Modernizing the Scientific Software Approach for the Fusion Analysis Code TRANSP

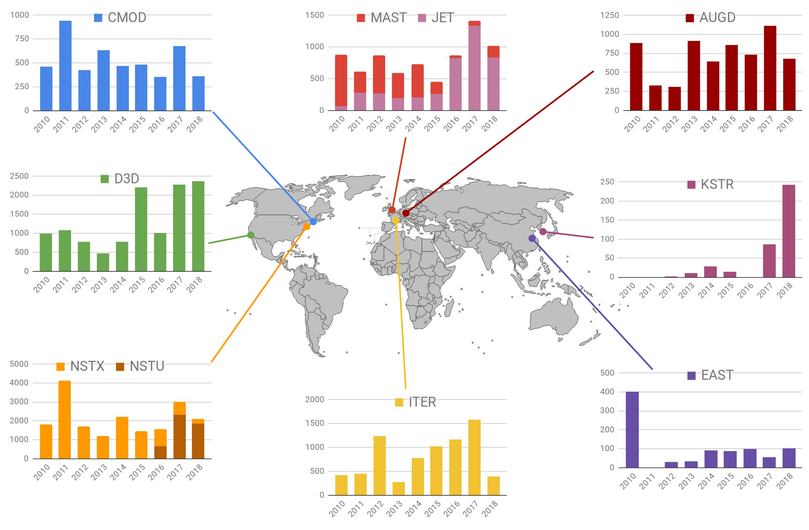
J.S. Sachdev, M. Gorelenkova, X. Yuan, J. Breslau, and F. Poli
Princeton Plasma Physics Laboratory, Princeton University Princeton, NJ, 08543
TRANSP DOI - 10.11578/dc.20180627.4

SIAM CSE19 Minisymposterium:
Software Productivity and Sustainability for CSE and Data Science
Poster DOI - 10.6084/m9.figshare.7758305

TRANSP is a valued code in the fusion community...

- TRANSP is a time-dependent 2D MHD equilibrium and 1D plasma transport solver for modeling tokamak fusion plasma discharges
- Used by hundreds of physicists and engineers at several research centers world-wide
- The physics capabilities of TRANSP has grown significantly over 30+ years of development
- Originally used for interpretation of experimental results but now includes predictive modeling capabilities
- Advanced usage:
  - Campaign planning and experiment design
  - BEAST mode - rapid between experimental shot analysis
  - TRANSP as a flight simulator for plasma control systems
  - Whole Device Model, including predict first
- TRANSP is under continuous development to provide users with new physics models and improved performance

TRANSP usage: over 62k simulations performed since 2010



TRANSP also used to model plasma discharges for other tokamaks including: ARIES, DEMO, FNSF, HI2A/HL2M, IGTR, JT60, KDMO, LTX, MST, RXFM, STEP, TCV, TFTR, WRK

…but the software approach needs a reboot

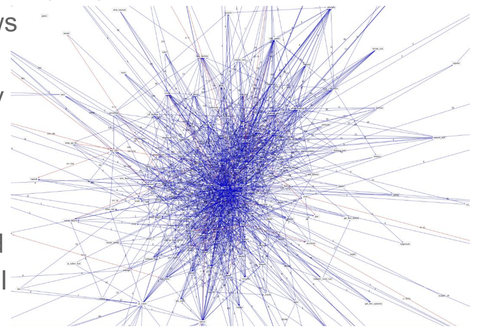
- Lack of attention to best software practices has lagged the advancements in physics models
- Result is a non-modular code making the implementation of new physics models challenging, particularly models that require extensive HPC capabilities
- 30+ years of development has led to significant repo clutter
  - Not used and unusable sections of code
  - Deprecated physics models not removed from repo or docs
- Homegrown build system based on csh and gmake
  - Very powerful but difficult to port to other computing facilities
- Homegrown regression testing suite instead of tools widely used and accepted by the community
- Versioning of subsections of code but not entire code
- Development done directly in trunk by multiple developers

Our mission

Update the software approach and restructure TRANSP to improve code flexibility while maintaining ability to couple physics modules that involve a wide range of spatial and temporal scales

TRANSP dependency graph

- Repo organized in sub-directories with each compiled as its own library (278) or test programs (259)
- Dependency graph † shows significant directory interlinking including many reciprocal links
- Hairball graph useful for quickly finding code not linked with executable and are candidates for removal
- Attempts to reorganize graph shown some parts of the repo are more modular than expected (e.g., MHD equilibrium tools)
- Directory reorganization will lead to a more levelized graph



† Architecture dependency graph generated using the Understand software from SciTools

Modernizing our scientific software approach

- Must continue physics model development; performing modernization with a step-by-step careful approach
- Adding regression tests to ensure no code breakage
- Objective is to restructure the code to result in a hierarchical dependency graph and improve modularity and portability

Code repository and versioning

- Moved from SVN to a private GitHub repository ✔
- Will open source TRANSP code after code restructuring is complete (as per DOE requirements)
- Development team adopting proper branch management with development in branches and pull requests for merges ✔
- Quarterly releases with version numbers and release notes for an update-to-date public release ✔

Code restructuring and clean-up

- Removed multiple deprecated physics models and not used sections of code ✔ (but ongoing)
- Over 100k lines of unneeded code removed from repo so far
- Restructuring code based on physics and streamlining the interface between the modules and core TRANSP code
- Remove 3rd party code from repo, pull latest version link to libs

Productivity

- Adopt a modern continuous integration approach for builds and regression testing (likely Jenkins)
- Eventually rework the build system

Portability

- Containerization of TRANSP for quick deployment to other computing facilities including cloud-based HPC ✔
- Using a recipe-based Singularity container (www.sylabs.io) ✔

Acknowledgement
The modernization of TRANSP was motivated by necessity; however, much of the strategies employed and plans made were inspired by a similar effort being performed by the xRage group at LANL. Their work was presented by C. Ferenbaugh during the IDEAS-ECP webinar on “Bringing Best Practices to a Long-Lived Production Code” presented 2018/01/17.

PPPL PRINCETON PLASMA PHYSICS LABORATORY
PRINCETON UNIVERSITY
U.S. DEPARTMENT OF ENERGY | Office of Science

* *Work supported by US DOE Contract No. DE-AC02-09CH11466*

TRANSP PPPL