APT ETC – Web Based Approach

APT ETC Development Team
February 28, 2001 – Version 1

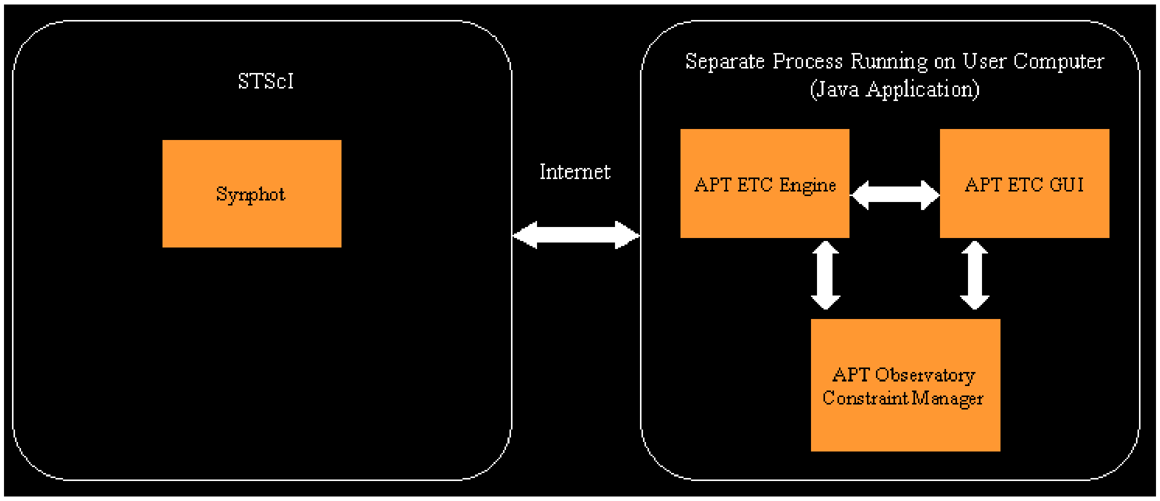
1. Introduction

This document describes the APT ETC architecture and four different approaches for extending the APT ETC infrastructure to support a web based ETC. Each technical approach is discussed and FTE estimates are provided. This document does not discuss the current issue of what terminology the ETC should provide the user (i.e., Phase 1, Phase 2, or some hybrid). It only addresses the technical issues of supporting a web based ETC. The following members were involved in putting this report together.

- Rob Douglas
- Perry Greenfield
- Tony Krueger
- Donald Mclean
- Chris O'dea
- Anand Sivaramakrishnan
- Hemant Shukla

2. Current APT ETC Architecture

The current APT ETC can be thought of as an engine and GUI, which are both written in JAVA and distributed as part of APT or as a standalone tool. The APT ETC calls Synphot for its signal to noise calculations. The APT Observatory Constraint Manager (OCM) tool provides instrument configuration checking for the APT toolset. The ETC uses the OCM to check legal instrument configurations and provide users with legal instrument choices. The APT ETC engine can support both batch processing of calculations and work interactively on a request-by-request basis. Figure 1 below illustrates this architecture.


STScI
Synphot
Internet
Separate Process Running on User Computer
(Java Application)
APT ETC Engine
APT ETC GUI
APT Observatory Constraint Manager

APT ETC Application Architecture

Figure 1.

3. Supporting a Web-Based ETC

This section describes four technical approaches for extending the APT ETC architecture that would support a web based ETC. With all of these approaches, changes can be made to the web based ETC at STScI and made available without the need to redistribute the APT software, since all the code runs at STScI or is served from STScI. Note however, that if anyone has downloaded APT or wants to use the non web version, they will need to download the tool. There is the possibility that the web based ETC will be a newer version than the downloaded version.

Web-Based Applets

In this approach the APT ETC runs as an applet in the web browser at the user site. An applet is a Java program that is executed from within a web browser whereas a Java application is a distinct process running under the computer's operating system. Applet's do not need to be downloaded by the user and installed, they are downloaded within the web browser. The browser must be set up to support Java. Application's on the other hand, need to be downloaded and installed by the user and Java must be installed on the user machine.This applet approach would essentially replace the CGI based GUI with the APT Java GUI interface accessible over the web running the existing APT ETC.

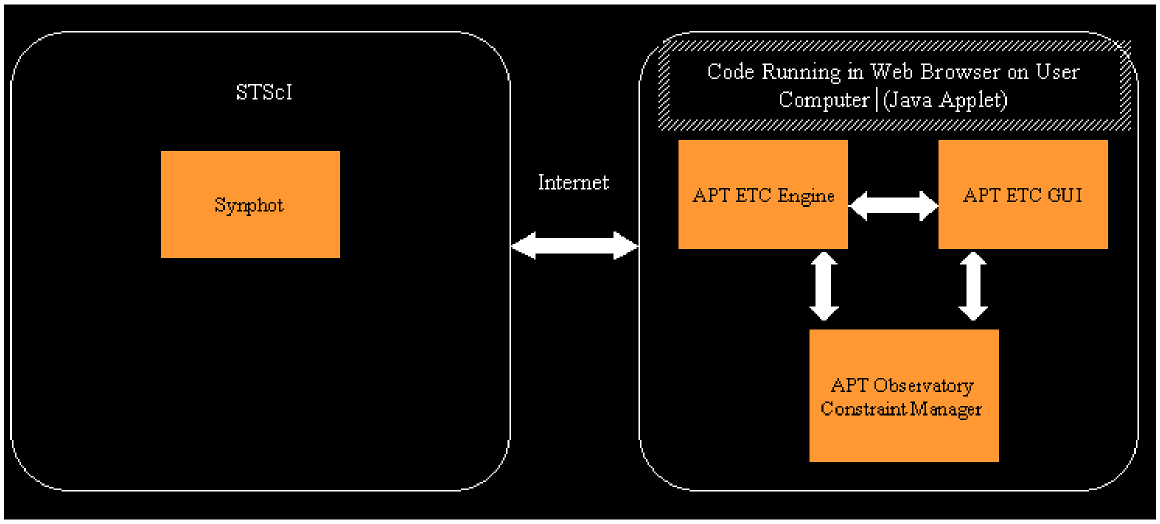
Pros

- Code reuse is very large. One base set of code, both the engine and GUI, are used to support both APT and the Web users.
- The GUIs will be nearly identical. Note, not all Java GUI features are supported in applets so some look and feel changes may need to be made.
- Immediate feedback in GUI for legal/illegal instrument configurations.

Cons

- High download time since the java code must be downloaded over the web into the browser to run as an applet.
- Not all web browsers support running Java applets.
- Can't read/write files to/from user machine. If the ETC needed to read a user-specified spectrum from file, this would not be possible.
-

Figure 2 below shows the applet architecture.

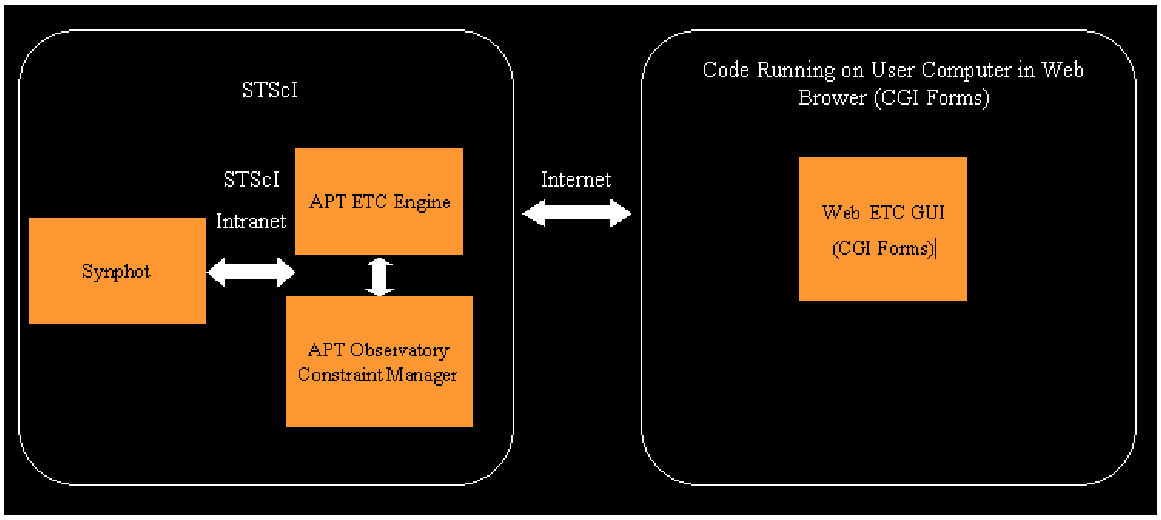

STScI
Synphot
Internet
Code Running in Web Browser on User Computer (Java Applet)
APT ETC Engine
APT ETC GUI
APT Observatory Constraint Manager

APT ETC Applet Architecture

Figure 2.

Web-Based CGI Form GUI

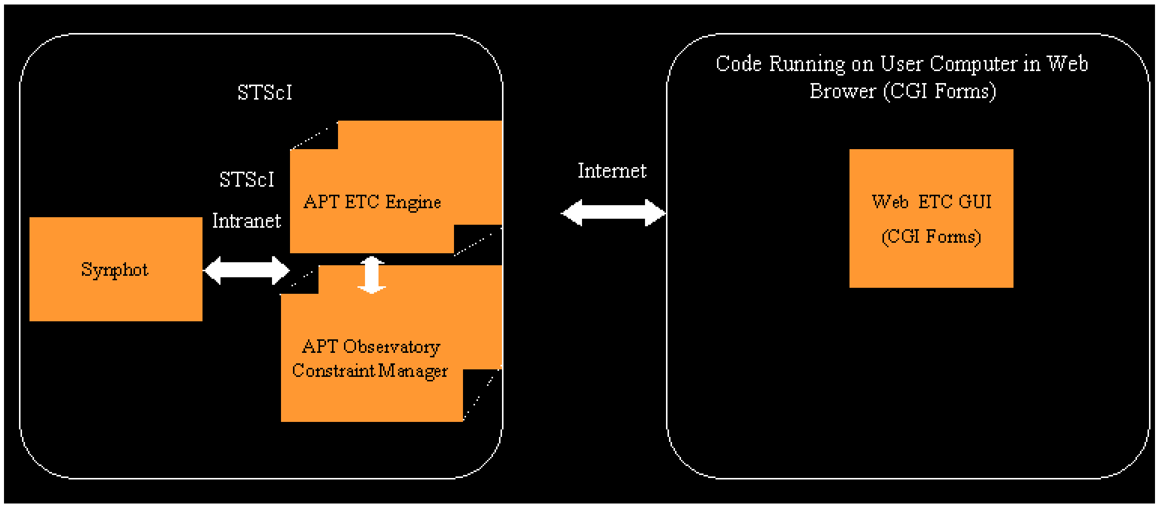
In this approach, their will be two separate ETC GUIs, one for the APT application and one for the web browser. The APT GUI would be the current APT GUI and the web browser GUI will be a CGI web GUI, like we have today. The APT infrastucture will support both the APT GUI and the CGI web GUI. We looked at two different approaches for how the CGI web GUI could work. One is called the static page approach and the other is the dynamic page approach. The static page approach is exactly the approach used by the web ETCs today. A user specifies there input parameters and then asks the ETC to calculate. If there are any instrument configuration errors, the user is notified after they ask the ETC to calculate. Dynamic pages would provide input and feedback on legal instrument configurations as the user is inputting their parameters before they ask for an ETC calculation. Figure 3 illustrates the architecture to support both static and dynamic web pages.


STScI
STScI
Intranet
APT ETC Engine
Synphot
APT Observatory Constraint Manager
Internet
Code Running on User Computer in Web Brower (CGI Forms)
Web ETC GUI
(CGI Forms)

APT ETC Web Based Architecture (servlets)

Figure 3.

The next question was how to provide server support from the APT infrastructure (ie., APT ETC engine and OCM) for the web based GUI. We looked at implementing the APT ETC engine and OCM as a Java servlet. In this approach, there will be one ETC server that handles all the ETC web GUI requests. We looked at a multiple servers approach, where each ETC web GUI request would spawn another APT infrastructure process. Figure 3 illustrates the sevelet approach and Figure 4 illustrates the multiple server approach.


STScI
STScI
Intranet
APT ETC Engine
Synphot
APT Observatory Constraint Manager
Internet
Code Running on User Computer in Web Brower (CGI Forms)
Web ETC GUI
(CGI Forms)

APT ETC Web Based Architecture (multi-process)

Figure 4.

After looking at the different combinations of WEB GUI and APT server models, there were three models we chose to evaluate.

Server is a Java Servlet with Static Web GUI pages

Pros

- Server Running at STScI
- Same WEB ETC GUI as today
- Same model for processing as today
- Can scale with the number of requests
- Can Distribute load to other servers if necessary

Cons

- Will require extending APT ETC engine to support servlets
- Need servlet server to support this approach
- Duplication of work keeping two GUIs current

Server is a Java Servlet with Dynamic Web GUI pages

Pros

- Server Running at STScI
- Same WEB ETC GUI as today
- Can scale with the number of requests
- Can distribute load to other servers if necessary
- Can provide more immediate feedback on legal/illegal instrument specifications upon entry

Cons

- Will require extending APT ETC engine to support servlets
- Need servlet server to support this approach
- Changes current WEB ETC processing model
- No techincal experience with Dynamic Web pages
- Duplication of work keeping two GUIs current

Server is multi-process with Static GUI pages.

Pros

- Server Running at STScI
- Same WEB ETC GUI as today
- No changes needed to APT ETC engine/OCM

Cons

- Need to have data conversion routines to convert data from GUI format to APT ETC format
- Could put a burden on STScI computing resources to support servers at peak loads
- Duplication of work keeping two GUIs current

Development Team Recommended Approach

This is the recommended approach from the **development** team.   It was felt that the running the server as a Java Servlet with Static Web GUI pages would be the best solution to provide a web based ETC that was the same as we have today with the best user response time.

4. FTE Costs to Support a Web-Based ETC

This section provides the costs to support the different technical approaches for providing a web based ETC. These estimates are just the developer costs associated with the task. They do not include testing costs from the Instrument Groups or from independent testing, documentation update costs, or GUI evaluation costs from users. The cost to develop is the initial FTE cost to extend the APT infrastructure to support a web GUI. The cost to maintain is the cost over and above the baseline APT support to maintain the web GUI after it is developed. The cost to maintain is given in FTE per year. The cost to develop is given in full time FTE. These are not schedule time, but level of efforts.

Approach	**Cost to Develop**	**Cost to Maintain**
Applets	4 FTE months	.25 FTE per/yr
Server is a Java Servlet with Static Web GUI pages (Recommended Approach)	3 FTE months	.15 FTE per/yr
Server is a Java Servlet with Dynamic Web GUI pages	7 FTE months	.25 FTE per/yr
Server is multi-process with Static GUI pages	2 FTE months	.15 FTE per/yr