Internal Technical Memorandum ITM-1998-01 (**DRAFT)**

Astronomer's Proposal Tools (APT) Project Management Plan

T. Krueger
January 4, 2000

ABSTRACT

This document describes the management plan for the APT project. Many of the ideas were taken from other documents ("Tools of the Future Working Group", "Role of Project Stakeholders...") verbatim. This document tries to put all the ideas into one document by which the APT project can be developed and maintained.

1. Introduction

The project to construct the APT is an ambitious, multi-year project that promises to greatly improve and streamline the preparation and validation of observing programs.

2. Background

Originally, Hubble proposals were submitted using the Remote Proposal Submission System (RPSS). This submission system was designed in the mid-1980s and represented the state of technology and experience in handling "service mode" observations. It provided the bare minimum of user support, for example, checking for syntax or spelling errors and some illegal configurations. When received at STScI, these proposals were processed to determine feasibility and schedulability. It was at this stage that most problems were discovered, and manual intervention by operations staff was necessary. RPSS was used until 1994. RPSS was neither conducive to efficient user support strategies nor was it observer/observatory staff user friendly. After three years of proposal preparation experience, an effort was initiated to improve upon the RPSS process, which led to the release of RPS2 for cycle 5 (1994) observing. RPS2 was designed to further two major goals:

- Improve the quality of proposals at submission time and thereby avoid the need for observers and operations staff to iterate. This was achieved by making some of the telescope operations constraints available to observers when they prepared their programs.
- Make routine the process of updating proposals after submission for scientific or operational reasons. This was achieved by dividing proposals into "visits" which can be independently planned and scheduled.

RPS2 was implemented using client/server technology which processed a proposal in batch, and then graphically displayed the results of the processing to the observer. In developing RPS2, we

opted in favor of the modest RPS2 architecture instead of a full interactive environment because of the shortcomings of Trans (there was no way to query Trans to receive a quick answer to a question) and the lack of suitable integration software like Corba and Java RMI which were still years away. Further, the simpler RPS2 architecture, which was a vast improvement over RPSS, could be provided to the user community quite rapidly. A fully interactive system was thus not cost effective in 1994. The vast majority of proposals are submitted today without feasibility or schedulability errors because observers are given sufficient information to remove these errors during proposal development. However, there are major areas with potential for improvement.

In the last three years, technological advances such as widespread use of the Internet, multi-platform visual development tools, and overall increases in the power of desktop hardware are allowing for significant improvements in user support tools that can be provided by an observatory. APT is envisioned as an integrated environment that will:

- Leverage off state of the art technologies;
- Provide modern user support tools;
- Achieve the goals stated in the Project Goals section below.

For the past two years their has been a collaborative effort (Scientist's Expert Assistant - SEA) between Goddard Space Flight Center and STScI to prototype visual and expert system technologies and how they help an observatory achieve the following goals in proposal preparation;

- Reduce the amount of observatory staff support needed for routine observations
- Self sufficiency of the observer (i.e., make it easier for observers to create accurate, feasible observations.

The SEA project is slated to end in March 2000 which overlaps with the APT project requirements definition phase. The evaluation of user SEA will begin in the Jan-Feb 2000 time frame. The APT project will use the SEA user evaluation as input into its requirements definition and functionalities. The APT project will also evaluate the SEA software for applicability and re-use in the APT project. The APT project does not promise to use any SEA software, until after a software evaluation is performed and then the project will decide what if any is re-usable. For information on the SEA project see http:// aaaprod.gsfc.nasa.gov/SEA/.

3. Project Goals & Vision

The APT project envisions an integrated tools environment that will achieve the following user support goals.

- Provide observers with tools that are more intuitive, visual, and responsive.
- Provide observers with tools that allow them to primarily spend time on scientific decisions and not on the mechanics of using the system.
- Have observers produce observations that do not require any manual intervention by our staff at least 80% of the time
- Provide documentation/help that is friendly, up to date, and easily accessible to users of varying levels of expertise.
- Provide an extensible framework which is responsive to change in both technology and observatory operations.

4. Technical Description

APT is envisioned to consist of two major components; the APT tools set that provides users with tools that help them prepare their science programs, and the integrated environment that unifies all the tools and makes them interoperable. It is envisioned that the APT tools will be developed to work as a stand-alone tool and in the integrated APT environment. The project will have to go through a process to determine what tools need to be in the system and how the system will work. Given our experience with RPS2 and the SEA work, some envisioned tools and functionalities are;

- Visual Target Tuner (VTT) allowing observers to visualize their targets and field of view.
- Exposure Time Calculators (ETCs) allowing observers to calculate their exposure times.
- Phase I Submission Form allowing observers to fill out their Phase I forms.
- Exposure Planner allowing an observer to layout their exposures in their orbits.
- Bright Object Checking allowing an observer to see any health and safety issues caused by their observation.
- Visit Planner allowing observers to visualize timing relationships between visits (e.g. BEFORE, AFTER, GROUP WITHIN) and to better understand unschedulable situations.
- Canned Observing Strategies which would help to automate the process of applying customizable observing strategies to observing programs
- Import of Data from Archives allowing observers to import details of an observing program from any of a number of mission archives to be used to form a new proposal.
- Integration with Online Documentation providing observers with the latest documentation
- Access to Execution Data providing the user with program feedback and help in scheduling decisions

5. The Role of Project Stakeholders

This section describes the role of the APT project users.

5.1 Categories of APT Users

Since the types of observing programs are so varied, it is important to incorporate a wide range of viewpoints to maximize the likelihood that all of the important APT functional requirements are identified at an early stage, which in turn will improve the changes of overall project success. It is useful to categorize the likely APT users because the success of the project will be determined by the extent to which is satisfies both internal and external needs; The needs of external users include: a more efficient means to specify an observing program, and efficient means to evaluate alternative strategies, and to have confidence that the program is correct. The needs of internal users include: a means to validate, format, and schedule observations optimally with minimal human intervention, and to design, constraint, test, deploy, and maintain a viable collection of tools over the course of several years.

Here are the anticipated APT project stakeholders;

- HST GOs and GTOs from the external community who have little or no knowledge of the HST scheduling system
- Instrument Scientists who are involved in preparing calibration programs or who serve as contact scientists for GO programs
- Program Coordinators who implement HST programs.

- Data Analysts that are associated with active instruments, who prepare Phase 2 calibration proposals
- Software Developers who must build and maintain the system and tis various components.
- STScI Management who monitor the project

One or more “champions” from each category should be identified to form a team(s) to provide input throughout the course of the APT project. Because the problem domain is broad, it is important that each category is represented, but at the same time, the number of users must be kept manageable if there is any chance of providing timely and focussed input.

5.2 Key Areas of User Input

The team(s) should provide input during the life of the APT project, in the areas of the process model, product conception, requirements definition, and testing and validation. Each of these areas are described below. What is perhaps most important is to specify what is expected from the user team(s). It may be best if there is one user team that is responsible for all the areas of input. They may choose to break themselves up into sub-teams, etc.

Product Conception Input

The APT project concept is collection of utilities that collectively enables the complete specification of an observing program. There are many issues a proposer must resolve in the process of preparing a viable observing program, not the least of which is exploring the multitude of possibilities and restrictions that constrain the science goals at hand. The problem domain is sufficiently broad and complex that it is very important to construct software tools that hide complexity where possible, provide the flexibility to consider alternative approaches where available, provide immediate and concise feedback where appropriate, and that integrate seamlessly with existing or planned applications to facilitate the specification of a program. It is highly likely that there are a number of possible ways to partition the various sub-tasks that must be completed to specify the observing program.

Product Conception input will help to ensure that the right products are developed for the APT, and that they form a functionally complete and well integrated set. Specifically, this would include the identification of one (or a few) process models by which observing programs are developed, and to articulate which are the right software functionalities (i.e., tools) to build to facilitate the process model(s), while achieving the overall APT project mission. The definition of the process model(s) will naturally lead to a discussion of new ways of formulating the current Pahse1/Phase 2 processes (which should provide the seeds of major innovations), while at the same time providing a vital context to evaluate whether the APT is developing the right tools.

Once the process model is understood, a set of applications (i.e., tools) can be identified. For each application (i.e., tool), the following should be articulated:

- The business case for each application to be developed (i.e., what are the direct and indirect benefits to the users and to STScI).
- A description of how each application conceptually fits into the system as a whole
- A description of the kinds of information that form the input to and output from each application.
- A set of high-level use-cases for each application
- Light-weight interface prototypes, if appropriate, to illustrate or amplify the points above

A valuable component of these reports is a discussion of alternatives that were considered but rejected, and why. It is entirely reasonable the product concept is based on the evaluation of components in one or more existing systems, such as RPS2, SEA, Starview2, etc.

Requirements Input

The specification of software requirements is perhaps the most difficult task to be undertaken by project teams, and yet it is one of the critical factors in project success. While the detailed specifications will be worked out collaboratively with users and APT software developers, this area is usually the most difficult because it requires a good deal of expertise in both the problem domain and in the software domain.

Requirements are needed to ensure that each product is accurately specified for the APT. That is, the requirements must be specified at a level where a software engineer knows what to build; when the product is built, it must be able to demonstrate whether the product meets the requirements. Since this process is an on-going process and users oftentimes don't know exactly what they want, different approaches will be used based on what is being built. For example, building an user interface may involve building a prototype and iterating with very little written documentation until the end, whereas, designing a straight forward algorithm may define its requirement fully up front. No matter what approach, the requirements should be documented, baselined and references to mock-up prototypes should be supplied. Prototypes are often very effective at capturing the essence of the intended functionality, while reducing the risk of mis-communication between users and developers. The requirements process is constantly changing, and continual review and input will be needed by the APT project. The concept of baselining requirements for APT does not mean that things will not change. Baselining means that the user and developers have agreed upon a set of requirements for proceeding in the next step or phase.

Testing/Validation Input

As early as possible, the stakeholders should be involved in developing test plans, including use-cases, and to evaluate the pre-release product. The test plan must demonstrate that the released product meets the implemented requirements, that the use-cases can be executed and that they meet ease-of-use and other criteria, and that any numerical calculations are accurate to within the prescribed tolerances. User documentation will be evaluated for accuracy and completeness.

6. Software Development Plan

The software development process usually consists the following stages, concept development, requirements definition, design, implementation, testing, and maintenance support. Whether you are building a throw away prototype to test concepts or delivery an operational system, the software goes through these different stages. For example a prototype, needs a concept, requirements, design, and implementation for the evaluation. The amount of effort put into each stage may be minimal, maybe without any review and is done totally be a single user/developer team. Delivering a system for operational use, will go through each of these stages, but much more formally. Some tools in APT may need to spend a fair amount of time in prototyping and trying to innovate, and some will be developed from the beginning in a more formal regimented manner. The APT project will determine on a product by product basis what needs to be done for the software development process. It is recognized that all operationally released products will go through a regimented development process to ensure the most bug-free and accurate product. The quality assurance plan

in this document describes a regimented process for managing operational software. Again how strict this plan is adhered to for prototyping or trying new non-operational things will be determined by the project based on the products goals.

Since APT is expected to be a collection of tools with infrastructure allowing the tools to communicate, small teams (one person to multi-person) will be formed to tackle the various tools and infrastructure issues. For example, their may be a Exposure Time Calculator Team, a Visual Target Tuner Team, a Infrastructure Team, etc. People may be assigned to more than one team. It is also envisioned that the teams will be responsible for the product from conception to and including the maintenance of the product. The original “Tools of the Future” working group report called for an innovator and fielder separation. This model had a lot of appeal from its conceptual point of view and its goals, but it was hard to envision how to actually implement this, particularly with the smaller development team than envisioned by the working group.

One of the key areas that needs to be addressed by the RPS2/APT software/user group is how do we phase out RPS2 and phase in APT. We make a recommendation to keep the two tools as decoupled as much as possible with the ability to import/export the input/output products between RPS2 and APT. For example, a user might use the APT Visual Target Tuner tool to pick their targets. They write an RPS2 file from the VTT, which can be imported into RPS2. They make changes in RPS2, and save their RPS2 file which can be imported back into the VTT for looking at the target. This approach poses the least amount of overhead and effort on the development process. One area that must not be forgotten, is by reducing the amount of software effort are we making it harder for the end user. The key is to minimize unnecessary development effort while not overburdening the user.

The project plans to set up a delivery schedule, where the latest changes are made available for evaluation or operationally. One idea currently discussed is monthly releases where whatever is tested and ready for release is made available. The project wants to try and avoid major releases at the Phase 1 / Phase 2 deadlines and the scrambling to meet those deadlines. For operational software, this means that testing and new development will always be going on concurrently.

7. Quality Assurance Plan

How much the quality assurance plan is adhered to will depend on whether we are releasing an experimental protoype or operational release. The two have different guidelines. If the release is a prototype, we may not go through all the quality assurance steps, whereas it is an operational release we certainly will. Operational software should be robust, give correct answers, clear diagnostics, and not crash or get into unusable states. Prototype software, depending on the goals of the prototype, may not adhere to the same criteria as operational releases. The project will determine what quality assurance steps are necessary when developing a prototype.

7.1 Document Guidelines and Configuration

Documentation shall be stored outside the operational firewall so that it is accessible to everyone on the team. We don’t envision a need for a configuration control mechanism such as CVS. All project documentation will be stored in one location accessible to all project members.

The project recommends that all documentation be prepared in FrameMaker whenever possible.

The project recommends that all presentations be done in PowerPoint whenever possible.

The documentation should be made available in HTML and/or PDF for the project web page.

7.2 Source Code Configuration

The primary programming language for the APT will be JAVA. The project will use the Concurent Versions System (CVS) for source code configuration. The repository for the source code will be stored outside the operational firewall.

7.3 Requirements Reviews

All requirements will go through a requirements review before proceeding to design/implementation. All requirement documentation will be made available to the all reviewers, before the review. The requirements will be presented at the requirements review, which is open to all interested parties.

7.4 Design Principles and Standards

The system should favor interactive approaches over batch processes. Performance and responsiveness will be considered in all design decisions.

The system should reuse existing components of other systems where appropriate (e.g. SEA components, Starview Components, etc.)

The system design will be object-oriented.

The system design should implement the Model, View, Controller (MVC) architecture.

The design will use a Unified Modelling Language (UML) tool. We have two tools available at STScI, Rational Rose and Together. We will pick a UML after we talk to individuals who have used both.

The system will be extensible (i.e., easy to add new features/tools).

The system should be platform independent wherever possible. Java is the leading candidate because it provides a number of advantages such as;

- interactive graphics and image processing support;
- support for accessing databases and catalog servers;
- user interface support;
- portability; and
- object oriented development.

The system will use a common data format. The markup language XML is the leading candidate since it is a world wide standard for data representation. The advantages to XML are:

- Support of automatic checking of documents for structure validity;
- availability of a rich array of tools to process and display XML documents; and
- availability of existing Java libraries that already support XML

7.5 Design Reviews

All designs will go through a design review before proceeding to implementation. Two individuals will be selected to be the primary reviewers and responsible for reviewing the design in detail before the design review. All design/requirements documentation will be made available to the all

interested parties, including the primary reviewers, before the design review. The design will be presented at the design review, which is open to all interested parties.

7.6 Coding Standards and Development Environment

The project will develop a set of coding standards. The RPS2 project is developing a set of Java coding conventions which are a blend of the GSFC and Sun Microsystems coding standards.

The project developers will be encouraged to use an Integrated Development Environment (IDE). There are three available at STScI Visual Cafe, Jbuilder, and Code Guide. We will evaluate them and pick one for use on the project. This evaluation is not expected to be much effort, since we have STScI personnel who are familiar with all of them.

7.7 Source Code Reviews

All source code will go through a code review before proceeding to developer testing. The code should compile and load without errors before it is reviewed. Two individuals will be selected to be the primary code reviewers and responsible for reviewing the code in detail before the code review. The primary reviewers should be familiar with the programming language being reviewed. After the code is compilable and loadable, a copy of the code will be made available to all reviewers. At the code review, the two reviewers will go through the code first with the developer. All other reviewers will provide their comments after the primary reviewers.

7.8 Testing & Release

The software will go through the following phases of testing. The amount of time for testing will depend on what is being tested. It is expected that testing will be occurring concurrently with new development. The actual implementation of this testing approach will need to be developed further.

Local Developer Testing

With CVS, each developer has their own individual copy of the software. Each developer will responsible for testing their software in the local area. They should feel that they made every effort to ensure that their are no bugs in their code before committing the changes to the CVS repository

Developer Testing

Once the software is committed to the repository it is available to all other developer’s and testers. Other developer’s will be updating to their local areas, thus getting the committed changes of all the other developers. When they run update system, they will be testing all the most recently committed changes from all the developers.

Independent Testing

At some point the testing group will pull a copy of the software from the repository for formal unit and integration testing.

Operational Release

Once independent testing has concluded, the software will be made available for beta or operational release.

7.9 Problem Reporting

We will use the STScI OPR system for reporting and tracking operational problems internally. We have set up APT problem reporting in the STScI help desk software. It is not clear what we will use the help desk software for just yet, but it is available.

8. Resource Management Plan

8.1 Firewall Issues

Currently part of the development team is behind the firewall and part is outside the firewall. We need to investigate whether the entire project team should be moved outside the firewall, remain in its hybrid state, or be moved behind the firewall. This will need to be evaluated.

8.2 Personnel

The following people will be working on the APT project. One of the key decisions that affects APT software/tester staffing is what level of effort to support RPS2. Prior to APT, RPS2's developer/tester team ranged from 3-4 FTE's. Continuing this support level will slow APT development. The APT/RPS2 software team will need to work with the RPS2 users to determine what level of support is truly needed by RPS2 such that we can free up software developers to work on APT.

Development/Testing Team

The development/testing team will be responsible for the development and testing of the APT project and the support RPS2. The development teams exact assignments have not been determined yet. This is just a description of the RPS2/APT development team resources. The following people will be supporting the APT project and the RPS2 project (including integrated schedule work & NGSS).

Person	Effort	When Available	Role
Tony Krueger	75%	Jan 1, 2000	Project Manager Developer
Chris Burkhardt	100%	Jan 1, 2000	Developer
Frank Tanner	100%	Jan 1, 2000	Developer
Jesse Doggett	100%	Jan 1, 2000	Developer
Rob Douglas	100%	Jan 30, 2000	Developer
Tom Donaldson	20% 100%	Jan 1, 2000 Oct 1, 2000	Developer
Hemant Shukla	50%	Jan 1, 2000	ETC Developer
Dick Shaw	50%	Jan 1, 2000	ETC Developer
Fred Romelfinger	60%	?	Developer
Alan Farris	20%	?	Developer
Donald McClean	50%	Mar 2000	Developer

Person	Effort	When Available	Role
Karla Peterson	100%	Jan 1, 2000	Tester, Require-ments

Requirements Team

The requirements team will be responsible for determining the requirements for the APT project. this team is currently being formed. See section 5 for more details.

8.3 Hardware

Each developer/tester shall have their own computer capable of running the software development environment. The software shall be developed on multiple platforms to help with the elimination of platform bugs and improve testing prior to release. The following hardware/OS systems will be needed to support the development team.

- At least one developer developing on a Sun Solaris computer
- At least one developer developing on a PC (DELL) windows or NT computer?
- At least one developer developing on a PC (DELL) Linex computer.

All other developers can be developing on whatever system they have available to them so long as it supports the development environment. We expect that most of this hardware will be available already at STScI. We know the Sun Solaris systems are available and that DELL's are at the institute.

8.4 COTS

At this time the majority of development tools are available in house. We may incur some small costs for COTS tools such as Linex or Integrated Development Environments (IDEs), but none of these are expected to be a major cost.

Here is a list of COTS tools expected to be used;

- Framemaker
- PowerPoint
- Rational Rose or Together (UML tool)
- Visual Cafe, JBUILDER, or Code Guide (IDE tool)
- Linex or Windows OS

9. Management Plan

9.1 Risk Management

Software prototyping should be used whenever possible to reduce risk. The project will have to identify areas of risk and when its applicable for prototyping.

Since part of APT's project is to innovate, their will be failure involved. Not every good idea is going to succeed. APT management will have to identify areas where failure could occur, identify what the cost of failure is, and make these clear to STScI management and the stakeholders before proceeding.

9.2 Security

There are no security issues related to this project.

9.3 Training

No formal developer training is planned. We have internal expertise with the existing COTS tools and software that we can go to for help. There will be startup costs for individuals who are not familiar with some of the tools and programming languages, but this will be budgeted for in the project schedule.

The STScI support staff (PCs, DAs, CSs) will be trained APT as new functionalities are released.

9.4 Project Reporting and Dissemination of Information

The project web page is www.stsci.edu/apt.

The project email distribution list is apt-list@stsci.edu

The reporting mechanism to ESS management has not been setup yet.

The development team meets Tuesdays from 3-5 weekly. Subscribe to apt-list to find out the meeting place an agenda.

10. Project Schedule

This section describes the project schedule. Currently we are evaluating the SEA, Starview, and Gator systems for applicability to APT. A first cut at the project schedule will be developed by mid January.