Center for Component Technology for Terascale Simulation Software
PI: Rob Armstrong

Quarterly Report for SciDAC ISIC – Quarter 1
December, 2001

Overview

During the first quarter of work within the Center for Component Technology for Simulation Software (CCTTSS), we have achieved significant advances throughout all four facets of the project: frameworks and infrastructure, parallel scientific components, parallel data redistribution and model coupling, and integration of the Common Component Architecture (CCA) into scientific applications. As an initial step in working toward our vision of the CCA as a foundation of DOE high-performance scientific application software communities, we have been quite active with general outreach, with particular emphasis on laying the groundwork for interactions with the TOPS and TSTT SciDAC ISICs as well as SciDAC application projects in climate modeling, computational chemistry, fusion, combustion, accelerators, and high-energy and nuclear physics. Additional highlights of initial work include the development of a suite of prototype parallel components and component-based scientific applications, which serve as a foundation for CCA tutorial material and provide a starting point for interfaces with several scientific applications projects. We demonstrated these components at the SC2001 conference and plan to deploy them through the CCTTSS web site in early 2002.

Integration of the CCA into Scientific Applications

Coordinator: David Bernholdt

We are taking a three-pronged approach to user outreach and application integration. During the current period, we have intensified the general outreach efforts that have been active since the CCA's inception. We have also begun laying the groundwork for interactions with external projects that use CCA technologies and organizing the Center's internal application focus activities in climate modeling and computational chemistry.

To summarize the more detailed list of our activities presented in Appendix A, during the past quarter, Center members submitted three papers and presented seven talks and seminars. We highlight the SC2001 conference as a significant coordinated Center activity during this period. Our primary presence in the Exhibition Hall was hosted in Oak Ridge National Laboratory's booth, where we showed three different parallel scientific applications as demonstrations of CCA technologies. Several of these demos were also presented in the Argonne booth, and additional demonstrations were shown in the Los Alamos, NCSA, and Research@Indiana booths. We also hosted a "birds of a feather" session on *Component-Based Software Development for High-Performance Computing*. Through our activities at SC2001, we were able to introduce many people to the Common Component Architecture, and it is likely that a number of these will become CCA users in the near future.

We also had direct interactions with a number of research projects (both SciDAC and non-SciDAC) that have already expressed strong interest in CCA or a definite intent to use CCA technologies. We are closely tied to the TOPS SciDAC ISIC through active CCTTSS members Lois Curfman McInnes (ANL) and Scott Kohn (LLNL), and to the TSTT SciDAC ISIC through Lori Freitag (ANL), who is a PI in that center as well as coordinator of the CCTTSS Scientific Data Working Group. Lori Freitag is also involved in ongoing meetings with the University of Chicago's FLASH Center, which is interested in CCA as they evaluate design issues for their flagship application code. David Bernholdt and Jim Kohl (ORNL) have begun discussions with Tony Mezzacappa (ORNL), who leads one of the SciDAC HENP applications projects on supernovae. David Bernholdt has also had preliminary discussions with Don Batchelor (ORNL), who leads one of the SciDAC fusion applications. Craig Rasmussen (LANL) is a co-investigator and the Center's liaison to the SciDAC HENP accelerator science project lead by Kwok Ko (SLAC) and

Robert Ryne (LANL). The CCTTSS, with Jaideep Ray (SNL) as primary liaison, has also been involved with the SciDAC CFRFS combustion center led by Habib Najm (SNL) from its inception. The CFRFS plans to use CCA technology, including an AMR structured mesh (GrACE) component, as the "glue" for their planned software research "facility."

During this period we also began our applications integration work within the Center. In November, David Bernholdt (ORNL), Lori Freitag and Jay Larson (ANL), and Craig Rasmussen (LANL) met at the National Center for Atmospheric Research with our collaborators from the main SciDAC Global Climate Research project lead by Robert Malone (LANL), as well as a number of researchers interested in CCA in conjunction with NASA's Earth System Modeling Framework. We developed a plan for the initial integration of CCA into the Model Coupling Toolkit (which will in turn become part of the Community Climate System Model (CCSM)) as an example/demonstration for the larger CCSM community. In December, Steve Benson (ANL), David Bernholdt (ORNL), Curt Janssen and Helgi Adalsteinsson (SNL), and Theresa Windus (PNNL) met at PNNL to launch the computational chemistry application focus. Collaborator Rick Kendall (Ames) and Center member Lois Curfman McInnes (ANL), who were unable to attend the meeting in person, joined by teleconference for some portions. Our discussions lead to a plan for the initial "componentization" experiments on the computational chemistry packages NWChem and MPQC and interfacing to optimization components based on the Toolkit for Advanced Optimization (TAO).

Component Frameworks and Infrastructure

Contributors:

- Scott Kohn (coordinator), Tamara Dalghren, Tom Epperly, Gary Kumfert (LLNL)
- Dennis Gannon, Ken Chiu, Alek Slominski (IU)
- Craig Rasmussen, Matt Sotille (LANL)
- Ben Allan, Rob Armstrong (SNL)
- Steve Parker, Todd Kalil (UU)

During the past quarter, our activities in frameworks and infrastructure focused on three primary areas: additions to the CCA component specification, improvements and integration of existing component frameworks, and enhancements to our language interoperability technology.

We extended the existing CCA component specification with two new interfaces: *BuilderServices* and *TypeMap*. *BuilderServices* is a new port interface that simplifies the construction of component applications through uniform creation, connection, and deletion mechanisms. It provides a standard way for application developers to access CCA framework services from their main program or script. To demonstrate *BuilderServices* in a scripted environment, we created a prototype Python binding to these services within the CCAFFEINE framework using SILOON to automatically generate the Python code. We expect to migrate this to use the SIDL language interoperability technology in the future. Our second interface addition to the standard, *TypeMap*, provides a type-safe and convenient mechanism for passing configuration information among co-operating components in an application. It can be thought of as a component version of the command-line arguments normally provided to a non-component application.

For our framework activities, we focused on the integration of our existing component framework technology and the demonstration new framework capabilities. The SCIRun and CCAFFEINE teams collaborated to combine the best capabilities from the two frameworks and work those capabilities into the CCA standard. We also completed a prototype implementation of a Grid-based CCA framework within a web services framework. Our implementation uses Globus to create components, SOAP (an XML over HTTP protocol) as a communication mechanism, SSL for security, and Grid standard public key security for authentication and authorization. We demonstrated this system for applications in chemical engineering, remote instrumentation, earth orbit debris computations, and linear algebra test cases. Finally, we began development of a graphical builder for a CCA-compliant framework based on previous work with the SCIRun framework. This new graphical builder will be used by computational scientists to create component applications using a visual programming metaphor.

We have begun to merge our existing language interoperability techniques into a CCA component framework. As a proof-of-principle, we implemented a prototype CCA framework using SIDL and Babel that demonstrated the ability to integrate software written in C++, C, Fortran, Java, and Python. This work is a first step towards integrating SIDL and Babel into existing CCA frameworks. Furthermore, the SIDL and SILOON/CHASM teams have begun to combine the automatic parsing capabilities of SILOON/CHASM with the general language interoperability capabilities of SIDL and Babel. This combined technology would enable the semi-automatic wrapping of legacy Fortran 90 and C++ components. Finally, we completed the Python support for SIDL and added over two thousand new test cases to improve the robustness and verify the correctness of the Babel language interoperability infrastructure.

Parallel Components for DOE Computational Science Domains

Contributors:

- Lois Curfman McInnes (coordinator), Steve Benson, Lori Freitag, Boyana Norris (ANL)
- Randy Bramley, Dennis Gannon, Ken Chiu, Alek Slominski (IU)
- Matt Sotille, Craig Rasmussen (LANL)
- David Bernholdt, Jim Kohl (ORNL)
- Ben Allan, Rob Armstrong, Jaideep Ray (SNL)
- Steve Parker, Todd Kalil (UU)

During the past quarter, CCA component developers have continued to incorporate into the CCA specification the most promising ideas that we have first explored in various institutional projects. This work involves both updating components to reflect recent changes in the specification and proposing modifications to the CCA standard to migrate ideas from various implementations. In addition, the CCA Scientific Data Components Working Group created a group charter and defined raw data and local array interfaces specifications that have been passed by the CCA Forum. The subgroup also developed prototype unstructured mesh query interfaces (in conjunction with the TSTT SciDAC center) and distributed array interfaces.

A working group with participants from ANL, ORNL, and SNL developed three component-based scientific applications that demonstrate the reusability and composability of various new prototype CCA-compliant components, including several that employ interfaces under development by CCA working subgroups for scientific data and parallel data redistribution. In addition, LANL developed several components and demos. These components and applications, which we demonstrated at the SC2001 conference, serve as a foundation for CCA tutorial material and provide a starting point for interfaces with several scientific applications projects. A brief overview of these components is below, while Appendix B contains a complete list with a short description of each. In addition, we will deploy the full source code and documentation through the CCTTSS web site in early early 2002.

These components provide baseline functionality for unstructured mesh management, finite element discretization, structured AMR, parallel linear algebra, time integration, unconstrained minimization, description of distributed data decompositions, redistribution of parallel data, and passing data to graphical front-ends for visualization. Two of the demo applications solve a time-dependent PDE: one uses adaptive structured meshes and the other uses unstructured meshes; the third application solves an unconstrained minimization model using structured meshes. All three applications employ the CCAFFEINE framework and re-use the following components: (1) an *ESIFactory* and parallel linear solver components that incorporate abstract interfaces under development by the Equation Solver Interface (ESI) working group, (2) a *DistArrayDescriptorFactory* component to describe parallel data layout, (3) a *CumulvsMxN* data redistribution component that collates data from multi-processor runs to a single processor for runtime visualization using (4) *VizProxy* component variants. These components leverage and extend parallel software tools developed at different institutions, including CUMULVS, GrACE, LSODE, MPICH, PETSc, PVM, SUMAA3d, TAO, and Trilinos.

Parallel Data Redistribution and Model Coupling

Contributors:
- Jim Kohl (coordinator), David Bernholdt (ORNL)
- Sue Mniszewski, Pat Fasel, Reid Rivenburgh, Craig Rasmussen (LANL)

The CCA's “MxN” Parallel Data Redistribution Working Group assembled an interface specification in July, 2001 that builds on months of ongoing group work and encompasses the existing functionality of both the CUMULVS (ORNL) and PAWS (LANL) systems. This MxN interface captures the fundamental operations required by parallel data coupler components, including a full range of potential synchronization and communication options.

Using this common interface specification, we developed two new MxN component prototypes based on CUMULVS and PAWS technology and demonstrated these components in several applications with the CCAFFEINE framework at SC2001. Both MxN components are single-threaded and allow synchronized communication and parallel data exchange among remotely executing components, across multiple CCA frameworks. The CUMULVS MxN component was used unmodified in three distinct applications developed for SC01 demonstrations (discussed above). The *DistArrayDescriptorFactory* component provided a generalized means for describing the data fields and their decompositions for the MxN component. A CUMULVS-based *VizProxy* component applied the MxN component to gather parallel data from each of the applications into a local array for visualization. The PAWS MxN component demonstrated a simple ping-pong example, where data was sent from a parallel “M” sender to a parallel “N” receiver and then back again, such that the number of processes “M” and “N” on each side was different (with M and N greater than 1).

The next phase of MxN research will continue developing these initial implementations to fully support the current MxN specification. We will also extend the specification to include a generalized and consistent mechanism for application/data discovery, as well as more low-level support interfaces for manipulating MxN data handles, communication schedules, and parallel data connections.

Appendix A
Details of CCTTSS Outreach Activities

[1] Benjamin A. Allan, Robert C. Armstrong, Alicia P. Wolfe, Jaideep Ray, David E. Bernholdt, and James A. Kohl, The CCA Core Specification in a Distributed Memory SPMD Framework, *Concurrency and Computation: Practice and Experience* (accepted, August, 2001).

[2] Matthew Sottile and Craig Rasmussen, Automated Component Creation for Legacy C++ and Fortran Codes, paper, First International IFIP/ACM Working Conference on Component Deployment, Berlin, Germany, June 20-21, 2002, submitted.

[3] C. R. Johnson, S. G. Parker, and D. M. Weinstein, Component-Based Problem Solving Environments for Large-Scale Scientific Computing, *Concurrency and Computation: Practice and Experience* (submitted).

[4] Dennis Gannon, Component Architectures for High Performance Grids, talk, The Tenth IEEE International Symposium on High Performance Distributed Computing (HPDC), San Francisco, California, 2001.

[5] Kate Keahey, Pat Fasel, and Sue Mniszewski, PAWS: Collective Interactions and Data Transfers, talk, The Tenth IEEE International Symposium on High Performance Distributed Computing (HPDC), San Francisco, California, 2001.

[6] Dennis Gannon, Component Architectures for High Performance Grids, invited talk, European Conference on Parallel Computing (EuroPar), Manchester, UK, 2001.

[7] Matt Sottile, Component Technology for Terascale Simulation Software, talk, LACSI Symposium Workshop on High Performance Numerical Libraries, 2001.

[8] Dennis Gannon et al., Programming the Grid: Distributed Software Components, P2P and Grid Web Services for Scientific Applications, invited talk, 2nd International Workshop on Grid Computing (GRID 2001), workshop at SC2001, Denver, Colorado, 2001.

[9] David E. Bernholdt, Component-Based Software for High-Performance Computing: An Introduction to the Common Component Architecture, seminar, Oak Ridge National Laboratory, Oak Ridge, Tennessee, 2001, Attended by ORNL representatives of two SciDAC ISICs and two SciDAC applications projects.

[10] David E. Bernholdt, Component-Based Software for High-Performance Computing: An Introduction to the Common Component Architecture, seminar, Pacific Northwest National Laboratory, Richland, Washington, 2001.

[11] SC2001 Conference Activities, Posters and demonstrations in the ANL, LANL, NCSA, ORNL, and Research@Indiana booths., 2001.

[12] Component-Based Software Development in High Performance Computing, ”Birds of a Feather” session, SC2001, Denver, Colorado, 2001, Speakers included: Rob Armstrong (SNL), Steve Parker (Utah), Alan Sussman (Maryland).

[13] Lois Curfman McInnes, Component Technology for Terascale Simulation Software, talk, TOPS Center Kickoff Meeting, 2001.

[14] David E. Bernholdt, Initial discussions with ORNL fusion researchers representing an ORNL-led SciDAC Fusion project and other non-SciDAC projects., 2001.

[15] Lori Freitag, CCTTSS representative to TSTT Unstructured Mesh Query Interface Working Group, 2001.

[16] Craig Rasmussen, Component Technology for Terascale Simulation Software, talk, Accelerator SciDAC Center kickoff meeting, 2001.

[17] Lori Freitag, University of Chicago FLASH Center, Bob Rosner PI, Ongoing discussions on design issues in their flagship application code.

[18] Gary Kumfert and Scott Kohn, Introducing Babel: a Language Interoperability Tool, Working Meeting with TOPS (David Keyes, Steve Benson, Andy Cleary, Matt Knepley, Barry Smith) and TSTT (Lori Freitag) representatives, Argonne National Laboratory, Argonne, Illinois, 2001.

Appendix B

Overview of SC2001 Demo Components
December, 2001

A working group with participants from ANL, ORNL, and SNL developed three component-based scientific applications that demonstrate the reusability and composability of various new prototype CCA-compliant components, including several that employ interfaces under development by CCA working subgroups for scientific data and parallel data redistribution. In addition, LANL developed several components and demos. A brief overview of these parallel components is below; complete source code and documentation will be deployed through the CCTTSS web site in early 2002. These components and applications, which we demonstrated at the SC2001 conference, serve as a foundation for CCA tutorial material and provide a starting point for interfaces with several scientific applications projects.

Two applications solve a time-dependent PDE: one uses adaptive structured meshes and the other uses unstructured meshes; the third application solves an unconstrained minimization model using structured meshes. All three applications employ the *CCAFFEINE* framework and re-use the following components: (1) an *ESIFactory* and parallel linear solver components that incorporate abstract interfaces under development by the Equation Solver Interface (ESI) working group, (2) a *DistArrayDescriptorFactory* component to describe parallel data layout, (3) a *CumulvsMxN* data redistribution component that collates data from multi-processor runs to a single processor for runtime visualization using (4) *VizProxy* component variants. These components leverage and extend parallel software tools developed at different institutions, including CUMULVS, GrACE, LSODE, MPICH, PETSc, PVM, SUMAA3d, TAO, and Trilinos.

The following components provide service capabilities:

- ***CheetahMPI*** – Craig Rasmussen, Sue Mniszewski, Pat Fasel, and Reid Rivenburgh (LANL) – Provides a high level interface on top of MPI to allow different components to partition MPI_COMM_WORLD into separate communication groups.

- ***CCAFFEINE Utilities*** – Ben Allan and Rob Armstrong (SNL) – Provide a range of prototypes for service components, including a simple ***Logger*** component to print diagnostics to a user-configurable file and a ***BuilderService*** prototype that renders the entire *CCAFFEINE* framework as a component to facilitate application building.

The following components provide capabilities for mesh management and discretization:

- ***TSTTMesh*** – Lori Freitag (ANL) – Provides prototype capabilities for querying unstructured meshes based on interfaces being designed within the TSTT SciDAC Center.

- ***FEMDiscretization*** – Lori Freitag (ANL) – Provides finite element discretization of diffusion and advection PDE operators and linear system assembly capabilities.

- ***GrACEComponent*** – Jaideep Ray (SNL) – Provides parallel AMR infrastructure, which follows a hierarchy-of-patches methodology for meshing and includes load-balancing; *GrACEComponent* is being used within a combustion code in the SciDAC CFRFS project.

- ***Various AMR Support Components*** – Jaideep Ray (SNL) – Provide support for AMR computations, including computation of the error incurred by advancing the solution and adaptors that translate a patch-based hierarchy into a *DistArrayDescriptor*, which can then used in parallel data redistribution and visualization.

The following numerical components provide parallel linear algebra, time integration, and minimization:

- ***ESIFactory*** **–** Boyana Norris (ANL) **–** Generates linear algebra objects that employ interfaces under development by the ESI working group; these vectors, matrices, and linear solvers are used as fundamental objects within *LinearSolver*, *IntegratorLSODE*, *TAOSolver*, and *TSTTMesh*, so that applications can use ESI-compliant implementations provided by external components; currently available underlying implementations are based on the PETSc (ANL) and Trilinos (SNL) libraries.

- ***LinearSolver*** **–** Boyana Norris (ANL) **–** Provides a prototype linear solver port that uses ESI vector and operator interfaces, so that applications can benefit from implementations that use newly ESI-enabled parallel linear solvers within the PETSc and Trilinos libraries. *LinearSolver* components implement functionality that generally cannot be defined by using only the ESI solver interface.

- ***IntegratorLSODE*** **–** Ben Allan (SNL) **–** Provides a parallel ODE solver, based on LLNL's ODEPACK and the ESI specification. The distribution also contains components that provide specialized mathematical operations on ESI vectors, generalized error/convergence specifications for implicit time-integrators, and logging to disk of errors/messages from different components in a parallel setting.

- ***TAOSolver*** – Steve Benson, Lois Curfman McInnes, and Boyana Norris (ANL) – Provides parallel unconstrained minimization solvers, which build on infrastructure within TAO (ANL) and use the ESI specification. We also developed sample components for function, gradient, and Hessian evaluation, which serve as starting points for interfacing with SciDAC computational chemistry applications.

The following components provide baseline functionality for describing distributed data decompositions, exchanging and redistributing parallel data, and passing data to graphical front-ends for visualization:

- ***DistArrayDescriptorFactory*** – David Bernholdt (ORNL) – Provides a uniform means for applications to describe dense multi-dimensional arrays and is based upon emerging interfaces from the CCA Scientific Data Components Working Group.

- ***CumulvsMxN*** – Jim Kohl (ORNL) **–** Builds on CUMULVS (ORNL) technology to provide an initial implementation of the "MxN" parallel data redistribution interfaces that are under development by the CCA "MxN" Working Group. *CumulvsMxN* is designed to span multiple CCA frameworks and to pass data between two distinct parallel component applications.

- ***PawsMxN*** – Sue Mniszewski, Pat Fasel, Reid Rivenburgh, and Matt Sottile (LANL) – Builds on PAWS (LANL) technology to distribute data between two data parallel components. The *PawsMxN* component uses MPI to move data in a scalable and efficient fashion and follows the evolving CCA "MxN" interface specification.

- ***VizProxy*** – Jim Kohl (ORNL) – Provides a companion "MxN" endpoint for extracting parallel data from component-based applications and then passing this data to a separate front-end viewer for graphical rendering and presentation. Variants exist for structured data and unstructured triangular mesh data as well as text-based output.