Towards Automating Operations of SGP VGOS Stations

Jim Lovell[1], Darryl Lakins[2]

Abstract The NASA Space Geodesy Project (SGP) aims at establishing a network of up to ten VGOS stations in the next several years. The goal is to have the network controlled and monitored from the Space Geodesy Network Operations Center (SGNOC) with the local operations teams focused on safety, maintenance, and the technical aspects of operations. To attain this high level of automation, a set of use scenarios and requirements were defined for the Next-Generation Field System (NGFS). The initial steps on the road to the NGFS and automated operations were made with additions to the current Field System (FS). *Fesh2* is an automated schedule file management module that checks the IVS data repositories for current master and session schedule files, downloads updated versions, and prepares stations for observing. In addition, work was done on an SGP Automation module that will take over local operator hands-on FS tasks before, during, and after an observing session. In this paper we describe the new modules as well as future steps.

Keywords Software, Observations, Operations, Scheduling, Correlation

1 Introduction

A core component of VLBI systems is the software responsible for handling site operations and at most IVS stations this is the NASA Field System. Development of the current Field System began over 40 years ago and it is currently in use at over 30 sites worldwide. It is an essential component of the IVS network and is also widely used for astronomical VLBI. With the advent of VGOS, the number of stations in the NASA Space Geodesy Project (SGP) network increases with up to ten planned, as well as new broadband feeds and receivers, digital back ends and high bandwidth networks. It was determined that a systematic approach is required to design a new system based on this extensive experience with the Field System. The Next Generation Field System (NGFS) will be built, so it can support VGOS stations, can be effectively maintained, is adaptable to future technological developments, and is capable of a high level of automation.

The NASA vision is for a Space Geodesy Network Operations Center (SGNOC) to coordinate operations of all geodetic techniques supported by the SGP, including VLBI (Figure 1). The SGNOC may not need to be confined to a single location but could be virtualized, capable of running in a follow-the-sun mode. Ultimately, in a "post-integrated operational network", a lights-out mode is envisaged where operation is fully automated and site on-call personnel are only required to resolve safety-related anomalies, respond to emergencies, or manage occasional hands-on tasks such as maintenance. The NGFS will handle communication with the SGNOC

In preparation for development of the NGFS we have examined future usage scenarios and, with the assistance of the IVS community, documented the software requirements necessary to achieve them. Usage scenarios encompass operations; station calibration (e.g., pointing, amplitude calibration, phase and cable-delay measurement); equipment and proce-

(1) NVI, Inc./NASA Goddard Space Flight Center, 7257 Hanover Parkway, Suite D, Greenbelt, MD 20770, USA
(2) NASA Goddard Space Flight Center, Greenbelt, MD 20771, USA

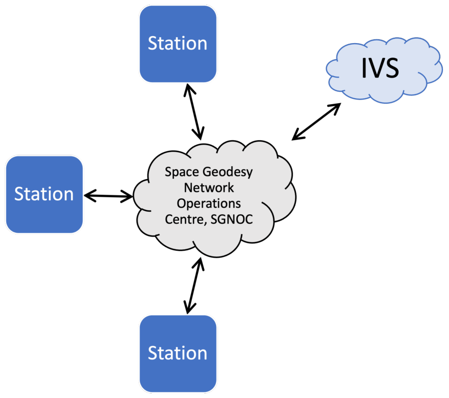


Fig. 1 The Space Geodesy Network Operations Center (SGNOC) provides coordination of station activities and a conduit to the IVS for schedules, logs, status reports, etc.

dure testing; antenna mechanical performance testing and monitoring; system sensitivity and stability tests; fringe checking; and maintenance tasks.

Work on a development strategy is now underway with coding and testing to follow in the coming few years.

In the meantime, we are developing some of the techniques required for automation using the existing Field System and a test VLBI Communications Centre (VCC), analogous to the VLBI component of a future SGNOC [1]. The new software will be tested and eventually deployed at the SGP stations, producing information on how well automation works in practice, which will help speed up development of and transition to the NGFS.

This work has started with two initial station-based software applications: *fesh2* and *SGPAutomate*. In the following sections we describe these, their current status, availability, and future work.

2 Fesh2

Fesh2, a progression of *fesh*, which is available as part of the Field System, is a Python package that provides automated schedule file preparation.

Fesh2 runs in a terminal window and regularly checks IVS schedule servers for new or updated versions of master and schedule files for the specified station. A check for the latest version of the master file(s) is done first, but skipped if the time since the last check is less than a specified amount. Similarly, checks on schedule files are only done if the time since the last check exceeds a specified duration. If new or updated schedules are found, they are optionally processed with *drudg* to produce `snp`, `prc`, and `lst` files. By default, once the files have been checked, *fesh2* will provide a summary and then go into a wait state before carrying out another check. *Fesh2* can also be run once for a single check or status report and not go into a wait state. *Fesh2* can also be run in a mode that gives a continually updating status report but does not process any files. Multiple instances can be run simultaneously. If *drudg* output `snp` or `prc` files have been modified at the station and a new schedule becomes available, *fesh2* will download the file but not overwrite *drudg* output, but it will warn the user.

Fesh2 can be run as a foreground application or as a service in the background. It is compatible with Python version 3.5 and above. Figure 2 shows the *fesh2* user interface. Is this case the AuScope station at Katherine is being managed. Both the 24-hour and Intensive master files are being monitored and any schedule for this station within the following 14 days is being monitored and processed.

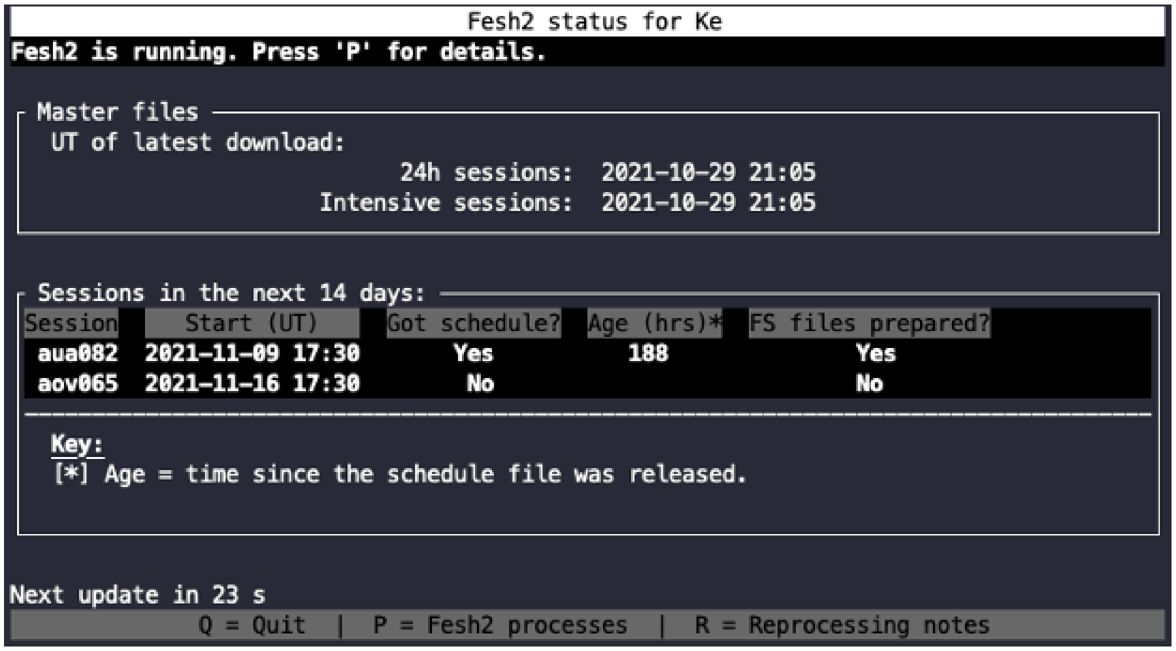
2.1 Availability

At time of writing (June 2022), *fesh2* is being prepared for release and will be made available via the NVI repository on GitHub [2].

2.2 From fesh2 to fesh3

While *fesh2* communicates with the existing IVS file servers to monitor and maintain schedule files, *fesh3* will adopt an approach much closer to the NGFS model. *Fesh3* will maintain a communications link with a test VCC server, receiving push notifications of schedule changes (which is more efficient than regular polling from the station to IVS servers) and providing feedback to the VCC of schedule status at the station.

Fesh3 is currently at a proof-of-concept stage and, given its dependancy on a test VCC server, will initially be tested and deployed at SGP stations only. However,



```
                              Fesh2 status for Ke
Fesh2 is running. Press 'P' for details.

 Master files
  UT of latest download:
                              24h sessions:  2021-10-29 21:05
                        Intensive sessions:  2021-10-29 21:05


 Sessions in the next 14 days:
 Session    Start (UT)      Got schedule?  Age (hrs)*  FS files prepared?
  aua082  2021-11-09 17:30       Yes          188              Yes
  aov065  2021-11-16 17:30        No                           No
 --------------------------------------------------------------------------
  Key:
  [*] Age = time since the schedule file was released.


Next update in 23 s
                Q = Quit  |  P = Fesh2 processes  |  R = Reprocessing notes
```

Fig. 2 The *fesh2* user interface, in this case for AuScope Katherine station. It shows the status of the latest master files downloaded, then a list of sessions over the following two weeks and their schedule status. In the case of aua082, the schedule file has been downloaded and the necessary Field System files produced with *drudg*. The schedule for aov065 is yet to be released but *fesh2* is regularly polling the IVS file servers and will download and process it once available.

it is hoped that the work on *fesh3* will inform NGFS development.

3 SGP Automation Software

At present, many of the session tasks carried out at the stations are done by hand via Field System commands which are time consuming and sometimes error prone. The *SGPAutomate* software is intended to provide automation of as many tasks as possible to allow local operations teams to focus on safety, maintenance, and the technical aspect of operations. Like *fesh3*, *SGPAutomate* is written in Python and is compatible with version 3.5 and above.

3.1 Code Structure

SGPAutomate interacts with the Field System server (`fsserver`) and is modular at the level of distinct operational tasks, which can be grouped to suit any defined scenario. For example, a typical IVS session consists of four *sequences* of tasks: pre-session checks, start of session tasks, in-session checks and monitoring, and post session tasks. Each task within each *sequence* is implemented in a separate module as defined in the software and can be used once or in multiple groupings as desired. Listing 1 shows part of the *SGPAutomate* configuration file that defines each *sequence* and the tasks to be carried out in each. Pre-session tasks start with a Network Time Protocol (NTP) check, `CheckNTP` (line 11), which checks that the computer time is synchronized with the NTP server. This is followed by opening a log file, checking hardware, and then carrying out a range of other system checks. The second *sequence* (line 21 onward) contains tasks that are carried out at the start of an experiment. It can be seen here that individual tasks can be re-used. In this case, `CheckNTP` is repeated. There is also a facility to continuously repeat a sequence, which may be desirable for regular system checks during a session. This may be configured by setting the `repeating` variable and specifying a time gap between repeats with `repeat_gap_min`.

```
[sequences]
    # Arrays contain a list of tasks (given above)
     in the order they should be executed for a a
    # specific activity. For example, the
     PreSession sequence contains all tasks to be
     carried out
    # before a sesion starts
    [sequences.PreSession]
        name = "Pre-session"
        description = "Procedures to be carried out
      before the session."
        repeating = false
        repeat_gap_min = 0
        tasks = [
            "CheckNTP", "StartFSLogFile",
            "CheckRDBE", "CheckMark6",
            "CheckMCI", "MountMark6",
            "CheckTiming",
            "InitializePointing",
            "SetModeAtten", "CheckRDBEs",
            "CheckPointing",
            "TestRecording"
        ]

    [sequences.StartExperiment]
        name = "Session start"
        description = "Procedures to be carried out
      at the start of the session."
        repeating = false
        repeat_gap_min = 0
        tasks = [
            "CheckNTP",
            "StartMulticastLogging",
            "SendReadyMessage", "StartSched",
            "SendStartMessage"
        ]
```

Listing 1 A section of a *SGPAutomate* configuration file defining procedure sequences which each consist of several tasks. See the text for a more detailed description.

The software will run each task in order and process output to decide if it was completed successfully or not. It will issue warning messages if there are problems and pause the sequence to allow for operator intervention if the task is unsuccessful.

SGPAutomate has a graphical user interface (GUI) and a limited text-based interface. Figure 3 shows an annotated screenshot of the current (unfinished) GUI. It consists of a status summary at the top followed by a tabbed section, one for each of the defined sequences. Each task is listed with the following features:

- The task can be selected for execution or not;
- A link is provided to documentation that describes the task;
- The user can choose to be prompted or not before continuing after the task;
- The task status (pending, done, or unsuccessful);
- If the task has been run, a link to the relevant part of the log is given; and
- It is possible to run each task individually when a sequence is not underway using the `[run]` button.

Typically a user will select a sequence, confirm the tasks to be carried out, and then press the `[Go]` button to commence. For tasks that cannot be fully automated (e.g., mounting a Mark-6 module) *SGPAutomate* will pause the sequence and notify the operator that manual intervention is required.

Lastly, the GUI shows a log of interaction with the Field System server which can be filtered to search for events of interest.

To assist stations wishing to add new tasks, *SGPAutomate* includes commented template Python code and accompanying documentation as a guide.

3.2 Next Steps

Work is continuing to implement as many tasks as possible. At the moment the focus is on tasks required to support a geodetic session, but the software could be easily expanded for other activities such as end-to-end system checks, pointing or SEFD calibration observations, or other maintenance tasks.

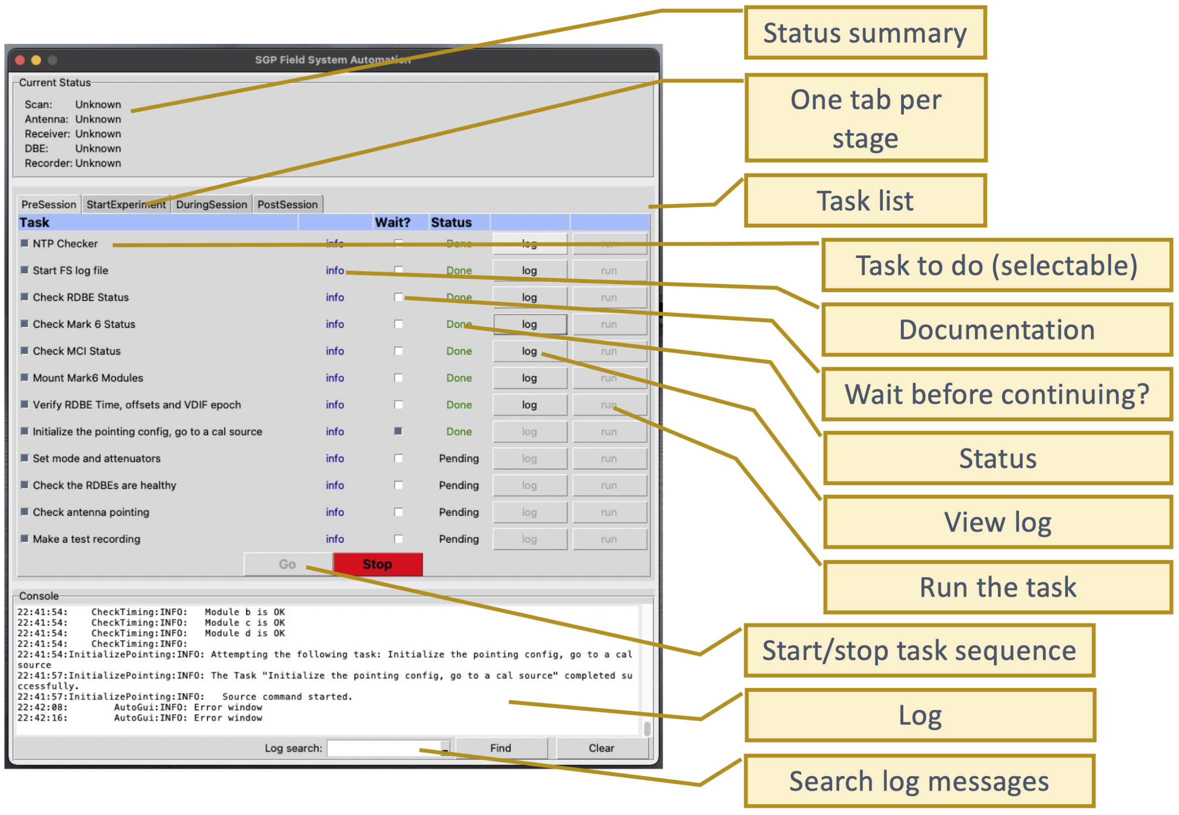
It is planned to add VCC communications to *SGPAutomate* in the future so that the station can automatically provide live status information to the operations center and bypassing (for example) the need to send ready, start, and stop emails.

It is hoped that, once the software has reached a higher level of maturity and has undergone testing, a version may be released more widely to accompany the Field System as an optional addition.

4 Conclusions

Fesh2, *Fesh3*, and *SGPAutomate* are placed outside the main FS distribution so they can be developed and maintained on a different timescale. However, attention should be made to the FS compatibility as commands and responses, which *SGPAutomate* uses, may change with new releases.

It should also be noted that similar developments are occurring elsewhere (e.g., [3, 4]) and there may be opportunities for coordination of efforts.



Current Status

Scan:	Unknown
Antenna:	Unknown
Receiver:	Unknown
DBE:	Unknown
Recorder:	Unknown

Task		Wait?	Status		
NTP Checker	info	☐	Done	log	run
Start FS log file	info	☐	Done	log	run
Check RDBE Status	info	☐	Done	log	run
Check Mark 6 Status	info	☐	Done	log	run
Check MCI Status	info	☐	Done	log	run
Mount Mark6 Modules	info	☐	Done	log	run
Verify RDBE Time, offsets and VDIF epoch	info	☐	Done	log	run
Initialize the pointing config, go to a cal source	info	■	Done	log	run
Set mode and attenuators	info	☐	Pending	log	run
Check the RDBEs are healthy	info	☐	Pending	log	run
Check antenna pointing	info	☐	Pending	log	run
Make a test recording	info	☐	Pending	log	run

Console

```
22:41:54:    CheckTiming:INFO:   Module b is OK
22:41:54:    CheckTiming:INFO:   Module c is OK
22:41:54:    CheckTiming:INFO:   Module d is OK
22:41:54:    CheckTiming:INFO:
22:41:54:InitializePointing:INFO: Attempting the following task: Initialize the pointing config, go to a cal
source
22:41:57:InitializePointing:INFO: The Task "Initialize the pointing config, go to a cal source" completed su
ccessfully.
22:41:57:InitializePointing:INFO:   Source command started.
22:42:08:        AutoGui:INFO: Error window
22:42:16:        AutoGui:INFO: Error window
```

Fig. 3 A development version of the graphical user interface for the *SGPAutomation* software. See the main text for a description.

References

1. M. Berube, J. Gipson, J. Lovell and D. Lakins 2022, “Improving IVS communication through a VLBI Operating Center”, these proceedings
2. NVI Inc GitHub web page `https://github.com/nvi-inc`
3. A. Neidhardt et al. 2019, “The Smart Observatory for Autonomous and Remote Observations”, In Armstrong, Baver, and Behrend, editors, IVS 2018 General Meeting proceedings, `https://ivscc.gsfc.nasa.gov/publications/gm2018`
4. A. Neidhardt 2019a , “A New Generation of Wettzell’s Remote Access to the NASA Field System using Web-based Techniques”, In Armstrong, Baver, and Behrend, editors, IVS 2018 General Meeting proceedings, `https://ivscc.gsfc.nasa.gov/publications/gm2018`
5. A. Neidhardt 2019b, “Install and configure the monitoring of EVN Jumping JIVE and IVS Seamless Auxiliary Data (operation, diagnostic, and analysis data)” `https://www.haystack.mit.edu/workshop/TOW2019/files/Seminars/Neidhardt_Sem1.pdf`