A Study of the Common Component Architecture (CCA) Forum Software

Daniel S. Katz, E. Robert Tisdale, Charles D. Norton

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

A growing problem in the development of large-scale multi-disciplinary scientific applications for high-performance computers is managing the interaction between portions of the application developed by different groups, possibly at different periods if code-reuse is desired. In the business world, component-based software engineering has been proposed as a solution. These technologies, including Microsoft's Component Object Model (COM) [1,2] and Sun's (Enterprise) JavaBeans (EJB) [3,4], may not be appropriate for scientific computing. To examine this issue, the Common Component Architecture (CCA) Forum [5,6] was formed.

The CCA Forum is developing a component architecture specification to address the unique challenges of high-performance scientific computing, with emphasis on scalable parallel computations that use possibly distributed resources. In addition to developing this specification, a reference framework, various components, and supplementary infrastructure, the CCA Forum is collaborating with practitioners in the high-performance computing community to design suites of domain-specific abstract component interface specifications. [7] The CCA methodology at its simplest requires a component-writer to add one call to the package being componentized. This call is a declaration of the ports the component supplies, and the ports that the component uses. These ports are the interfaces that are used for interaction with other components. The example applications in this presentation include port definitions, and this will be discussed in the presentation.

NASA's ESTO-CT (the Earth Science Technology Office's Computation Technologies) project has so far been successful in achieving its goal of "*Demonstrating the power of high-end, scalable, and cost-effective computing environments to further our understanding and ability to predict the dynamic interaction of physical, chemical, and biological processes affecting the Earth, the solar-terrestrial environment, and the universe*" [8]. However, the impact on software development for most scientists in the broader community has been limited. The software developed by the grand challenge teams was targeted for a specific application and implemented to achieve specific objectives. The ability to extend functionality and foster reusability is buried within the collective knowledge of small teams. In the long term this will hamper progress toward supporting and sustaining a development program to produce new science results through community collaboration. This was recognized by NASA management, and the project's current work emphasizes frameworks and interoperability for large-scale high performance scientific software development and maintenance. This is an appropriate focus as the coupling of existing and newly developed codes represents the next major milestone needed to advance the project plan goal statement. The material in this presentation is being developed as a part of the ongoing study of the CCA Forum's technology by the ESTO-CT project.

The contributions of this presentation will be qualitative and quantitative examinations of the CCA software as applied to two examples. Both examples include unstructured adaptive mesh refinement (AMR). The reason for this choice is that AMR libraries are used by most of the ESTO CT applications, which are generally physical, grid- or mesh-based simulations. The first example uses a single processor. The process of modifying existing Fortran 90 code consisting of a driver routine and an AMR library [9] into two components is described in detail. The performance of the original application, and the componentized version are measured and compared. This will show that there is some amount of start-up cost associated with using the component version, and that calls from one component to another have a very small increase in time, comparable to an additional function call being used to wrap a function. The second example will involve parallel components, and will again discuss the procedure used to transform the code into components, as well as comparing the performance of the two versions. In this example, the dynamic data redistribution inherent in the AMR routines will cause a fair bit of communication. As of the date this abstract is being written, the timing comparisons for the parallel code have not yet been made, but they are expected to show relatively close timings.

After attending the presentation discussed herein, the programming and performance implications of using the Common Component Architecture should be clear. While the examples provided do not fall in the realm of embedded computing, they are certainly relevant to high-performance computing, whether embedded or not. The lessons discussed in this presentation should enable a listener to decide if the CCA is an appropriate path for a future project.

[1] Microsoft COM Web page, see http:///www.microsoft.com/com/about.asp.

[2] R. Sessions, COMand DCOM: Microsoft's Vision for Distributed Objects, John Wiley & Sons, 1997.

[3] R. Englander, Developing Java Beans, O'Reilly, 1997.

[4] R. Monson-Haefel, Enterprise JavaBeans, O'Reilly, 1999.

[5] R. Armstrong, D. Gannon, A. Geist, K. Keahey, S. Kohn, L. C. McInnes, S. Parker, B. Smolinski, "Toward a Common Component Architecture for High-Performance Scientific Computing," *Proceedings of High Performance Distributed Computing*, pp. 115–124,1999.

[6] Common Component Architecture Forum, see http://www.cca-forum.org/.

[7] B. Norris, S. Balay, S. Benson, L. Freitag, P. Hovland, L. McInnes, and B. Smith, "Parallel Components for PDEs and Optimization: Some Issues and Experiences," in review as an invited paper in a special issue of *Parallel Computing*, Feb. 2002.

[8] ESS Project Plan, June 2000 (Project homepage: http://ct.gsfc.nasa.gov/)

[9] Charles D. Norton, John Z. Lou, and Thomas Cwik, "Status and Directions for the PYRAMID Parallel Unstructured AMR Library," in 8th Intl. Workshop on Solving Irregularly Structured Problems in Parallel (15th IPDPS), San Francisco, CA 2001.