Center for Component Technology for Terascale Simulation Software (aka Common Component Architecture) (aka CCA)

Rob Armstrong & the CCA Working Group
Sandia National Labs
rob@cca-forum.org
cca-forum@cca-forum.org

Component Technology for Terascale Simulation Software:

Bringing Components to High Performance Computing

http://www.cca-forum.org/ccttss

CCA

Common Component Architecture

Participants:

ANL

Indiana Univ.

LANL

LLNL

ORNL

PNNL

SNL

Univ. of Utah

Partners:

Climate Modeling

Quantum Chemistry

Combustion Modeling

Currently DOE Apps Are:

- One-off, stove-pipe, few participants

But Also:

- Scalable, and low latency
- Large & rich legacy investment

Challenges

- Practical parallel HPC component model
- Integrate legacy software investment
- Connection to Grid components
- Multi-language

Goals

- Enable plug-and-play parallel simulations
- Establish component "marketplace" with applications partners
- Extend commodity component technology for HPC

Today:

Single-purpose, monolithic, tightly-coupled parallel codes

Rob

High Performance Component Framework

Scientific Components

CCA

"MxN" Parallel Data Redistribution

Applications Integration

Component-Based Scientific Application

Discretization Engine

Implicit Solve

Visualization

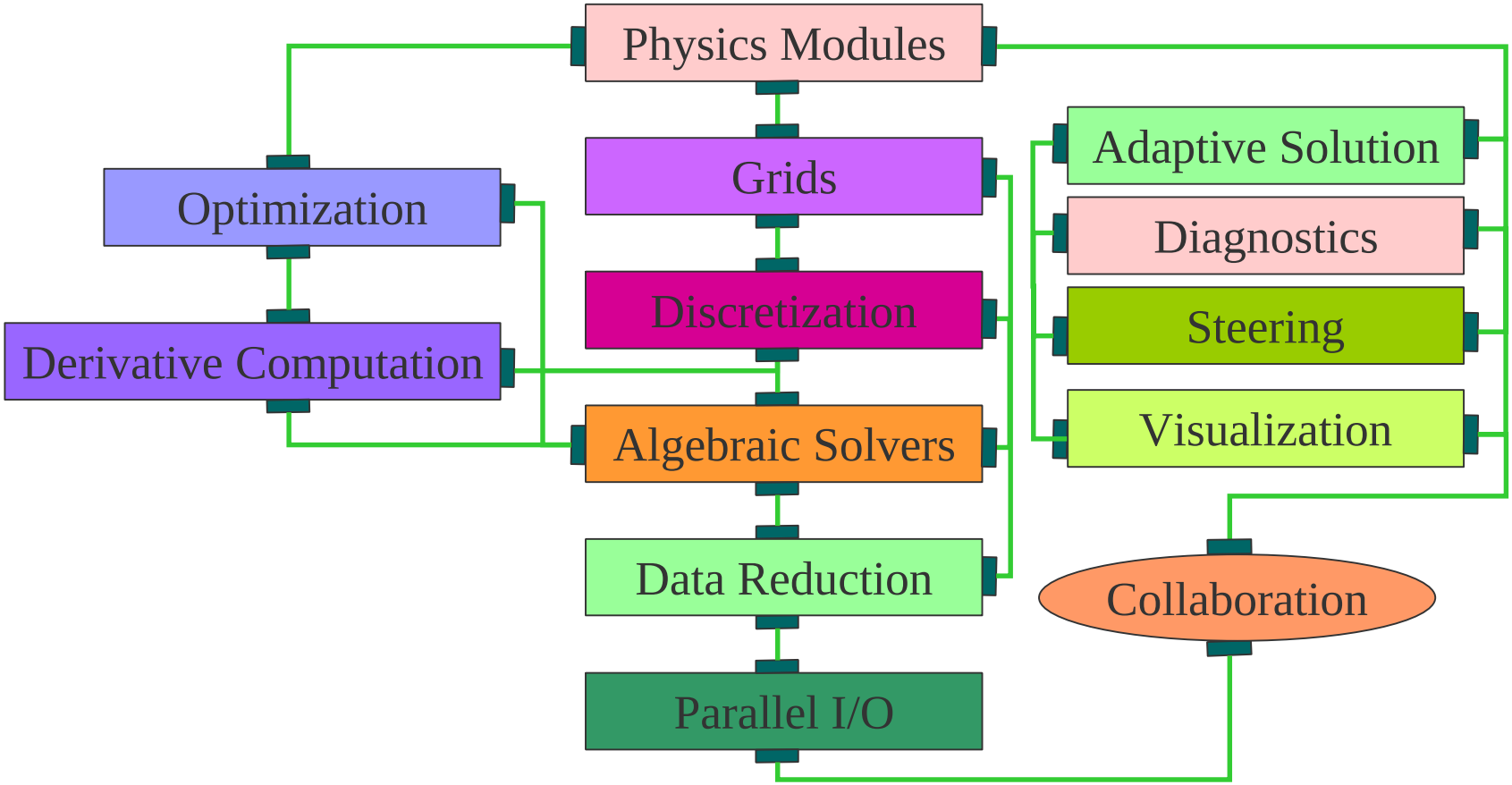
Ports

Link

Common Component Architecture: what we do for you & to you

- Gain
  - Users:
    - Choice of useful software for high-performance computing (HPC)
    - In a near mix-and-match, plug-and-play fashion
    - Independently developed code works together in parallel
  - Developers:
    - Produce code that will find a wider audience
    - With consequent fame and maintenance drudgery
    - Use other people's code and take credit for it
- Pain
  - Can't continue develop software "unconstrained"
  - CCA Constraints:
    - Provide the fences that make good neighbors
    - Are small when wrapping well-designed legacy code
    - Identify and disallow behaviors that produce incompatibility

21st Century Application



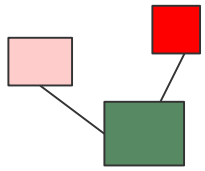
By definition components are independent software modules that are composable...

- **Component** - Encapsulated software object that provides a certain functionality or service and can be used in conjunction with other components to build applications

  A component consists of
  - an API (abstract interface) and
  - one or more component implementations
  - common behavior among these components

- **CCA uses a peer component model** - No particular component assumes it is “in charge” of the others. Allows the application developer to decide what is important.

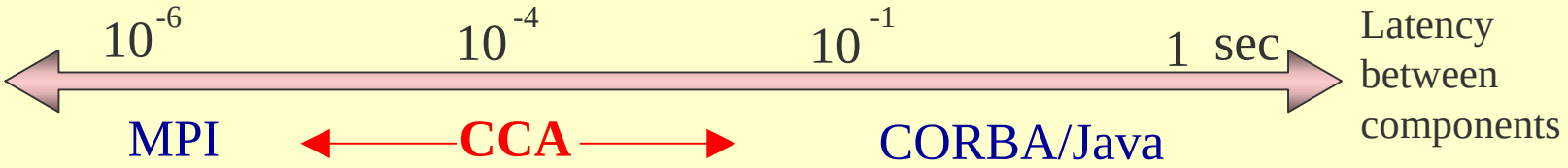
CCA Motivation



Desire to build Science Applications by hooking together drag-N-drop components.

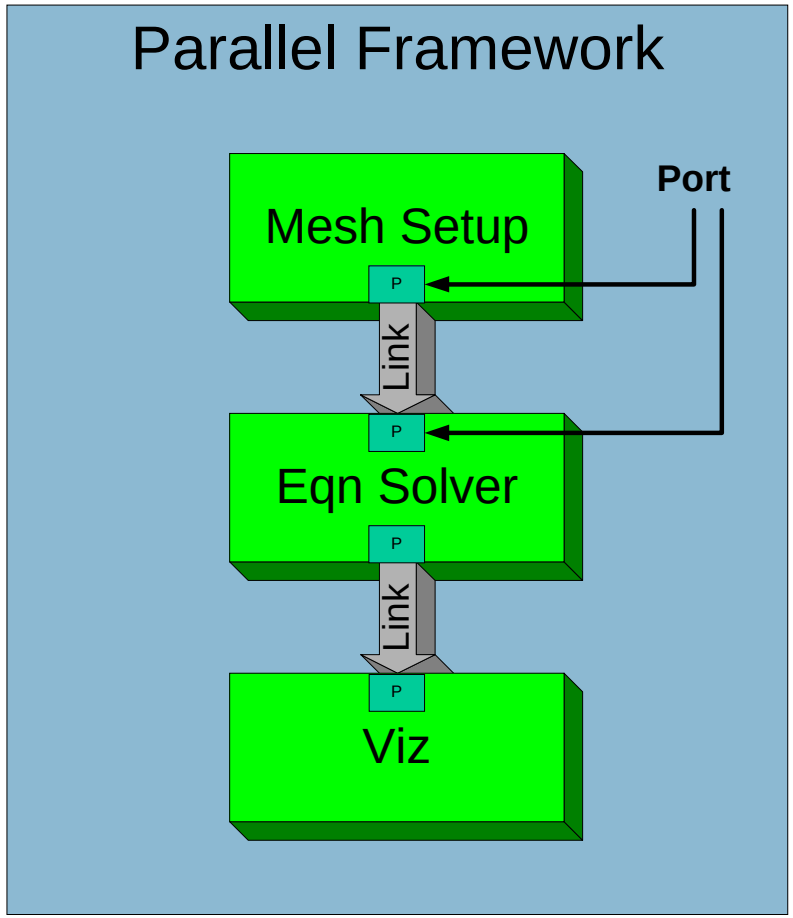
DOE Common Component Architecture provides a mechanism for interoperability of high performance components developed by many different groups in different languages or frameworks.

Existing Component Architecture Standards such as CORBA, Java Beans, and COM do not provide support for parallel components.



What have we developed?

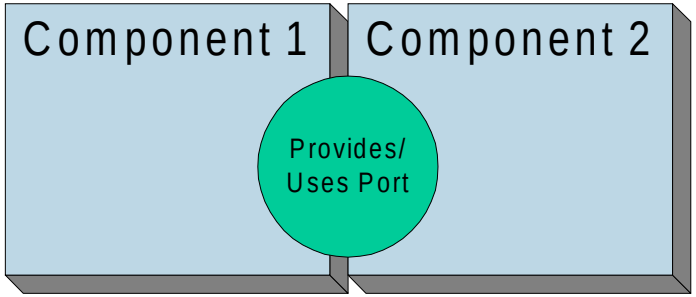
- Metaphor from visual programming (e.g. AVS)
  - instead of data-flow ports are linked with interfaces
  - provides/uses design pattern
- Scripting support for applications
  - create Python applications out of components
- Scientific IDL spec.
  - provides language interoperability only - not part of the component arch. *per se.*
  - introspection.



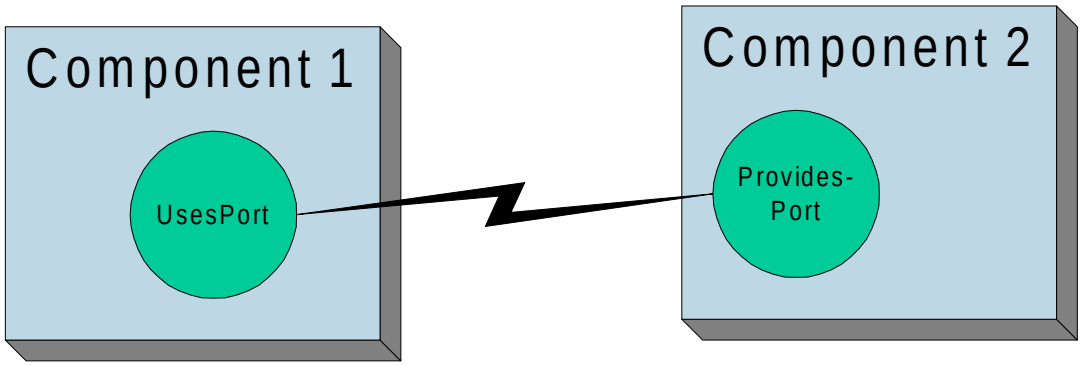
http://www.cca-forum.org/spec

Ports preserve the high-performance of direct connections plus the versatility of distributed object systems

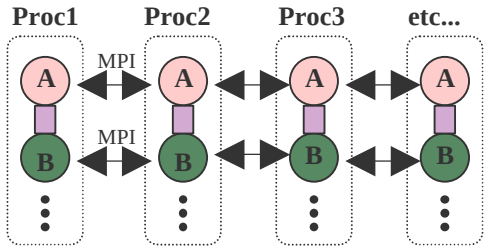
- Allow directly connected interfaces: the next component is one function call away.

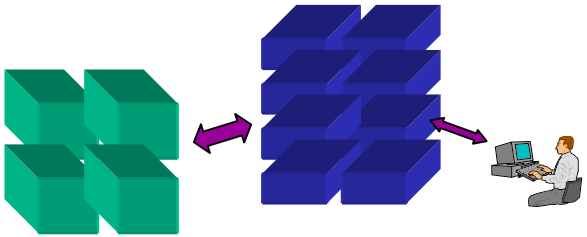


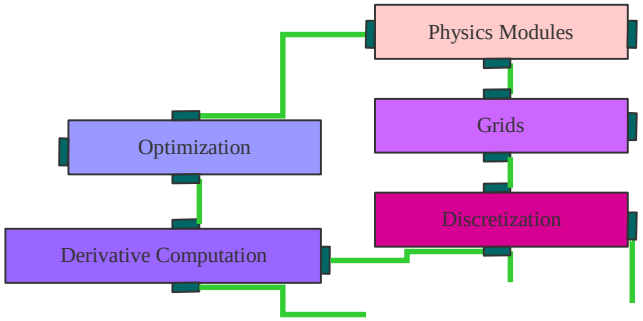
- Adapters will create network-distributed objects out of the same components without altering them.



CCA develops components & supporting code for direct use in apps



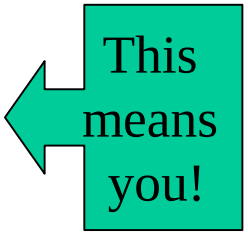




- Reference Framework
  - One unified framework
  - Parallel and grid aware
- MxN
  - Data re-decomposition components
  - Framework plug-ins
- Components and Applications
  - Various component prototypes
    - service capabilities, discretization, mesh management, linear algebra, vizualization, etc.
  - Collaboration with other groups to define common interface suites
    - E.g., scientific data interfaces (structured and unstructured)
- Details: http://cca-forum.org/ccttss/report/Dec01

Collaborations with Other SciDAC Groups

- CCA does not pretend to be experts in all numerical algorithms, just to provide a standard way to exchange component capabilities.
- Medium of exchange: interfaces
  - Equation Solver Interface Forum (ESI) has made headway toward defining interfaces for linear algebra
    - http://z.ca.sandia.gov/esi
  - **Need experts in various areas to define sets of domain-specific abstract interfaces**
    - scientific application domains, applied math, visualization, etc.
      - e.g., compressible flow component
      - e.g., Newton-Krylov methods



CCA aids other centers with their component oriented applications

- CCA tutorials
  - Recent: PNNL (Dec 2001) & prior to CCA Forum meeting in Santa Fe (last week)
  - Planned: adjunct to future CCA Forum meetings (quarterly).
  - Materials to be placed on web: http://www.cca-forum.org/tutorial
- Direct interactions with application centers and other ISIC's (http://www.cca-forum.org/ccttss/report/Dec01/outreach.pdf).
- Example code:
  - Specification site has toy examples:
    - http://www.cca-forum.org/spec
  - SC01 code is available
    - Several heavy-weight applications.
      - Not toys, takes some work to build:
      - http://www.cca-forum.org/cca-sc01

Some Current Interactions between CCA and Other SciDAC Groups

- Chemistry/Combustion modeling
  - *A Computational Facility for Reacting Flow Science* (PI: H. Najm) - Develop a CCA-based software research "facility" for reacting flow science
  - *Advanced Software for the Calculation of Thermochemistry, Kinetics, and Dynamics* (PI: A. Wagner) – Develop component-based solutions for the coupling of electronic structure and kinetics software
- *Collaborative Design and Development of the Community Climate System Model for Terascale Computers* (PI: R. Malone)
  - Develop component-based software design methodologies and integrate the CCA with current componentization efforts within the climate community
- *Terascale Optimal PDE Simulations ISIC (*PI: D. Keyes)
  - Create suites of interoperable solvers and define common interfaces for PDE, eigenanalysis, and optimization components
- *Terascale Simulation Tools and Technologies ISIC* (PI: J. Glimm)
  - Create interoperable meshing and discretization technologies and define common interfaces for scientific data components
- Additional SciDAC interactions in progress (see http://www.cca-forum.org/ccttss), and we are seeking new SciDAC collaborations also