CCA-PIO & CCA-LSI:
On Design Common Component Architecture Interface for High Performance Scientific Computing Libraries

Fang Liu, Yongquan Yuan, and Randy Bramley
{fangliu, yyuang, bramley}@indiana.edu
Extreme Computing Lab, Indiana University, Bloomington, Indiana

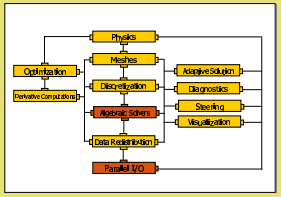
Project Overview

This work targets the emerging use of software component technology for high-performance scientific parallel and distributed computing. The use of Common Component Architecture (CCA) framework provides opportunity to encapsulate the complex scientific software and to reuse them easily. BABEL compiler from LLNL provides language interoperability among scientific packages.

This poster presents an effort on designing minimal set of interfaces for HPC parallel file I/O libraries and HPC linear solver libraries. Preliminary experiments show that overhead introduced by adding the CCA layer on original software package is very small.

Motivating Simulations

The large-scale scientific simulation often combines use multiple software packages developed by different groups in different computer languages. For example, computational astrophysics, chemistry and fusion, each has challenging resolution and complexity requirements that demand massively parallel computing resources and a range of sophisticated software. CCA provides an application composition model based on reusable components. Common interface needs to be defined to facilitate the interaction between components.



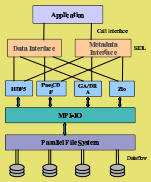
This diagram presents computational phases in typical multidisciplinary scientific applications.

CCA-PIO Design

What CCA-PIO stands for?

A CCA Common Parallel File I/O Interface.

Design Architecture



Why CCA-PIO is needed?
•High-level, easily used parallel I/O interfaces to manipulate structured data such as arrays to facilitate transferring objects between components that might even be running on different numbers of processes
• Interface, programming language, and amount and type of metadata required by different I/O libraries differ widely
•Files written by one data library interface are not directly accessible by another

SIDL Interface

CCA-PIO uses SIDL to provide a universal, minimalist set of parallel I/O interfaces for data and metadata, spanning multiple data libraries.

```
package ccaPio version 0.1 {
    interface dataStore{
      int createArrayTemplate(...)
      int openFile(…);
      int closeFile(…);
      int writeArray(…);
      int readArray(…);
    }
    interface metaStore {
      int getArrayTemplate(…);
      …
  }
}
```

dataStore interface

• Setup array template
• Operate files
• Manipulate data

metaStore interface

• Get information about data.

CCA Parallel I/O Component

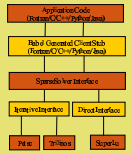
• Provides interface compatibility between components
• Provides language interoperability (Fortran, C, C++, Java, Python)
• Enables parallel I/O data coupling across components
• Maintains data portability over platform
• Allows re-use of existing data libraries

CCA-LSI Design

What CCA-LSI stands for?

A CCA General Sparse Linear Solver Interface.

Design Architecture



Sparse linear solver is the most essential and computational intensive part of HPC applications. Our design tries to decouple the HPC applications from the solver libraries they use by adding Sparse-Solver interface between the application and solver libraries. Application user don't have to learn individual underneath solver libraries, but simply use our interface and implementation.
Currently, the implementation is done for the Trilinos, Petsc and SuperLu.

SIDL interface

CCA-LSI uses SIDL to provide a minimal set of interface for parallel sparse linear solver libraries.

```
Package ccaLsi version 0.1 {
  interface SparseSolver{
    void Initialize(…);
    void Finalize();
    int SetupMatrix(…);
    int SetupRHS(…);
    int SetupOptions(…);
    int Solve(…);
  }
  iterface Iterative extends SparseSolver {
        …
  }
  interface Direct extends SparseSolver {
        …
  }
}
```

SparseSolver interface:

•Setup linear system
•Setup solve parameters
•Solve

Iterative and direct subinterfaces:

•Specializations for algorithm, parameter settings.

CCA Parallel Linear Solver Component

Implementation of the CCA-LSI provides a reusable CCA linear solver component which relieves the application developer from changing application code when changing the solver libraries. Along with Babel generated client stub, It provides language interoperability.

This work is supported by DOE grant