Recent Progress:

1. An AMQP only solution decided upon for PY3<->IPY communication
2. A more realistic AMQP code process has been written and tested to send / receive many messages as a batch between Python3 and Ironpython. The coded experiment reveals the parallel processing advantage inherent with AMQP.
3. Refactor started.
4. Code packaged as PSLTDSim

   Can/will be posted on PYPI (python package index) once more complete.
5. Simulation runs in IPY after refactor (expected).
6. Added simple code timing.
7. Terminal output reduced for non-debug runs.

   Leads to speedup of ≈2 when running from Visual Studio

   Only about ≈1.5 speedup from terminal runs

   Equates to a simulation 6 to 14.75 times faster than real time (using 0.5 sec time step)
8. New Fact: Github is now owned by Microsoft - allows for free private repos.
9. GitHub repository updated:

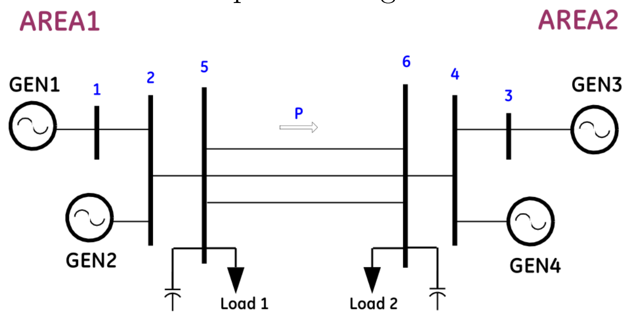
   `https://github.com/thadhaines/`

Current Tasks:

1. Continue refactor and incorporation of AMQP into simulation process.

   Flow chart various processes to more clearly define where, and how, communication can be coded efficiently.
2. Experiment with ODE solver in numpy / scipy `odeint` in Python 3

   Make a tgov1 for LTD simulation
3. Define Agent actions for AGC/LFC (ACE calculations), Shunt Control, β calculation
4. Work towards experimenting with a multi-area model:



5. Investigate line current data (add branch section agents to model)
6. Refine data output - keep quickplotter in mind - Dictionary structure, variable naming, functionality, meta... May be less of an issue since plotting can take place from python3.

Future Tasks: (Little to No Progress since last time / Things coming down the pipe)

1. Add Ramp perturbance Agent
2. An agent for every object: Shunt, SVD, Branch, Transformer, Power Plant, ...
3. Think about having all simulation parameters be defined in one file that is fed into simulation. i.e. Have all perturbance, AGC, LTD, etc., related code in a text file so python doesn't have to be a 'user' thing.
4. Investigate Runge-Kutta integration. (probably inside scipy...)
5. Enable multiple dyd files to overwrite / replace previously defined agents/parameters
6. Identify System Slack bus programmatically (maybe just assume in first area?)

   or calculate system slack error differently... An average of slack error?

Current Questions:

1. Overview of planned PSLF scenarios? $\rightarrow$ Similar to Heredia but on Wecc/MiniWecc Scale?
2. Is there more available/relevant event data that may help us to verify simulations of specific instances (wind ramps or other behavior) that the novel research will focus on?
   (Heredia paper data helpful for some wind ramp data context)
3. Any progress on Wecc single gen per bus system, and/or miniWecc Area definitions?