

Telescope Ground Support” in Proceedings of the 1986 Goddard Conference on Space Applications of Artificial Intelligence and Robotics.

*ST Proposal Entry Processor System Requirements*, 1987, Space Telescope Science Institute.

Portions of the proposal Selection function could be converted to a Telescience environment. This would eliminate the need for mailing proposals to referees and even for the Telescope Allocation Committee (TAC) to meet in the same room when selecting proposals. Once the proposals were in machine readable form, the distribution to referees and the collection of referee scores could all be handled electronically. A non-Telescience but still useful software tool would aid the TAC meeting by identifying scenarios where accepting proposals with the highest referee rankings violates resource limits or other constraints. The actual TAC meeting itself is not so easily replaced by distributed users communicating electronically. Existing technology does not provide the flexibility and high bandwidth which can sometimes be achieved in a face to face meeting.

## **7. Conclusion**

The Proposer Entry Processor (PEP) system at STScI is a set of software tools which support the Entry, Evaluation, Selection, and Transformation of HST Proposals. With the addition of the Remote Proposal Submission System (RPSS), the PEP Entry Subsystem can now operate as a Telescience environment and provides the HST user with a faster and more responsive method of specifying exposures to be executed by the Hubble Space Telescope. Adding feasibility and proposal preparation tools to RPSS will further aid the user in creating efficient and scientifically productive HST proposals. Telescience operations of PEP can increase the system's usefulness while reducing its operating costs. Telescience can do more than just save trees.

## **Acknowledgements**

The following people, while not authors of this paper, were instrumental in the development of the PEP system: William Cohen, Marc Damashek, Tom Hornick, Steve Lubow, Don Rosenthal, Steve Shore, Lyle Sutton.

## **References**

- Call for Proposals and Proposal Instructions*, 1985, Space Telescope Science Institute, STScI SC-02.
- Hornick, T., Cohen, W. and Miller, G. 1987, "A Natural Language Query System for Hubble Space Telescope Proposal Selection" in Proceedings of the 1987 Goddard Conference on Space Applications of Artificial Intelligence and Robotics.
- Jackson, R. 1987, "Expert Systems Build By The 'Expert': An Evaluation of OPS5" in Proceedings of the 1986 Goddard Conference on Space Applications of Artificial Intelligence and Robotics.
- Johnston, M. 1988a, "Automated Telescope Scheduling" in *Coordination of Observation Projects*, Cambridge University Press.
- Johnston, M. 1988b, "Automated Observation Scheduling for the VLT" in preparation.
- Miller, G., Rosenthal, D., Cohen, W. and Johnston, M. 1987, "Expert System Tools for Hubble Space Telescope Observation Scheduling" in Proceedings of the 1987 Goddard Conference on Space Applications of Artificial Intelligence and Robotics.
- Rosenthal, D., Monger, P., Miler, G. and Johnston, M. 1986, "An Expert System for Hubble Space

necessary to have the observer completely specify the observations needed, which will be executed when conditions are best.

It may no longer be sensible to grant a user a fixed block of time and have the user directly control the facility in real time. The HST experience with proposal forms, syntax, remote entry, and validation are all directly relevant where the user does not make most of the telescope operation decisions in real time.

*Telescience* has been described as providing direct, iterative, and distributed user access to the remote device. Clearly Telescience is not possible for the user directly controlling HST. However, some of the software systems which take the proposer's information and convert it into spacecraft commands can have Telescience aspects.

With the development of RPSS and with the distribution of RPSS software to remote user sites, the PEP Entry Subsystem now operates as a Telescience environment. RPSS allows the HST user to Enter, Edit, Validate, and Report on their proposal. RPSS allows the person most familiar with the scientific requirements to create a set of spacecraft activities to be scheduled in SOGS/SPSS.

Additional software tools to aid the user in preparing a HST proposal could also be added to the RPSS system. Such a tool might aid the user in selecting the configuration, mode, filter or grating, and optional parameters. This tool would be akin to the AI system, considered in Section 2, which would take the data requirements and determine the spacecraft activities. Another tool might combine the user's information about the target brightness with the choice of instrument and calculate an exposure time for the desired signal to noise ratio. In fact, prototypes of both of these tools have been developed at the Space Telescope European Coordinating Facility (ECF).

What of the Evaluation Subsystem functions? Can these be made more direct, interactive, and distributed? A Duplication function might be provided as a part of an HST Archive query system. The PEP Duplication tool could be enhanced to take a RPSS file and check for duplications against the archives. However, a RPSS version could not check for duplications against other GO proposals. Such a capability would require GO access to other GO proposal data, and this would violate the data privacy requirements.

The Resource Usage tool should be distributed if the users are to verify that their proposals meet any TAC imposed resource constraints. However, there would be a danger that the user would attempt to fine tune the proposal to reduce the resource usage and due to extra timing restrictions make the proposal much harder to schedule. The Resource Usage estimate is made without creating an actual schedule and without knowledge of the actual orbital events. The orbital events seldom match the statistical assumptions of the tool, and any attempt at micro-scheduling are doomed to failure.

The Feasibility tools are the ones most suited to being made available to the user and in a distributed fashion. These tools would help the user to identify inconsistent or impossible to schedule observations at an early enough stage to allow modification by the person who knows the scientific needs best. The sooner these tools are added to RPSS, the sooner the proposer will be able to know that they have asked for legal AND feasible spacecraft activities. There will always be feasibility problems which only appear when actually scheduling the activities, but the more problems which are caught earlier, the better.

loading the proposal in the data base is all outside the control of the user. While RPSS allows proposals to be entered in the PEP EDB, the user has only one-way, one-time, single-proposal access to the EDB. It is not possible for a RPSS user to corrupt large portions of the PEP data base.

#### **5.4 RPSS Hardware**

The RPSS node is a MicroVax2 with 7 megabytes of memory. It has two 75 megabyte disk drives. Additional memory and disks are being purchased in order to reduce execution time for the Validate function and to give the users larger disk quotas.

#### **5.5 Advantages for the RPSS User**

It may seem like much more work for the GO to use RPSS instead of just sending in his proposal on paper forms. However, there are advantages for the GO using RPSS. First, there is the accuracy of the proposal. With RPSS, the GO knows that the proposal the Institute has in its data base is an exact match to what the GO entered. There is no risk of entry errors or of unreadable entries on the forms. Target coordinates are a prime example of where entry errors might have disastrous results.

Another advantage is fast turn-around on errors. A GO using RPSS can find and fix problems with the proposal quickly. A GO relying on paper forms must wait for STScI to enter, validate, evaluate errors, contact the GO, and make corrections. If the error correction process drags on too long, then the proposal may miss scheduling opportunities.

When changes to the proposal are needed, the GO with a RPSS format copy of the proposal can make the quickly changes and re-submit the proposal via RPSS.

With RPSS, the person who knows the most about the scientific requirements, the GO, is the same person who finds the legal syntax to express those requirements. The scientific staff at STScI can only make an educated guess about what the proposer wants, and sometimes those wants can not be easily communicated via letter or phone.

STScI realizes several benefits from having GOs use RPSS. The Institute will have less staff time spent on proposal entry and correcting syntax and validation errors. This allows more time to be spent on evaluating and scheduling the proposals. Less computer resources, I/O, CPU, and Memory, will be needed on the main STScI computers if most of the proposals received have been validated on RPSS, and can be automatically loaded into the PEP data base.

With the entire process of proposal entry and validation being done more quickly, less time will elapse between when the GO writes the proposal and when the GO receives their HST data. Even for Astronomy with its billion year time scales, fast turn around is important.

In the area of proposal processing, the next generation of space based and ground based astronomical instruments have needs quite similar to those of HST. The sophistication of the NASA Great Observatories (e.g., AXAF) and of the European VLT require more than a cursory reading of a user manual. To adequately exploit the time variation in the capabilities of the new facilities, it may be

which writes two error files, `by_line.err` and `by_message.err`, in the observer's directory and sends VaxMail to the observer's account when it is done. Because of the large CPU resources required by this function, the command adds an entry to a batch queue which allows only one process running at a time. This prevents the RPSS computer from being bogged down by simultaneous VALIDATE jobs.

If the observer has a copy of the RPSS software on a local VAX, they can perform the syntax and validation checking at their institution. Otherwise they must transmit the erroneous section of the proposal file to their home institution, edit it, transmit back to the RPSS computer, and rerun the command.

Once all errors are corrected the observer must use the SUBMIT command on RPS. For those observers with their own copy of the RPSS software, they will have to transmit the proposal to the RPSS computer and then use SUBMIT. This command runs both the CHECK and VALIDATE commands and will not accept a proposal with any CHECK or VALIDATE errors. This requirement forces the observer to create a Validation error-free proposal following the Proposal Instructions syntax.

If SUBMIT accepts the proposal, it will concatenate the proposal into one file, and place that file in a secure area. SUBMIT will also notify the proposer by sending a VaxMail message, which includes the remote proposal ID. The observer will need this ID number when sending the signed copied of the coverage to STScI.

When STScI receives the signed coverage with remote ID, they will notify the PEP Data Base Administrator (DBA). The DBA then fetches to remote proposal from the secure area and loads it into the PEP EDB. As part of the load procedure, the remote file is compared to what was entered previously into the EDB. The proposal title and principle investigator's name must match before the file will be loaded.

After the proposal has been loaded, a VaxMail message will be sent to the observer's RPSS account. By logging into their account regularly, the observer will know when the proposal has been loaded.

### **5.3 Security**

Since the proposals are on a computer with public access, security of the data is a key feature of the RPSS design. The proposals are kept secure by using very tight file protections and limiting the commands a proposer may execute. RPSS users are confined to their own directories, and no file can ever be made publicly readable. As a further protection of the system, only a few VMS commands are permitted, and, of those permitted, the power of the commands has been greatly reduced.

For SUBMIT to work, it needs special privileges to write the proposal file to the secure area. This is not a privilege that a RPSS user should have, so SUBMIT is split into two parts. The first part is activated when the user types the SUBMIT command. This part "wakes-up" the second part and tells it which file to process. The user has no access to the second part, and the first part is smart enough to preserve the security of the system.

The process of storing the proposal file in a secure area, converting it to data base commands, and

A special keyword is *include*: <filename>. It is used to break a large proposal file into several smaller files. Since file transfer can be very slow, this keyword allows a small portion of a large proposal to be transferred. This is especially useful when editing on the user's computer and validating on the RPSS computer.

## 5.2 The RPSS Process for Phase II Observers

The successful proposers (Observer) will receive a letter from STScI announcing their selection. This letter includes instructions on how to access the RPSS node, their account name, and password. They can reach the RPSS computer via three different networks: TELENET, SPAN, and ARPANET. TELENET uses the x.25 protocol and files can be transferred using KERMIT. SPAN and ARPANET use the Decnet protocol and files can be transferred using the COPY command or VaxMail. To increase the account security, the password given to them is pre-expired, and the observers must change their password after logging into the account. If they forget, they must contact STScI to re-enable the account.

Waiting in the new account will be two VaxMail messages. The first message gives them some instructions on how to use RPSS. The second message contains the proposal information which has already been entered by STScI, i.e., the Coverage and General Form. They can execute the VaxMail command EXTRACT/NOHEAD <filename> to create the RPSS remote proposal file in their RPSS directory.

The remote proposal file is normally transferred to the observer's home institution computer and edited there. Although the RPSS system supports on-line editing, the limits of current packet switched networks can make on-line editing very slow and suitable only for very small changes. The proposer edits the RPSS remote proposal file to enter all the information for the Target List and Exposure Logsheet sections. Once all the proposal data is entered into the RPSS format file, the file is transferred back to the observer's account on the RPSS computer.

The observer can also execute a command on the RPSS computer which generates a complete (albeit, empty) template file. The template file has all the valid keywords and is a blank proposal, waiting to be filled in. The observer may fill in or replicate the sections needed and delete the sections or lines that do not apply.

Once the proposal has been entered into the RPSS format file in the observer's account back on RPSS computer, it must be checked for valid RPSS file syntax and for valid proposal syntax. Invalid RPSS syntax can be misspelled keywords or improper format for a keyword's value. The RPSS syntax is checked by executing the command

*CHECK <filename>*

which writes an error file, syntax.err, in the observer's RPSS directory.

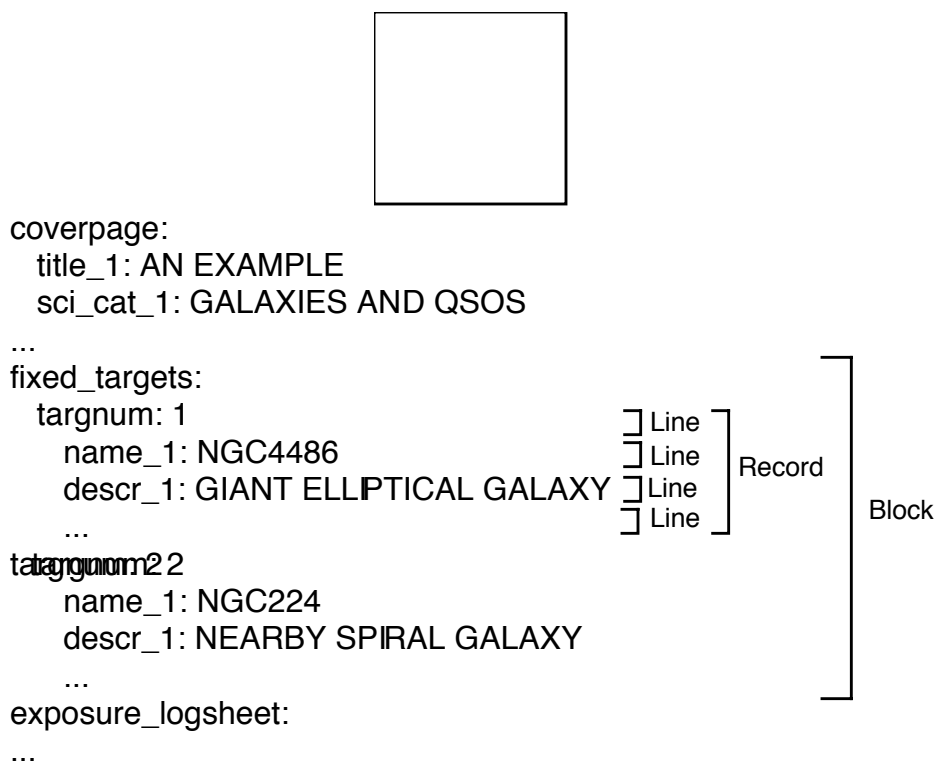
Invalid proposal syntax can include an invalid filter for a certain instrument or an unrecognizable Special Requirement. The proposal syntax is checked by executing the command

VALIDATE <filename>

Proposal Form	RPSS Block Keywords
Coverpage	coverpage abstract
<code>_form_text</code>	general_form_address
Target List	fixed_targets generic_targets solar_system_targets
Exposure Logsheet	exposure_logsheets
Scan Data	scan_data

A record is analogous to single line on the Target List or Exposure Logsheet and is a logical collection of RPSS lines. The start of a record is signaled by the use of a record keyword. For example, the Fixed Target List paper form can contain several different targets, each with a line of data on the paper form. The fixed\_targets block would contain several records, each corresponding to one target on the paper form. There will be as many records in this block as there are targets on the form.

The following diagram of an abbreviated proposal shows the Block, Record, Line structure, using the actual keywords.



be used on a completely non-HST problem. Once the initialization files are created, the users can create their own commands and procedures with this meta-tool.

## 4.5 Transformation

The Transformation Subsystem, described in Rosenthal, Monger, Miller, and Johnston (1986), converts the information in the IDB into the representation required by the ground system data base (PMDB). It is an expert system written in OPS5 and C and currently contains about 550 rules. It provides an interface between the Astronomer-friendly syntax of the Proposal Forms and the complex and voluminous syntax of the ground system.

The subsystem is designed and built in an environment where the users were developing and would continue to develop the procedures for populating the SOGS input data structures from IDB data structures in PEP. Transformation has been in operation for more than two years and is continually being enhanced to deal with additional proposal syntax and with new procedures. Converting the data using a rule based expert system has proven to be a very effective way of meeting the rapidly evolving requirements of the users.

An alternative method of entering proposal data into the PEP Entry Data Base is via the Remote Proposal Submission System (RPSS). The goals for RPSS were to provide an easy to use system with wide user access. The system is consistent with the paper forms and allows the users to detect and fix syntax problems with their proposals.

A standalone computer was used to provide the PEP Entry Subsystem functions to users logged in to RPSS from remote sites. The separate computer provides necessary processing power, network connections, and security provisions. Based on the successful experience with this concept, we are beginning distribution of parts of the RPSS software to remote sites for local usage.

Under the current procedures for proposal processing, the prospective HST user will send the Coverpage, General Form, and the Observation Summary Form, to STScI in the period called Phase I. The Director and TAC will select proposals, based on this information. The successful proposers (General Observers or GO's) will then use RPSS to transmit their Target Lists and Exposure Logsheets to STScI, in the period called Phase II. The following sections describe the operation of RPSS in more detail.

### 5.1 RPSS File Format

The RPSS representation of the proposal forms was intended to resemble the paper forms as closely as possible and to be simple to use. The RPSS remote proposal file is an ASCII flat file, with each line containing either a RPSS keyword and optional value or else a comment, i.e.,

*Keyword : Value*

The RPSS remote proposal file is organized in **blocks**, **records**, and **lines**. **Lines** are the smallest element are grouped into **records**. **Records** are grouped into **blocks**. **Blocks** correspond to different parts of the proposal forms. The valid **blocks** are:

subscription of the available resources, e.g., total spacecraft time, data volume, etc.

The tool is written largely in OPS5 and uses the same set of rules used by Transformation to assemble exposures into the scheduling aggregates, as described in Jackson (1987). The overhead times for Earth occultations, slews, and guide star acquisitions depend on the number and durations of these scheduling aggregates. The earth occultation overhead times are calculated assuming a conservative estimate of the average viewing time. The Resource Usage tool's overhead time estimates could not be significantly improved upon without actually scheduling the proposal's exposures.

### **4.3.3 Feasibility**

The purpose of the Feasibility tools is to identify problems with exposures before the exposures are scheduled or executed. By identifying these problems early, the proposer and STScI staff have more time to devise problem-free exposures which meet the proposer's scientific needs.

Currently there is a tool to check that exposures are not requested at times which conflict with the sun distance or moon distance limits, and a tool to verify that uncalibrated filters or entrance apertures are not used. The next tools to be implemented will:

- Determine if and when guide stars are available for an exposure;

- Check that the proposer has not requested exposures which are syntactically legal but which are inconsistent, absurd, or missing crucial related exposures;

- Verify that the exposure time is consistent with the instrument used, the target brightness, and the signal to noise ratio and verify that the instrument will not be damaged by an overexposure.

The list of possible Feasibility tools is almost endless.

### **4.4 Selection**

The Selection function is provided by a set of tools which are used to assign proposals to referees and to support the decision making process of the TAC (Time Allocation Committee) and the STScI Director on which proposals to select. The most important tool is called TACOS, a natural language interface to a single table data base and described in Hornick, Cohen, and Miller (1987). The TACOS user can create querying or editing commands either in real time or by procedures stored in the user's initialization file. A potential query might be the average referee score of all Solar System proposals. Potential data editing could be modifying referee scores or entering the TAC priority of a single proposal.

TACOS is used by the TAC to track the resources allocated and balance the accepted program between the various subdisciplines and proposers. For example, the TAC must make sure that European Space Agency member nation proposers receive 15% of the HST observing time and that real time spacecraft contacts

The TACOS tool is written in Common LISP and can be used on any single table database. The syntax, grammar, and data base structure are all determine by initialization files and the tool could

to be accepted or rejected by staff having this level of access to the system. With these security features and the backups of the EDB, PEP is able to limit access to proposal information and to recover from erroneous user changes.

This subsystem has been extensively used in entering the 307 Guaranteed Time Observer proposals, the approximately 250 Orbital Verification and Science Verification proposals, and more than 100 General Observer proposals. Since its initial delivery, the PEP Entry Subsystem has been enhanced to deal with additional forms and with new syntax, and has proven to be a successful and operational system.

### **4.3 PEP Evaluation**

The purpose of the PEP Evaluation subsystem is to evaluate proposals and assist STScI staff in reviewing the technical feasibility of proposals. This includes identifying possibly redundant exposures, spacecraft resources consumed by the exposures, and impossible to implement exposures. The Evaluation Subsystem has three general functions: Duplication Checking, Resource Usage, and Feasibility.

#### **4.3.1 Duplication Checking**

Since HST time is such a scarce resource, it is important to check for the possibility of duplications with either previously executed and archived exposures or concurrently proposed exposures. However there are more than 10,000 exposures scheduled each year and the number of possible pairs of “duplicate science” exposures is very large, of order  $N^2$ . The purpose of the Duplication tool is to identify a small number of possible duplications for more detailed evaluation by a human scientist. No proposed exposure is rejected without a scientist evaluating the significance of, or the need for, what appears to be a scientific duplication. There could be situations where duplicate science is necessary, e.g., confirming suspected time variability of a phenomenon.

Scientific duplication is assessed both by positional similarities and by instrument usage similarities. Due to the wide field of view of certain HST detectors and to the uncertainties in the coordinates provided by the users, there are degrees of position matching criteria. Similarly, since equivalent scientific information can be obtained with different instrumental setups, there are several degrees of instrument matching.

The Duplication tool is written in C, OPS5, and the data base query language, IQL and is described in Jackson (1987). The position matching and instrument matching code is written in OPS5, which allows for rapid development and modification of the algorithms to meet new input syntax or new definitions of “duplication”.

#### **4.3.2 Resource Usage**

Individual proposals can vary quite widely in the resource overheads needed to execute the same total exposure time. For example a one second HRS exposure in the middle of a string of other HRS exposure on the same target would take one second to execute. But a lone one second WFPC exposure could take 10 minutes to slew the spacecraft, 12 minutes to acquire guide stars, 1 second to expose the CCD, and 4 minutes to read out the data. The Resource Usage tool estimates how much of the limited spacecraft and ground system resources an individual proposal consumes. This resource information is used by the Time Allocation Committee (TAC) to prevent the over-

not all the operational requirements are known in advance.

All the code was developed in a VAX environment under the VMS operating system, i.e., the same hardware environment as SOGS. The computer languages were chosen to match the needs of the subsystem. The Entry subsystem is largely written in C and uses the flexible data structures available there. A large portion of the Transformation subsystem was written in OPS5, a rule based production language ideal for handling large numbers of special cases. The core of the Selection subsystem was written in Common LISP, a language well suited for natural language processing. The ability to choose the best language for the task has help to quickly create the software tools needed for each subsystem.

## **4.2 PEP Entry Subsystem**

The Entry Subsystem takes the user's information from the proposal forms or RPSS files (described in Section 5) and enters it into the Entry Data Base (EDB). The Entry Subsystem was designed to be robust and to allow entry of any alphanumeric information on the proposal forms. It was also designed to be easy to use and require a minimum of training for the entry clerk.

The ease of use was met by having the clerk enter data into terminal screen templates which closely resemble the proposal forms. The robustness was met by having the entry tools allow any alphanumeric information on the form to be entered and stored in the database. The ability to enter the entire contents of the paper form means that illegible characters or incorrect syntax will not halt the entry process.

Since the EDB information is in the form of free text and thus can contain either valid or invalid syntax, there is a Validation tool which verifies that the user has followed the Proposal Instructions and has used only legal syntax. For example, Validation will verify that the user has requested a filter which exists for the desired instrument configuration and operating mode. The Validation tool takes the free text input, generates error messages describing any illegal syntax, and populates the Internal Data Base (IDB) relations. Whereas the EDB relations contain free text and are organized along the lines of the proposal forms, the IDB relations contain specific numeric or character values organized in a hierarchy which describes exposures and targets, their properties, and their interrelations. If the Proposal Forms and syntax is a computer language, then Validation is a compiler.

The EDB provides an insulating layer between the proposal forms and the IDB representation of the proposal information. Should the proposal forms be changed, none of the subsystems which get their input from the IDB will be affected by the change. The only changes would be to the EDB, to the Entry tools, and to Validation which takes the EDB information and populates the IDB.

The screen based entry tools can also be used to search the EDB for information in a specific proposal. There is also a very powerful interactive query language and report writer which can be used to make complex queries and format the results of the queries.

Another requirement of the PEP system was to provide proposal security and change tracking. The security is provided by allowing only people in certain lists the ability to read or edit proposal information. When a user edits a line on a form, a record is sent to a proposal history relation which tracks who changed what and when. There is also a Signoff facility which allows pending changes

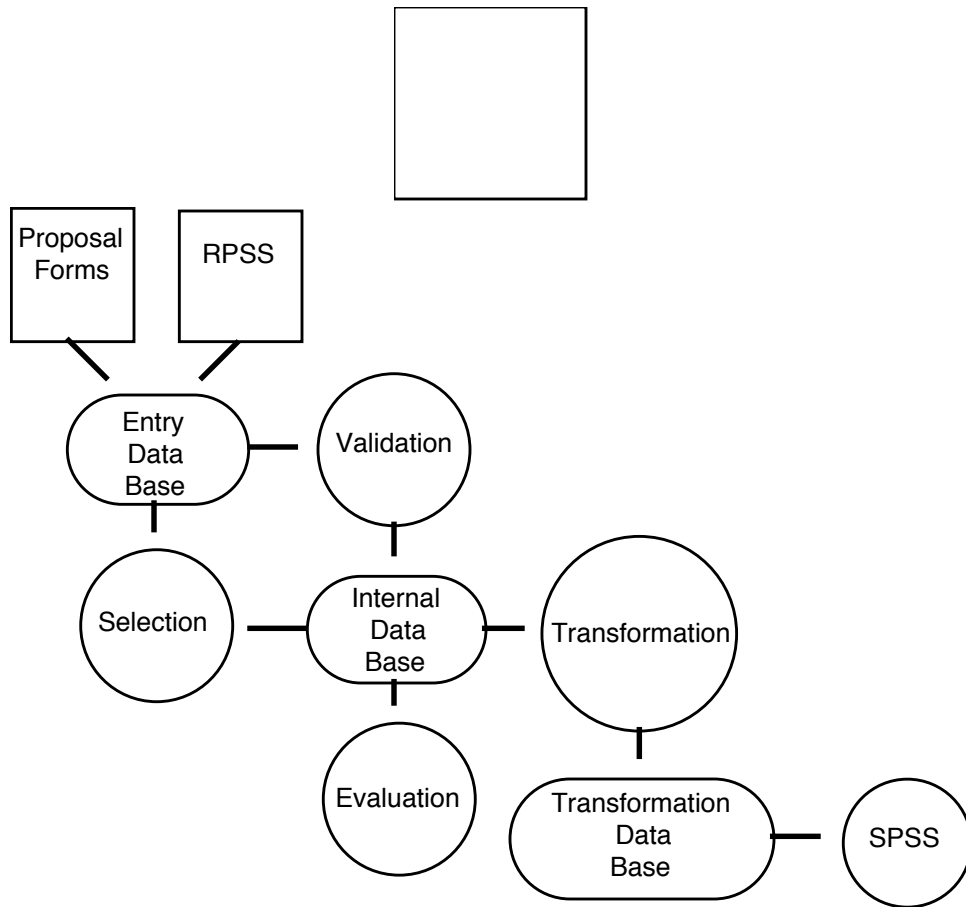


Figure 1 - Overview of PEP System

The Entry subsystem consists of RPSS (described in Section 5) and other software tools and provides entry and editing capabilities for the Entry Data Base (EDB) and the Validation function which populates the Internal Data Base (IDB). The Evaluation subsystem is a set of tools which uses IDB information to check feasibility problems, find duplicate exposures, etc. The Selection subsystem uses EDB and IDB information to aid the proposal selection process. The Transformation subsystem converts the IDB information into the SPSS data representation which is stored in the Transformation Data Base (TDB) for later transmission to SPSS.

#### 4.1 PEP Development Environment

The PEP software was developed in a rapid prototyping environment where the developers had as much understanding of the system requirements as the users had. This combination of a rapid, iterative code enhancement cycle and a high level of domain expertise by the developers provided a working PEP system quickly and economically.

The Entry Data Base (EDB), Internal Data Base (IDB), and Transformation Data Base (TDB) are relational data bases in a Britton-Lee BL700 data base machine. The interface between the BL700 and the VAX's is provided by Signal Technology's Omnibase, a fourth generation language tool. The extensive querying capability provided by Omnibase is very useful in an environment where

*Do For Targets <numbers>*

or

*Repeat <lines> Every <time> For <number> More Times*

to either execute an Exposure Logsheet line for a number of targets or to execute a number of lines at the stated interval.

Not only does this eliminate bulk and repetition, it can also make the proposer's intent more clear to both humans and software. This subroutine or do-loop ability is functionally similar to block structured computer programs.

There are two additional forms which are used in the small number of proposals where additional information is needed. The Proper Motion/Parallax Form is used to specify the apparent motions of fixed targets when these motions are significant (e.g. could affect target acquisition). The Scan Data Form is used when the HST Pointing Control System scanning capability is used, i.e., when the Spatial Scan special requirement is stated.

While the forms and syntax are more complicated than proposal forms for ground based telescopes, they provide a compact and expressive representation of the series of activities and decisions desired by the user. The information required for software to execute an observing program with a ground based telescope would be similar in volume to that required for HST.

#### **4. PEP System Overview and Design**

The PEP system was designed to support the Entry, Evaluation, Selection, and Transformation of HST observing proposals as described in the *ST Proposal Entry Processor System Requirements* (1987). The process by which proposal information flows through the PEP system is illustrated in the following diagram.

The Exposure Logsheet provides a powerful mechanism for expressing the observations to be done at the positions specified on the Target Lists. It provides both a simple to use form for the common types of observations, and yet a powerful means of expressing complicated programs with many interdependencies. This form ties the exposure information to the target list information and contains all the information provided by the user describing the spacecraft activities desired.

For simple observations, the user specifies on this form the *Target Name*, *Configuration*, *Operating Mode*, *Spectral Element*, *Entrance Aperture*, *Flux Reference Number*, *Number of Exposures*, and *Exposure Time*. For more complicated observations, the *Optional Parameters* allow the user to change the instrument settings from the default values. This is especially useful for onboard target acquisition modes, where the nature of the field determines the best set of search parameters. The use of default settings allows the HST user to ignore those settings which are not relevant to his needs and allows STScI to use the best values based on recent experience without having to alter the contents of the Exposure Logsheet whenever an improved value is determined.

The *Special Requirements* section of each Exposure Logsheet line allows the user to specify:

- Relationships between exposures, e.g.,
  - Early Acquisition For <line>
  - Same Orientation For <line> As <line>
  - Calibration For <line>
  - Real Time Analysis For <line>
  - After <line>
  - Sequential <lines> Within <time>

- Additional properties of an exposure, e.g.,
  - Position Target <X,Y>
  - Spatial Scan
  - Critical Observation
  - At <date> +/- <range>
  - Dark Time

- Branches and conditional exposures, e.g.,
  - Branch To <line> If <condition>
  - Conditional If <condition>
  - Select <number> of <lines> Or <lines> ...

Each exposure on the Exposure Logsheet is labeled with a line number. The <line> in these Special Requirements refers to a single line number or a range of line numbers.

These *Special Requirements* must be described using syntax which is listed in the Proposal Instructions. The syntax limitations allows the user's needs to be processed by software automatically and without any human interpretation.

The Exposure Logsheet provides several constructs which allow the user to express a set of exposures in a succinct fashion. The *Sequence Definition or Usage* column can be used to Define sub-routines of exposures and then to Use such subroutines. In the Define lines, some of the columns can have placeholder symbols, which get substituted with specific values from the Use lines. In the *Special Requirements* column, the user can state

### **3.1 Coverage and General Form**

The Cover Page is an “executive summary” of the proposal, and contains the proposal title, scientific category, the principal investigator, the number of targets to be observed, the amount of exposure time requested, a scientific abstract, and the amount of funding requested.

The General Form largely expands on the Coverage information. There are General Form sections listing all the investigators and their address. There are other sections in which the proposer describes in detail the scientific justification of the project, why HST is needed, why special scheduling or calibration requirements are necessary, what the data analysis plans are, etc.

The information on these forms are used primarily in the proposal selection process and for contacting the proposers, but they also provide a written description of the proposer's intended observations. The only limitation on the content of the forms is that PEP cannot enter graphical data or non-alphanumeric characters, e.g., Greek letters.

### **3.2 Target Lists**

The HST proposal forms allow for three categories of targets: Fixed, Solar System and Generic, each with its own Target List. This division is necessary since the specification of target position is different for each class.

Fixed targets are defined by a specific position on the sky. The proposer may specify a particular right ascension and declination (along with uncertainties), or an offset in coordinates from another fixed target. Specification of extended or area targets (e.g. nebulae, galaxies) is also possible.

The positions of Solar System targets are specified in one of a number of ways, including standard names (e.g., Jupiter), orbital elements (e.g., a new comet), or positions relative to other solar system objects (e.g., satellite or planetary surface features). The positions can also be restricted to specific time intervals or periods when certain planetary events occur (e.g., maximum elongation of a satellite). The Moving Object Support System (MOSS) was developed by JPL and converts these position and time specifications into a time series of position vectors which are used by SOGS in scheduling the observations.

Generic targets provide added flexibility to the proposer. These targets are identified by general target characteristics or broad locations in the sky. Examples of generic targets include targets of opportunity (nova, supernova, comet, etc.) or certain types of targets in large regions of the sky (e.g. any field within 10 degrees of the north galactic pole.) Generic targets are useful when it is unduly restrictive or impossible to select a specific target at the time of proposal submission.

Each Target List requests a target name, target description, anticipated HST acquisition problems and target brightness data. Target names and description are important in the construction of useful astronomical archives and in understanding the proposer's intent. The STScI Proposal Instructions give detailed guidelines for naming and describing targets. Target brightness data is requested so that exposure times can be independently verified and adjusted to compensate for orbital conditions (e.g. scattered earthlight)

### **3.3 Exposure Logsheet**

Fortran or Pascal. Similarly, in specifying the spacecraft activities, the proposer could specify the actual spacecraft command loads on a timeline or else could specify very high-level commands in a “friendly” language.

When specifying detailed spacecraft activities, it is all too easy to request physically impossible, unpermitted, or internally inconsistent activities. This problem is much less likely to occur when specifying higher level commands. The complexities of modern spacecraft and of spacecraft operations require detailed knowledge which is costly to acquire. The syntax used in specifying the activities must be validated to assure that only legal syntax was used and that the activities are feasible.

### **3. HST Proposal Forms**

Users of ground based telescopes and previous spacebased telescopes have traditionally been granted specific time periods in which they control the telescope in real time. The proposal forms for these facilities have only contained information needed for proposal selection and for providing necessary staff and hardware resources at the telescope. The forms have not contained a detailed description of exactly what observations are to be done. The situation with HST is quite different. Real time communication with HST is available about 20% of the time, due to HST being in a low earth orbit and sharing the 1 MHz Tracking Data Relay Satellite data link. Thus the proposer must specify on the proposal forms all the information needed to execute the observations. An analogous situation for a ground based telescope would be if the astronomer had to write a computer program which would command the telescope and perform all the desired observations.

Planning and scheduling of HST observations is currently performed with the Science Planning and Scheduling System (SPSS) of the Science Operations Ground System (SOGS) developed by TRW. SPSS requires its input to be in a syntax and form which is set by the design of the SOGS software and internal data structures. A scientist would need to have a great deal of specialized knowledge about the internal operations of SOGS in order to properly describe his series of desired exposures in the SOGS syntax. The burden on the scientist would be much too great, and the probability of error in the specification of the exposures would be much too high.

To meet this limitation of SOGS, the STScI developed the HST proposal forms, described in the *Call For Proposals* and *Proposal Instructions* (1985), with the following goals:

- Be oriented towards the user community - easy to understand, and concise and logical in the amount and sequence of data requested

- To accommodate both simple and sophisticated observations

- To allow the proposer to specify what data should be collected without becoming needlessly encumbered by telescope and instrument specific details

- To allow data entry by entry clerks directly from the submitted forms with a minimum amount of training.

The following sections describe the Proposal Forms, i.e., the Coverpage and General Form, the Targets Lists, and the Exposure Logsheet.

cution. Observers are not in continuous communications contact with the HST. Telescope and instrument operations must be carefully examined to ensure the health and safety of the spacecraft.

A second reason is that important gains in scientific efficiency can be achieved. Johnston (1988a,b) has addressed the need for automated scheduling of ground and space based telescopes. Interleaving of observations from different programs, instead of block scheduling of time can lead to increased scientific return by minimizing instrument changes and calibrations, and accommodating programs which require short observations over long time periods. Many observatories offer “service observing” where observatory staff members execute observations for the proposer. At ground based observatories, it is all too common for a program requiring excellent atmospheric conditions to be executed during a time of mediocre seeing, while a program with less stringent requirements happens to occur during the best seeing conditions.

Computer science, in particular, the field of artificial intelligence (AI) has matured sufficiently that we can begin to tackle these problems. Although there is still much debate over whether computers exhibit any form of intelligence, it has been clearly demonstrated that computers are solving problems which formerly required highly trained humans and that traditional (non-AI) computing paradigms are inadequate to deal with these tasks.

The thesis of this section is that the observing proposal form and proposal processing must be considered in view of the above. A proposal is not simply used by an observatory to select observers. The proposal form must contain sufficient information that the observatory can enhance the observing process and increase the scientific productivity of both the observatory and the observer. This includes proposal selection, evaluating proposals for feasibility and efficiency, implementation and data archiving.

The process of elaborating the scientific program can be thought of as progressing from *Asking Questions* to *Identifying Data Needed* and finally to *Specifying Instrument Activities*. Before describing the “Astronomer oriented” vocabulary by which the HST user characterizes their programs, it is useful to consider various possible vocabularies. A user might describe their programs at three levels of abstraction: Answers, Data, and Spacecraft Activities

A scientist could simply ask a question, e.g., “What is the relationship between mass of a galaxy and its central velocity dispersion?” Software would determine the data required and either locate the data in archives or else determine the telescope activities needed to obtain the data. However, such software does not yet exist and is beyond the state of the art of AI today.

The next level of abstraction would be for the user to specify the data they wanted. The characteristics of astronomical data include: spatial range, spatial resolution, spectral range, spectral resolution, time range, time resolution, and signal to noise ratio for the flux. The software would search the archives and determine the best choice of spacecraft activities which would generate the desired data. The software faces a much more limited range of possible inputs than when answering a more general question, but it would still need detailed knowledge about the spacecraft capabilities and how to obtain the required data. Such a software tool could be built with today's technology, but it would be a significant undertaking.

Instead of having the proposer specify the questions or the data, the proposer would specify the spacecraft activities needed to obtain the desired data. Here, an analogy with computer languages becomes relevant. A programmer can work in assembly language or in a high-level language like

## 1. Introduction

The PEP System at the STScI is an interface between the HST user and the planning and scheduling software. The purpose of this system is to accept a high level description of an astronomical observing program and to produce from this the parameters necessary for the planning and scheduling software. Additionally, PEP provides tools for technical evaluation and selection of observing proposals. Traditionally, the processes of solicitation, selection, evaluation and implementation have been largely manual. The tools provided by the Pep system provide a novel approach and may be useful as a model for proposed ground and space-based observatories.

This paper describes the Pep system and how it assists in proposal processing. Section 2 describes from a general point of view the process of soliciting and implementing an astronomical observing program and describes some of the high level requirements of this process. Section 3 describes the STScI proposal forms which allow an astronomer to describe an observing program at a high level, without being needlessly burdened by implementation details. Section 4 examines the architecture of the Pep system and provides an overview of the major subsystems: Entry, Evaluation, Transformation and Time Allocation Committee support. Section 5 discusses the PEP Remote Proposal Submission System in detail. The last section discusses PEP in the context of Telescience and applications to other observatories.

## 2. The Observing Process

Consider first the observing process, from the formulation of a scientific problem through the collection of data: An astronomer poses a question, decides what observations would answer the question, identifies a telescope and instrumentation which are capable of obtaining these observations, plans the observations, and finally executes the observations and collects the data. For example, an astronomer may ask "what is the distribution of stellar ages?" Knowing that lithium abundance is correlated with age would lead to observing red giant stars to obtain lithium spectral line profiles. An observing proposal for time on a telescope with a suitable aperture and spectrograph would be submitted. Prior to observing, a sample of red giants stars would be picked, and parameters such as wavelength ranges and exposure times would be determined. Lastly, observations are made, perhaps adjusting the program to compensate for changes in weather, instrument performance, etc.

This is, of course, a very brief and idealized sketch of what is often a long and complicated process! The essential point is that during this process, the expression of the observing program must undergo a number of transformations, from general descriptions to specific implementation details.

For many observatories, there has been no need to explicitly enumerate these observing steps as one person, the observer, was responsible for most aspects of the program. The transformation from general program to specific instrument and telescope operations was implicitly performed by the observer, often "on-the-fly". This is particularly true for classical, ground-based telescopes, and to a lesser extent for space-based telescopes.

There are two reasons why it may be necessary or desirable to consider the observing process more explicitly: First, the operation of some telescopes is so complex that an observer has neither the time nor the motivation to acquire the necessary expertise to implement the program singlehandedly. HST serves as a good example of this. The input to the planning and scheduling software requires a detailed understanding of the HST, orbital conditions, and the software data structures. The ground system requires that most observations are planned in detail weeks in advance of exe-

# **The Proposal Entry Processor: Telescience Applications for Hubble Space Telescope Science Operations**

Robert Jackson<sup>1</sup>  
Astronomy Programs, Computer Sciences Corporation

Mark Johnston  
Space Telescope Science Institute<sup>2</sup>

Glenn Miller<sup>1</sup>, Kelly Lindenmayer<sup>1</sup>  
Astronomy Programs, Computer Sciences Corporation

Patricia Monger, Shon Vick, Robin Lerner, Joel Richon  
Space Telescope Science Institute<sup>2</sup>  
3700 San Martin Dr.  
Baltimore, MD 21218

The Proposal Entry Processor (PEP) System supports the submission, entry, technical evaluation review, selection and implementation of Hubble Space Telescope observing proposals. This paper describes the PEP system, concentrating on features which illustrate principles of telescience as applied to the HST. These principles are applicable to other observatories, both space and ground based.

The PEP proposal forms allow a scientist to specify scientific objectives without becoming needlessly involved in implementation details. The Remote Proposal Submission System (RPSS) allows proposers to submit proposals electronically via Telenet, SPAN and other networks. RPSS performs syntax and semantic checks on proposals. PEP uses a fourth generation database system to store proposal information and to allow general queries and reports. The Transformation subsystem uses an expert system written in OPS5 to cast a scientific description of an observing program into parameters used by the planning and scheduling system. The TACOS system is a natural language database which supports the proposal selection process. Technical evaluations for resource usage and duplicate science are performed using rulebased systems.

- 
1. Staff member of the Space Telescope Science Institute
  2. Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration