Intelligent AT Fault Diagnosis System Report to date

R. Landau

November 19, 1992

1 Introduction

The goals of the Fault Diagnosis System are to monitor the operation of the AT and to detect and diagnose failures and problems as accurately and in as short a time as possible. This would increase the uptime and data quality for scientists using the array and thus their productivity.

The AT is an array of six radiotelescopes connected as an interferometer. Each antenna is outfitted with about 1000 monitor points that report the health and stability of specific components and signal pathways in the array. There are an additional 350 monitor points from the central site (common to all antennas). It is the data from these points as well as the astronomical data (visibilities) produced by the array that the Fault Diagnosis System (FAULTY) acts on. Each of these sources typically reports every 10 seconds. The system also tries to do preventive maintenance by monitoring trends in selected monitor points.

The AT thus appeared to be an ideal environment for the application of statistical process control and the use of a knowledge system. The problem was well defined, although complex. The AT's operators and users are professional scientists so feedback on the operation of a diagnostic system would be of high quality. And the AT is completely under our control so that test periods can be provided and faults can be induced at needed stages in the project's development.

Consequently, the application was conceived as an IISE Demonstrator Project to demonstrate the benefits of sharing the knowledge and active participation of colleagues from other divisions of CSIRO, in particular the Divisions of Mathematics and Statistics (DMS) and Information Technology (DIT). The AT provides an excellent demonstrator platform for such a knowledge-based process-control system. It has high visibility and is highly regarded for its engineering excellence. No aspect of its operation is classified, and it is seen by a broad international spectrum of users and visiting VIP's. *[The previous two paragraphs are from R. Ekers.]*

The system is also designed with an eye to its generalization and application to other large instruments or to manufacturing processes.

The system can be conveniently factored into four sections: the knowledge base—containing information about the monitor points and failure modes of the AT, the data paths—programs that acquire, process, and archive the data, a diagnostic section—the intellegent core of the system, and a user interface—a user-friendly (we hope), interactive, means for display and investigation.

In what follows I describe the evolution of *Faulty* to its present state from its start eighteen month ago and anticipate the next six months of effort.

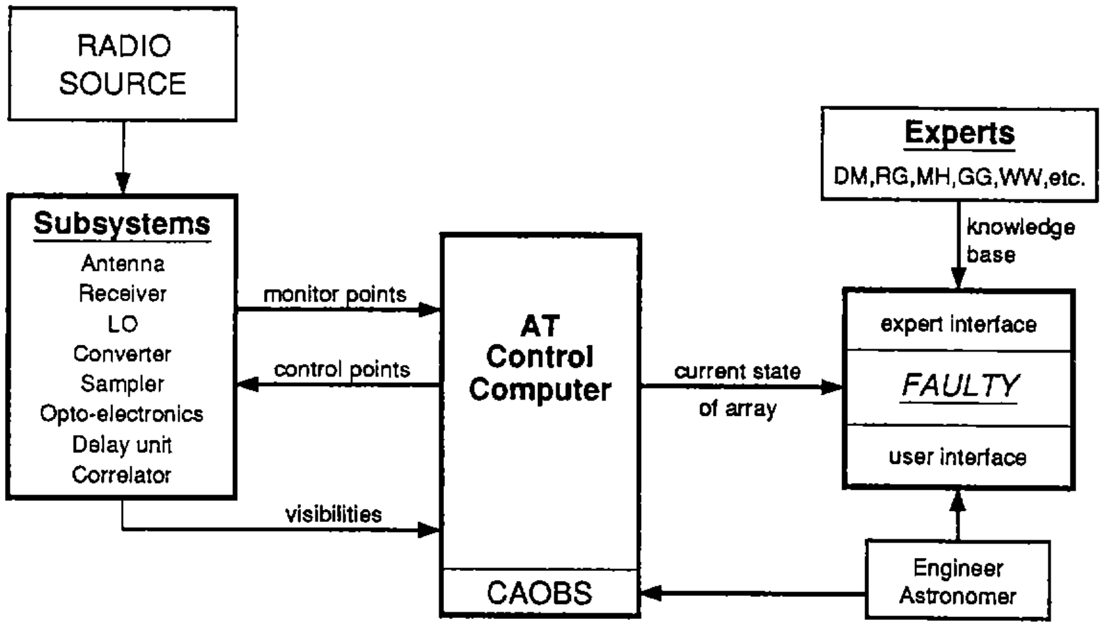
2 Conceptual stages

The project was conceived as a natural development of the monitor system devised for the Compact Array. This scheme provided ready access to the monitor data and found favor with the technical staff. In time, with increasing familiarity and faith in the reliability of the data, ambitions grew—surely expertise could be added to the system to provide diagnostic assistance

An initial interdivisional meeting was held at Narrabri with Bob Golomb (DIT), Doug Shaw (DMS), John O'Sullivan (RP), and Alan Young, Mal Sinclair and Ron Ekers (ATNF). A wide range of concepts were explored and realistic expectations were established. The meeting also exposed DIT and DMS to the realities of the Compact Array monitor system. *[The previous two paragraphs are from M. Kesteven]*

In the beginning, I knew nothing about expert systems or X-windows. The first months were spent trying to determine if the diagnostic section could be built as an expert system and if the user interface could be done in X. At this time Linda Strickland and Margot Nichols, librarians at DIT and RP, were invaluable, copying material and arranging inter-library loans.

When the array fails, many monitor points may report errors at once and it takes an intelligent program to digest all the information. This behaviour and the scale and complexity of the system suggested an expert system. Furthermore, it was soon apparent that the system would rely heavily on the knowledge of the experts who designed the eight subsystems of the AT. The diagram below shows the interaction the system would have with the AT, with the users (both astronomers and engineers) and with the experts.



AT Fault Diagnosis System - Interactions

Their input comes in three phases: initially they supply the information about monitor points—what they monitor, what their normal operating range is, and what other conditions this range might depend on. Later the experts describe the ways they know the array can fail and their favorite diagnostic sequences when it does. Finally they accumulate experience using the first versions of the system. Their refined diagnoses allow the programmer to produce the next version. Thus some kind of expert system seemed appropriate.

However all of the diagnostic expert systems described in the literature were of much smaller scale than ours would have to be, and they all ran fairly slowly, largely because they were inference engines: they followed a large set of rules that tried to *deduce* what fault had occurred. I was able to develop a scheme of "signature analysis" which incorporates the experts' knowledge but which *calculates* rather than infers the fault. The subsystems are divided into logically diagnosable modules (usually one or two physical modules) with approximately 32 monitor points each. For any fault in one module, each monitor point is either in range or out of range, a binary condition. This sequence of 32 conditions is the signature of that fault for that module. A dictionary of faults can thus be built up for each module. The current (perhaps failed) state of the system also has a signature which can be compared with each dictionary entry in turn. A perfect match needn't be found—the closer a fault is to the current state the more probable it is. If no fault is sufficiently close, the current state is a new fault which can be named and inserted in the dictionary. Timings showed that a signature could be compared and a "closeness" calculated in 33 μsec, so that the entire array could be diagnosed by this method comfortably on a 10-second cycle. Furthermore, it would be a simple matter to induce faults in the system, record the signature that resulted, and add that to the dictionary.

At this stage I discussed building the core of the diagnostic section around a signature analyser with Craig Lindley at DIT. He suggested it was a sensible idea and helped orient my thinking about expert systems in general. In August 1991 the biennial meeting of the *International Joint Conference on Artificial Intelligence (IJCAI-91)* was held in Sydney. Several papers on diagnostic systems were presented, confirming the impression that only small applications were actually being tried out. The workshops, papers, and corridor discussions persuaded me that it was unlikely I was ignoring any important area relevant to our diagnostic project.

At DIT I also met an X-guru with whom I had a useful discussion and who directed me to the best reference books on X, which I was able to buy. He suggested that X was the appropriate resource with which to build the sort of interface I had in mind, but that accessing it from Fortran might prove difficult. Subsequent experience with Digital's inplementation of the Fortran binding to X has proved him right. VMS and Fortran were chosen for compatibility (they are the basis of the AT control and acquistion software) and because it would allow other programmers to maintain the system easily.

Several months were then devoted to developing tools to run separate tasks in detached processes, to synchronize them, and to pass data from one to another, all procedures heavily dependent on calls to the system services of the VMS operating system. I then felt in a position to assemble the pieces.

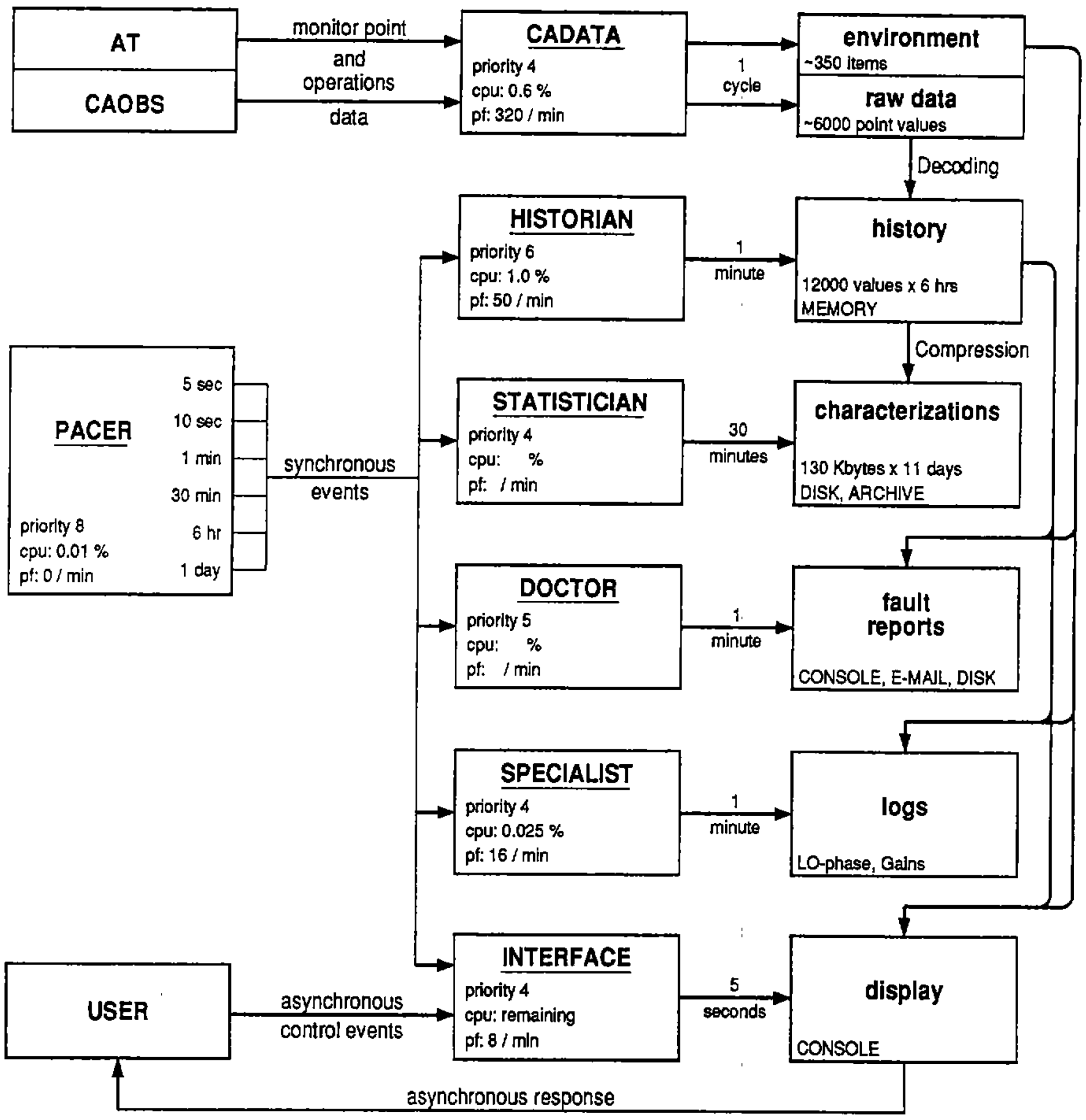
3 The Structure of *FAULTY*

Seen from inside the computer, *Faulty* consists of seven tasks running in separate "detached" processes. These are the underlined names in the boxes of the *Tasks* diagram on the next page.

The tasks are synchronized by event flags; that is, they are typically hibernating, and wake up to perform their function only in response to an event generated by another task. The exception is the Interface, which also reacts to user input. One task, Pacer, running at high priority, does nothing but raise event flags at particular times of day, which other tasks are waiting for. A flag every ten seconds, for example, starts Historian reading the raw monitor points data into a history array in memory. After one minute Historian raises flags for Doctor to start a diagnostic cycle and for Specialist to write another record to each of its log files, which are closed at midnight in response to another of Pacer's flags. CADATA is a utility written by Dave McConnell that makes the raw data from the telescope available to any application wishing to use it. Raw data is updated every observing cycle (between 5 and 32 seconds) so CADATA, although logically part of *Faulty*, does not make use of the event flags.

The tasks share data amongst themselves by writing and reading "global common sections", regions of memory or disk that are part of the virtual address space of each participating task. There are three of these: the raw and environmental data that are overwritten by CADATA every observing cycle; the history array, containing a six hour record of the decoded raw data, maintained by Historian; and a global section of parameters that glue the tasks together.

Seen by the astronomer or engineer, *Faulty* has two faces, one visible and one hidden. The



AT Fault Diagnosis System - Tasks

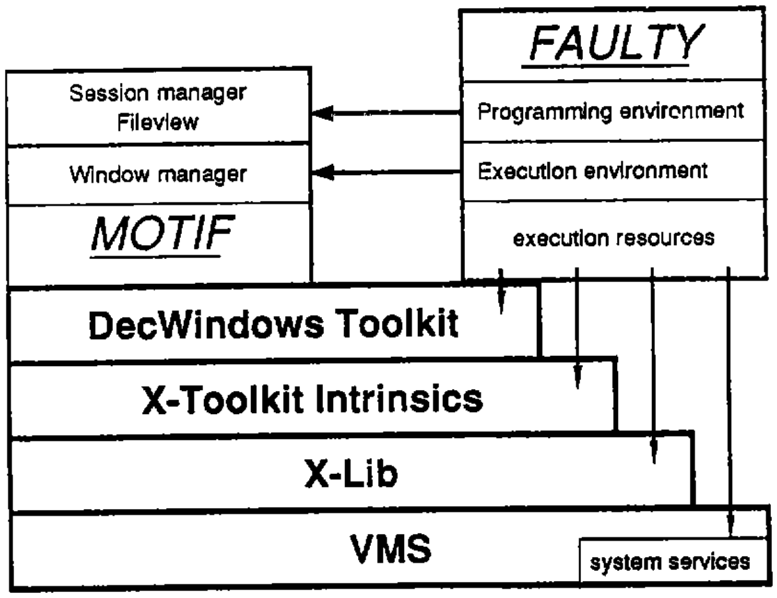
Interface is present whenever a user is logged on and provides graphical and statistical tools for investigating the current state of the array and its past history. This itself has already proved a powerful diagnostic facility. The data available are the high and low value of each monitor point over each minute for the last six hours; and, for the last 11 days, a statistically robust characterization of each point over 30-minute intervals. This characterization gives a good description of the point with emphasis on possible aberrant behaviour. It includes the median value over 30 minutes, the maximum and minimum, the interquartile width (a robust analog of the standard deviation), the trend in time over that interval, a count of jumps and spikes, and an estimator of intermittancy—the number of times the value has gone out of range and then come back in. In developing these estimators I have had the help of Doug Shaw of the Division of Maths and Stats. The task Statistician characterizes each point of the array in this way every half hour.

To uncover causal effects it is often useful to estimate the correlation of one monitor point with another. A robust analog of correlation is the non-parametric degree of association. Algorithms for calculating the degree of association and its uncertainty have been developed for both the minute-ly data and the 30-minute characterizations.

The hidden face of *Faulty* is Doctor, the task that performs the fault diagnosis. It runs in the background regardless of the presence of the interface. Doctor consists of three sections: Its core is the signature analyser discussed above. This is surrounded by a pre-filter and a post-filter. The pre-filter is aware of the observing environment and can decide, for example, that if the observer has just changed frequencies not to be concerned by abrupt changes in total power. It can also decide which module to diagnose first and which power supplies to check before the dictionary of faults is consulted. The post-filter recognizes repeated diagnoses of the same fault (before the fault is corrected) and supresses the generation of a fault report in these cases.

The report generator is the third section. A fault report contains a captured state of the system from 12 minutes before to 4 minutes after the fault, together with all probable and possible diagnoses with their levels of likelihood. The severity of the fault determines how it is announced. All faults are logged. A synopsis of most faults is e-mailed to the appropriate engineer. Severe faults generate messages on the console if a user is running the Interface (we expect the Interface to be running as a matter of course during both maintenance and observing); otherwise they can notify the observer through the observing program, CAOBS.

Seen by the programmer, *Faulty* can be described by the *Platform* diagram below.



AT Fault Diagnosis System - Platform

Faulty runs under VMS on a VaxStation 3100 with 32 Mbytes of memory and one Gigabyte of disk. The tasks execute programs written in Fortran from which calls to the operating system are made for services such as process creation and inter-process communication. The initial state of the User Interface is described in a C-like language which builds a hierarchy of X-windows with associated attributes (widgets). Widgets receive user events (clicks of the mouse button, for instance) which are queued by X and eventually generate subroutine calls that do the work appropriate to the event. This may include manipulating the widget by calls from Fortran to the X-toolkit. Graphics are drawn in windows with calls directly to the X-lib collection of graphics primitives.

Numerous bugs. incompatibilities between X and Fortran, and errors of documentation are met by imaginative programming.

4 Current status

Version 1.4 of *Faulty* has been released and is available for use as I work on version 1.5. These are the functions it is currently performing:

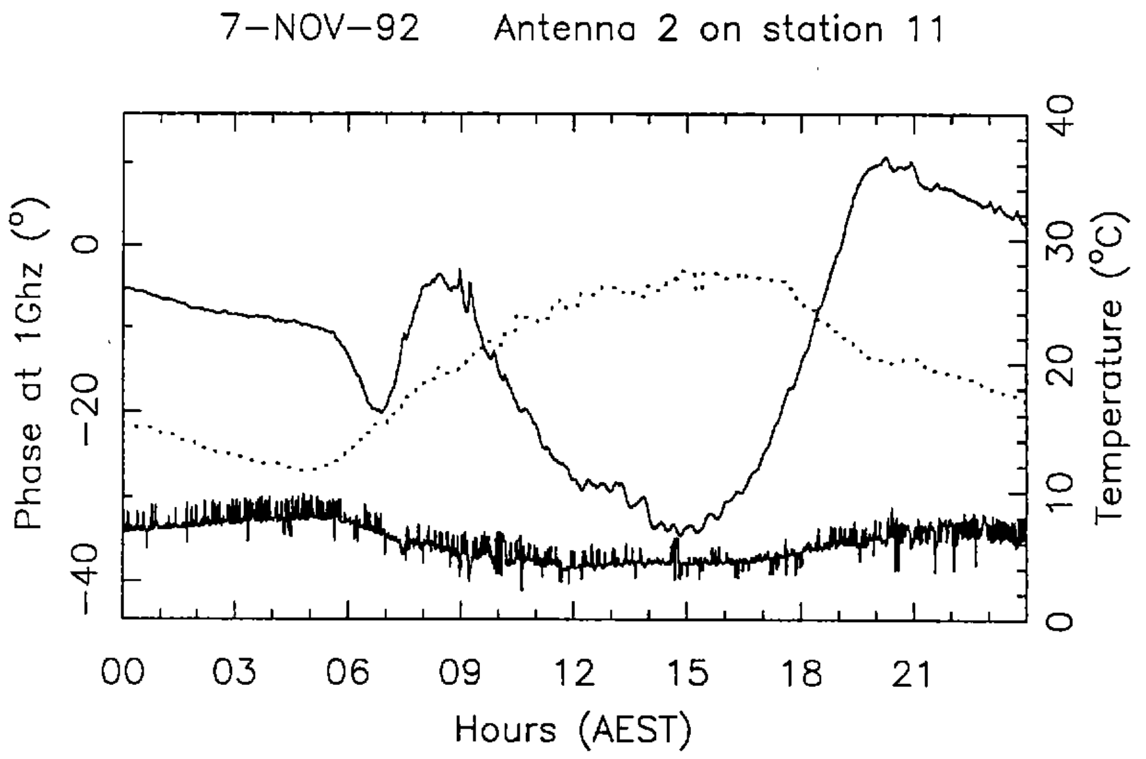
Historian continuously produces its history file from the raw data provided by CADATA.

Doctor diagnoses only the receiver subsystem at this time, using a dictionary of 72 faults, determined largely by inducing them during a recent maintenance period and capturing the resulting state of the array with *Faulty*'s snapshot function. Faults were also induced in the Local Oscillator (LO) subsystem, and this will be the next diagnostic section added.

Specialist runs continuously writing data every minute to each of two log files. It was never part of the original conception of *Faulty*'s functions, but was put together in a few days at the simultaneous and independent requests of two engineers because it was simple enough to assemble, and has since proved useful enough to incorporate.

The gain log monitors the gains in the four frequency channels at each antenna. It was originally intended to search for unexplained gain changes that might have affected observing a day or more before they were brought to the attention of the engineers. Since then, as an unanticipated result, the detailed variability seen in the output plots has suggested a better way to apply the gain calibration.

The LO phase log monitors the LO phase stabilization system, watching the round trip phase to each antenna together with the ambient temperature and two power levels in the stabilizer modules. A sample output is shown below.



The power levels are monitor points, the temperature is an environment variable originating from the site's weather station, and the round trip phase is read from a PC in the screened room. The purpose of the log is to examine these disparate sources looking for rare jumps in phase, uncorrelated with temperature, that might be produced in the stabilization module.

We can configure Specialist to add other special-purpose logs without difficulty.

Statistician can generate a statistical characterization over any time interval selected by the user, but currently lacks its archive facility.

The diagnostic section of *Faulty* relies on the intelligent use of the upper and lower limits for the monitor points, and these have been set with some abandon in the past. It is now possible for Statistician to calculate enough statistical information over a "normal" period of operation to set these limits automatically. Power supply voltages and currents are particularly suitable for this procedure. There are also a fair number of points (receiver amplifier biases are good examples) whose limits should be different from antenna to antenna because some components are adjusted differently in different modules. A single pair of limits, set to include all these variations, are too wide to detect many types of abnormal behaviour in any one receiver. A new points database has been compiled which will accomodate antenna-specific limits, and Statistician will again be used to determine them.

The User Interface is usable and friendly, but not complete. It primarily lacks the facility for displaying the block diagram of a module that has been found faulty, indicating good and bad monitor points. The problem arises in part because the needed diagrams are in many different computer-readable and occasionally unexchangeable formats in many different locations and in part because Digital's own DDIF format into which the other formats may have to be converted, itself is not easily displayed in X.

Output from the Interface appears in three ways.

The last six hours of data from selected monitor points is redrawn every minute. Corresponding points for different frequency channels are plotted in different colors in one window. There are currently two windows, but eventually there will be six. Hardcopy output can be produced, such as that on the next page, showing excess LO phase noise on point L42IF (part of the phase stabilization system) outside the 10-minute period around 14:40 hours. A simultaneous plot of antenna elevation (upper frame) shows that the elevation drive motor was stopped for these ten minutes. This coupling is currently being investigated.

Numerical output requested by the user (statistics, for example) currently appears in a separate window from the Display. Eventually the display will fill the screen and present all output.

The Interface also controls the appearance of various widgets to indicate the state of the array: the observing frequencies, for instance, or the locations of the antennas. These are updated at various times in response to event flags from Pacer.

The selection of monitor points to be displayed is also still crude. Eventually they will be selected from a menu—presently their names must be known and typed. Except for this, the interface is, by intention, a mouse only affair, so that it can be used equally by adept and inept typists. The following input features are available:

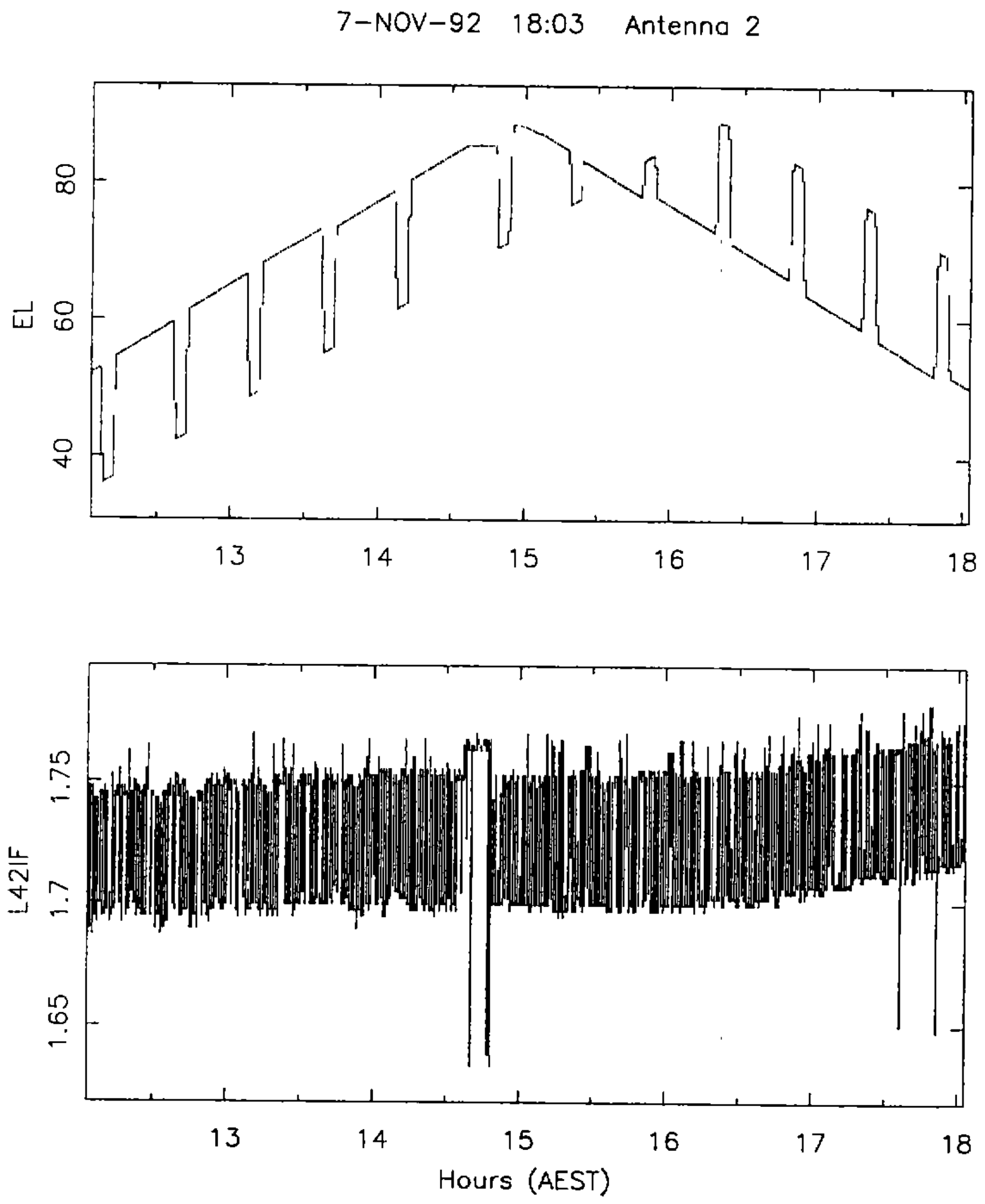
Antennas and frequency channels can be selected from icons around the edge of the display window.

Functions are begun by clicking on their buttons.

Interaction with the graphical displays is controled by the cursor.

Four separate states of the Interface can be saved and recalled for later use.

Other users learning to drive the interface have had no difficulties so far.



In addition to the ~1000 specific monitor points in each antenna there are eight general monitor points wired to BNC connectors to which any voltage source can be connected. *Faulty* is currently monitoring the phase of the 128-MHz sampler clock with a vector voltmeter connected to one of these points in an effort to understand an intermittent problem known as "early-normal" that has plagued the array since its beginnings. We are looking for correlated variations between this signal and monitor points in the LO system.

5 Future developments

To complete *Faulty* the following will need to be done:

Doctor: Add the remaining subsystems to the diagnostic section. This will require more fault simulation experiments. (Some of these are scheduled for the December maintenance period.)

Interface: Expand the display section from two to six graphics windows.
Add routines to display block diagrams. George Graves is helping me collect the relevant diagrams.
Add a menu to select monitor points for display.

Statistician: Add the archiving facility to catch data more than eleven days old.

Database: Install the new points database with antenna-specific limits. This is scheduled for December.

Training: Find a "Friend of *Faulty*" and train him or her to maintain the system after June 1993. Adding new faults to the fault dictionaries will be made as automatic as possible; nevertheless, modifying the pre- and post-filters may take some skill.

Full diagnosis of the array will also require the availability of some hardware, not currently in place. Monitoring the correlator, delay units, and part of the opto-electronic subsystems will need an Antenna Control Computer (ACC) in the screened room. Monitoring the conversion subsystem requires the C31 and C41 interface modules. Monitoring the antenna drives, the power system, the turret, the subreflector, and the pedestal room power supplies requires the installation of datasets in the antenna pedestal rooms. Each of these pieces of equipment is scheduled to be available in June 1993. Until then *Faulty* will be able to diagnose the receivers, the LO, the samplers, and the part of the opto-electronic subsystem that resides in the antennas.

I believe *Faulty* would benefit significantly from the experience accumulated by its engineer users in a trial period following its release. Most other computer programs are debugged until they work and then they cannot be improved much, although they can be expanded or ornamented. *Faulty* is in the nature of a probabalistic classifier (of diagnoses). Even if it functions, its diagnoses are likely to be refined as its algorithms and filters are improved. The first trials with *Faulty* are likely to produce some very good and other wholly bizarre diagnoses. I imagine that a two or three month trial period followed by a one or two month programming period will produce a more intelligent, incisive and useful diagnostic system.

6 Interdivisional participation and external interest

My contacts with DMS and DIT, described in sections 2 and 3, have been useful, though not extensive. A general problem was to identify needs of the project, arising naturally, not created artificially, that could be satisfied by people in the other divisions. Only two were recognized, and these led to modest, fruitful, exchanges of ideas.

More specific problems also arose. In the case of DMS, I found my colleague enthusiastic but forgetful. Promised material was not sent, and calls were often not returned. After a while I usually let the matter drop, although others have suggested that I ought to have been more persistent. Augmenting the problem is the insistence of the two specialists in charge of the network at the AT and at DMS that each arrange the mailer protocol his way. It has been impossible for two years (without a deeper knowledge of network resources) to send e-mail from ATNF to DMS (try mailing anyone@syd.dms.csiro.au, for example). Nevertheless, Doug and I are still in communication and refining some of FAULTY's estimators.

In the case of DIT, my contact was helpful, but it was made clear that his time was limited and almost a personal favor to me because his managers did not consider the AT project in line with DIT's goals.

Other organizations have expressed an interest in our project over the course of its development. Those that have led to meetings or exchanges of documents are:

- Jason Stryomilio and a group of engineers at the Anglo-Australian Observatory wished to include a fault diagnosis system in the 400-element spectrometer they were designing for the AAT. They were glad to describe their instrument to me and discuss its suitability for a modified system similar to *Faulty*.

- Andrew Jennings and Adam Kowalczyk of Telecom Research Laboratories were interested in our approach to fault diagnosis. However they are testing a neural network scheme for diagnosing the phone system, so there is not much overlap.

- Allan Wells and Dennis Jarvis of DMT were interested in the details of our signature analysis method, and were contemplating the design of a toolbox of routines to standardize and represent a knowledge base (like our monitor points database) in an accesible and extensible manner and onto which a signature analyzer could be easily layered.

- The publishers of *Artificial Intelligence in Australia : Research Report* have asked for a description of our application for inclusion. This is an annual survey (since 1976) providing communication among practicioners of the art.

- Helen Sim and I have collaborated on an article in ICON (September 1992).

We may have an opportunity of presenting our results to the next biennial *International Joint Conference on Artificial Intelligence (IJCAI-93)* to be held in August.