The SciDAC Center for Technology for Advanced Scientific Component Software (TASCS)

http://tascs-scidac.org

Lead PI: David Bernholdt
Oak Ridge National Lab, bernholdtde@ornl.gov

<table>
<thead>
<tr>
<th>Institution</th>
<th>Institutional Lead PI</th>
<th>Institution</th>
<th>Institutional Lead PI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANL</td>
<td>Lois McInnes</td>
<td>SNL</td>
<td>Rob Armstrong</td>
</tr>
<tr>
<td>Binghamton U</td>
<td>Madhu Govindaraju</td>
<td>Tech-X</td>
<td>Sveta Shasharina</td>
</tr>
<tr>
<td>Indiana U</td>
<td>Randy Bramley</td>
<td>U Maryland</td>
<td>Alan Sussman</td>
</tr>
<tr>
<td>LLNL</td>
<td>Tom Epperly</td>
<td>U Oregon</td>
<td>Matt Sottile</td>
</tr>
<tr>
<td>ORNL</td>
<td>Jim Kohl</td>
<td>Virginia State U</td>
<td>Kosta Damevski</td>
</tr>
<tr>
<td>PNNL</td>
<td>Jarek Nieplocha</td>
<td>past participants</td>
<td>LANL, U Utah</td>
</tr>
</tbody>
</table>

General contact point (PIs and technical leads):
tascs-leads@cca-forum.org
Past, Present, Future

• The Common Component Architecture (CCA) is a community effort, with SciDAC projects in the vanguard

• CCTTSS (SciDAC1) brought the CCA from idea to prototype stage
  – Core ideas well-developed
  – Useful implementations, but not polished
  – Numerous users, but not “standard”

• TASCS is bringing the CCA from prototype to production for general computational science users
  – Robustness of tools
  – Usability of components and component environment
  – Developing an ecosystem (component toolkit) around CCA
  – Providing new features and capabilities for scientific software developers via the component environment

• Still many open issues and opportunities beyond those TASCS is addressing
  – Must gather experience before we can determine appropriate abstractions
We’re Different
(from Most SciDAC Projects)

• We’re trying to fundamentally change the way scientific software is developed and used

• Current “culture” of computational science emphasizes short-term heroism over long-term sustainability
  – CCA impact on performance, simulation size is second-order

• CCA (and most software engineering) requires a significant initial investment (intellectual and in software)
  – Longer-term payback
  – Significant social and sociological issues

• Groups choosing to make the investment in CCA tend not to be the ones issuing press releases for the fastest/biggest/*/est
  – PR challenge
CCA Toolkit: Making it easier to create components, and making available a suite of real, useful components

User Outreach & Application Support: Broaden awareness and adoption of component technology and the CCA

Component Technology Initiatives: How can we exploit the component environment to provide computational scientists with better ways to develop their software?
# Institutional Involvement

<table>
<thead>
<tr>
<th>Initiative</th>
<th>ANL</th>
<th>BU</th>
<th>IU</th>
<th>LLNL</th>
<th>ORNL</th>
<th>PNNL</th>
<th>SNL</th>
<th>Tech-X</th>
<th>UM</th>
<th>UO</th>
<th>VSU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiatives</td>
<td>C</td>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HPC</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQV</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CQoS</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td>3</td>
<td>3</td>
<td>C</td>
<td>3</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enhancements</td>
<td>3</td>
<td></td>
<td>C</td>
<td>3</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Usability</td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toolkit</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>C</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Outreach</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>C</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

C = Coordinator  3 = Participant
CCA Environment Activities

Supporting and improving the foundation of the CCA environment

• **Core Tool Support and Maintenance**
  – Keeping the core software tools running in the face of change

• **Enhancements**
  – Extending the CCA environment (specification, core tools) with additional features/capabilities required by customers and other activities in TASCs

• **Usability**
  – Making HPC component technology more accessible to users through automatic wrapping, testing tools, better debugging support
CCA Tool Chain

- **Onramp**
  - (Semi-) automatic componentization of legacy code
  - Under development

- **Bocca**
  - Component development tool ("command line IDE")
  - Initial release, additional development planned

- **Babel**
  - Language interoperability & component middleware
  - In production

- **CCA-Spec-Babel**
  - Implementation of CCA spec functionality using Babel
  - In production

- **Ccaffeine**
  - HPC framework with optional GUI
  - In production
Environment: Core Tool Support & Maintenance

Keeping the core software tools running in the face of change

- Helpdesk and bug tracking
- Technical documentation
- Ports for high-end and other platforms
- Continuous integration and regression testing for tools
- Develop CCA (specification) conformance tests
- Automated conformance testing
Environment: Support
Recent Accomplishments

• Helpdesk and bug tracking implemented on all tools and tutorial
• Extensive expansion of documentation in tutorial format
• Fully ported to commodity Linux, Mac
• All tools working on Cray, build still “special”
• Partially ported to IBM Blue Gene
• General “static application generator” nearly complete
  – Important on systems that don’t support dynamic linking
Environment: Enhancements

Extending the CCA environment (specification, core tools) with additional features and capabilities required by customers and other activities in TASCs

- Filling out the CCA specification with crucial new services
- Building bridges to other related technologies (e.g., Kepler)
- Interoperability and integration of CCA foundational tools
Environment: Enhancements
Recent Accomplishments

• Establish CCA Review Board (CCARB) to ensure rigorous multi-stage specification process

• MPIService & EventServices are nearing CCARB approval

• Babel gains initial F2003 support and multi-language structs in C, C++, Python, & F2003

• Demonstrated CCA/Kepler interoperability (CBHPC 2008)

• Prototype of BabelRMI using Proteus

• New stable Babel 1.4.0 release
Environment: Usability

Making HPC component technology more accessible to users through automatic wrapping, testing tools, better debugging support

• CCA Onramp
  – (Semi-) automatic wrapping of existing code into the CCA environment based on source code annotations
  – Allows “mixed use” of code in both original form and as components
  – Refactor application into componentized domain-specific framework

• Debugging and testing
  – CCA abstractions and capabilities simplify designing, developing software but can make it harder to debug, test
  – Capturing and documenting multi-language debugging approaches, developing testing tools
CCA Onramp

- Source code annotations to specify componentization (components, interfaces)
- Generate SIDL for interfaces, components
- Use bocca to generate component skeleton
- Generate code to adapt between existing code and Babel bindings
Environment: Usability
Recent Accomplishments

• Onramp created and released for C-based codes.
  – Fortran and C++ support in near-term plan.

• Source annotation specification draft available for public comment and contribution.
  – Reference implementation in Onramp tools.

• Distribution of Onramp through SciDAC outreach site.
  – As of October 2008, third most downloaded project

• New versions of PDToolkit static source analysis tools to support CCA efforts (e.g. new query interface)

• Evaluation of automated build and test harnesses (e.g. CruiseControl)

• Establishment of routine tools testing on Cray
CCA Toolkit

Making it easier to create components, and making available a suite of real, useful components

- Provides instant application prototyping
  - Scientists can try out ideas quickly in a parallel HPC setting.
  - Develop those ideas incrementally, concentrating on particular components to increase performance or fidelity

- Provide tools to make componentry easy
  - Bocca automatically generates a CCA component template as easy as giving them a name
Simple Bocca Example

# create an empty but buildable CCA skeleton
bocca create project myproj
cd myproj
./configure

bocca create port myJob
bocca create component myWorker –provides=myJob:job1

# fill in public functionality
bocca edit port myJob

# fill in implementation
bocca edit component –i myWorker

# compile application
make
CCA Toolkit
Recent Accomplishments

• Components ready to use for:
  – Linear Algebra, Equation Solvers, Optimization, MPI
    • PETSc, SPARSEKIT2, and more
  – Parallel decomposed meshes

• More components being packaged to meet Toolkit requirements

• Initial release of bocca
  – Incorporated into tutorial
  – Significant productivity gains
Component Technology Initiatives

How can we exploit the component environment to provide computational scientists with better ways to develop their software?

• **Computational Quality of Service**
  – Adaptation of running component applications in response to changing conditions (performance, accuracy, etc.)

• **Software Quality and Verification**
  – Extend component interface definitions to express contracts, extra-functional characteristics, with automatic verification

• **Support for Emerging HPC Hardware and Software Paradigms**
  – Use CCA interfaces to manage MCMD applications on massively parallel systems
  – Integrate software on hardware accelerators into CCA applications
Initiatives: Computational Quality of Service (CQoS)

Adaptation of running component applications in response to changing conditions (performance, accuracy, etc.)

Collaborators: Teaming with TASCS Software Quality and Verification (SQV) Initiative, University of Oregon TAU team, PERI, TOPS. Close interactions with SciDAC scientific applications teams to motivate and validate this work:

- **Combustion:** parallel mesh partitioning in combustion
  - (BES) CFRTS SciDAC project (H. Najm PI)

- **Quantum Chemistry:** resource management in quantum chemistry
  - (BES) SciDAC project (M. Gordon PI)

- **Fusion and Accelerators:** efficient, robust, and scalable linear solvers (with TOPS)
  - (FES) FACETS SciDAC project (J. Cary PI) - Newton-Kryov solvers in parallel edge plasma simulations
  - (HEP) COMPASS SciDAC accelerator project (P. Spentzouris PI) - solvers in beam dynamics simulations
Initiatives: CQoS Approach

- **Analysis**: performance monitoring, problem/solution characterization, and performance model building
- **Control**: Interpretation and execution of control laws to modify an application’s behavior
- **CQoS database components**: Manage interactions with performance and runtime databases to facilitate analysis and decision making
- **Leverage external tools** where appropriate, e.g., Anamod (Eijkhout et al.)
Initiatives: CQoS
Recent Accomplishments

• Collected requirements from four motivating applications
  – Initial implementation focus: component composition and configuration
    for quantum chemistry and combustion
    • See, e.g., http://wiki.mcs.anl.gov/cqos/index.php/Quantum_Chemistry_CQoS_Activities
• Designed initial CQoS API and software suite testbed
  – Initial API focus on CQoS database query and management
  – Testbed applications for parallel nonlinear solvers, mesh partitioning in
    combustion, quantum chemistry
• Implemented performance database query
  and management components
• Determined overall strategy for CQoS software
• Developed prototype models and control laws
  – Preliminary version for combustion, in progress for quantum chemistry
• In progress: running base performance experiments
Initiatives: Software Quality and Verification (SQV)

Extend component interface definitions to express contracts, extra-functional characteristics, with automatic verification

- **Collaborators:** Working in conjunction with CQoS activity

- **Approach**
  - Identify and define relevant component characteristics and constraints
  - Pursue suitable semantic representations leveraging work on interface contract enforcement capabilities integrated into Babel middleware toolkit
Enforcable Contracts for SIDL Interfaces

- Current component architectures define syntax of interfaces
- Extend interface to include semantics (behavior) for more complete definition
  - “Design by contract”
  - Help ensure component performs correctly
  - Help ensure component is used correctly
- Selective enforcement to control impact

```plaintext
package vector version 1.0 {
  class Utils {
    static double norm(in array<double> u,
                        in double tol,
                        in int badLevel)
    require /* Preconditions */
       not_null : u != null;
       u_is_1d : dimen(u) == 1;
       non_negative_tolerance : tol >= 0.0;
    ensure /* Postconditions */
       no_side_effects : is pure;
       non_negative_result : result >= 0.0;
       nearEqual(result, 0.0, tol)
       iff isZero(u, tol);
  }
}
```
Initiatives: SQV Recent Accomplishments

• Integrated refactored version of latest prototype enforcement approach into the Babel source code repository

• Developed example implementations of a specification with contracts in C++, FORTRAN 77, Fortran 90, Java, and Python

• Developed example client for the above implementations in C++

• Demonstrated in regression tests for Babel Release 1.4.0:
  – C or C++ client → C, C++, FORTRAN 77, Fortran 90, Java, and Python implementations
Enable effective use of CCA on emerging HPC architectures

- **Multiple-Component-Multiple-Data (MCMD) CCA Technology** to support apps based on multiple levels of parallelism for massively parallel systems
  - Motivated by multiple apps: chemistry, climate, materials modeling, fusion, etc.
  - Used in NWChem

- Simplify use of systems with **heterogenous** node architectures (e.g., IBM RoadRunner)
  - Couple/swap component codes on accelerators and general purpose CPUs
  - Bioinformatics/proteomics applications in EMSL Mass Spectrometry facility

- **Fault Awareness** in CCA, in collaboration with CIFTS
  - Bridge between CIFTS Fault Tolerance Backplane (FTB) and CCA Event Service
  - SWIM (fusion) MCMD simulations as initial model application
Multiple-Component Multiple-Data Use Cases

NWChem (chemistry)

Co-Op (materials)

SWIM (fusion)
Initiatives: HPC
Recent Accomplishments

• **Multiple-Component-Multiple-Data (MCMD) CCA Technology**
  – Developed processor teams specs and prototype implementation
  – The initial scope extended to support coupling independent jobs in subsurface simulations e.g., hydrology and geophysics

• **Simplify use of systems with heterogenous node architectures**
  – High performance event service developed and demonstrated
  – Coupled component codes on accelerators and general purpose CPUs (e.g. Polygraph on FPGAs)

• **Fault awareness**
  – Prototype fusion demonstration application nearing completion
User Outreach & Application Support

Broaden awareness and adoption of component technology and the CCA

• Application support
  – In tandem with Environment Support activity for tools

• User outreach and support
  – Tutorials, coding camps, etc

• Community outreach
  – Papers, presentations, workshops, collaboration server, etc.
How We Work with Others

- **R&D Partners**
  - Motivation and test beds for specific TASCS R&D activities
  - TASCS support built into *Component Technology Initiatives*

- **Collaborators**
  - Jointly funded w/ partner projects through *Outreach*

- **“Walk-Up” Users**
  - Modest support through *Outreach*

- Help Desk supported through *Environment & Outreach*

- Designated points of contact (often “embedded”), frequent interactions, coding camps, site visits, etc.

- **CCA Forum** ([www.cca-forum.org](http://www.cca-forum.org))
  - CCA standards body and user community

- Tutorials, presentations, publications, workshops, etc.
Outreach: Recent Accomplishments

- Many collaborative interactions (23+)
  - See http://tascs-scidac.org/collaborators
- 46+ papers to date (presentations not yet cataloged)
- 6 tutorials
  - ACTS Collection Workshop (annual), Supercomputing 2007, SciDAC 2007, Para’08
- 5 workshops organized
  - HPC-GECO/CompFrame (’06, ’07), CBHPC’08, Para’08, CBC’08
- Host majority of CCA Forum meetings
- Established separate tascs-scidac.org web site
- cca-forum.org collaboration services activities in progress
  - Migration from LBNL to ORNL, Improved web site, EPrints server about to go into production
Major New Users and Future Opportunities

New Users
- GWACCAMOLE (groundwater)
- FACETS (fusion)
- COMPASS (accelerator physics)
- Community Surface Dynamics Modeling System (CSDMS)

Future Opportunities
- GNEP Nuclear Energy Advanced Modeling & Simulation (NEAMS)
- DoD CREATE
- HPC Application Software Consortium (HPC-ASC)
Exascale Challenges

• Code complexity!

• Multiphysics code coupling

• Massive parallelism

• Composition of parallel {components, libraries, etc.}

• Exposure and use of extra-functional properties of software components
  – Performance, parallelism, contracts, etc.

• TASCS has some relevant R&D activities and is working with relevant applications

• TASCS will not fully “solve” these problems, but we’re gaining experience
Summary

• TASCS is part of the community developing and using the Common Component Architecture

• Long-range focus on improving the ability to develop, manage, and use increasingly complex high-performance scientific software

• Broad R&D effort
  – Production quality tools
  – Making componentry more accessible
  – Technology initiatives on new capabilities for software developers

• Engaged with many applications, extensive broader outreach