NOTE 198
Report of the AIPS++ Scientific and Technical Advisory Group

STAG, (ed. R. Braun)

1996 November 14

A postscript version of this note is available.

1 Introduction & General comments

The AIPS++ scientific and technical advisory group (membership is included at the end of this document) met for the first time in Socorro on 4, 5 and 6 November 1996. Presentations by various AIPS++ project members were interspersed with both public and private discussion. In this document, we summarize the recomendations of the advisory group; beginning with some rather general issues and then proceeding to more specific issues for which feed-back was requested.

The AIPS++ project has clearly made substantial progress towards the implementation of a powerful modern problem solving environment. The project now needs to focus on making the transition from the research/ development/ prototype stage to the production environment. A comprehensive plan is needed for this phase of the project, with a clear timetable for delivery of functionality and a realistic indication of manpower requirements.

2 Project Management

Overall, we are happy with the current management of the AIPS++ project under Tim Cornwell. Tim has the right combination of scientific and software expertise, combined with a pragmatic approach to software development, to get the job done. However, more attention needs to be paid to

project definition. In particular, (1) a comprehensive development plan should be formulated, (2) the target audiences and contents of the releases through 1998 need to be specified in more detail, and (3) a date should be set (e.g. the mid-1997 release) for completing the first stable version of the programming interfaces.

We are concerned that the various application areas are currently somewhat lacking in global focus. We can appreciate the difficulty of beggining with a very extensive list of user requirements and then translating these into a concrete development path which results in full functionality in a finite time. Even so, this is exactly what is necessary. A "blueprint" must be made for each application area which sets out how to proceed, in an organized way, from zero functionality to full functionality. Some of the expertise to draw up a realistic development path already exists within the AIPS++ group, particularly in the areas of synthesis imaging and soon (we hope) in the area of VLBI. We suggest that more use might be made of external "contact groups" for the various application areas. Enlisting the assistance of highly motivated and knowledgeable end–users might be quite effective at focusing the development path. The lack of global development focus is especially evident in the area of Single Dish applications. It seems necessary to us that a high level person carry out a global system design in this area at the earliest possible date.

The lack of manpower available in all application areas is a major concern. This needs to be addressed in the near future.

3 Content and Audience of Releases

There are a number of critical issues which must be addressed concerning upcoming releases of the AIPS++ package. In particular, the content and audience of each planned release must be well defined to ensure that they will be effective. Furthermore, it is vital that each release be "billed" in such a way that no unrealistic expectations are generated.

3.1 January 1997 Release

It is our view that the so–called "β release" scheduled for January 1997, must already be as bug–free as possible. In particular the problems of memory management, and Glish–related crashes should be solved, before even this limited distribution of the package is undertaken. The functionality need not be excessive, but what is present should work well. The audience of this release is presumed to be only a very small number of "friendly" sites

beyond the currently participating consortium members. One of the primary purposes we identify for this release is actually the definition of a concrete deadline which should serve to focus the efforts of the group and lead to a stable development environment at the earliest possible time.

Although we understand the increased logistics impact, we suggest that both source code and documentation be included with the initial binary release, even if there is no intention of supporting local building. The likely recipients should be able to look more closely at the workings of the package. The very limited functionality of this release make it critical that prospective users have the ability to get their data both in *and* out. The ability of the package to write at least calibrated single–source UVFITS files should therefore be given a high priority. We believe this would greatly enhance the number of early users.

3.2 Mid–1997 Release

The release scheduled for mid–1997, which has been referred to as the first "full release", might be better referred to as a "Limited Public Release". The term "full release" seems to us to imply a much greater degree of functionality than can be envisioned for this early date. This first public offering of the package will, in our view, be critical for defining the community perception of the project. For this reason, we feel that it is important that a high priority be given to further development of the user interface(s). Simple "auto–GUIs" should be reinstated to enable novice users to setup processing steps without the need for mastering a verbose and non–intuitive syntax. The command line interface should also be carefully evaluated and if necessary repackaged to insure a high degree of accessibility to the user community. The package needs to "feel" good to new users. This property is more critical, in the short term, than an enhanced functionality. During 1997, new users will have more tolerance for a reduced functionality, than for the frustrations which arise from an awkward interface.

At the same time it seems very desirable to have completed at least one high level application package which thoroughly exercises all of the major system components; including interactive image display and graphics. Only then will it become clear what the "standards" (in terms of user interface, documentation, performance, etc.) should be for new applications within the system. These "standards" need to be defined as soon as possible so that standard applications can be generated within the extended user and programmer community.

The clear limitations of functionality of this release make it important

that data can be brought into, and removed from, the AIPS++ environment. A capability for both reading and writing multi–source UVFITS seems essential in this regard. If it is deemed impossible to reconcile the different table functionality of UVFITS with that of AIPS++, then it would be sufficient to provide calibrated output.

3.3 Subsequent Releases

Based on the solid base of the mid–1997 release, increasing functionality will presumably accompany each subsequent release. A global plan to plot the development of all major processing components which are currently envisaged should be made to help in planning for a timely completion.

Planning for the early support of LINUX and NT platforms should receive a moderately high priority.

3.4 User Support & Feedback

It is important that the target audience (astronomer/user or developer) be understood before a release takes place. Is the purpose of the release to support outside development, or to provide end–users with complete scientific packages to do serious work? Is the purpose of a release merely to involve the outside community and get feedback from users to guide further development? These decisions need to be made early in the planning phase of a release to guide development. The announcement of a release should make it clear to prospective users what type of user (end–user or developer) the release is targeted for. Probably several releases over a span of 1–2 years will be required to meet all goals for the baseline system and the emphasis of each planned release should be made clear so that prospective users can decide when to get involved. We feel this is essential, not only to focus development, but to avoid disappointing users who may expect a release to provide something that it does not.

Further planning by the AIPS++ group is needed to prioritize development for the next 1–2 years and specify the goals and contents of each release. Our impression is that the January 1997 release will have limited impact outside the AIPS++ consortium sites and will serve mainly to get a snapshot of the system out for testing. The mid–1997 public release is unlikely to contain enough scientific applications software to be interesting to most astronomers, but it will provide the user community with an early look at the system and should provide developers with the first stable and fairly complete version of the development environment. It is important that this

version contain one or two complete scientific applications to demonstrate that a complete development environment exists, and to provide working examples of typical AIPS++ applications. A further release sometime in 1998 should provide the first fairly comprehensive scientific applications, as well as an update to the development environment.

Some mechanism for technical and user support should be in place by the time of the mid–1997 release. At a minimum this should consist of a project email address for technical support and perhaps a hotline for telephone support. Additional resources may need to be identified for platform support, preparation of distributions and patches, version control, and technical and user support.

A list of user-related concerns which we believe require attention in time for the mid–1997 release is given below.

- Provide GUIs for most functions.
- Polish user interaction with Glish to have a better feel.
- Standardize the look and feel of different applications, whether they are composed of Glish commands, part of the standard packages or e.g. AipsView.
- Provide context sensitive help.
- Provide context sensitive defaults.
- Provide tutorials for the (limited) advertised functionality.
- Provide global and data-specific history logging and editing tools.
- Provide means to save and reload parameter values.
- Provide plotting capabilities with publication quality. This includes control over fonts, sizes, weights and labeling.
- Provide reasonable (perhaps tunable) memory usage and high performance (well within a factor of 2 compared to existing packages).

3.5 Performance & Testing

It is especially important for a first package release to avoid establishing a bad reputation which gets passed on by word–of–mouth and which may take years to overcome.

There was significant concern that the performance associated with some AIPS++ operations is unsatisfactorily slow. Regrettably, few hard numbers on the speed performance of AIPS++ code are available, and the few that are, are not entirely encouraging. No systematic performance evaluation of, for example, the synthesis package has been performed. Such evaluation needs to be performed, and the code needs to be tuned so that it is no worse than about a factor of two slower than present packages, preferably much better. This problem was discussed extensively with Cornwell, Glendenning and Garwood. Clearly, the code can be sped up but it is unclear that manpower is available to implement all of the changes needed on a reasonable timescale. The committee feels that performance is an important aspect of AIPS++ and inadequate performance could result in poor acceptance by the user community.

The memory requirements of AIPS++, both for a programmer (256 MB) and a user (64 MB), tend to be larger than the workstations currently used by most astronomers. Although hardware and compiler improvements may diminish this concern over the next few years, the AIPS++ group needs to continue to investigate ways to reduce memory bloat. Memory leaks must be plugged.

The alpha testing of AIPS++ and the AIPS++ interface (Glish) has been minimal so far, yet it is a very important step towards acceptance of AIPS++ by the user community. To date, it appears that virtually no consortium astronomers have acted as friendly novice users or "alpha" testers. AIPS++ will not be a real system until it is in production use on a daily basis within NRAO and the other consortium sites. New releases should be beta–tested within the consortium for several months prior to any major release, by having the software used on a daily basis for routine data processing by the scientific and technical staff. This is in addition to the alpha–testing performed by the programming staff during development. We recommend the aggressive recruitment of consortium scientific staff for in–house testing.

4 AIPS++ Sub-Assemblies

4.1 Visualization

AipsView has a reasonable overall functionality but has inadequate publishable line graphics functionality (e.g. choice of fonts, tiling, annotation). It is probably best to freeze development and concentrate on the Glish/TK widgets and the image display library.

Completing the Glish/TK interface should have very high priority. This should be in place for the January 1997 release. It is not completely clear whether it will yield adequate line–drawing functionality from Glish. The group felt it was important to keep searching for more capable substitutes for pgplot on a 1 to 2 year timescale.

The image display library should receive fairly high priority, since interactivity of DOs depends on it. Early 1998 might be a reasonable target date for completion.

The group did not feel that it was advisable to adopt Java in the short term, but felt it important to keep track of commercial developments in this area.

4.2 Glish/User Interface

Glish is one of the best features of AIPS++ and it provides a powerful capability for object oriented programming at the level of interpreted scripts. However, Glish as it currently stands, is not well suited as a CLI, and something more approachable is needed to avoid turning off users, especially novice users. GUIs provide one way of addressing the problem. Both application GUIs and auto–generated GUIs for controlling Glish level modules or objects will help. If much interactive use of the CLI is anticipated then a streamlined command line interface to Glish is needed, similar to that provided by existing astronomical data system shells and Unix shells. It should be easy to do simple, common operations without a lot of typing. Addressing this shortcoming has a very high priority before the mid–1997 public release. We wish to stress, however, that we are not recommending construction of a complete layer on top of Glish.

There is also a need for real feed-back from novice users to evaluate user satisfaction. It would be very fruitful for the developers to spend some time with existing systems, like IDL, to gain insight into successful CLI design.

Clearly it is critical that Glish be made robust against crashing.

Some suggestions of specific methods to reduce unnecessary typing are:

- shorter class names,
- minimum matching
- default contexts (if you almost always type `im.plot()`, let the `im.` be default as indicated by a prompt)
- alternatives to the `method()` syntax when there are no arguments

- default (AIPSrc?) instantiation of most obvious objects

Specific goals for the mid–1997 release might be a number of standard Glish closure scripts that bundle standard imager functionalities. These should be accompanied with documentation, reliable failure modes, and GUIs. The logging of processing history from both Glish and DOs should also be enabled.

4.3 GUIs

There should be an auto–GUI or standard GUI for all DOs and all "standard" Glish scripts (preferably in time for the January 1997 release).

We recommend development of simple GUIs created by DOs to enable interaction with asynchronous processes (preferably in time for the mid–1997 release).

Work should be done on developing a standardized GUI for major applications.

4.4 Tasking

A capability needs to be added to allow large jobs to execute in the background (similar to ampersand in Unix). This should include provision for monitoring the progress of background jobs, interacting with jobs which require input, aborting jobs, and passing data back to the foreground context.

Ideally AIPS++ should not be tied to Glish as the only mechanism for managing intermodule (distributed object) communications. More powerful messaging system standards are sure to be important in the future, e.g. CORBA. The messaging subsystem should be abstracted and isolated so that new messaging system technology can be substituted in the future. In addition to protecting AIPS++ from technological evolution in messaging systems, this is important if AIPS++ is to be an open system capable of communicating with external non–AIPS software.

4.5 Parallelization

Some modest level of effort needs to be devoted to the discussion of parallelization, including the use of multi–threads, but this is clearly a goal for a future release. Taking advantage of specific software development talent on an as-needed basis is probably realistic at this point.

Within the modest overall priority of this topic, most attention should be given to supporting parallelization within small multi-processor machines,

which are likely to become increasingly common within the general user community.

4.6 Single Dish Data Analysis

Some progress is being made at last on the single dish data analysis front. The committee was shown a prototype application (SDCalc) which had a good user interface for operations on single scans. While we felt this was a good prototyping experience we felt it lacked the scope necessary to deal with multi-dimensional datasets and the high data rates which will, in general, be needed. The committee strongly recommends a multi-dimensional approach which is more solidly based on the standard AIPS++ MeasurementSet. In particular, we envision users will need sophisticated bulk–analysis tools to cope with large datasets with several of the following dimensions: two spatial, one or more frequency, one polarization, one beam and one time. The flexibility of the MeasurementSet can accommodate these needs more easily than a FITS datacube approach. A multi-dimensional user interface approach will make this flood of data more comprehensible to the astronomer.

One area of synergy which appears bypassed is the interface between traditional single dish processing and the AipsView visualization world. For example, AipsView is capable of selecting and creating spectra from datacubes. These spectra, or collections of spectra, should be written into a MeasurementSet which could then be manipulated by SDCalc.

There was substantial concern that the envisioned needs of the GBT might not be met on an appropriate timescale.

4.7 Help/Documentation

All AIPS++ releases, starting with January 1997 should contain full application oriented documentation to enable users to run AIPS++ effectively. This documentation should include fully–worked examples using a test data set shipped with the release, and a description of the methodology being used by each application. The documentation should preferably be shipped with the software, so that network problems do not get in the way of the user. There should be an extensive tutorial on general AIPS++ and Glish use, including annotated scripts. Specific requirements include:

- Module/Object/Method listings which are more hierarchical. All the user now sees is a very flat listing of methods.
- Context oriented help from Glish.

- Complete postscript versions of all the web documents.
- A search engine to enable program developers to find relevant code.

4.8 Synthesis and VLBI processing

Polarimetric self–calibration and NNLS imaging are the jewels of the January release. This capacity is unique. A good implementation of these is most important. The speed of the processing, however, is a major concern. Conventional self–calibration and deconvolution should not be a factor of 5 slower than comparable systems in the January release, and preferably no slower than a factor of 2 by the mid–1997 release. If these speed requirements can be met, then spectral–line, multi–IF/multi–frequency processing should be addressed. Mosaicing and wide–field imaging are very desirable but should have lower priority in both January and mid-1997 releases.

Extending the above synthesis processing to include the functionality and user friendly interface of DIFMAP would be valuable for the VLBI community, as well as being of wider appeal among synthesis imagers. Once a better user interface has been produced, this should be comparatively inexpensive in relation to the product that can be delivered. DIFMAP should be used as a reference for performance.

Planning for VLBI processing is in its infancy within the project. The current requirements and planning for VLBI processing do not appear to be well integrated within the project. We appreciate the lack of manpower currently commited. However, longer term planning for VLBI is sorely needed.

4.9 Measures

The arrival of the measures toolbox was a welcome development. Full support of VLBI and pulsar applications will be important additions for the coming months. There is some concern regarding the need for extensive retrofitting of code to introduce the toolkit at all the necessary levels. Tools like this would ideally have been available in the early stage of application development. At this moment several application development projects can still profit from continuing effort in this area, notably VLBI and interferometer calibration.

We can see good reason to make some of the functionality of this system available at the user interface level in an early stage. Priority might be given to making the unit system available, in particular the use of physical constant and unit conversions from Glish. Furthermore, coordinate trans-

formation (e.g. from B1950 to J2000) would be interesting as well as Doppler calculation (e.g. topocentric to LSR).

It is important that all physical constants, coordinate systems and methods of conversion be fully documented with appropriate references.

4.10 General Toolkit

The AIPS++ library provides the programmer with a rich and versatile toolkit. At present, the main pending developments in the Table System part of the library appear to be ensuring that features such as multi-user synchronization and locking are implemented.

There are two critical items with regard to the libraries: Firstly, it is vital that memory leaks be hunted down and eliminated. Otherwise mysterious system crashes are sure to occur. Secondly, because there appear to be misgivings that AIPS++ applications are slow compared to the competition, every effort should be made to ensure that the library classes are optimally designed for maximum speed.

The time may be nearing when the core functionality of the library should be frozen, with a switch of priority to writing applications.

5 STAG Membership

Robert Braun (NFRA) chair
Jayaram Chengalur (NCRA) not present for this meeting
Roger Foster (NRL)
Dennis Gannon (Univ. Indiana)
Walter Jaffe (Univ. Leiden)
Lee Mundy (Univ. Maryland)
Bob Sault (ATNF)
Lister Stavely-Smith (ATNF)
Dave Shone (NRAL)
Doug Tody (NOAO)
Huib Jan van Langevelde (JIVE)
Tony Willis (DRAO)
Al Wootten (NRAO)