Systems Modeling

Prepared by: The Skyentists

Date: 10/17/19

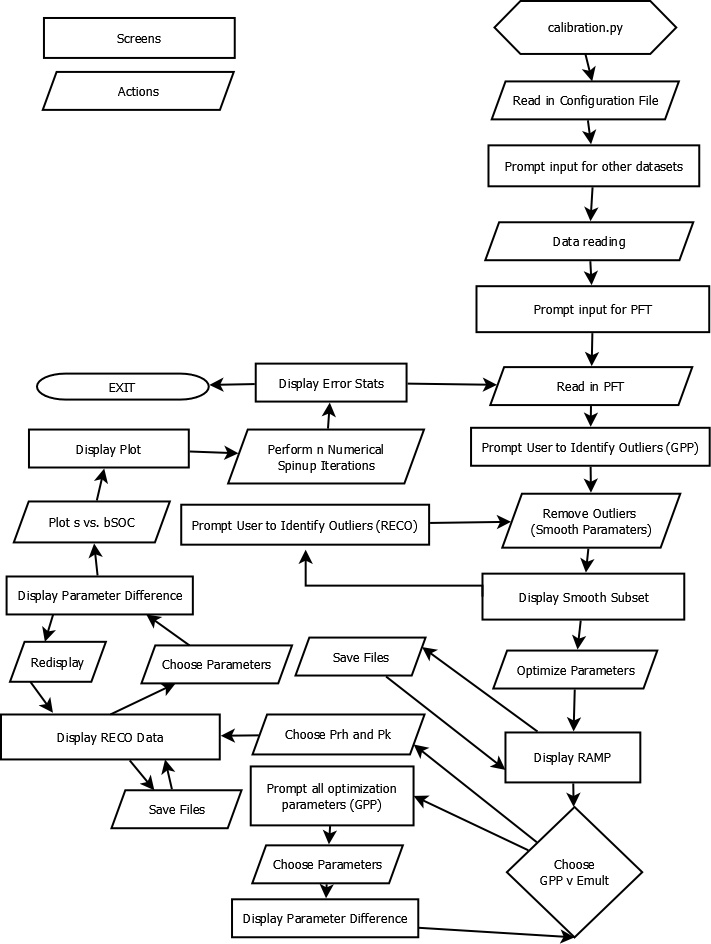
## Document Overview

We create models in software development because workflows can be too complex to design through coding alone. If software developers jump right into coding as soon as they understand the requirements of a system, they could code themselves into a corner if they poorly designed their workflow. A systems model lets the user see the workflow of a system in an abstract manner, allowing for easy modifications without massive changes in codebase. Different models focus on different parts of a system as they complete a business task or process. A commonly used model is Swimlane which focuses on users interacting with other users in a system. Another popular model is the Entity Relationship model, which mimics an Object-Oriented programming approach with classes, attributes, and actions. The rest of the document contains two systems models with the first model, a State Transition Network, focusing on a single user interacting with a system and a second model, a Data Flow Diagram, that focuses on data flow.

## Model #1 – State Transition Network

The state transition network shown in Diagram 1 is a model that shows the different states that the user will interact with during the calibration process. This calibration process helps the user edit the values of the parameters that will be used for in an advanced climate change model. Each square box represents a display and each parallelogram represents an action the user can take. For the most part this diagram moves in one direction. The user inputs data necessary for calibration and the software uses that input accordingly. Occasionally there are small roundabouts in the workflow when the user has the choice to review changed plots or save figures. We chose this model to help visualize our user interface. Each square box represents a different screen we will need to design. We will have to design many displays, but each display seems relatively simple to create. Through making this diagram, we learned that most of the calibration software works in the backend, away from the user’s eyes. We also learned that each screen will have a simple display, but if designed poorly will be hard to understand for those will little knowledge in climatology.

Diagram 1. This model displays the screens that the user will interact with throughout the software within a state transition network.



## Model #2 – Data Flow Diagram

The Data Flow Diagram (DFD) is a type of model that demonstrates the ways the data flows throughout the system and the processes performed to alter this data. The model has a key of circles for processes, arrows for data flows, sandwiches for data stores, and rectangles for external entities. The DFD allows a high-level view on the data and allows for sub-models to be created for more complex functions. This model provides an accurate description of the calibration process due to the program having a plethora of functions and back-end work to transform the data. A DFD also goes hand-in-hand with the functional requirement analysis since it is easy to spot missing functions or requirements that need to be performed on the data.

When creating the DFD, the team learned that the data will be going through many complex processes and data stores during the full calibration process. There was also the identification of more complex functions than previously thought throughout the model creation. After creation and discussion, there are less data stores than were present in the first draft of the model.

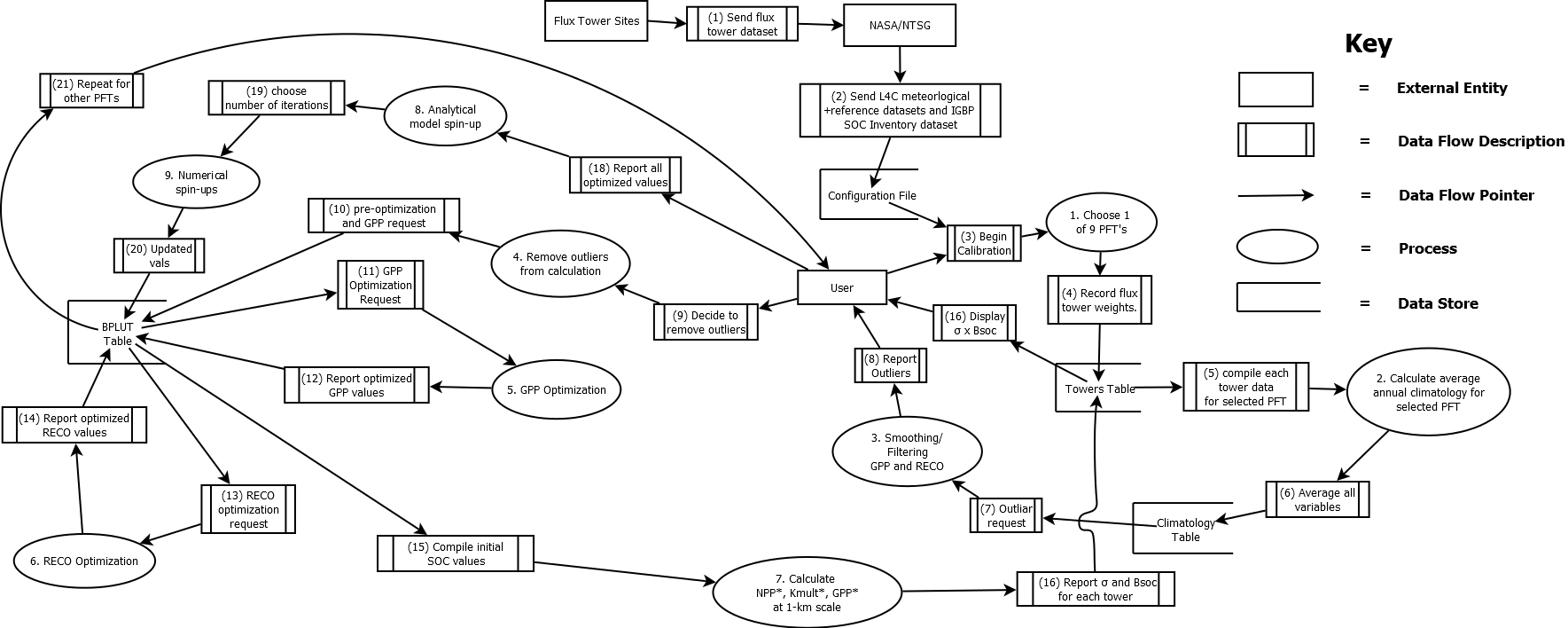
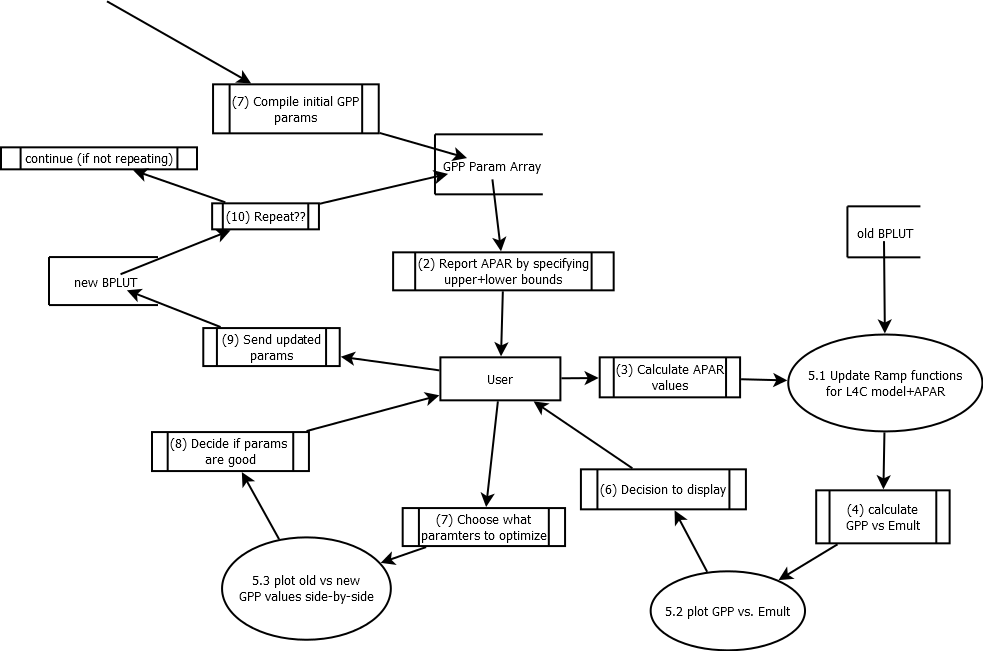
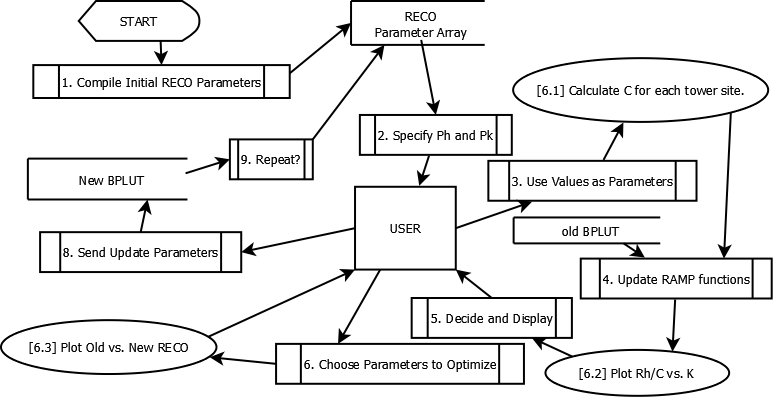
Diagram 2. This diagram shows the highest-level view of the calibration process in a data flow diagram. This details the processes that are performed on the data from the start of the process to the end of the process.

Diagram 3. This diagram displays the GPP optimization process as a whole from the beginning step, compiling the initial GPP parameters, to the final step, sending off the optimized GPP parameters to RECO optimization process (number 5 on Diagram 2).



Diagram 4. This diagram displays the RECO optimization process, similar to GPP optimization in Diagram 3. The entire RECO process from initializing parameters to sending off the optimized RECO parameters to the BPLUT table (number 5 on Diagram 2).

## Execution and Acknowledgement

The team members hereby indicate by their signatures below that they have read and agree with the specifications of this document.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| [Insert name & date here] |  | [Insert name & date here] |  | [Insert name & date here] |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| [Insert name & date here] |  | [Insert name & date here] |  |  |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| [Insert Client name & date here] |  | [Insert Client name & date here] |  | [Insert Client name & date here] |