

Center for Simulation of Wave Interactions with MHD

March 3, 2006
DOE – Germantown

D. B. Batchelor, L. A. Berry, S. P. Hirshman, W. A. Houlberg, E. F. Jaeger, R. Sanchez – ORNL Fusion Energy

D. E. Bernholdt, E. D’Azevedo, W. Elwasif, S. Klasky – ORNL Computer Science and Mathematics

S. C. Jardin, G-Y Fu, D. McCune, J. Chen, L. P Ku, M. Chance, J. Breslau – PPPL

R. Bramley – Indiana University, D. Keyes – Columbia University, D. P. Schissel – General Atomics,

R. W. Harvey – CompX, D. Schnack – SAIC, J. Ramos, P. T. Bonoli – MIT

Unfunded participants:

L. Sugiyama – MIT, C. C. Hegna – University of Wisconsin, H. Strauss – New York University, P. Collela – LBNL

H. St. John – General Atomics, G. Bateman, A. Kritz – Lehigh Univ.

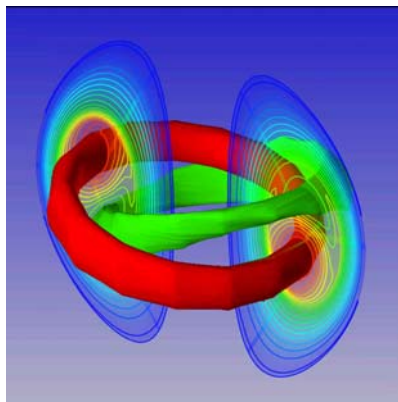
- **Program logic, progress, plans – Batchelor**
- **Design approach for SWIM computational Framework – Bramley**
- **Physics targets, research issues – Jardin**



SWIM brings together two mature sub-disciplines of fusion plasma physics, each with a demonstrated code base using the most advanced computers

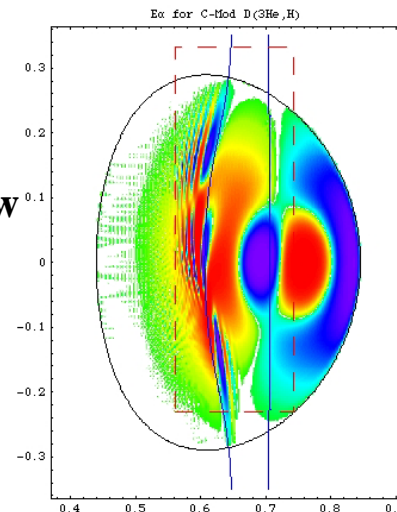
Extended MHD – CEMM

- MHD equilibrium
- Macroscopic fluid instability
- Current and magnetic field evolution



High power wave-plasma interactions – CSWPI

- Plasma heating
- Externally driven current or plasma flow
- Non-Maxwellian particle distributions



Fluid equations, extended to include non-ideal and kinetic effects
(10^{-5} sec $<$ τ_{MHD} $<$ 10^{-1} sec)

Plasma wave equation ($\tau_{\text{RF}} < 10^{-7}$ sec), coupled to slow evolution of plasma velocity distribution ($\tau_{\text{FP}} > 10^{-2}$ sec)

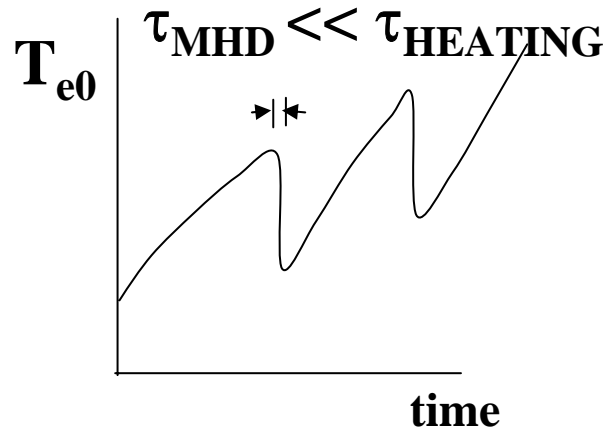
Why couple these particular two disciplines?

- Macroscopic instabilities can limit plasma performance
- RF waves can mitigate and control instabilities

SWIM has two sets of physics goals distinguished by the time scale of unstable MHD motion

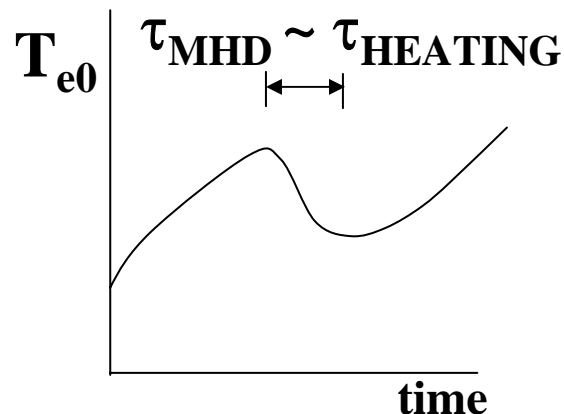
Fast MHD phenomena – separation of time scales

- Response of plasma to RF much slower than fast MHD motion
- RF drives slow plasma evolution, sets initial conditions for fast MHD event
- Example: sawtooth crash



Slow MHD phenomena – no separation of time scales

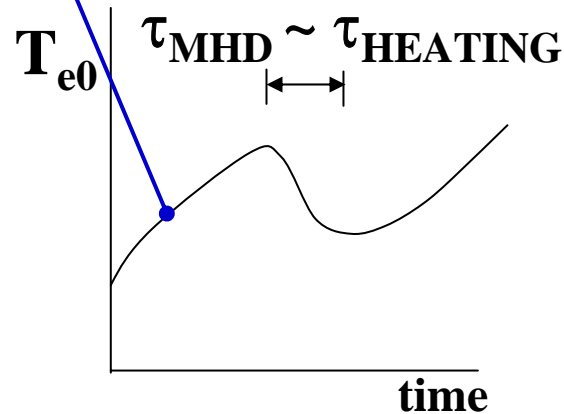
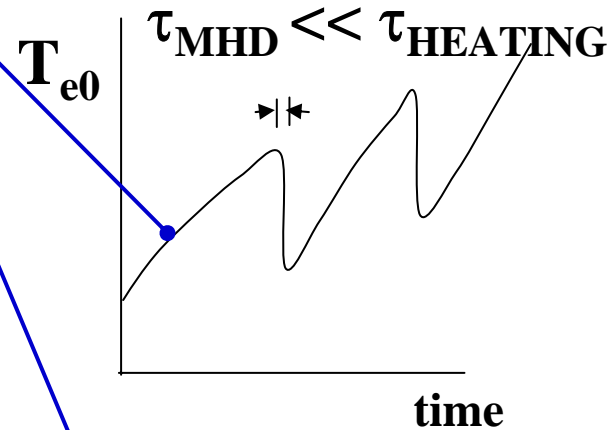
- RF affects dynamics of MHD events \Leftrightarrow MHD modifications affect RF drive plasma evolution
- Deals with multi-scale issue of parallel kinetic closure including RF – a new, cutting edge field of research
- Example: Neoclassical Tearing Mode



We are approaching these regimes in two *campaigns* of architecture development and physics analysis and validation

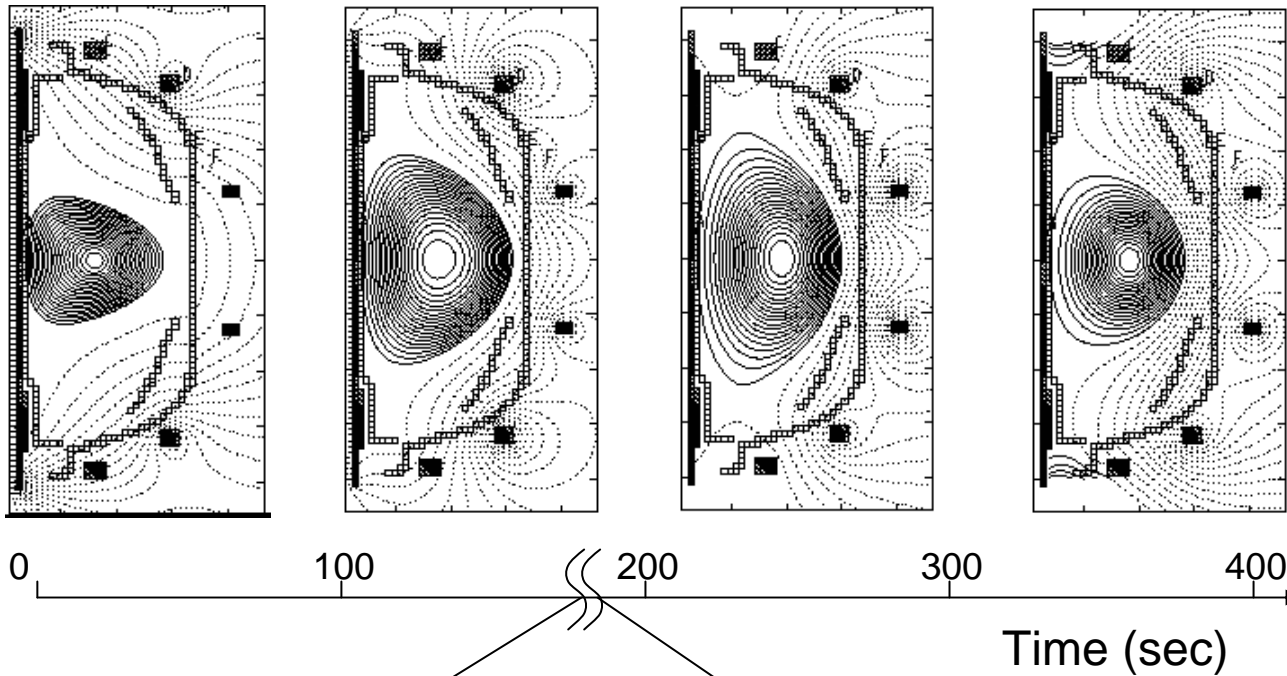
Simulation of plasma evolution requires complete model – Integrated Plasma Simulator (IPS)

- Heating and current drive sources
- Particle sources
- Transport
- Magnetic field evolution



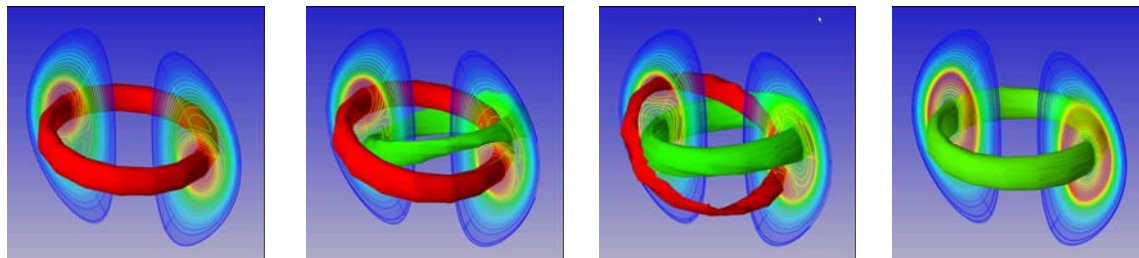
Integrated Plasma Simulator will allow coupling of virtually any fusion fusion code, not just RF and MHD, and should provide the framework for a full fusion simulation

Simulation of plasma evolution requires complete model – Integrated Plasma Simulator (IPS)



- Plasma evolves through a series of 2D axisymmetric equilibrium states
 - Heating and current drive sources
 - Particle sources
 - Transport
 - Magnetic field evolution
- Instabilities occur as instantaneous events

180.0001 180.0002 180.0003 Time



- 3D Extended MHD simulation starts and ends in axisymmetric state

Accomplishments to date – meetings, designs, etc

Principal investigators + PTRANSP project leaders meeting (August 2005 – PPPL)

Preliminary assessment of computational framework requirements, component and codes

⇒ Adopted an inter-component communication approach using a common **Plasma State** component

General organizational workshop on integrated plasma simulation (November 2005 – ORNL)

Open meeting – many attendees not directly in the SWIM project

Detailed analysis of components – component definition documents

- High-level mathematical definition of which equations must be solved by any component instance;
- Specifications of the data objects shared among different component categories, at the physics level;
- Example workflows defined which will initiate the fast MHD campaign.

Physics research needs identified and prioritized

⇒ **Component Definition Documents** (available at web site)

Principal investigators + key component developers IPS initial design meeting (Jan. 2006 – PPPL)

Fast start – initial loose component coupling, file based communication

Wrapper script design – minimal impact on physics code developers

Common look for Plasma State component

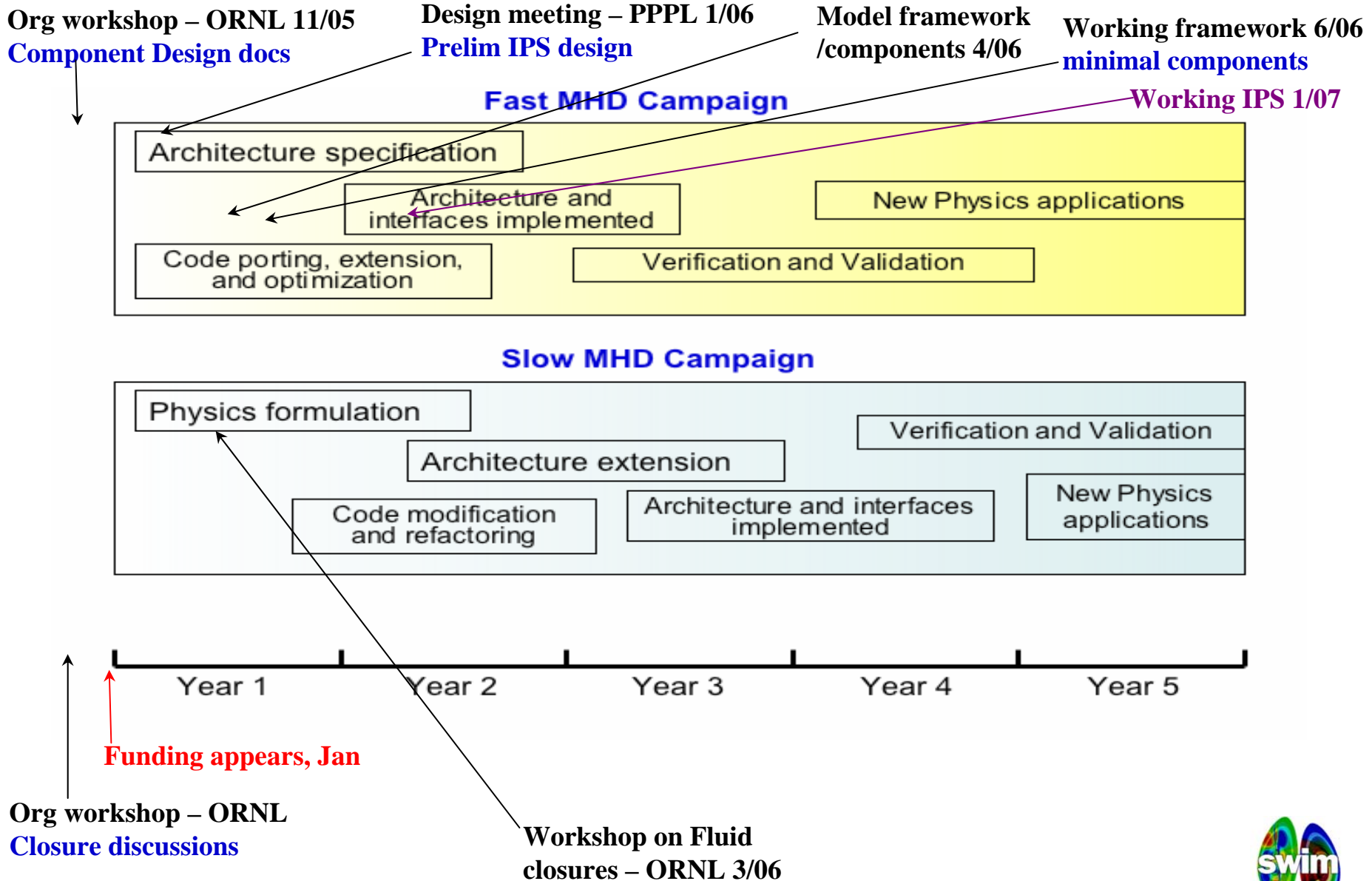
⇒ **Initial design report** (on web site, Bramley's presentation coming up)

Workshop on fluid closures (March 2006 – ORNL)

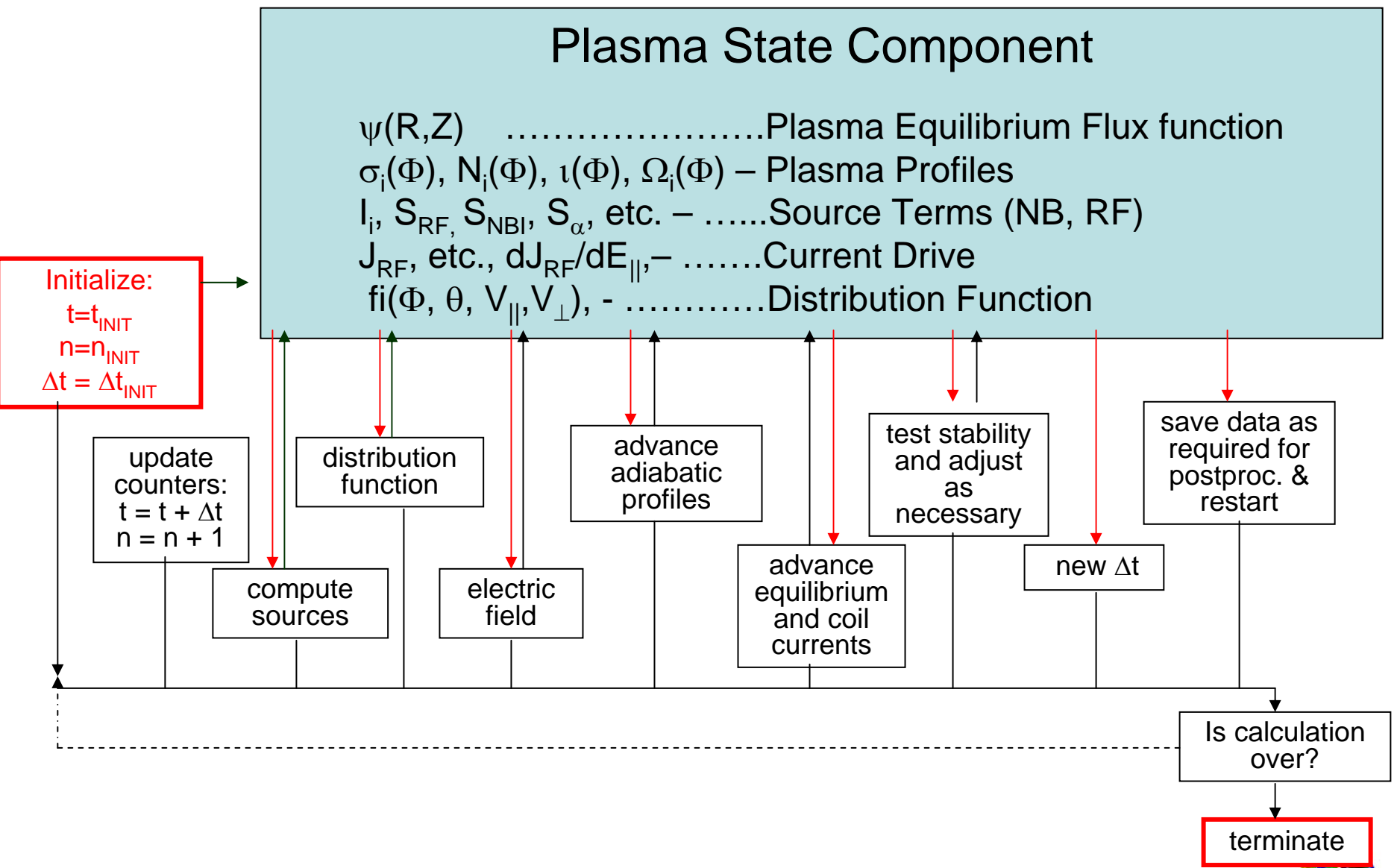
Open meeting – plasma theorists, non-fusion attendees



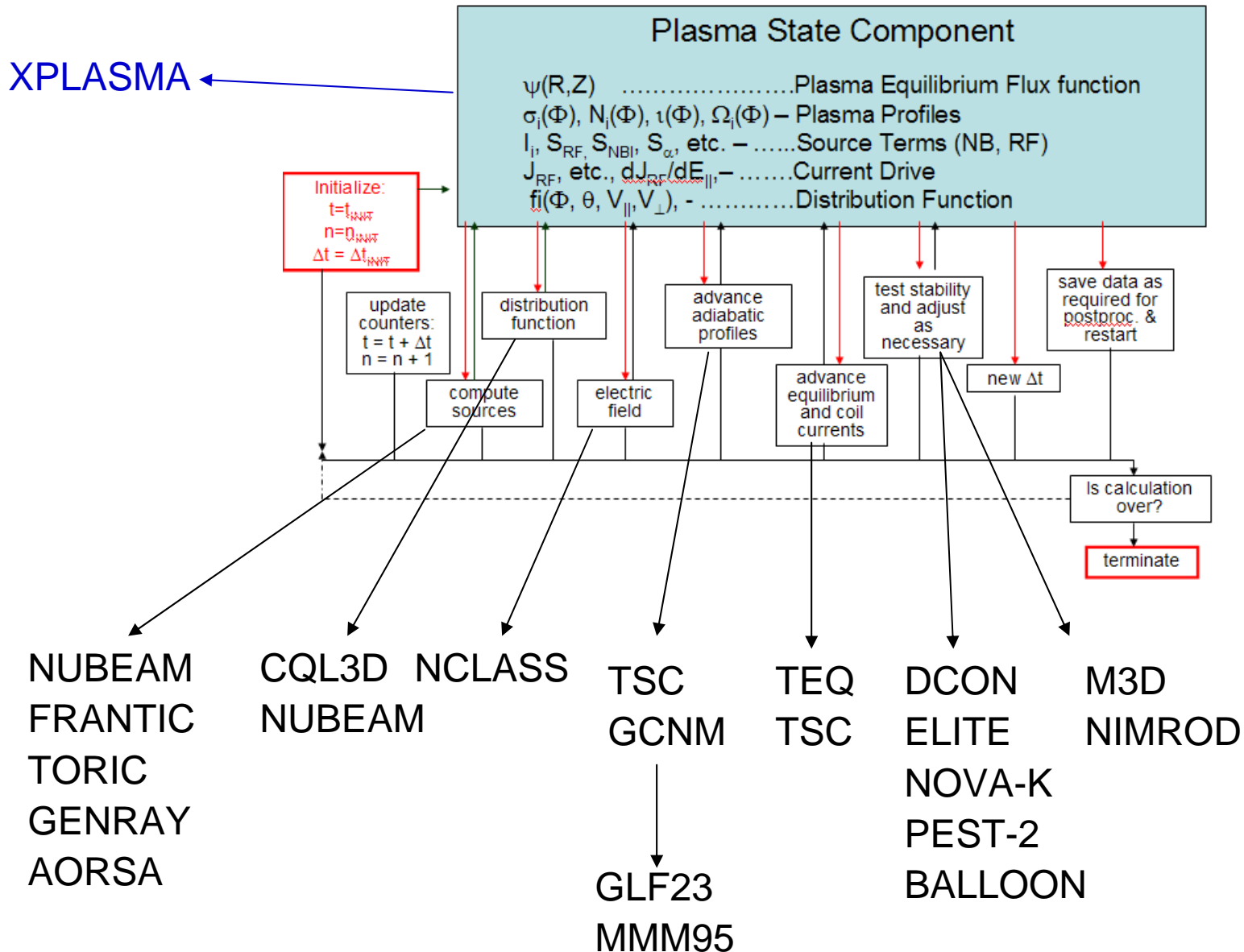
SWIM Work-plan and Timeline



Integrated Plasma Simulator design – Plasma State component plays a central role



Components are being implemented using existing fusion codes



Our objective is that the community adopt the SWIM architecture as the backbone for computational fusion research

- **The SWIM project is open – it is designed to be easy and beneficial for non-team-members to include their own models in the SWIM platform**
- **We engaged the community at an early stage**
 - **Defining component functionality and interface standards**
 - **Inclusion of additional codes (e.g. TORIC, NIMROD, NCMN) – we budgeted to support his kind of activity**
 - **Design documents posted on the web**
- **First project meeting (Nov. 2005, ORNL) attracted many participants from outside the team – both fusion and CS/math**
- **Organizing a community-wide workshop to address fluid closure issues of importance to CEMM and SWIM (March, 2006, ORNL)**
- **Engaged with international effort – possible joint meeting at ITPA Confinement Data Base and Modeling**

We are coordinating with new SciDAC projects in areas where they are complementary to SWIM

Framework and CS Design Principles

1 Staged, evolutionary approach

- Interfaces are still evolving
 - But major advances made at meetings in November and January
- No “SWIM branch” versions of constituent codes (instead, SWIM adapters added to the standard development path)
- Keep modularization as long and as far as possible (sufficient onto the component the evil thereof)

2 Beg, borrow, steal, **reuse** rather than rebuild ...

3 Target highest end computers from the start

Framework and CS Design Principles: Reuse

- Recycle user interaction system from LEAD weather prediction project

Workflow MyLead Component Monitor Hel

Add Node Remove Node Connect/Disconnect

Composer

Port Information Notification

Time	Component	Status	Message
07/19/05 14:20:45.760	Workflow	Started	
07/19/05 14:20:46.937	ADaM_FeatureExtraction	Invoking	
07/19/05 14:20:47.786	ADaM_FeatureExtraction	Started	Host: 146.229.234.73
07/19/05 14:20:48.277	ADaM_FeatureExtraction	INFO	install location of the lead tools is /home/grid5070/production/adam-services
07/19/05 14:20:48.589	ADaM_FeatureExtraction	INFO	the local temporary directory is /tmp
07/19/05 14:20:49.72	ADaM_FeatureExtraction	INFO	the input url of the data file is gridftp://frozone.itsc.uah.edu/home/grid5070/production/adam-service...
07/19/05 14:20:49.331	ADaM_FeatureExtraction	INFO	the output url to push the output files is gridftp://frozone.itsc.uah.edu/tmp
07/19/05 14:20:49.441	ADaM_FeatureExtraction	INFO	the site name is KOUN
07/19/05 14:20:49.685	ADaM_FeatureExtraction	INFO	the local work directory is /tmp/nexrad_1121800848241/3DMesocycloneDetection
07/19/05 14:20:50.118	ADaM_FeatureExtraction	INFO	attempting to copy input data file to /tmp/nexrad_1121800848241/3DMesocycloneDetection/input.data
07/19/05 14:20:52.288	ADaM_FeatureExtraction	Received a file	From gridftp://frozone.itsc.uah.edu/home/grid5070/production/adam-services/FeatureExtraction/inpu...
07/19/05 14:20:52.470	ADaM_FeatureExtraction	DEBUG	attempting to run the detection
07/19/05 14:20:53.972	ADaM_FeatureExtraction	Consumed a file	gridftp://frozone.itsc.uah.edu/home/grid5070/production/adam-services/FeatureExtraction/input.dat
07/19/05 14:20:54.965	ADaM_FeatureExtraction	Finished computation	0 sec
07/19/05 14:20:55.84	ADaM_FeatureExtraction	INFO	attempting to push output files to gridftp://frozone.itsc.uah.edu/tmp



Framework and CS Design Principles: Reuse

- Recycle data management methods from CIMA

CIMA X-ray Crystallography

Home IUMSC ChemMatCARS Purdue Minnesota Myers Hall Request an Account

Home News About Contact

IUMSC

ChemMat-CARS
Univ. of Chicago at APS

Purdue
Crystallography Center

IUB Myers Hall

X-ray Cryst. Lab
Univ. of Minnesota

Other Collaborating
Laboratories

CSAF
University of Sydney, AU

NCS
Southampton, UK

Other CIMA
Application Portals

Common Instrument Middleware Architecture

This portal demonstrates some of the functionality that will be available to the crystallographers collecting data in participating laboratories (left hand side menu). Researcher can follow the progress of an experiment using a simple web browser connected to the Internet. The IUMSC instrument is being used as a test system for the Instrument Middleware project, see <http://www.instrumentmiddleware.org>

The work being described on these pages are supported by NSF Cooperative Agreement OCI-0330566 and MRI CDA-0116050

Login

User Name

Password

Remember my login

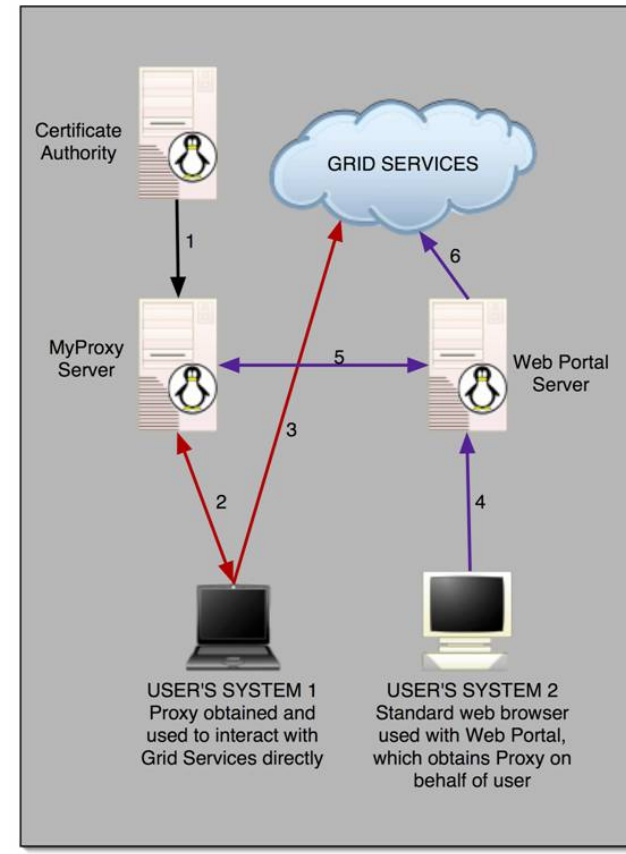
Login

[Forget your password?](#)

powered by gridsphere

Framework and CS Design Principles: Reuse

- **Recycle authentication from myProxy** (authorization *technology* is not a problem – developing a community standard is the main difficulty)



Framework and CS Design Principles: High end systems

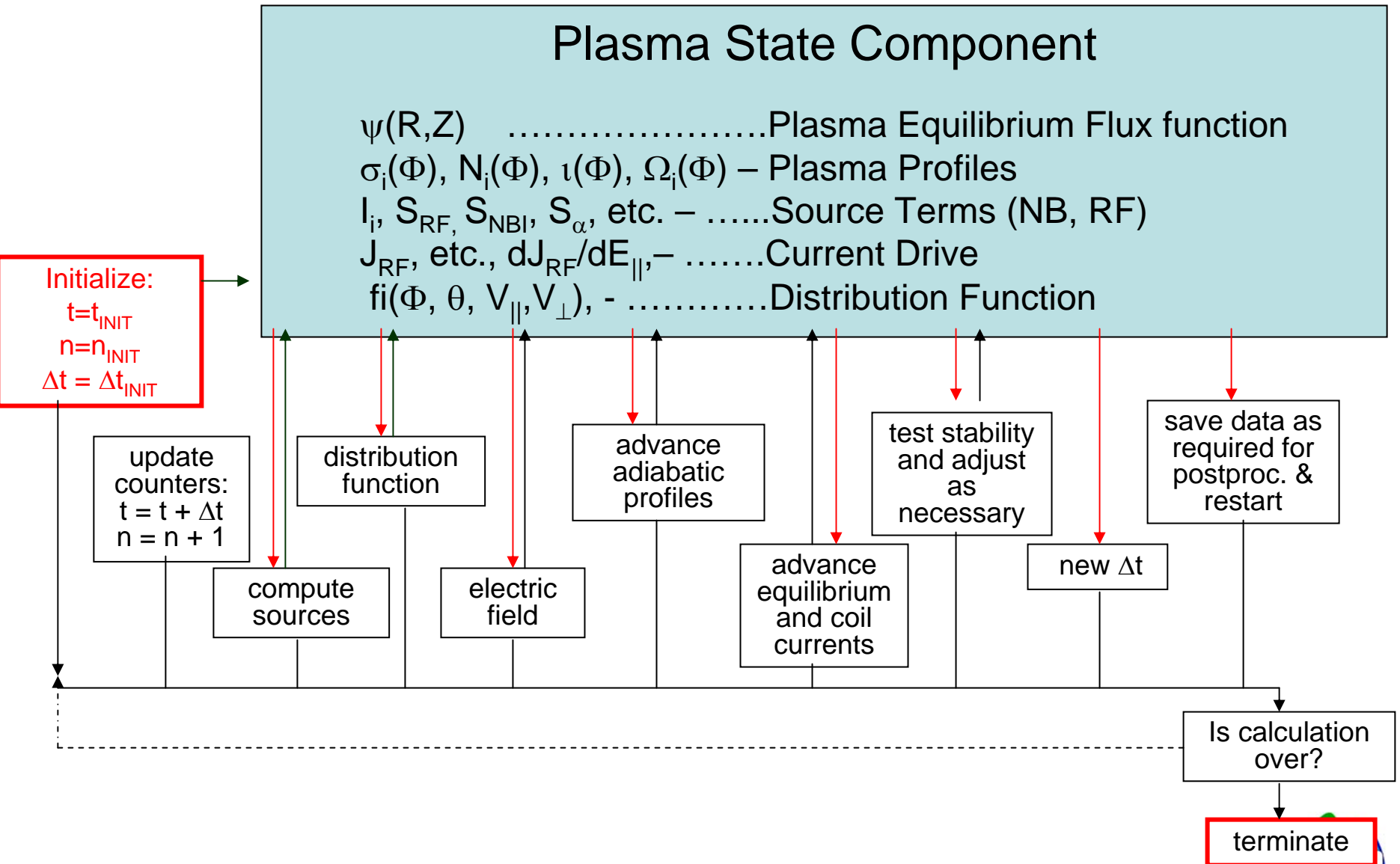
- **Highest end systems must be handled from the start, not as an afterthought**
 - Some **site authentication** policies limit what can be automated on ultrascale systems
 - **Reliability**: insulate individual component executions on leadership-class machines from failures in
 - SWIM framework
 - Other SWIM apps components running as part of the job
 - WAN failures or slowdowns
 - Limits **languages** that can be assumed for scripting, managing SWIM
 - Limits systems capabilities:
 - No **sockets** from compute nodes
 - **Ports** limited to/from head nodes
 - Abilities to create **new processes or threads**
 - **Batch system interaction** for multiple jobs

Have solutions to these, but prohibits some cool CS tools

Framework and CS Stages: I

- Define components on most abstract/minimalist level (**done**)
 - **Plasma state** is central object a SWIM run develops
- Define interface interactions needed (**done ...**)

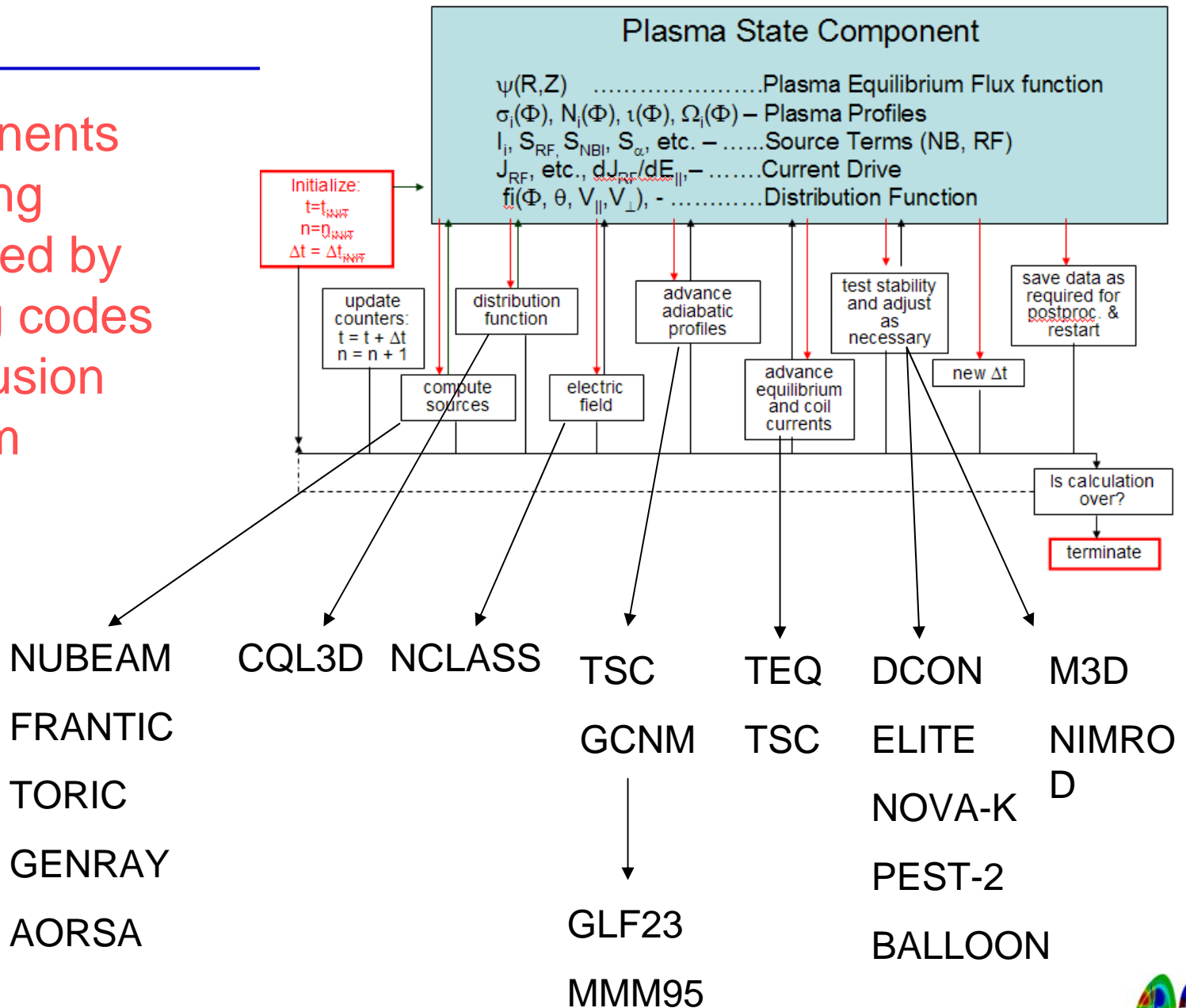
Basic Time Loop and Central Role of Plasma State:



Framework and CS Stages: I, ctnd

- **Define codes to be used as components and roles they play (done ...)**

Components are being populated by existing codes in US fusion program



Framework and CS Stages: I, ctnd (being done now)

- **Port codes to shared, controlled platform**
 - Not a violation of the “highest end” principle; mix of compute requirements in any multicomponent SWIM run
 - Each app component needs **configure, build, run system** for porting
 - Each app component needs **sample data** sets
 - Testing of installation data sets
 - Default settings for use in SWIM
 - Separate out a hierarchy of input needed by each app component; data
 - Specific to a **tokamak**
 - Specific to a **shot**
 - Specific to a **simulation run**
 - Specific only “internally” or **locally** to a component
 - Decomposition identifies what is common across all components in a SWIM run versus what can be changed in a given component

Framework and CS Stages: II

- **Start with a strictly **file-mediated exchange** of data between application components**
 - **SWIM distinction of fast/slow MHD allows initial campaign with lower inter-component data exchange performance requirements**
 - **Allows **isolating of any bugs** or problems, and allows us to pack the complete set of problem-causing inputs to a component manager**
 - **No changes needed to the application component code; SWIM overall interactions handled by scripting**
 - **Eliminates issues of re-entrancy and overall system state consistency**

Framework and CS Stages: II, ctnd

- Use run scripts to launch, monitor, manage each **app component**
 - Only four required functions: *Initialize, Step, Finalize*
 - Notifies framework constantly about
 - Queue submission
 - Queue status
 - Run completion status
 - Names/locations of output results
- Overall **simulation control script** to coordinate the app components and manage files used for data exchange, results
 - Until decomposition of global versus local input data and parameters is done, relies on end user assuring consistency
 - Once that decomposition is done, the overall script supplies global data to each component which then
 - Can run a local code to convert to native input format, or
 - make modifications in code to directly accept them

SWIM sample execution model

Simulation Control File(s)

Select specific component codes (e.g. TORIC vs AORSA)

Code-specific initialization data

Fusion machine specific data

Simulation control data:

Computer environment settings

Plasma initial conditions

Plasma source and control, time sequences (events)

Simulation Controller

Computer environment initialization

Initialize plasma state

Initialize component

Advance Sequence: [t → t+Δt]

Step RF

Step Particle Source

Step Fokker Planck Solver

Solve $j_{||}(E)$

Step Profiles

Step Magnetics & Equilibrium

Test linear stability

Apply Reduced MHD if unstable

Evaluate Simulation Control

finalize, take another *step*, or adjust and restart current step

Typical component

RF Component

RF_step:

Get_plasma_state

Convert plasma state to code-specific form and write code-specific initialization data

Launch Implementing code: (i.e. AORSA, TORIC, or GENRAY)

Convert code specific output to plasma_state form

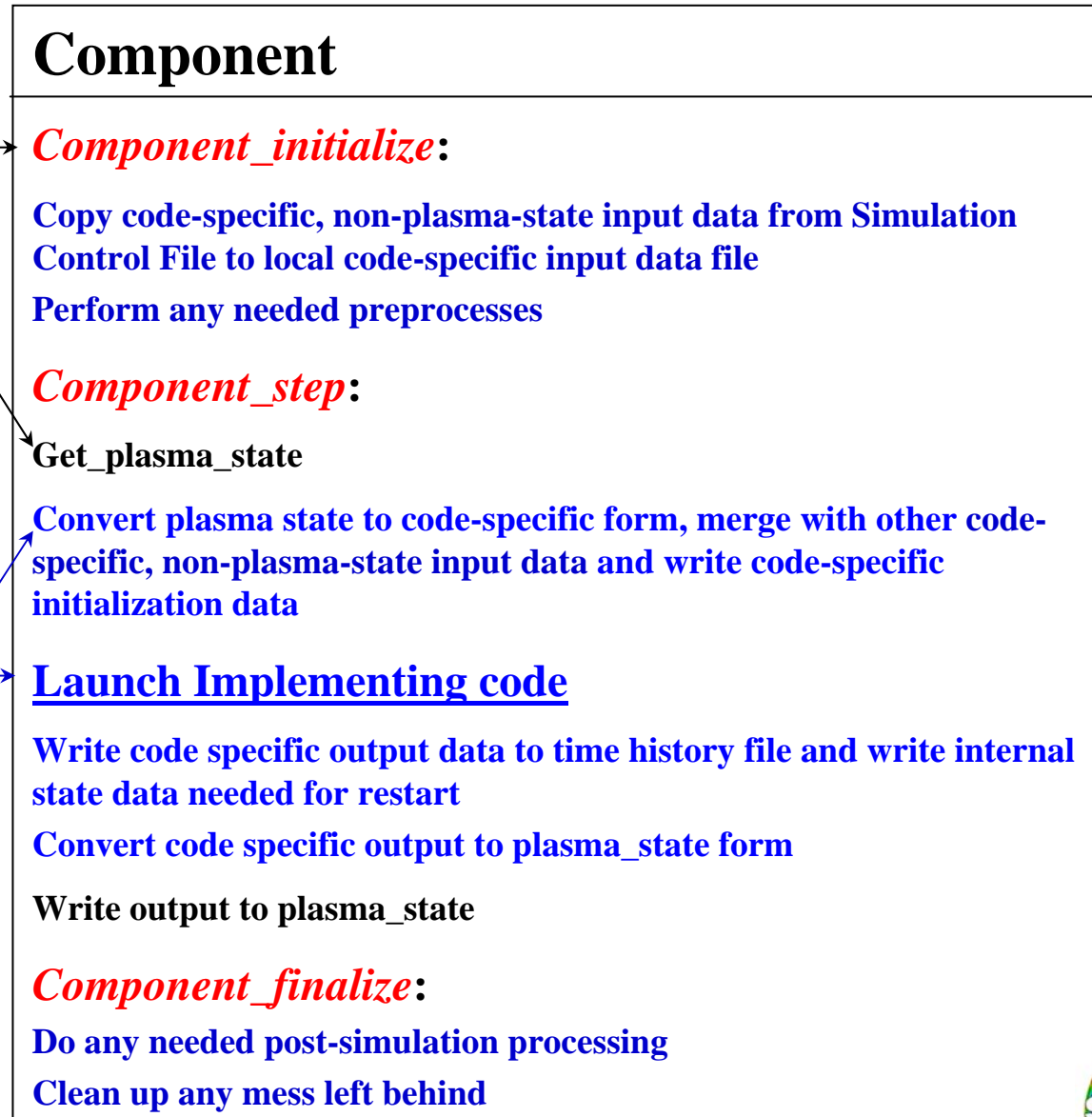
Save internal state of component

Write output to plasma_state

Typical component interfaces and work flow

Things in black are generic, provided at component level

Things in blue are code specific. Require action by code developer



Physics Program Principles

- **Use existing physics codes as much as possible (reuse)**
- **Concentrate on a small number of physics problems**
 - **Are important for ITER**
 - **We can make a unique contribution**

Physics Goals

- **Routine (IPS) simulation of Entire Discharge**
 - with access to “best available” component models
- **Sawtooth Instability**
 - better understanding, including role of energetic particles
 - RF stabilization/destabilization predictive tools
 - improved MHD event model (e.g. Porcelli/Rosenbluth) for IPS
- **Neoclassical Tearing Mode Stabilization**
 - better understanding (including interaction with sawtooth)
 - RF stabilization predictive tools
 - develop improved IPS models

Critical physics issues have to do with calculating distribution function and closures

- **Same Fokker-Plank code should couple to full-wave RF code and have neoclassical transport of fast ions**
 - **Modification of CQL3D**
- **Need to learn how to transfer distribution function data from Fokker-Plank code to stability codes**
 - **Both particle and continuum representation**
- **Need to develop generalized neoclassical theory for a geometry with islands in the presence of intense RF**
 - **Fluid Closures Workshop May 22-24 at ORNL**

Summary

- **Our objective is that the community adopt the SWIM architecture as the backbone for computational fusion research**
- **The component-oriented architecture, with a flexible control level, will allow coupling of virtually any fusion code, not just RF and MHD, and should provide the framework for a full fusion simulation**
- **Addressing ultra-scale computing from the start**
- **Three physics goals**
 - **Unique Integrated Plasma Simulator for comprehensive long-time simulation of tokamak discharges**
 - **Sawtooth instability stabilization and destabilization**
 - **Neoclassical tearing mode stabilization**