**nScope** : a python based scientific work bench with MDSplus data visualization tool

S. Shiraiwa, PSFC, MIT

**Motivation**

Computer modeling of experiment is critical component of data analysis on C-Mod. It is done by using many tools (os, languages, platforms) for:
- Collecting and preprocessing data
- Running simulation codes
- Visualizing results

Problems of this style of data analysis include:
- Error-prone and fragile work flow
- Poor recording capability (error-prone and fragile work flow)
- Unnecessarily time-consuming
- Difficult to transfer experience to others

**nScope Basics**

**Work environment**:
- Project tree browser
- Tree data viewer
- Python shell and shell variable viewer
- Multi-panel test editor with syntax highlight
- Graphics window (Figure window)
- Debugger
- Access to Mercurial repository

**Tree objects**

In nScope, a user works on a tree object called a "project". A "project" is place to hold experimental data, data processing scripts, and other information.
- A project has a tree structure, allowing a user to organize his/her work
- A tree data viewer to access project data intuitively
- A tree is built automatically upon user’s action, such as viewing MDSplus data, so that a user can access data later.
- However, a user can also build a tree manually.

Example: Above surface plot is shown as a child object of event1 in a tree browser. Objects' internal data (tree data) are accessible in a tree variable viewer.

**nScope data browser in nScope**

In nScope, scope is implemented as a subclass of figure window, providing many desirable features:
- Overplot of multiple discharges, multiple signals in single/multipage
- Generating input files
- Support various plot: line, errorbar, image, vector field, triangular plot
- Supports various image output formats
- Fully undo/redo-able annotation
- Scriptable from python in no-window mode
- Object-oriented design to expand and embed it in your data analysis
- Multi-threaded access to hide network latency

**Multiple ways of overplotting**

**In-panel data analysis both by TDI + Python**

Data read in panels via TDI
- Unlimited number of variables can be read in a panel
- Server-side data processing is possible (TDI)
- Further data processing on client (python)
- This example will write flatness length to title.

**Direct publication quality graphics generation**

Edit graphics properties
- Colors, line style, font size
- Legend
- embed EPS picture
- and more

Edit can be done by the GUI, but also is scriptable.

Graphics remembers data, therefore, can extract original data anytime later.

**Support various plots**

- Multiple signals, discharges
- Same data from different trees
- Multipage plot
- 3D surface

**Data access and computer modeling in nScope**

nScope has various tree objects implemented to construct a simulation workflow.

**Object oriented design to easy expand and build a custom app.**

**nScope** has various tree objects implemented to construct a simulation workflow.

- Model. provides a place to define each work flow.
- Script. is a python script, but it already knows how to work with the project trees.
- Data. allows for accessing variables in formats (.nc, .mat, .sav, .namelist, .grf, .aff, ...).
- Solver. defines how to solve a model and how to store solutions in tree.
- A parametric solver and an optimizer are also built-in.

"Parameter" defines common variables.

Physics model can be linked to construct a complicated simulation workflow (this example couples models for three physics codes and multiple diagnostic data).

In addition, nScope is:
- Written in 100% pure python, developed on wxPython widgets and matplotlib.
- Under GNUv3 license.
- Wiki (under construction) : http://nscope.psfc.mit.edu/

This work supported by USDoE awards DE-FG02-99ER45512 and DE-AC02-08CH11466.