NANOSTAR MODULE

Activity Profile

Interreg Sudoe
EUROPEAN UNION
NANO STAR
European Regional Development Fund

Contents

1. INTRODUCTION

NANOSTAR is a European SUDOE project. One of its goals is to develop a software suite to enable student teams and university staff to perform preliminary analyses and design small satellites.

Nanostar software suite (NSS) needs a visualization module that allows the team to communicate internally, to check and validate mission analysis and scenario and to get pretty communication outputs. . This visualization module shall interact in terms of input/output with other software in the NSS.

The goal of this module is:

- to facilitate satellite mode management (the activity profile of the spacecraft during the mission)
- to ease NSS interaction with VTS, a multi-purpose visualization tool developed by CNES.

2. OVERVIEW.

The activity profile module consists on a user friendly interface (front-end) for manipulating a schematics representation of an activity profile, i.e., a sequence of working modes for a satellite. In particular, the infrastructure allows the user to carry out the following tasks:

- Timeline creation, manipulation
- Drag and drop of satellite modes on the timeline
- Visualize or hide eclipse scheduled on the timeline.
- Navigate through the timeline.

The activity profile package is organized in a standard project folder structure as follows:

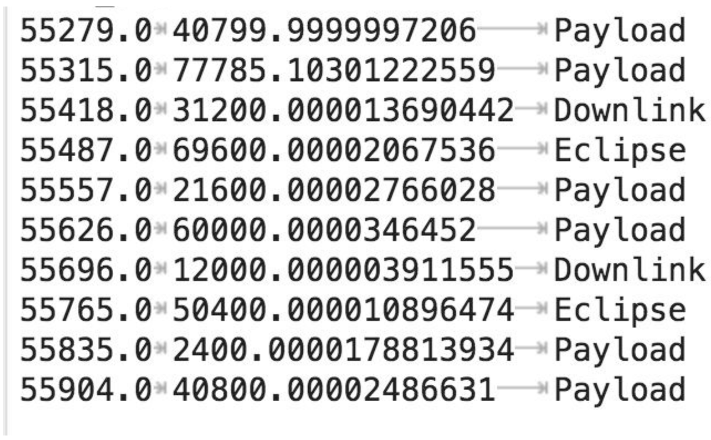
- doc/
- README.md
- src/
- test/

where the doc folder contains the documentation files, which is briefly summarized into the README.md file. The software is distributed along with open source MIT license. The source code can be found in the src directory.

3. INPUTS.

The following inputs have been defined. Some of them are optional, other are required:

- Satellite position data with the CCSDS standard OEM data in a .txt format [**optional]**.
- NSS database project exported as a json file [**required**].
- Mode listing data with the CCSD OEM in a cvs format file [**optional**]. An example of a user-provided mode-listing OEM File is shown in **Figure 1**.



```
55279.0	40799.9999997206	Payload
55315.0	77785.10301222559	Payload
55418.0	31200.000013690442	Downlink
55487.0	69600.00002067536	Eclipse
55557.0	21600.00002766028	Payload
55626.0	60000.0000346452	Payload
55696.0	12000.000003911555	Downlink
55765.0	50400.000010896474	Eclipse
55835.0	2400.0000178813934	Payload
55904.0	40800.00002486631	Payload
```

Figure 1: Example of a user-provided mode-listing OEM fil

4. OUTPUTS

The following outputs have been defined:

- Activity profile with the CCSDS standard OEM in a txt file [
- Satellite position data with the CCSDS standard OEM data in a .txt format.
- Satellite eclipse data with the CCSDS standard OEM data in a .txt format
- VTS file (.vts) (respecting the VTS input/output format)

5. INSTALLATION AND USAGE

The Activity profile module is programmed using python 3.7 programming language to be run within the JupyterLab computing environment. The interactive visualization is build upon the dash framework. Dash is a productive Python framework for building web applications. Written on top of Flask, Plotly.js, and React.js, Dash is ideal for building data visualization apps with highly custom user interfaces in pure Python. It's particularly suited for anyone who works with data in Python.

The following python packages has to be installed before running the web-based application in Jupyter Lab:

```
$ pip install plotly==4.5.0
$ pip install dash==1.8.0
$ pip install dash-html-components
$ pip install dash-core-components
$ pip install json
$ pip install numpy
$ pip install flask
$ pip install dash_bootstrap_components
$ pip install jupyter dash
```

As a first step, open the file *MAIN_activity_profile.ipynb* run it. Thereafter, a pop-up window will appear. It is depicted in **Figure 2**. There are two main plots:

- In the upper figure the sequence of spacecraft modes is illustrated with respect to time for a default scenario. Each color represent a different working mode. The boxed and draggable and resizable, allowing the user to interactively modify the mode sequence. By hovering on the upper side of lower side of each rectangle, the label for each mode will be displayed (e.g., payload, eclipse, downlink)

- In the lower figure, a set of parameters describing the evolution of the semimajor-axis, the inclination, the eccentricity, the cartesian position and velocity, the visibility with respect to the ground station, and the eclipses during the trajectory. The specific parameters to be plotted can be selected with the dropdown menu.

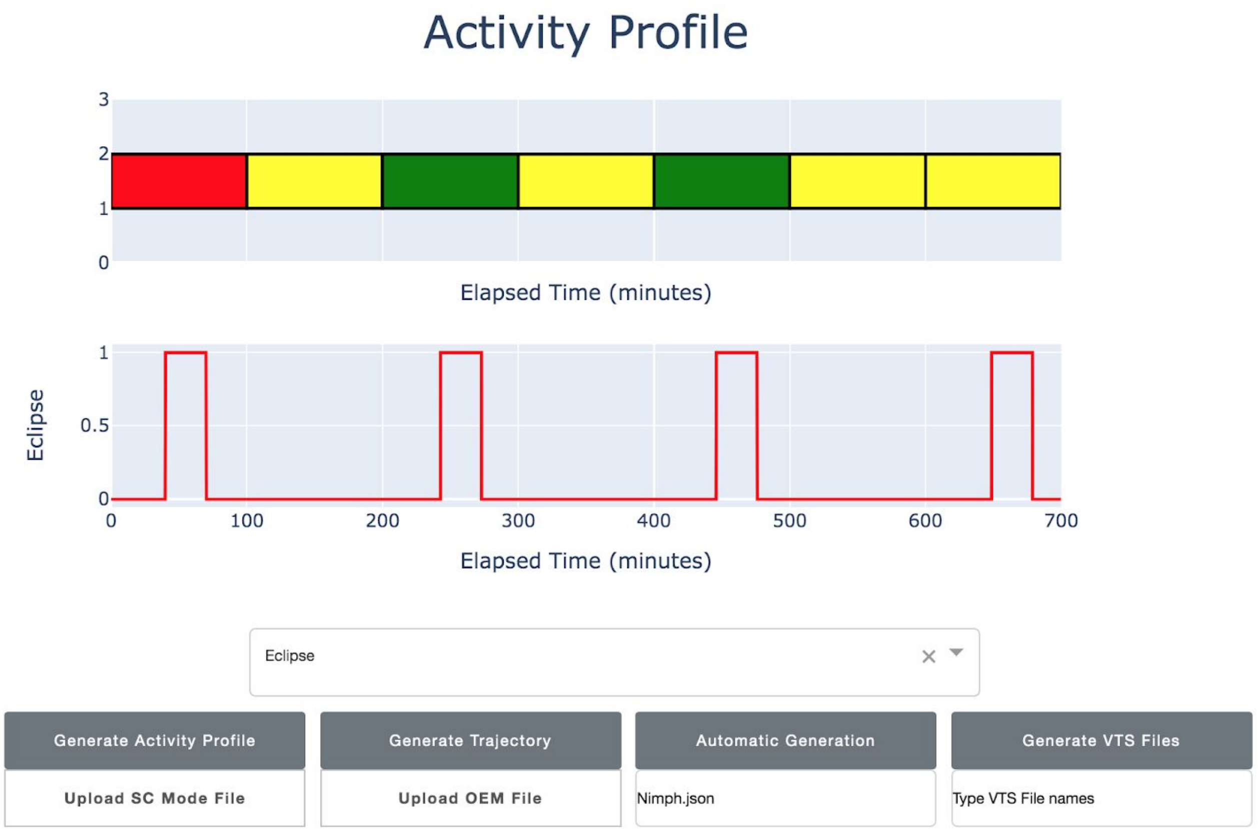


Figure 2: Activity profile default main window.

The are two different ways of utilizing the tool:

- In the former case, the user can provide an OEM file describing the sequence of modes and a OEM describing position of the satellite over time. First, the user has to load the oem file (with .txt extension), by clicking on the button **Upload SC Mode File.** Then, when clicking on the **Generate Activity Profile**, the new activity will be displayed in the upper figure. In order to generate the orbital element plots, the user has to be upload the ephemerides file (with .txt extension) by clicking on **Upload OEM File**. then, by pressing the **Generate Trajectory** button, the new eclipse, visibility and orbital elements are plotted in the lower figure. The result for the provided example included in the distribution is depicted in **Figure 3**.

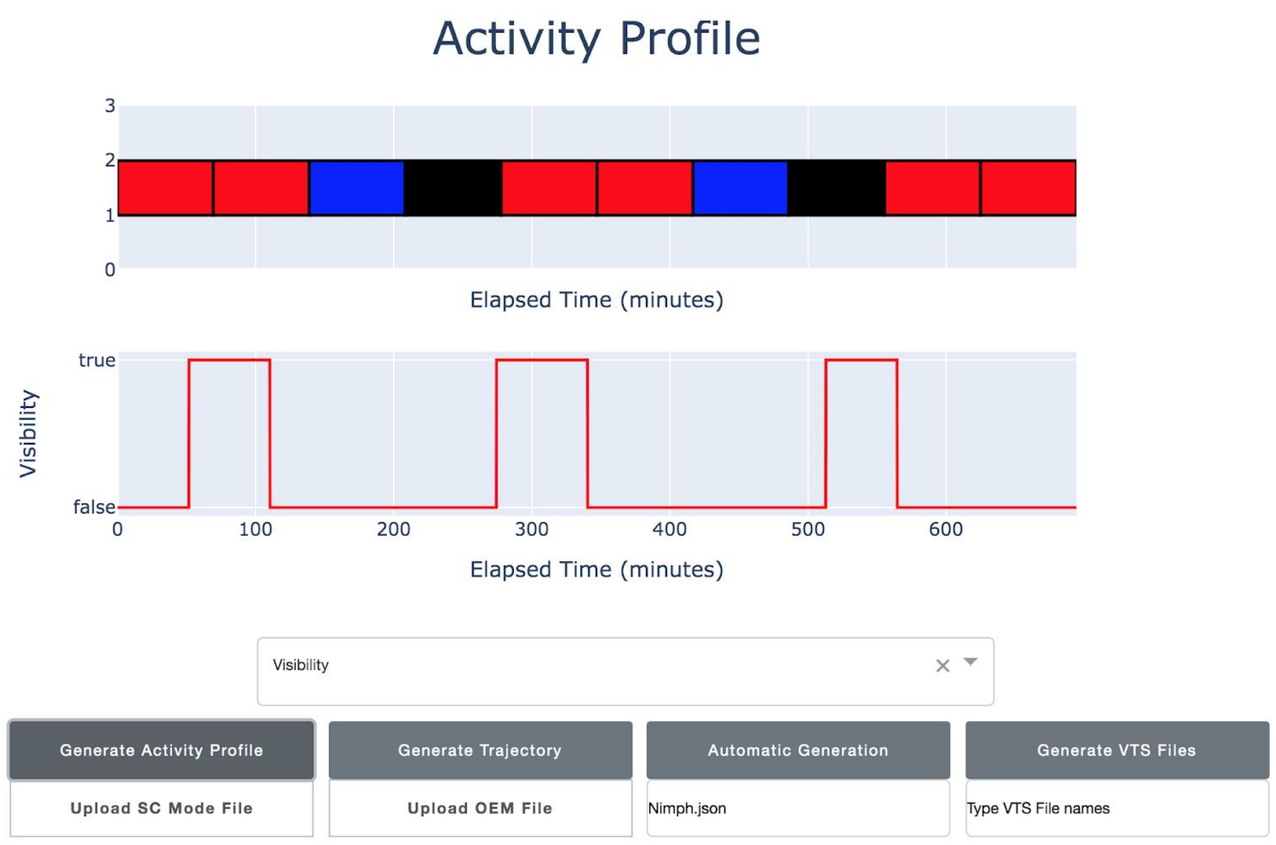


Figure 3: Activity Profile main window after loading the default SC Mode File

- In the second scenario, the application is able to automatically generate a feasible sequence of modes. Eclipse modes happen when in Earth's shadow, downlink modes are adjusted to happen when the satellite is visible from the ground station as long as there is no eclipse. The remaining time is distributed among the available payload modes. The satellite orbit parameters, In order to generate this profile, the user has to provide a .json file with orbit parameter data, the available modes, and the ground stations. This files follows the NSS standard, and can be directly downloaded from the NSS database. The name of the json File has to be included in the input box (where **Nimph.json** is the default project). Then, the user has to click on the button **Automatic Generation** to visualize the result. The labels for each mode will be displayed while hovering, the different boxed cab de dragged and dropped. **Figure 4** below exhibit the output obtained by using the default file.

- Notably, every MEM mode file (csv) generated by the activity profile application can be used as an input fo the tool.

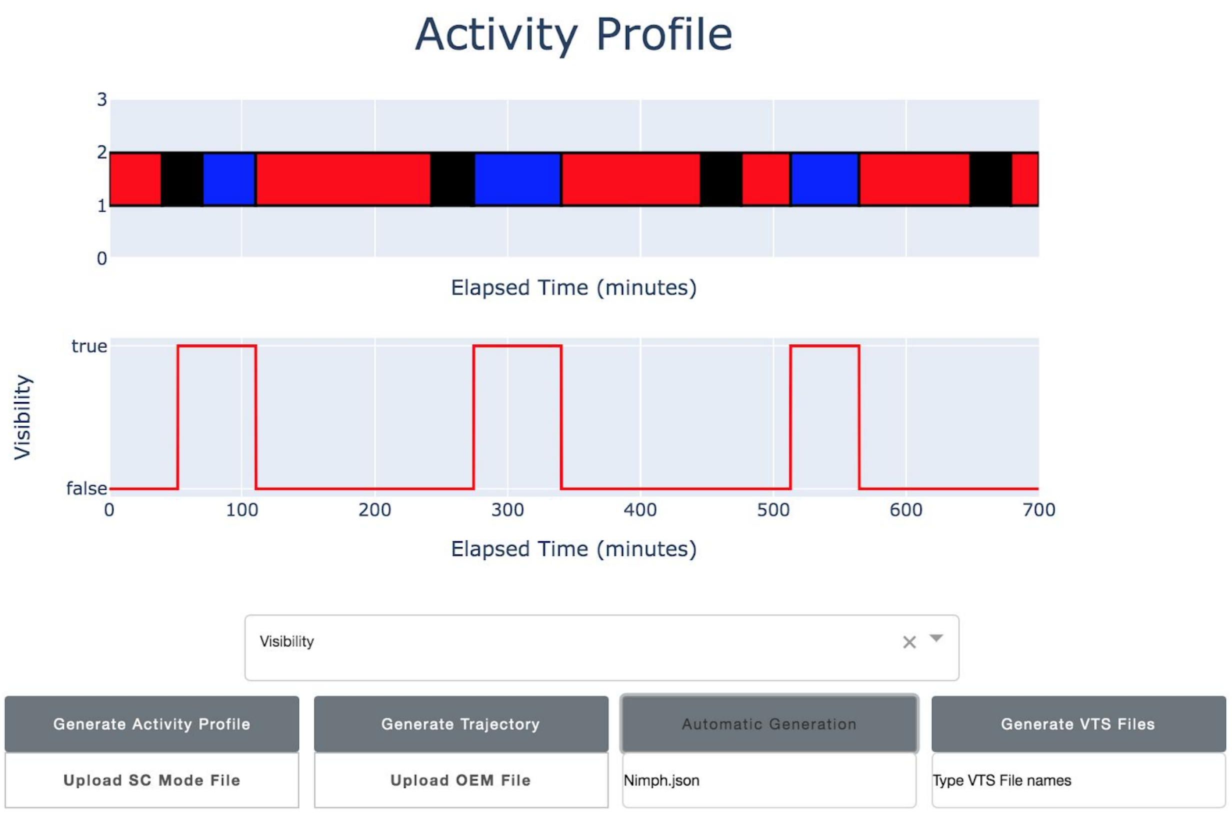


Figure 4: Activity Profile main window after the automatic activity profile generation

Finally, the user is able to generate the required VTS Files. They include the new spacecraft OEM activity profile file, and the .vts file that can be loaded in the VITS time loop tool. The user has to type the desired filename in the Input box **Type VTS File names** and click on **Generate VTS Files** to output the files.

- The application can also be run via the command line promt by running the file MAIN_activity_profile_SERVER.py:

  ```
  $ python MAIN_activity_profile_SERVER.py
  ```

  The webbrower will be opened automatically in the port where the dahs application is deployed.

  The same operations as in the JupyterLab app can be done.

- In the server application, a download link for retrieving the satellite MEM file in csv format can be found. This can be useful for manually editing the cvs file and later upload it back to the server with the new configuration.