

Overview of CCSM Infrastructure - Implications for ACCESS

Tony Craig

National Center for Atmospheric Research
Boulder, Colorado, USA

Presented at 18th Annual BMRC Modelling Workshop
ACCESS - Challenges and Opportunities
28 Nov - 1 Dec, 2006
Melbourne, Australia

What is Infrastructure?

- Model Infrastructure (Model Software)
 - High Level Design
 - Coupler, Coupling Software, Coupling APIs
 - Utilities (timing, performance, memory tracking, etc)
 - I/O library
 - Communication (MPI)
 - Build and Run Scripts
 - Physics APIs
- Project Infrastructure (Process)
 - Control
 - Organization, Roles and Responsibilities
 - Test and Validation Strategy
 - Communication (meetings, web pages, documentation, wiki)

Outline

- Overview of CCSM and CCSM Software Engineering
- CCSM Model Infrastructure
- CCSM Process Infrastructure
- Recommendations

CCSM Overview

- CCSM = Community Climate System Model (NCAR)
- Designed to evaluate and understand earth's global climate, both historical and future.
- Coupled system with atmosphere, ocean, sea ice, and land component models
- Multiple configurations
 - Plug and Play; active, data, and dead component models
 - Multiple resolutions (T31/4x5 to half degree)
 - Various modes; Present Day, IPCC scenarios, biogeochemistry
- Large community; scientists, software engineers, model developers, data users, managers, students, working groups

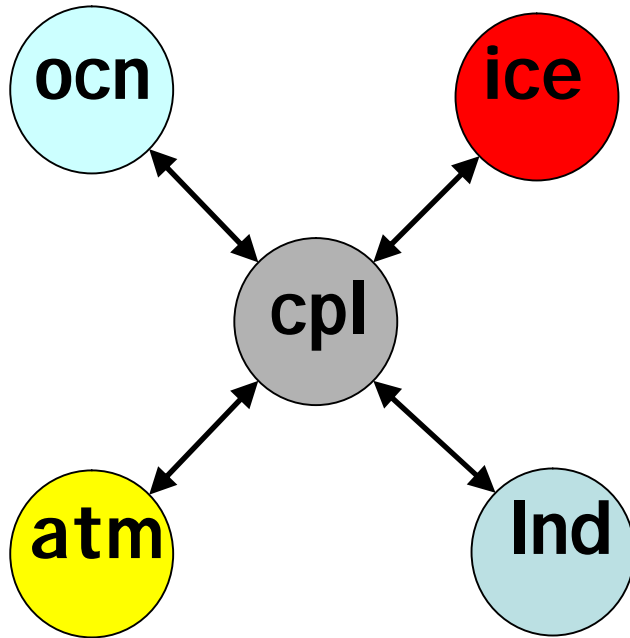
CCSM SE Overview

- **Good science top priority, facilitate capabilities**
- Fortran 90 (mostly), 500k lines of code
- Regular code releases (every few years)
- Scope - development, production, coordination, data archiving, data movement, process
- Software Engineering Working Group (broad community group)
- CCSM Software Engineering Group (~10 people based in Boulder)
- Community project; dozens of developers, hundreds of (model and data) users, thousands of others
- Collaborations are critical
 - University Community
 - SciDAC - DOE
 - ESMF - NASA/DOD

Role of CCSM SE Group

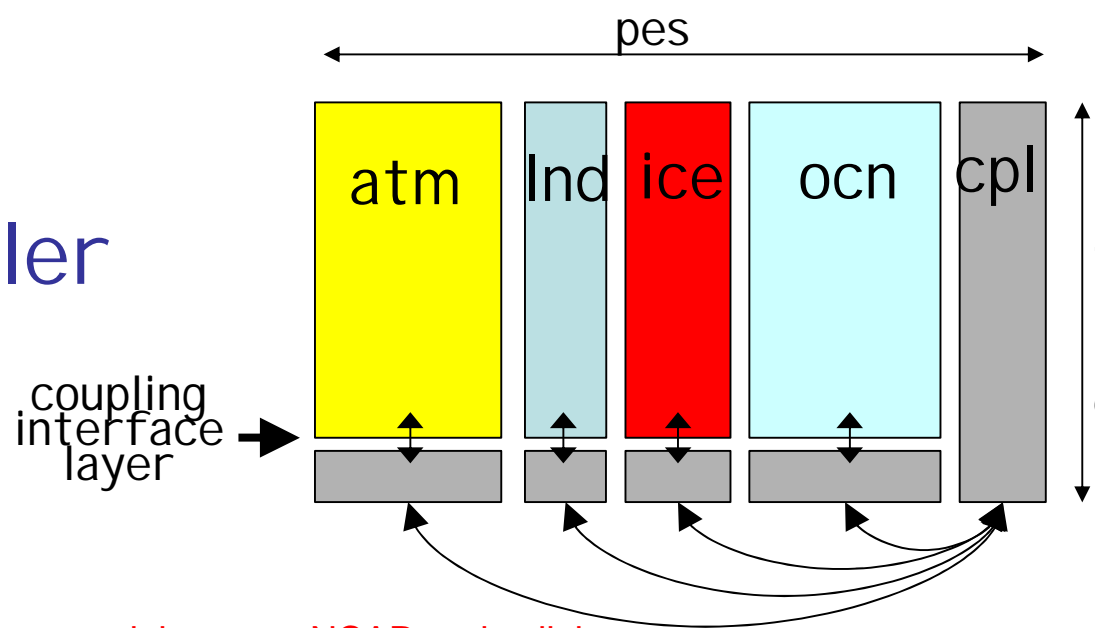
- Model development (science and IT)
- Coupler development, high level design
- Testing
- Performance, portability and hardware issues
- Build and run scripts
- Process and Control Infrastructure
- Modern coding practices (F90+)
 - Including standardize and improve model interfaces
 - Introduction of APIs
- Manage Code Releases
- Community Support
- CCSM SE Group is under resourced

CCSM "Hub and Spoke" System



- Components can run as a separate executables or combined in a single executable
- Each component is on a unique set of hardware processors
- All communications go through coupler
- Coupler
 - developed for CCSM, cpl6
 - communicates with all components
 - maps (interpolates) data
 - merges fields
 - computes some fluxes
 - has diagnostic, history, and restart capability

CCSM cpl6 coupler



- Recent redesign (cpl6)
 - Required close collaboration/teamwork between NCAR and collaborators (DOE/Argonne - MCT)
- Create a fully parallel distributed memory coupler
- Implement M to N communication between components
- Improve communication performance to eliminate any potential future bottlenecks as a result of increased resolution
- Improve coupling interfaces, abstract communication method away from components
- Improve usability, flexibility, and extensibility of coupled system
- Improve overall performance

Use of Shared Code

- Coupling infrastructure layer
- Constants (g, pi, freezing temp, “heats”)
- Physics; sun angle computation
- System calls (cd, rm, cp, env)
- File management; unit numbers, stdin/stdout
- Timing tools
- Configure/Make
- Build and run scripts automatically generated from command line scripts
 - Configuring the model; science, pes, machine, resolution
 - Namelists
 - Input data and prestaging
 - Generating run scripts including machine dependent batch settings and model startup
 - Archiving model output

Portability

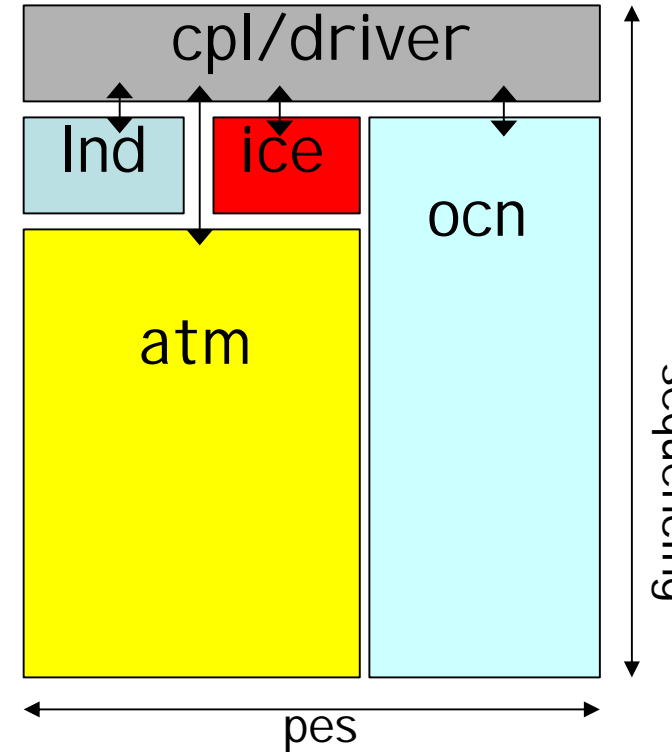
- Benefits
 - More robust code
 - Ability to take advantage of resources when they become available
 - Supports the community
 - More flexibility with respect to procurements
 - Need to have portable build and run scripts too
- Vectorization

Hardware

- Current architectures
 - Cray X1E, NEC SX - vector, high performance nodes/processors, “classic supercomputer”
 - Cray XT3, IBM Bluegene - massively parallel, small memory per pe, micro kernels, single/dual core, “next generation supercomputer”
 - IBM Power X, SGI Origin/Altix - distributed shared machine
 - Linux clusters - “commodity supercomputers”, every machine is unique, hardware/software challenges
- Performance
 - with 2 degree atm/land resolution (192x96) and 1 degree ocn/ice resolution (384x320), get O(10-40 yrs/day) on O(200 pes)

CCSM Design - Future Directions

Past	Future
Multiple Executable	Single Executable
Concurrent	Mixed sequential/concurrent (improves optimization of scaling and load balance)
Coupling in infrastructure	Coupling at driver level
1-2 degree resolution	0.1 degree resolution (technology must lead)
Moderately parallel, MPI/OpenMP, O(100) pes	Highly scalable, MPI/OpenMP, 1000s of pes (performance and memory scaling critical)



Project Infrastructure - Control

- Control risk; reproducibility, reliability, and robustness important
- Control access, source code repository; trunks, branches, rules
- Change review process (CRBs, gatekeepers, etc)
- Require testing and documentation of all code changes
 - Differentiate bit-for-bit, roundoff, non-climate, and climate changes
- **Keep it lightweight, be flexible, and allow the process to evolve**

Project Infrastructure - Tools

- Repository (CVS, Subversion)
- Model version tracking and documentation, tags
- Bug tracking (Bugzilla)
- Model runs database (MySQL)
- Bulletin Boards (vBulletin)
- Web Pages, Project Wikis
- Test automation (scripts)

Project Infrastructure-Communication

- Meetings, Telecons
- Web pages, Wikis, Hard and soft documentation
 - Public
 - Internal
- Document
 - Process
 - Access
 - Goals and plans
 - User, Science, Development Model Guides
 - Coding Standards
 - Porting and performance info
 - Model versions, Test results
 - Model runs and results

Recommendations for ACCESS

- Roles and responsibilities of leaders, groups, individuals, and the community must be clearly defined, teamwork important
- Good communication is critical
- Lightweight control processes
 - Code Repository, Change Review
 - Test Infrastructure, Validation
 - Documentation
 - Bug Tracking
- Maximize code reuse, adopt standards where appropriate
- Performance portability is worth the effort
- Work with UK Met Office as closely as possible to reduce duplication of effort in science, model infrastructure, and process.

The End