Support of the new VLBA systems at Green Bank

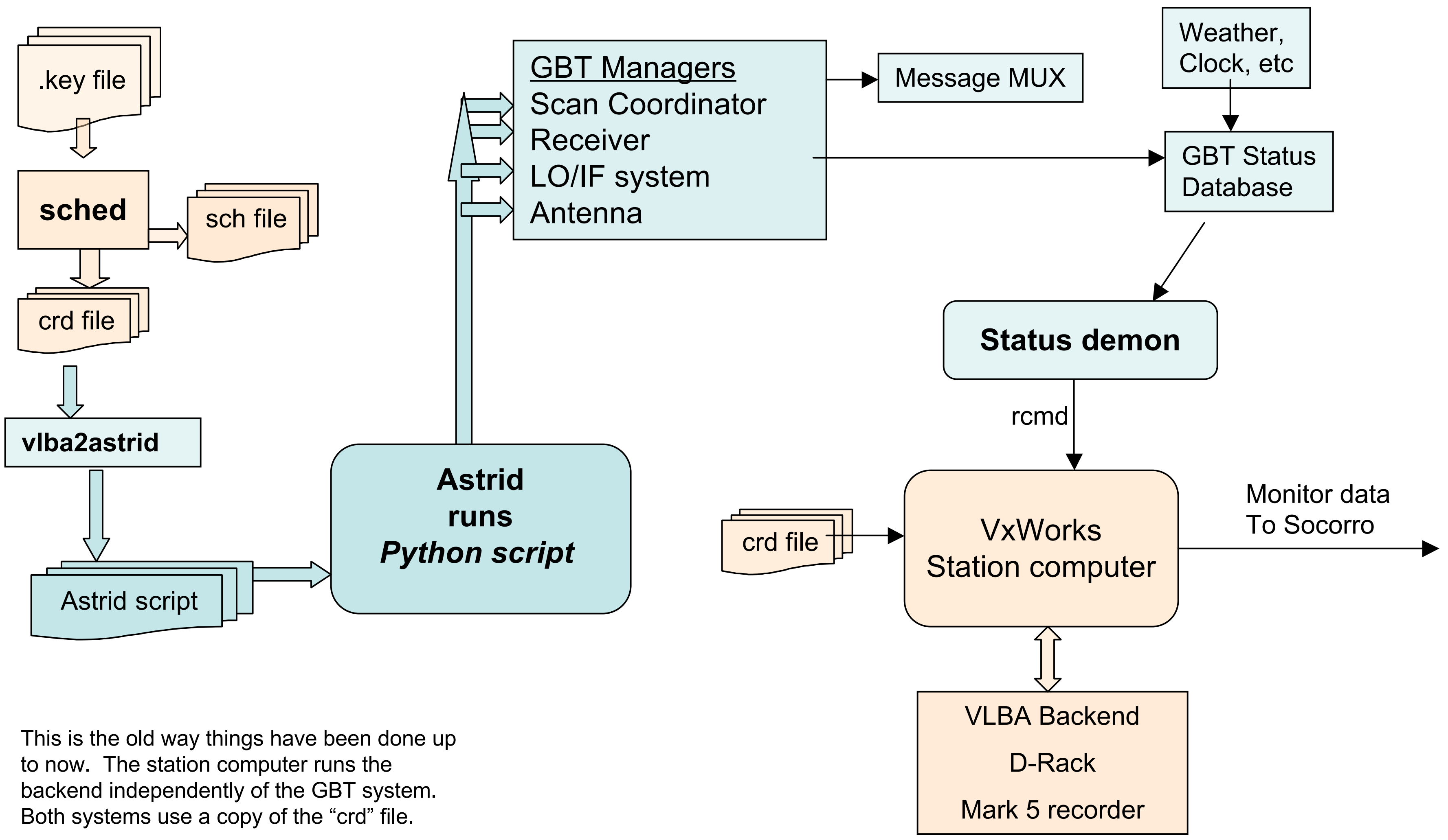
- how its done now
- GBT systems
- alternate plans

Goals

- GBT appears to the user as a VLBA antenna
- Integrate new VLBA back end fully into the GBT system

Plan zero

The way it works now.



This is the old way things have been done up to now. The station computer runs the backend independently of the GBT system. Both systems use a copy of the “crd” file.

GBT Observing

Observations consist of two parts: configuration and scans.

- configuration phase (all under computer control):
    - select the receiver
    - set LO frequencies
    - Make IF connections
    - Set backend modes (channels, sample rate, etc, etc)

- scan phase
    - the backend is commanded to start recording data, coordinated with the antenna tracking the correct source.
    - The configuration cannot change during a scan.
    - Configurations may change between scans.
    - The user may run several data taking scans using the same configuration.

- coordination of the devices is done by a special Manager program called the Scan Coordinator, which insures that the antenna tracking and the data recording start and stop together.

Astrid

- The Astrid user interface is a GUI which lets the observer manage and execute the observing scripts. Each observing script is put into a database referenced by the project code. A script may be selected and run singly, or several scripts may be queued up and run successively.

- These scripts are simply Python programs. There is a suite of Python functions specifically for controlling the telescope in various ways, and for sending commands to the various devices.

- for VLBA projects, we might imagine that we would add a python function that ingests the output of “vex2py” and runs it in Astrid. Processes in Astrid would translate the code and generate commands for the various Managers, which in turn control the devices, e.g., antenna, LO, backend.

Managers

A GBT Manager is a piece of software which controls a device or a related group of devices.

• Managers have standard access methods which may be used to control and monitor the devices. Access is through RPCs from other processes which may be on any node in the local network.

• Sending commands to a device is effected by setting "parameters" in a Manager. Monitoring of status and feedback information is done by connecting to samplers.

• Messages and alerts from a device are sent to a message MUX which may alert the user or the telescope operator of error conditions.

• To properly integrate any backend into the GBT system, it is beneficial to create a Manager for that backend.

A VLBA Manager

A Manager for the VLBA back end

- would command the 4x4 switch, the RDBE, and the Mark5 to the correct modes as specified by the user.
- It would tell the recorder to start and stop recording.
- It would collect GBT status information and multicast it.
- It would listen for messages and alerts from the VLBA back end devices.
- It would provide status information to the GBT operator through the standard system of sampler displays.

Possibilities for a Manager

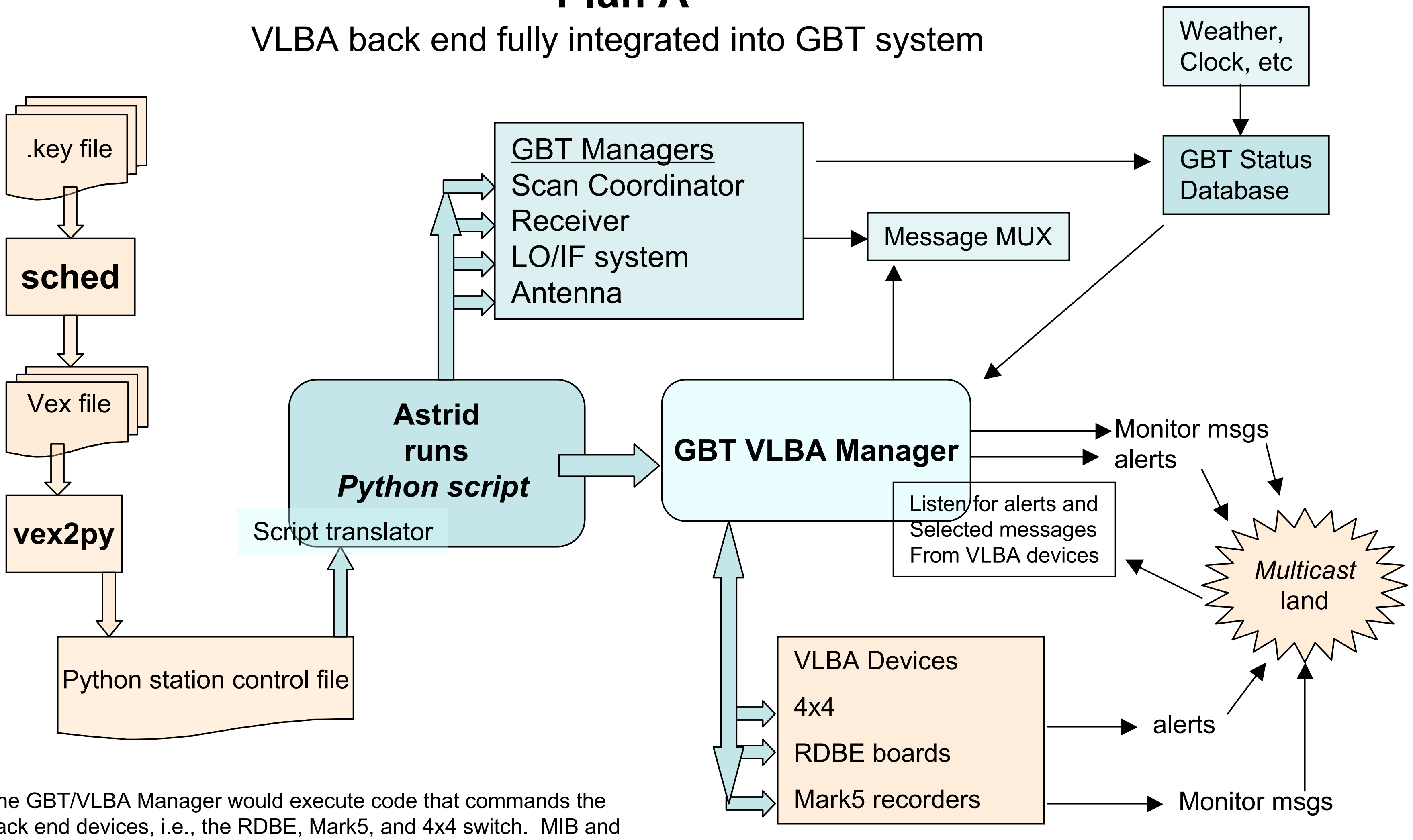
There are a couple of methods I can think of for implementing such a Manager.

a. The control of the 4x4 switch, RDBEs and Mark5s can be done directly by sending MIB commands and XML commands to the devices. In this scenario, we would not use the VLBA Executor. (That’s plan A)

b. Alternately, we could use the Executor program but require it to accept commands through a socket connection, much as the existing VxWorks computer does with its rcmd system (plan B) The Manager would send commands to the Executor via a socket, which would respond by sending the appropriate commands to the RDBEs, and Mark5s.

Plan A

VLBA back end fully integrated into GBT system

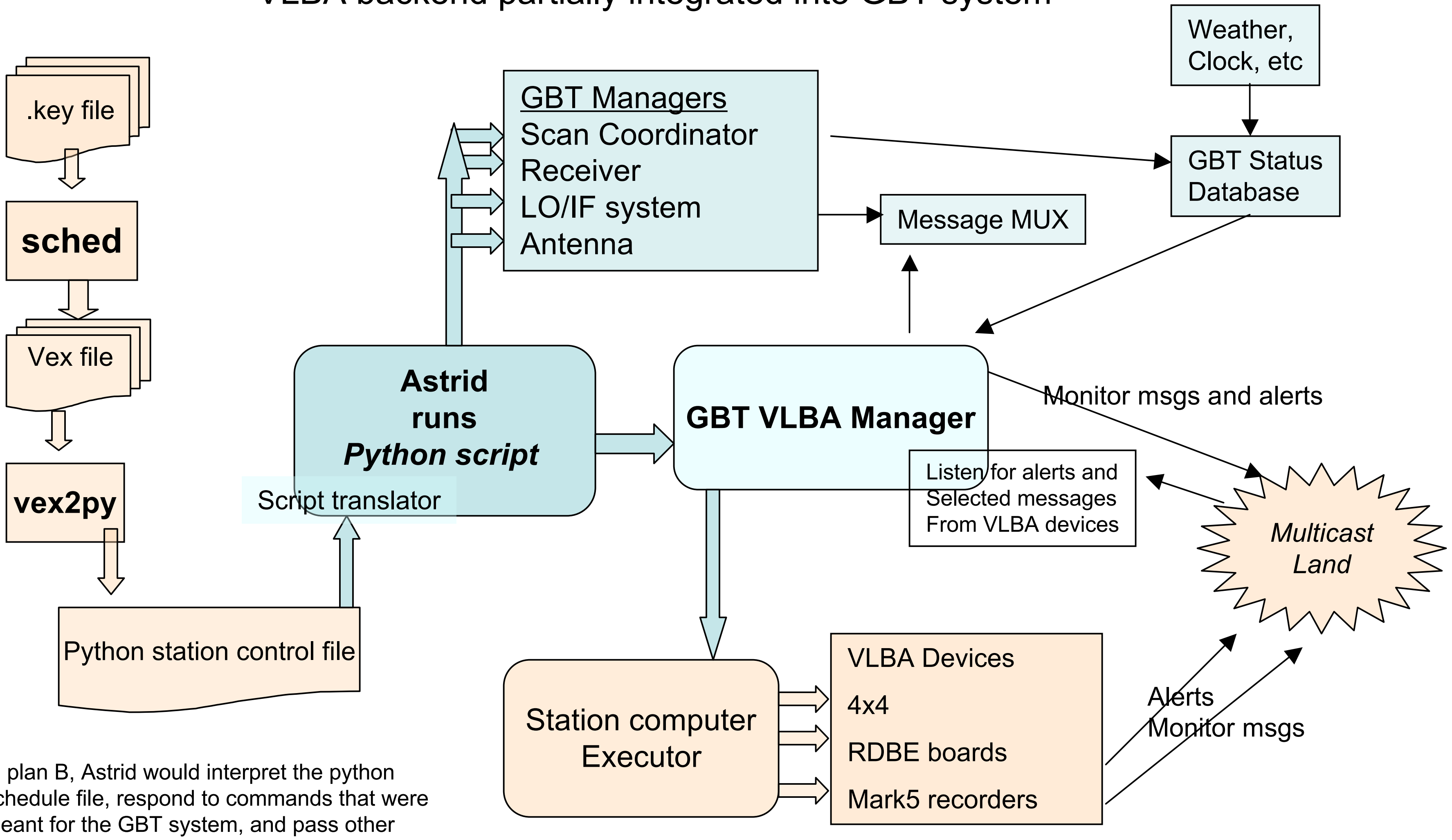


The GBT/VLBA Manager would execute code that commands the back end devices, i.e., the RDBE, Mark5, and 4x4 switch. MIB and XML commands would be sent to the device controllers directly.

To do this, we could copy parts of the code that translates Executor commands into VLBA device commands and use them in the Manager.

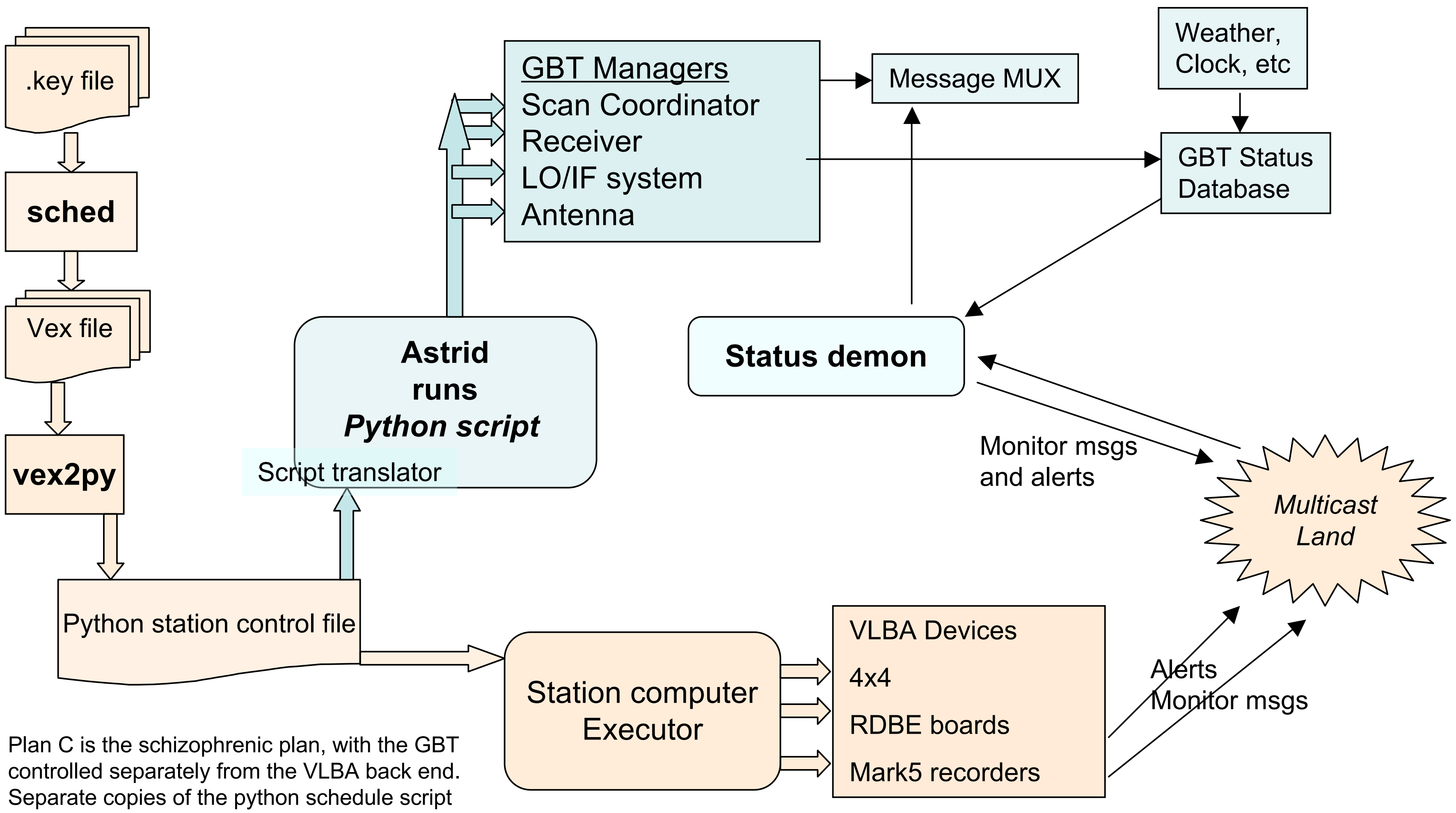
Plan B

VLBA backend partially integrated into GBT system



In plan B, Astrid would interpret the python schedule file, respond to commands that were meant for the GBT system, and pass other commands to the VLBA station computer executor via the Manager, which would in turn generate commands for the backend devices.

Plan C
VLBA backend runs independently



Plan C is the schizophrenic plan, with the GBT controlled separately from the VLBA back end. Separate copies of the python schedule script are run by Astrid and the VLBA executor.

A status demon collects the monitor data from the GBT system and multicasts it out to the VLBA monitor system. It may also collect messages and alerts from the VLBA devices and send them to the messge MUX..

Implementation Plan

1. Initial Implementation (“plan C”)
   a. Receiving, parsing, creating, sending multicast messages, alerts, and monitor data.
   b. Translating VLBA control files to Astrid scripts.

2. Approaching full integration (plans “B” and “A”)
   - adapt code from VLBA Executor for use in VLBA Manager.