

Component-Based Software Development for High Performance Computing

David E. Bernholdt

**ORNL and the Center for Component
Technology in Terascale Simulation Software**

bernholdtde@ornl.gov

<http://www.csm.ornl.gov/~bernhold/>

What are Components?

- Units of computational functionality, large enough to do something *useful*, but not large enough to do *everything*, with a well-defined interface to the outside world
- In an OO sense, a collection of objects that can stand alone
- Size of a component can vary greatly depending on context, effort invested, etc.

Examples of Components

Context: NWChem parallel computational chemistry code

- Coupled Cluster
- Density Functional Theory
- Hartree-Fock
- Integral evaluation
- Basis set
- Geometry optimizer
- Global Arrays
- ...

Context: Newly-created componentized Hartree-Fock code

- Fock matrix builder
- Eigensolver
- Density matrix builder
- Property analyzer (density)
- Integral evaluation
- Basis set
- Geometry optimizer
- Global Arrays
- ...

Why Use Components?

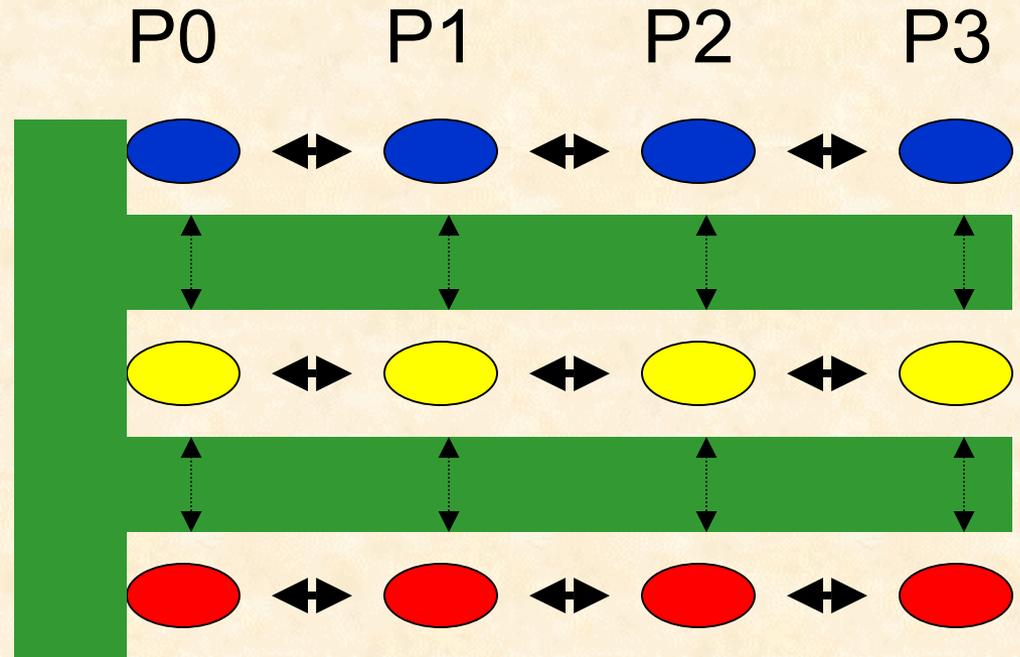
- Components improve software interoperability and re-use
- Components serve as building blocks to simplify software construction
- Components make it easier for developers to focus on creation of software in their area of expertise – a rich repository of components gives access to many experts
- Elsewhere in the computer industry, component technology is in wide use and is revolutionizing software development
 - CORBA, COM/DCOM, EJB, etc.

Components for High Performance?

- Commodity component environments are not suited to high performance computing
 - No concept of parallel computing (designed mainly for uniprocessor *distributed* computing)
 - Heavyweight – significant performance penalties even when components are running on same machine
- The **Common Component Architecture** is intended to provide a component model suitable for high-performance simulation applications
 - Preserve HPC performance
 - Bring benefits of component-based software development to HPC

The Common Component Architecture

- Supports parallel and distributed computing
- Minimalist standard
- Meant to work with commodity component models
- Currently have draft specifications (incomplete) and early prototypes
- Running parallel reaction-diffusion+visualization demonstration (SC00)



- Components: Blue, Yellow, Red
- Framework: Green
- Different components on same processor talk to each other via framework
- Same component on different processors talk to each other through their favorite communications layer (i.e. MPI, PVM, GA)

Component Interfaces

- Ports are interface definitions. Component writers code to port specifications
- Components either provide or use ports. If you *provide* a port, that means you implement it
- In current parallel framework (CCAFFEINE) components are loaded into the same address space (“direct connect” model)
- “Getting” a port means getting a pointer to the function lookup table for the component providing the port
- Calls between components are equivalent to a C++ virtual function call (low overhead)

CCA Participants and Status

- Primary developers at the moment come from
 - ANL, Indiana U, LANL, LLNL, ORNL, PNNL, SNL, U Utah
- Through informal activity over the last three years, we've developed (incomplete) draft specifications and prototype implementations
- CCA Forum is standards body, meeting quarterly, mailing list
- Center for Component Technology for Terascale Simulation Software (CCTTSS) is new SciDAC Enabling Technology Center
- CCA Forum & CCTTSS develop/standardize CCA itself (framework, essential services, etc.). Hopefully domains will organize their own “standards bodies” for CCA-compliant interfaces

CCTSS Plans

- Unify parallel and distributed frameworks
- Flesh out framework specification (esp. framework services)
- Language interoperability (incl F77, F90)
- Develop a “suite” of components to bootstrap adopters (numerical, data objects, parallel programming models, etc.)
- MxN data redistribution (coupling of parallel components)
- More than 20 projects/proposals plan to use CCA
- CCA proposal will include applications integration work in computational chemistry and climate (technology “push”)
- Work with other projects wishing to adopt (technology “pull”)

<i>Institution</i>	<i>Frameworks</i>	<i>Parallel Components</i>	<i>MxN</i>	<i>Applications</i>
ANL Lois Curfman McInnes	Low-level services	Data Components Optimization Nonlinear Solvers		Climate
Indiana Dennis Gannon	Distributed Framework	Linear Solvers		
LANL Craig Rasmussen			Component	
LLNL Scott Kohn	Language Interoperability Component Repository		Framework	
ORNL David Bernholdt, Jim Kohl		Fault Tolerance Visualization and Steering	Component	Climate Liaison
PNNL Jarek Nieplocha		Data Components		Chemistry
SNL Rob Armstrong (Lead PI)	SCMD Framework	Data Components		Chemistry
Utah Steve Parker	Builder Service Thread Safety	GUI Component	Framework	

CCTTSS Chemistry “Push”

- Participants: (ORNL), PNNL, SNL + Ames, ANL
- Focus codes: NWChem (PNNL), MPQC (SNL), GAMESS (Ames), POLYRATE/DIRDY (U Minn), TAO (ANL)
- Planned activities:
 - High-level interfaces between electronic structure methods in NWChem, MPQC & optimization methods in TAO
 - Automated protein/ligand binding studies (above + MM/MD, new drivers)
 - Automated PES discovery/mapping tools (collab. with ANL-lead SciDAC Chemistry project; NWChem, MPQC, GAMESS, POLYRATE/DIRDY, new drivers)
 - Lower-level interoperability between electronic structure packages (i.e. property evaluation and/or solvent models in NWChem, MPQC, and GAMESS)

Learning More, Becoming Compliant, Getting Involved

- www.cca-forum.org, cca-forum@z.ca.sandia.gov, quarterly meetings (next: 14-15 June, Yountsville, CA)
- I am “Applications Integration” Lead for CCTTSS
- Plan now for CCA-compliance (not a burden)
- Study and **comment on** proposals that directly relate to your interests
 - “CCA Data” working group is currently designing common interfaces for basic scientific data objects, i.e. local arrays, distributed arrays, data distributions, unstructured meshes
 - Meant to encompass existing and new packages
- Form a group to develop CCA-compliant “standard” interfaces for the materials community