed, 18 Mar 92 00:44:27 EST

Late night musings on your note.

First, I think this is the kind of discussion I think we need to get into for the next stage of the design; thanks for the time you put into writing it up. You should think about sending it out generally or discussing it in the morning.

As I understand your points, they are summarized something like:

1. The current system is cumbersome

Absolutely true; for this prototype we didn't worry at all about being concise, although we (well, I should say "I") agree that the next system needs to have more "expressive power."

2. Conceptually complicated

This I think is only partly true. You are becoming frustrated because you can't consistently use coordinates the way you'd like to. I think this is because you are extending the prototype beyond its original intention. Our explicit goal was to make image classes to allow us to do a simple clean in a particular way we were thinking of. You are trying to do something else and are discovering that you don't have the infrastructure to do it in a convenient way. We realized when writing it that there were useful accesses that we weren't writing. You broke the prototype, but we knew it would break if extended. Don't forget the limited scope we were charged with when we started this prototype.

I am personally pretty sympathetic to the view, though, that having three kinds of user-visible coordinates overwhelms the benefits the additional system gives you. This is something that I suspect you'll be able to give us insight into. (Whether or not all types are useful internally to the class is not relevant to the class user).

3. Not efficient

Guilty; I don't think we should be worrying about efficiency with this prototype.

4. Breaks Encapsulation

Not guilty; getting and setting a buffer (scrolling window/subimage/...) of values no more breaks encapsulation than doing so with a single value. The only way we can hope to have an efficient system is to do things exactly like this.

It is certainly true that the exact details of how we want to "buffer" pixel accesses and tie those pixels to the coordinates needs to be thought out more completely (I think Bill Cotton's prototype might have something to say here).

5. Insufficient communication

I agree. I think part of this can be improved by changes in organization (for instance, I think all astronomical structures should be in one "group," I've suggested to Geoff that a reorganization into three groups - support, structure, user interface - be done. Furthermore I think everyone should be a member of two groups and participate (at about a 3:1 ratio) in both groups). The organizational changes won't be enough though, and we'll need to be more explicit in making our "needs" to other groups known.

One amusing way to avoid surprises might be for an "independent" person to write the testbed program, or at least to extend it.

6. The image class needs to be rethought

Yes. I still tend to think that what's happening is that you're breaking the prototype in unsurprising ways. Nevertheless we're all (I

presume) in agreement that these classes (like everything) needs to be rethought out in a serious way for the next stage of the project.

This is fun.

Brian

From: Lloyd Higgs
Subject: Reply to Mark Holdaway
Date: Wed, 18 Mar 92 16:54:48 EST

Mark Holdaway has compiled a series of comments on perceived deficiences in the the difficulties that he and Sanjay had when they tried to use them in implementing a CLEAN procedure. On the whole, his comments are valid, but I disagree with some of them in detail. My personal response to the five areas of his not

The Coordinate System is Cumbersome:

I think this is a fair comment, but I don't think that the corollary is that it (they) should be discarded. Clearly the user should be shielded almost entirely from Pixel coordinates. Unfortunately, methods using these coordinates abound and probably should be removed or made private. Certainly equivalent overloaded methods using ImPixel coordinates should be created.

Mark's example of the difficulty in accessing reference values and deltas from the CoordSys object is valid, but was just not provided for in the prototype class. There it was assumed that details of coordinate conversion should be shielded from the user -- one gives it a coordinate and asks for the equivalent coordinate in another system. If one wants efficient methods to get at detailed parameters within the coordinate-system structure, new methods are obviously required. I was slightly disturbed that this example looked as if one were trying to make the "new" system perform exactly as if one was merely translating AIPS or SDE methods to aips++. We don't want to have our choice of new ways of doing things being based entirely on the way they are done in AIPS. Nevertheless, the comment is valid, and more discussion between the image-handling and uv-data groups would have resulted in the provision of more useful methods.

Conceptually Complicated and Uncleanly Divided:

I disagree with the first part of this statement but agree with the latter. As stated above, Pixel coordinates should be essentially invisible to the applications programmer (unless he really needs them!).

Coordinate Wish List:

I am not sure whether parameters such as OBSRA and OBSDEC belong amongst the attributes of a CoordSys class or whether they belong in some table of parameters attached to an Image. The current contents of a CoordSys class are nearly divorced from observational parameters; they just define a transformation from one system to another. The reference coordinates and deltas, are of course observation related.

The attributes of a CoordSysType object have not been developed at all in the prototype. Axis contents would have to be either specific attributes in the class, or implicitly defined from the CoordSysType identification.

Efficiency versus Encapsulation:

I am sure that trade-offs in this area will have to be made in a real system. One problem with direct access to the data is, of course, that one then HAS to deal with Pixel coordinates.

Recommendations:

I agree that more dialog between the image-handling and uv data groups should have taken place before the prototype was implemented. However, we must remember that many of the problems arose when the latter group was asked to implement the CLEAN algorithm, while in the original prototype design exercise, the former group was going to do this. The methods

implemented were based on a CRC CLEAN exercise carried out by the image-handling group. Changing horses towards the end of the prototype obviously caused some of the problems.

I am not sure that it is the ImPixel coordinate system which is causing the confusion -- I think it is the Pixel coordinate system!

In conclusion, I still think that the basic concepts developed for Images are sound, but much has to be done in the area of implementing friendlier methods. Areas where a lot of new thinking is required, however, is in the area of regions of interest (rectangles, polygons, or bit masks; numbers of the

Lloyd Higgs