Progress and Plans for Physics Integration/Frameworks

J.R. Cary
For the FSP "Frameworks Team"
March 25, 2010

Covering

- Physics composition
- Task composition (workflow)
- Development processes
- Production computing

Outline

- Mission
- Activity breakdown
- Accomplished
  - Outreach
    - Stakeholders
    - Questionnaire
    - 3-projects meeting
    - Planning meeting
  - Initiation of framework effort design
- Next steps

Project broken down into (1)

- **Physics composition** (enabling "the big run") is getting to the multi-physics, computational, dynamics application
  - Often what is submitted to the job queue
  - Secondary computations (e.g., EFIT preparation, are part of the next item)
- **Task composition** (enabling the preparation for and analysis of "the big run") the generally acyclic set of tasks that take one from initial question to physics understanding
  - "Workflow composition tools": bash, python, VizTrails, Kepler
  - Task utilities: for viz, data analysis, …
  - But can include using multiple "big runs" for optimization and design

Project broken down into (2)

- Development processes
  - Version control
  - Build systems
  - Test systems
  - Package management
  - Package distribution
- Production computing
  - Simplified input file preparation
  - Automatic data analysis
  - Job monitoring
  - User support
  - A production facility?

Addressing the PAC Charge

Charge to the FSP Program Advisory Committee (FSP PAC)

[FSP PAC Meeting, March 25-26, PPPL]

(1) Science Drivers -- Regarding the current set of proposed science drivers and associated science development road-maps, please comment on: (a) their appropriateness for the FSP; (b) the priorities for addressing them; and (c) whether they adequately cover the key areas needed for progress in MFE.

(2) Community Engagement -- Has the FSP program definition team defined and begun implementation of an effective community outreach plan?

(3) FSP Mission -- Regarding the FSP mission, please respond to the following questions:
(a) Has the FSP mission been defined and articulated in a clear and compelling way?
(b) Is the defined program scope (i.e., what will and will not be included in the program) appropriate and well focused?
(c) Has the FSP been appropriately placed into the context of other MFE program elements and the relationship to them adequately defined?

Outreach
(Community Engagement)

- Stakeholders
- Questionnaire
- 3-projects meeting
- Report on proto-FSPs
- Planning meeting
- Initiation of 3-projects dialog

Outreach: developed presentation for Software Integration and Support

- Presentation
  - Integration leads to need for framework activities
  - Presented by Jeff Candy at General Atomics
- Planning presentations
  - Colorado
  - Texas
  - Wisconsin
- Have learned
  - General support for FSP
  - Some desires for alternative/competing approaches (GA, UMd)

List of stakeholders developed

- Started from science drivers in December
- Developed list of researchers
- Sent list out and asked for more names
- Both *providers* and *users*
- Need to update based on current science drivers

Turbulence on transport timescales	
Developers:	
Candy	candy@fusion.gat.com
Barnes	michael.barnes@physics.ox.ac.uk
Srinath Vadlamani	srinath@txcorp.com
Rognlien/Cohen (for edge version)	rcohen@llnl.gov
Users	
Glen Batemann	bateman@lehigh.edu
Chris Holland	chholland@ucsd.edu

Questionnaire developed as first step to understand status

http://www.mcs.anl.gov/research/projects/fsp/fsp_survey/index.php

- Users and providers in each of the areas of
  - Physics composition: Users = modelers, Providers = code developers
  - Task composition (workflow): Users = modelers, Providers = workflow developers
  - Development processes
  - (Production computing came later)
- Received 1300 answers in total
- Path forward = balance of what users do (practice) and how that might be improved (vision)
- Many inputs and lessons

3 projects meeting: familiarization among the proto-FSPs

- **Goal: hands on to running the software of the 3 proto-FSPs (CPES, FACETS, SWIM)**
- **3 days long, one for each proto-FSP**
- **Codes run by each of the others to understand what all provide**
- **Selected neutral party (Brian Van Straalen) to develop comparison**
  - **Physics motivation**
  - **Implementations**
  - **Infrastructure**
  - **applicability**

	CPES	FACETS	SWIM
shared address space MPMD	no	yes	yes, but not enabled by framework
dynamic load balancing (not demonstrated in any framework yet)	Yes. Actor can be terminated and restarted with new processor count	Yes, FcComponent can be deleted and re-intitialized with new MPI_Communicator	yes. on every step() call a component can use a different fraction of batch allocation
single binary incorporating all components	no	yes	no
Allows one to use multiple binaries for code coupling.	yes	yes for pre-processing and post-processing but not in simulation code itself	yes
Works with a mixture of MPI and non-mpi codes	yes	no, but a Component can have trivial MPI modifcation	yes

proto-FSP comparison matrix

	CPES	FACETS	SWIM
single batch queue job per simulation	no. a batch submission done per Actor. Actors can be composite and share a qsub	yes. All components integrated into single executable	yes. IPS runs on head node of batch job and executes apruns inside this single batch job
Scalable, metadata rich, I/O	yes. ADIOS API, ADIOS high performance implementations	yes. HDF5 API, HDF5 high performance implmentations	limited parallel IO. Metadata gathered through adapter calls
Low Latency Coupling	Prototyped (ADIOS + DataSpace in-memory)	In memory coupling (shared address space) communicating through single InterfaceRank	Framework bypassed. shared address space approach
Large Bandwidth coupling	Yes. Files on disk can handle bandwidth. DataSpace coupling has unknown staging memory limits	Yes, get/set double[] can move large data in-memory. Possibly limited by available buffer space on InterfaceRank	IPS design focused relies on Plasma State being small. auxillary files can be used for large data movement.
Wide Area Network data movement	DataMover. works asynchronously and at near peak network capability. Interacts with Kepler workflow	scp and gridftp. FACETSGUI pull/poll model	scp. gridftp, ?
input file preparation GUI	Kepler job submission and input file staging with DataMover	FACETStudio. builds input config files and sanity checks variable. batch shell script still hand-written	Hand written driver python code. hand-written batch shell script.
			Adopting the FACETSGUI input generation approach
job monitoring	MySQL database, eSimMon Dashboard	progress files	ElVis monitoring of logging
Provenance mechanism	Kepler automatic logging to database. checksums	auxillary attributes stored in HDF5 files	automatic logging from wrapper codes
programming languages (excluding external dependencies)	C, Java	C++, bash	Fortran, Python
What happens when I have implemented a component incorrectly ?	ADIOS read error. better locking still needed on read-before-write error, or stale data	incorrect virtual function override=compile error	State Adaptor to catch write to wrong variable. locking against stale data needed.

proto-FSP comparison matrix

	CPES	FACETS	SWIM
ported to Blue Gene/P	ADIOS ported. Kepler can monitor a job running on BGP	yes	no
ported to Cray XT5	yes	yes	yes
version control	internal svn, revision control of component projects, tar file archiving	Central svn repository. All code, bilder, gui, scripts, support libraries, source bourne shells	central svn repository for IPS runtime code and Adapters and config scripts. Specific component code left in native repo
regression testing	testing before major releases	comprehensive 24/7 regression tests. reporting	Informal.
bug tracking	email to help@nccs.gov	email list	email list
documentation	Kepler user manual. ADIOS User manual. EFFIS Tutorial	FACETS build instructions, FACETS testing instructions. FACETS porting instructions	Proposal, some papers, SciDAC talks, hands on tutorial
			User's Guide on project website
website	http://www.cims.nyu.edu/cpes/	https://ice.txcorp.com/trac/facets	http://www.cswim.org

FSP Planning Workshop hosted in Boulder, CO - March 15-19 (last week)

https://ice.txcorp.com/trac/fspfrmwrkplan/wiki/FspPlanningWorkshop

- Broadened to cover all areas
  - Science drivers
  - Advanced components
  - Software integration and support
  - Data management
- 3.5 days
- All talks available at

https://ice.txcorp.com/trac/fspfrmwrkplan/wiki/FspPlanningAgenda

- Report now being written

Agenda and Goals of Boulder FSP Planning Workshop

- Day 1: Science Drivers, Use cases, Other successes, Requirements and Vision
- Day 2: Approaches to the Physics (composition)
- Day 3: Cross cutting areas of Advanced Components, Physics integration, Task Composition, Development Processes, Production Computing, Data Management
- Day 4: Presentation of Findings

- Overall goal: identify gaps and approaches to each part of FSP

What did we learn about Software Integration and Support?

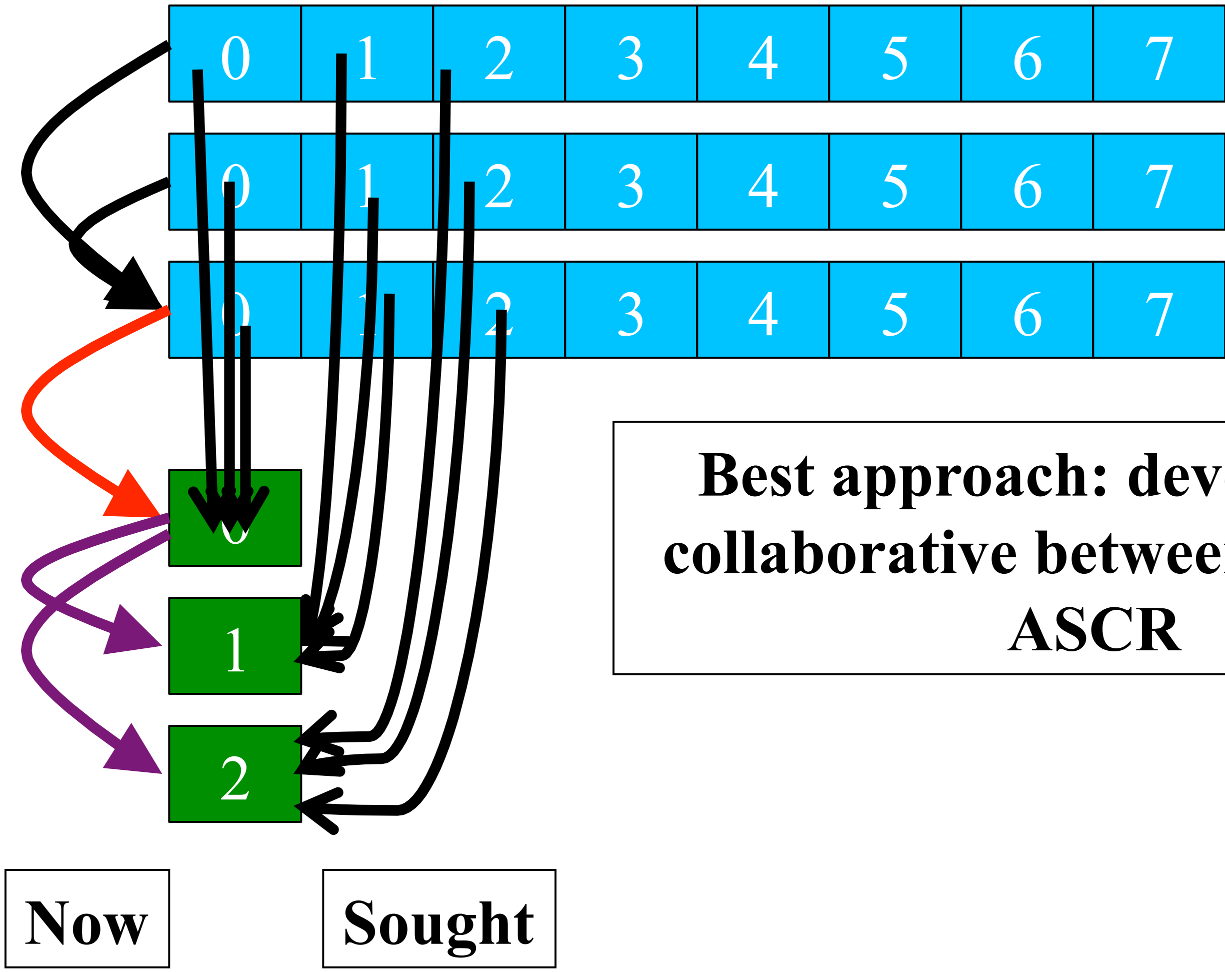
- An area-focused physics integration framework can work (Cecelia Deluca)
  - ESMF has been successful (3rd funding cycle)
  - Expect to do some major refactoring (year 3 in their case)
  - And some major restructuring (some folks are present to do research rather than produce)
  - At a cost of about $4M/year for the ESMF central team (high-performance coupling infrastructure)

What did we learn in Physics Composition?

- Our basic understanding of the couplings to support is correct (scale, region, physics, temporal)
- Generic methodologies exist for low-dimensionalities
- Lots of one-off solutions for 2D/3D
- Some current physics components not in desired form (e.g., free boundary equilibria not including circuit equations)

Opportunity for ASCR: generic framework infrastructure or lesssons for multiple offices

There are ASCR opportunities for enabling software development



Best approach: development collaborative between FES and ASCR

Physics implementations generally need lots of work to fit in framework

- Tests
- Provide provenance [sufficient information to reproduce, version info, what external libraries were used to build them, compilers (if built separately) and version]
- Documentation on what is solved, the algorithm, and how to interact (inputs, outputs)
- Self-describing, cross-platform binary output, with increasing amounts of metadata (viz, provenance, semantic)
- Restart capability (check point at specifiable time)
- Source code
- Error codes
- Specifiable input and output file names
- No hardwired I/O; settable log files (not to stdout)
- No embedded graphics
- Should work on common set of multiple platforms
- No specification of precision on compilation line

Desired by some

- Isolated definition of communicators
- Works on all LCFs, Linux, OS X

Who does this?
(Later)

Each of the proto-FSPs providing value

- **Integrated modeling**
  - **CPES: high-performance edge computations**
  - **FACETS: Core-edge integration demonstrated**
  - **SWIM: Core modeling with parallel sources**
- **Software infrastructure**
  - **CPES: Tunable I/O**
  - **FACETS: Run management (input file validation, generic job submission, visualization)**
  - **SWIM: Universal data structures for core modeling**

Overall development plans

- 1-2 years: generic software for 0D, 1D couplings, advanced components continue with one-off solutions for 2D, 3D couplings
- 5-7 years: (but started immediately) software for 2D, 3D couplings
  - Multiple kinds of grids
  - Interpolation
- Throughout: adaptation to rapidly evolving computer architectures

What did we learn in workflows?

https://ice.txcorp.com/trac/fspfrmwrkplan/raw-attachment/wiki/FspPlanningAgenda/workflow_breakout.pptx

- **Need tools for dealing with a large set of legacy data formats (iterdb, ufiles, eqdsk)**
- **Mandating viz tools not likely, but support a subset, at least one for routine, one for exploration**
- **Workflow tools can be used to enforce (some of the) provenance requirements**
- **Tools should be useful for verification and validation**
- **Desire for programmable tools (power users want to edit scripts)**

What did we learn from development processes?

https://ice.txcorp.com/trac/fspfrmwrkplan/raw-attachment/wiki/FspPlanningAgenda/DevelopmentProcesses-BreakoutReport.ppt.pdf

- General recognition that we need improved processes
- Users want federated software repositories for distribution with local control
- Cross platform important with moving LCFs
- Tools now becoming available for new era of larger projects (e.g., Bilder from FACETS)
- Must do testing
- Need global FSP bug tracking
- FSP will need non-scientists funded to take care of much of this

THIS is who does this?

What did we learn in production computing?

- **Strong desire for input file checking before submission (have 1)**
- **Workflow tools should be ubiquitous (clusters, LCFs)**
- **Will need monitoring tools (have 2)**
- **For non-LCF applications (control, cluster) need execution through matlab, python**
- **Some desire for an FSP facility with hardware**
- **Virtualization for debugging important**
- **Security should be minimized**

For exascale, will be crucial to address performance

- Advanced physics components must perform well at scale
- This becomes even more difficult in an integrated system
- Will need to advance physics components and integration in dynamic load balancing
- Performance regression testing will be critical

How do we incorporate changes into the integrated computing capability?

- ESMF model: advisory committee periodically sets the capabilities requirements
- Propose similar committee with representation of experimental facilities, theorists, users, computationalists, ASCR stakeholders
  - Propose upgrades
  - FSP management would review based on resources
  - Come to agreed upon position

Process for software integration and support

- Gather input (1st stage done, questionnaire, 3-projects meeting, planning workshop)
- Assess state of applicable software
  - We have already identified several technologies that show promise for shared use and future support (ADIOS, Run Management, Plasma State):
  - Lessons learned from proto-FSPs
- Engage larger design team
  - Frameworks team plus others
  - Prototype solutions
  - Need guidance on amount of framework unification
- Present plan for review

Timeline for frameworks planning

- Assessment of fusion software (proto-FSPs)
  - April 30, 2010: Charge completed, committee selected
  - July 2010: Meeting on what was learned from proto-FSPs?
- Development of plan for software integration
- Review of software integration plan: October 2010