Three little components

Don Batchelor
SWIM Project Meeting
September 23-25, 2008

- minimal_state_init
- generic_driver
- monitor

See our fun website at: www.cswim.org

minimal_state_init component

1) Generates a complete set of (almost) empty Plasma State files
2) Extracts TOKAMAK_ID, RUN_ID, SHOT_NUMBER from config file and loads into equivalent state variables
3) Sets ps%t0 = ps%t1 = start time of simulation t_0

1) **It is complete, and is probably adequate, but could easily be adapted**
2) **There is no real physics code – consists of a python component script and a fortran executable to interact with the Plasma State**
3) **It runs in the framework and has been tested by initializing "model simulations" as well as the C-Mod minority (maybe others)?**
4) **Could be used for as is regression testing? Only needs config files for input.**
5) **Binaries are generated by Makefile in component directory of svn. Make install moves them to ips/bin directory**
6) **Requires no attention from the user (unless he wants to initialize differently) ?**
7) **Simulations planned using the component? – probably all of them**
8) **Are there extensions to the components planned? None planned now. But might fix it to go out and bring in existing Plasma State files for initialization.**
9) **How many distinct versions of the component are there? As of now there is 1**

monitor component

1) **Collects selected data from the plasma state each time step and writes it in a netCDF file with time as an unlimited dimension – generates time series, plasma states are time slices**

2) **What it collects as functions of time is:**

Thermal species data: $n_e(\rho)$, Te(0), Ti(0), ne(ρ), Te(ρ), Ti (ρ)

Direct thermal species heating by ICRF: Pe_icrf (ρ), Pi_icrf (ρ), Pe_icrf_total,Pi_icrf_total

Collisional heating of thermal species by minorities: Pmin_e(ρ), Pmin_i(ρ)

Parallel and perpendicular energy density of ICRF minority ions: Eperp_mine(ρ), Eperp_mini(ρ), Eperp_mini(ρ), Eperp_mini(ρ)

3) **The framework picks up the monitor.nc file and moves it to the W3 directory so the portal can display the results with ELVis as the simulation proceeds**

monitor component

1) **It works but we want it to collect a lot more variables**
2) **There is no real physics code – consists of a python component script and a fortran executable to interact with the Plasma State**
3) **It runs in the framework and has been tested by monitoring "model simulations" as well as the C-Mod minority**
4) **Could be used for as is regression testing? Only needs an ELVis template file for input.**
5) **Binaries are generated by Makefile in component directory of svn. Make install moves them to ips/bin directory**
6) **Requires no attention from the user (unless he wants to initialize differently) ?**
7) **Simulations planned using the component? – probably all of them**
8) **Are there extensions to the components planned? The whole thing should be rewritten. It's tedious and error prone to write netCDF code every time you want to add something. It should be made configurable by the user.**
9) **How many distinct versions of the component are there? There are several of them. Every time I add some variables in increment the suffix so it won't break existing components – we are at monitor_comp_3 now**

generic_driver component

1) Runs a simple simulation involving – EPA, RF, FP and MONITOR components
2) Calls INIT function of each component
3) Cycles through time loop: calls pre_step_logic internal python method, calls STEP function for each component in reverse order from listing above, stores output in /simulation_results

1) **It is complete, and is a good starting point for any more complicated simulation**
2) **Python component script only, no fortran**
3) **It runs in the framework and has been tested by running "model simulations" as well as the C-Mod minority**
4) **Could be used for as is regression testing? Only needs config files for input.**
5) **No binaries to worry about**
6) **Requires no attention from the user (unless he wants to execute differently) ?**
7) **Simulations planned using the component? – a lot of simulations can use it as is**
8) **Are there extensions to the components planned? None planned now.**
9) **How many distinct versions of the component are there? As of now there is 1**