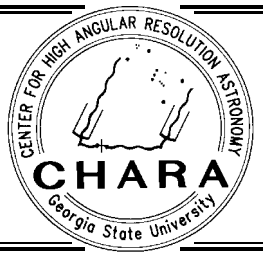

CHARA Technical Report

No. 40 5 October 1996

Preliminary Software Issues

M. Collins & T.A. ten Brummelaar (CHARA)

1. INTRODUCTION

No modern corporation would attempt to build a factory in a piecemeal fashion and business leaders know that presuming that a complex software project will "fall together" at the last minute usually results in mediocrity, if not chaos and failure. Like factories and buildings, complex software projects must be integrated into the larger enterprise while being carefully planned and coordinated. Thorough planning and good coordination of a project with average coders will consistently beat disorganized, unplanned coding, however bright the coders may be. Far from being a peripheral issue to the success of the overall project, the computer system is a major part of the CHARA system. This includes supporting many of the machines involved such as pointing the telescopes and carrying the "data pipeline" all the way from initial data acquisition right up to the final publishable paper, which is the ultimate goal. But there are also the "interface issues" which means planning the extent to which the computers will help and organize the operator by organizing information, listing options and giving feedback as to exactly what is happening.

2. WHAT NEEDS TO BE DONE?

Software related tasks for the early phases include:

- Start a high level software analysis which requires that everyone understand:
  - A business analysis (ultimately: what is important and what is our strategic vision?).
  - Outputs (deliverables) from the system.
  - Inputs and feedback into the system.
- Produce a list of tasks that the software must handle.
- Group together like tasks and set priorities.
- Come up with data dependency diagrams to show the order in which data must be produced.

[1]Center for High Angular Resolution Astronomy, Georgia State University, Atlanta GA 30303-3083
Tel: (404) 651-2932, FAX: (404) 651-1389, Anonymous ftp: chara.gsu.edu, WWW: http://www.chara.gsu.edu

- Build a tentative list and description of C++ style "objects" needed and the data that must be passed between them (do not write the full code yet, just list the public variables and member functions).
- Diagram the computer (hardware) layouts and give a brief description of what each one does (especially those likely to be facing a heavy workload).
- Choose the network operating system.
- Start writing up software coding standards for all coders, including examples.
- Test the speed of the Fast Fourier Transform and other time-critical functions to determine if special hardware will be needed.
- Rough out (in pencil for now) any needed screen diagrams.
- Write up the pull-down menus in the Windows-like environment of the command console and what each option does (or should some parts look like an Internet browser with lots of hot-links?). This will force people to think through the interface issues.
- Design and write a prototype telescope controller for Hard Labor Creek here in Georgia. This will also help estimate worker-hours needed to write and test the rest of the software.

3. INTERFACE ISSUES

While it is too early to decide if the interface should look like MS-Windows, we can start to look at big picture issues. For example, should the interface emphasize ease of use and minimal effort by the operator or should we presume we will have a well-trained "power user" more at home with a simple command line? Just how much operator intervention do we expect in each observation of an average target? To push down operating costs in the long term, we want to avoid having specialized and expensive labor performing tasks that could be done by lower cost labor. Ultimately, only astronomers (and engineers) can analyze, interpret, and publish CHARA's information but (theoretically) any computer literate operator could point the computer controlled telescopes at their targets to collect data.

4. GENERAL APPROACH TO SOFTWARE DESIGN

Sometimes it is easiest to start at the accomplished goal and track backwards to see how we got there. For example:

```
goal:   many papers were published
because there was much information to publish
because much data was quickly and thoroughly analyzed
because the data was organized in an easy to use form
because there was plenty of good data needing organization
because each available night, many good targets are observed
because the machines work efficiently on their targets
```

```
because the machines and their interfaces were easy to use and efficient
because the machines (and software) were well integrated
because the systems (including the computer systems) were well designed.
```

Note that this shows that the "analysis computers" here at Georgia State are an important part of the "CHARA information pipe-line." We will need to have a great deal of data storage space, easy-to-use reformatting programs, organizing programs, analysis programs, and so on, to turn the data into the goal of published papers. Clearly, each step of the information pipeline needs to be elaborated in the context of the goal.

5. CONCLUSION

It is important that some time be given to thinking about the overall issues in the CHARA Array control system. A great deal of time and effort can be saved in the future if the global issues are resolved early on. The main issues are:

1. What is the primary goal of the Array, and what should the control system do to help achieve that goal?
2. What is the system breakdown? What inputs and outputs does each subsystem need?
3. What are the sizes and transfer speeds required for this data?
4. What are the communication standards?
5. What is the coding standard?
6. What is the error trapping/reporting protocol?
7. What sort of interface is appropriate?
8. Which OS should we use (DOS/Windoze/OS-2/Unix/VxWorks/In-house) ?
9. Which programming language should be used (C/C++/ FORTRAN (only kidding folks)).
10. Who exactly is going to write all this code?

There is a danger of finding ourselves on the mountain with a lot of hardware and no way to control or integrate it, or having a system only one person knows how to work, and that person may not be there all the time.