The Advanced Computational Testing and Simulation Toolkit (ACTS):

What can ACTS do for you?

L. A. Drummond[1] and O. A. Marques[2]
Lawrence Berkeley National Laboratory
National Energy Scientific Computing Center (NERSC)
One cyclotron Road, Berkeley, CA 94720
http://acts.nersc.gov, acts-support@nersc.gov

Abstract

During the past decades there has been a continuous growth in the number of physical and societal problems that have been successfully studied and solved by means of computational modeling and simulation. Distinctively, a number of these are important scientific problems ranging in scale from the atomic to the cosmic. For example, ionization is a phenomenon as ubiquitous in modern society as the glow of fluorescent lights and the etching on silicon computer chips; but it was not until 1999 that researchers finally achieved a complete numerical solution to the simplest example of ionization, the collision of a hydrogen atom with an electron. On the opposite scale, cosmologists have long wondered whether the expansion of the Universe, which began with the Big Bang, would ever reverse itself, ending the Universe in a Big Crunch. In 2000, analysis of new measurements of the cosmic microwave background radiation showed that the geometry of the Universe is flat, and thus the Universe will continue expanding forever.

Both of these discoveries depended on high performance computer simulations that utilized computational tools included in the Advanced Computational Testing and Simulation (ACTS) Toolkit. The ACTS Toolkit is an umbrella project that brought together a number of general-purpose computational tool development projects funded and supported by the U.S. Department of Energy (DOE). These tools, which have been developed independently, mainly at DOE laboratories, make it easier for scientific code developers to write high performance applications for parallel computers. They tackle a number of computational issues that are common to a large number of scientific applications, mainly implementation of numerical algorithms, and support for code development, execution and optimization. The ACTS Toolkit Project enables the use of these tools by a much wider community of computational scientists, and promotes code portability, reusability, reduction of duplicate efforts, and tool maturity. This paper presents a brief introduction to the functionality available in ACTS.

1.- Introduction.

The ACTS Toolkit is a set of computational tools developed primarily at DOE laboratories and is aimed at simplifying the solution of common and important computational problems. These tools are freely available with minor licensing agreements for their use and distribution. The use of the tools reduces the development time for new codes and the tools provide functionality that might not otherwise be available. All this potential cannot be achieved, however, if the tools are not used effectively or not used at all.

[1] LADrummond@lbl.gov
[2] OAMarques@lbl.gov

The ACTS Project brings together software development from DOE laboratories, and in some cases in collaboration with universities, to the hands of a scientific community that is broader than the community in a single DOE laboratory. Consequently, ACTS has promoted reusability over duplication of efforts and tool interoperability over "hard-wired" and intrusive library interfaces that only serve a specific project. By analogy with a commercial company, ACTS has put the formerly uncoordinated research and development performed at various places into one distribution channel. ACTS complements DOE research and development efforts by adding technical support, quality assurance and marketing.

ACTS is bringing tools to higher acceptance levels among computational scientists and institutions. This provides the necessary means for individual tool projects to interact with more users, so the tools can gradually mature, becoming both more robust and portable to state-of-the-art high performance computing environments.

ACTS Tools are mostly library with interfaces to programs written in Fortran, C and C++. They are primarily designed to run on high performance computing architectures. Portability and performance are both considerations in their design and adaptability to emerging computer technologies. At the National Energy Research Scientific Computing Center (NERSC), we are currently providing a comprehensive support to the users of tools under ACTS. This support includes training and education, help prototyping scientific and engineering codes, and independent software evaluation of the tools.

This paper presents a short introduction to the ACTS Toolkit. For more materials and references the reader is advised to visit the ACTS Information Center [4]. In Section 2, we present a list of the tools currently available in the ACTS Toolkit. A few examples of successful implementations of scientific simulations using the ACTS Toolkit are presented in Section 3. In the last two sections, 4 and 5, we summarize lessons learned through ACTS support, and future plans.

2.- Available Functionalities in the Toolkit.

The following is a list of tools currently available in the ACTS Toolkit [5]. These tools provide solutions to some numerical problems, scientific data representation, data manipulation, as well as support for code execution, library tuning and interoperability. We categorize them by their functionality in 5 different groups, Numerical Tools, Tools That Support Code Development, Tools That Support Code Execution, Tools That Support Library Development, and Interoperability.

2.1. Numerical Tools

- ***Aztec*** (developed at Sandia National Laboratories, SNL, [5], [6], [7]) is a library that provides algorithms for the iterative solution of large sparse linear systems arising in scientific and engineering applications. It is a stand-alone package comprising a set of

iterative solvers, preconditioners and matrix-vector multiplication routines. Users are not required to provide their own matrix-vector multiplication routines or preconditioners in order to solve a linear system. The Aztec library is written in C and is also callable from Fortran. It is portable to most parallel platforms since it uses MPI to perform data communication. Overall, the package was designed to be easy to use. The user may input the linear system in a simple format and Aztec will perform the necessary transformations for the matrix-vector multiplication and preconditioning. After the transformations, the iterative solvers can run efficiently. If the input matrix is suitably partitioned, the efficiency can be further enhanced.

Aztec in now in the process of being superseded by the Trilinos solver framework [8]. Aztec's main strength is that it is a small and stand-alone package that contains all the components needed for the solution of a real sparse linear system of equations without requiring matrix-vector multiplications or preconditioning routines from the user. The most commonly used schemes to solve a large linear system on a distributed parallel machine include four major components: a mechanism to express the linear system, a matrix-vector multiplication routine, a preconditioner routine, and an iterative method to compute the solution by using the matrix-vector multiplication routine and the preconditioner. Aztec has carefully implemented all these components and overall exhibits good parallel efficiencies on many test problems.

- ***Hypre*** (developed at Lawrence Livermore National Lab, LLNL, [5], [9]) is a library for solving large, sparse linear systems of equations on massively parallel computers. The main features of this library are: scalable preconditioners, implementation of a suit of common iterative methods (these include Conjugate Gradient and GMRES for symmetric and unsymmetric matrices, respectively), intuitive grid-centric interfaces, and dynamic configuration of parameters. Hypre works for users with different levels of expertise and has user-defined interfaces for multiple languages.

- ***OPT++*** (developed at SNL [5]) is an object-oriented nonlinear optimization package for serial architectures. It solves optimization problems of the form $\min_{x \in R^n} f(x), h_i(x) = 0, i = 1,..., p, g_j(x) \geq 0, j = 1,..., m,$ in which the user specifies the function f (and, when available, its first and second analytical derivatives) and the functions h and g. OPT++ provides four solution algorithms: a Newton method, a finite-difference Newton method, a Quasi-Newton method, and a nonlinear conjugate gradient method.

- ***PETSc***, the Portable, Extensible Toolkit for Scientific computation (developed at Argonne National Laboratory, [5],[10],[11],[12]), provides sets of tools for the parallel (as well as serial), numerical solution of PDEs that require solving large-scale, sparse linear and nonlinear systems of equations. PETSc includes nonlinear and linear equation solvers that employ a variety of Newton techniques and Krylov subspace methods. PETSc provides several parallel sparse matrix formats, including compressed row, block compressed row, and block diagonal storage.

PETSc is one of the most popular and matured tools currently available in the ACTS Toolkit. PETSc has a growing number of application users and development projects that re-use some of the functionality available in PETSc (like TAO and SLEPc [13]). The PETSc team has long worked on the development of interoperable interfaces with other ACTS tools, and perhaps has implemented the most tool-to-tool interoperability interfaces.

- ***PVODE*** (developed at LLNL, [5]) actually refers to a trio of closely related solvers: PVODE, for systems of ordinary differential equations, KINSOL, for systems of nonlinear algebraic equations, and IDA, for systems of differential-algebraic equations. These solvers have some modules in common, primarily a module of vector kernels, and a generic linear system solver based on a scaled preconditioned GMRES method. PVODE is a solver for large systems of ordinary differential equations on parallel machines. It contains methods for the solution of both stiff and non-stiff initial value problems. Integration methods include the variable coefficient forms of the Adams and backward differentiation formula methods. The linear systems that must be solved during the implicit time stepping are solved with iterative, preconditioned Krylov solvers. The user can either supply a preconditioner or use one that is included in the PVODE package. PVODE is an extension of the sequential package known as CVODE, which has been widely distributed and used [14].

  The PVODE trio is most useful for solving large differential and nonlinear algebraic systems that arise in a variety of applications. Important DOE applications include chemical kinetics, atmospheric chemistry, semiconductors, and structural or mechanical systems.

- ***ScaLAPACK*** (developed at the University of Tennessee, Knoxville; University of California, Berkeley; and ORNL, [5], [15]) is a library of high performance linear algebra routines for distributed-memory message-passing MIMD computers and networks of workstations supporting PVM or MPI. It is a continuation of the LAPACK project [16], which designed and produced analogous software for workstations, vector supercomputers, and shared-memory parallel computers. The goals of both projects are efficiency (to run as fast as possible), scalability (as the problem size and number of processors grow), reliability (including error bounds), portability (across all important parallel machines), flexibility (so users can construct new routines from well-designed parts), and ease of use (by making the interface to LAPACK and ScaLAPACK look as similar as possible). The ScaLAPACK library contains routines for solving systems of linear equations, least squares, eigenvalue and singular value problems. They can also handle many associated computations such as matrix factorizations or estimation of condition numbers.

  ScaLAPACK is intended for use in large-scale applications that require numerical manipulation of large dense or band matrices. Within NERSC's user community, we have seen various routines from ScaLAPACK being used. The applications using ScaLAPACK include material sciences, computational chemistry and astrophysics.

- ***SuperLU*** (developed at the University of California, Berkeley; and NERSC, [5]) is a general-purpose library for the direct solution of large, sparse, nonsymmetric systems of linear equations on high performance machines. The library is written in C and is callable from either C or Fortran. The library routines perform an LU decomposition with numerical pivoting and triangular system solves through forward and back substitution. The LU factorization routines can handle non-square matrices but the triangular solves are performed only for square matrices. The matrix columns may be preordered (before factorization) either through library or user-supplied routines. This preordering for sparsity is completely separate from the factorization. Working precision iterative refinement subroutines are provided for improved backward stability. Routines are also provided to equilibrate the system, estimate the condition number, calculate the relative backward error, and estimate error bounds for the refined solutions.

  The developers of SuperLU have participated in many of the activities organized by the ACTS Toolkit Project and have provided us with feedback for the ACTS Information Center. SuperLU is a general-purpose software with a wide range of potential end users.

- ***TAO***, the Toolkit for Advanced Optimization (developed at ANL, [5]), focuses on large-scale optimization software, including nonlinear least squares, unconstrained minimization, bound constrained optimization, and general nonlinear optimization. There are a variety of software tools for solving the aforementioned problems; however, only TAO offers an Object-Oriented solution that provides a flexible optimization toolkit capable of addressing issues of portability, versatility and scalability in many computational environments. The algorithms in TAO place strong emphasis on the reuse of external tools where appropriate. TAO's design enables bidirectional connection to lower-level linear algebra support (such as parallel sparse matrix data structures) that is available in toolkits like PETSc.

  TAO is rapidly acquiring a growing community of users and serves as a good example of quality software being developed on top of existing reliable and robust software.

2.2. Tools That Support Code Development

- ***Global Arrays*** (GA, developed at PNNL, [5],[17]) is a library for writing parallel programs that use large arrays distributed across processing nodes and that offers a shared-memory view of distributed arrays without destroying their NUMA characteristics. The library has both Fortran and C interfaces. Originally developed to support arrays as vectors and matrices (one or two dimensions), it currently supports up to seven dimensions in Fortran and even more in C. GA offers two types of operations: collective operations (require participation and synchronization of all processes) and local operations (may be invoked independently by all processes). GA

also comes with a visualizer that uses trace files to animate array access patterns. Its main purpose is to analyze the impact of distribution on performance.

GA is meant to complement rather than replace the message-passing model, and is interoperable with MPI. GA was originally developed for use in the NWChem computational chemistry package [18], but has also been used in other chemistry packages such as GAMESS-UK, Columbus, Molpro, Molcas, and SOCI.

- ***Overture*** (developed at LLNL, [5]) is a set of Object-Oriented tools for solving computational fluid dynamics and combustion problems in complex moving geometries. It has been designed for solving problems on a structured grid or a collection of structured grids. It can use curvilinear grids, adaptive mesh refinement, and the composite overlapping grid method to represent problems involving complex domains with moving components. Overture programs are written at a very high level, using data-parallel array expressions in the style of High Performance Fortran (HPF). They can achieve high performance (comparable to Fortran) thanks to a preprocessor (C++ to C++) called ROSE. Effectively, ROSE is a replacement for the expression template technique of POOMA. Overture has aggregate array operations and tightly integrated graphical features based on OpenGL. AMR++, a package that directly supports adaptive mesh refinement methods, is built on top of Overture.

  In October 2001 the Overture Project was named one of the 101 most important discoveries supported by the DOE Office of Science in the past 25 years.

2.3. Tools That Support Code Execution

- ***CUMULVS*** (Collaborative User Migration, User Library for Visualization and Steering, developed at ORNL, [5],[21]) is a software framework that enables programmers to incorporate fault-tolerance, interactive visualization and computational steering into existing parallel programs. The CUMULVS software consists of two libraries, one for the application program, and one for the visualization and steering front-end (called the "viewer"). CUMULVS handles collecting and transferring distributed data fields to the viewers and oversees adjustments to steering parameters in the application. It also manages the dynamic attachment and detachment of multiple independent viewers to a running parallel application. In addition, CUMULVS provides a user-directed checkpoint/restart mechanism to enable users to integrate fault tolerance to a running parallel application.

  Among others, CUMULVS is very effective for monitoring and steering remote parallel execution and allowing the construction of attractive user interfaces that have greatly helped users that are not familiar with all the particulars of a given code and/or the complexities involved in parallel computing.

- ***Globus*** ([5],[19],20) provides a bag of services for the creation of computational Grids and tools with which applications can be developed to access the Grid. Globus

itself provides the following core services: communication services, resource allocation and process management. It also provides a standard interface to local resource management systems, allowing computational Grid tools and applications to express resource allocation and process management requests in terms of a standard application programming interface (API) and protocol, without constraining individual sites to any specific resource management tool. A standard protocol (LDAP, the Lightweight Directory Access Protocol) allows for discovering, publishing, and accessing information about the configuration and status of the computational Grid.

Globus developers were awarded the 1997 Global Information Infrastructure (GII) Next Generation Award for their work in advancing the technology and application of high performance distributed computing. The Globus technology is now used worldwide and it has provided a leading infrastructure for the deployment of and collaborations around Grid technology. There are three main kinds of users of this technology: the developers of Grid services, Grid-enabled application developers and users of the applications. Nowadays, all three kinds of users have to rely on their institutional agreement and provisions to work with the Globus technology, therefore the current support to Globus users requires a larger orchestration of personnel and resources than any of the other ACTS tools.

- ***PAWS*** (Parallel Application Work Space, developed at LANL National Laboratories, [5]) provides a framework for coupling parallel applications within a component-like model. Central to the design of PAWS is the coupling of parallel applications using parallel communication channels. The coupled applications can be run on different machines, and the data structures in each coupled component can have different parallel distributions. PAWS is able to carry out the communication without having to resort to global gather/scatter operations. Instead, point-to-point transfers are performed in which each node sends segments of data to remote nodes directly and in parallel.

  With the abrupt urge to couple multi-disciplinary codes there are high and increasing demands for tools like PAWS. The tool is currently under consideration by several research groups and we foresee the integration of functionalities available in other coupling efforts and PAWS through the use of component technology.

- ***SILOON*** (Scripting Interface Languages for Object-Oriented Numerics, developed at LANL National Laboratories, [5]) gives scientists the ability to rapidly prototype and solve problems on high performance parallel computers. SILOON provides tools and run-time support for building easy-to-use external interfaces to existing numerical codes. The developers hope to enable scientists and other application programmers to easily access existing object-oriented scientific frameworks and numerical libraries written in C, C++, and Fortran. SILOON is a middleware package that may be used with the most popular scripting languages: Perl, TCL, and Python. It generates “glue code” to bind external library interfaces and provide run-time support and a

computational server that is controllable under the scripting languages of choice. The first versions of SILOON were targeted for the POOMA and Overture frameworks.

- ***TAU*** (Tuning and Analysis Utilities, developed at the University of Oregon and LANL National Laboratories, [5],[22]) is a set of tools for analyzing the performance of C, C++, Fortran and Java programs. It is normally used in three steps: (1) instrument the program by inserting TAU macros into the program (this can be done automatically for C++ programs); (2) run the program, a trace file is then automatically generated; and (3) view the trace file with a TAU visualizer (RACY) or a third-party visualizer (such as VAMPIR). TAU collects much more information than what is available through *prof* or *gprof*, the standard Unix utilities. Also available through TAU are: per-process, per-thread and per-host information (supports *pthreads*), inclusive and exclusive function times, profiling groups that allow you to organize data collection, access to hardware counters on some systems, per-class and per-instance information, separate data for each template instantiation, start/stop timers for profiling arbitrary sections of code, and support for collection of statistics on user-defined events.

  TAU can be used by anyone writing a C, C++, Fortran or Java application and that wants to understand where the performance bottlenecks are. The hurdles of inserting profiling functions is minimized by two factors: only a single statement is needed to profile a function, and profiling can be disabled completely so that profiling insertions don't need to be taken out. Users of TAU have been enthusiastic about its usefulness.

2.4. Tools That Support Library Development

- ***ATLAS*** (Automatically Tuned Linear Algebra Software, developed at the University of Tennessee, Knoxville, [5]) and ***PHiPAC*** (Portable High Performance ANSIC, developed at the University of California, Berkeley, and the International Computer Science Institute) are tools for the automatic generation of optimized numerical software for modern computer architectures and compilers. These tools have both initially focused on level-three BLAS operations (matrix-matrix multiplications) and also a few routines from LAPACK, which have high potential for optimization. Traditionally, the optimization of these routines has been a tedious, architecture-dependent, hand coding process. Codes automatically generated by ATLAS and PHiPAC have been able to meet and even exceed the performance of the vendor-supplied, hand-optimized BLAS on a range of platforms. ATLAS and PHiPAC achieve performance through loop unrolling, explicit removal of unnecessary dependencies in code blocks, and use of machine-sympathetic C constructs. Code generators are parameterized and scripts are used to find the optimal choice of parameters for a given architecture and compiler. Research continues on these tools to extend them to new BLAS 3 operations and other basic linear algebra calculations, including sparse operations.

Our comments: ATLAS and PHiPAC are of primary interest to computer architects and administrators and are not tools that the application programmer would use, although the technology that they are enabling may well eventually spill over into tools for application programmers.

- ***PADRE*** (Parallel Asynchronous Data and Routing Engine, developed at LANL, [5]) is a C++ layer for interfacing with libraries that distribute data on parallel computers. PADRE attempts to provide a uniform interface to several parallel decomposition libraries (KeLP, Multiblock PARTI, etc.) and the communication mechanisms used by them.

- ***PETE*** (Portable Expression Template Engine, developed at LANL, [5]) is an extensible implementation of the expression template technique (C++ technique for passing expressions as function arguments **Error! Reference source not found.**). This technique uses C++ recursively-defined templates for transforming certain kinds of C++ statements into other statements with the same effect but higher performance. Evaluating array expressions in a single loop is of fundamental importance for performance, for example. Using simple C++ overloading it is difficult (practically impossible) to obtain this kind of translation. Furthermore, PETE has the advantage that the entire transformation is done by the C++ compiler, without requiring separate tools.

2.5. Interoperability

The Objective of the Common Component Architecture Forum (CCA Forum,[23],[24]) is to define a minimal set of standard features that a High-Performance Component Framework has to provide, or can expect, in order to be able to use components developed within different frameworks. Such standard will promote interoperability between components developed by different teams across different institutions. The Center for Component Technology for Terascale Simulation Software (CCTTSS) is dedicated to the development of a component-based software development model suitable for the needs of high-performance scientific simulation, particularly the CCA. The Center is funded by the U. S. Dept. of Energy (DOE) as an Integrated Software Infrastructure Center (ISIC) under the Scientific Discovery through Advanced Computing (SciDAC, [25]) program and includes members from Argonne, Livermore, Los Alamos, Oak Ridge, Pacific Northwest, and Sandia National Laboratories, Indiana University and the University of Utah.

3.- Some Highlights in High-Performance.

A number of high performance scientific applications are already using the ACTS tools. Here we summarize a few examples. For some cases, the results achieved could not have been realized otherwise.

- Collaborators at the Lawrence Berkeley National Laboratory (LBNL), Lawrence Livermore National Laboratory (LLNL), and the University of California at Davis have obtained a complete solution of the ionization of a hydrogen atom by collision with an electron, the simplest nontrivial example of the long-standing unsolved problem of scattering in a quantum system of three charge particles [1]. In this application, SuperLU was used to construct preconditioners for the solution of unsymmetric linear systems of order up to 1.8 million. SuperLU is a general-purpose library available in the ACTS Toolkit that provides for the direct solution of large, sparse, nonsymmetric systems of linear equations on high performance machines.

- On April 26, 2000, the international BOOMERANG (Balloon Observations of Millimetric Extragalactic Radiation and Geophysics) consortium, led by Andrew Lange of the California Institute of Technology and Paolo de Bernardis of Università di Roma La Sapienza, announced results of a very detailed measurement of the cosmic microwave background radiation (CMB). These results revealed that the curvature of the Universe is not positive or negative but flat [2]. Much of the data analysis was performed by Julian Borrill (LBNL) employing a software called MADCAP (Microwave Anisotropy Dataset Computational Analysis Package [3]) which uses several routines from ScaLAPACK. ScaLAPACK is one of the tools available in the ACTS Toolkit intended for linear algebra calculations.

- *Achieving High Sustained Performance in an Unstructured Mesh CFD Application,* http://www.mcs.anl.gov/petsc-fun3d. Kyle Anderson, William Gropp, Dinesh Kaushik,, David Keyes, Barry Smith [27, 28]. This application is based on FUN3D [26] code that is used in airplane, automobile, and submarine applications for analysis and design This code involves sparse, unstructured data that imply memory indirection with only modest reuse. This application won the Gordon Bell Award at SC 99. The results of this research is wide applicability to other implicitly discredited multiple-scale PDE workloads of interagency, interdisciplinary interest.

- Prometheus [29] is freely available as an unstructured multigrid equation solver for large scale (106-109 degrees of freedom). These softwares were developed using PETSc, to provide high performance, cross platform, support for iterative solvers for discretized partial differential equations and ParMETIS [30], from the University of Minnesota for parallel mesh partitioning. Future work will include the integration of functionality available in Trilinos/Petra.

More details on these codes and additional examples of scientific applications using tools from the ACTS Toolkit can be found in the ACTS Information Center.

4.- Lessons Learned

As a result of the user support and education activities provided by ACTS, we have learned various lessons that are valuable to the computational science community, and in

particular the software development and support projects. Here we summarize the most important issues:

- *There is still a gap between tool developers and application developers which leads to duplication of efforts*. Without projects like ACTS, application developers will continue to design and implement codes using techniques that are already available from other sources. Quite often these new developments are far from optimal because of the application developers' inexperience with all the different issues that lead to optimal performance. In many cases, application developers only consult sources like Numerical Recipes **Error! Reference source not found.** that do not address platform optimization and parallelism, algorithm robustness, and language specific optimization issues.

- *The tools currently included in the ACTS Toolkit should be seen as dynamically configurable toolkits and should be grouped into toolkits upon user/application demand*. Based on the particular needs of an application, users generally benefit from only a subset of the functionality available in ACTS; therefore, they may only need to install a subset of the ACTS tools in their computing environments. A future agenda for the ACTS Toolkit must include an automatic mechanism that will allow a friendlier and dynamic installation of the tools based on particular demands.

- *Users demand long-term support of the tools*. One of the main concerns that users have expressed to us is the longevity of support from tool developers and required evolution of the software as the hardware technology continues to evolve and the complexity of the scientific application continues to grow. Inasmuch as a solid base of reusable tools is utilized inside newer tool developments, users are guaranteed the evolution of the tools; thus long-term support and functionality are also guaranteed.

- *Applications and users play an important role in hardening tools*. The main parameters for maturity are portability, robustness, acceptance, and long-term support. It is particularly the interactions with real users and real applications that have made the software mature, portable, robust and better documented. In turn, mature software will be widely accepted inside a given scientific community. The current framework of the ACTS Toolkit has promoted the tools to a wider national and international audience, thus increasing not only the visibility of the tools worldwide but also the range of users and applications.

- *Tools evolve or are superseded by other tools*. As technology continues to advance, there are some tool functionalities that are either no longer needed or are improved as direct consequences of the user demands. An example of these changes in the ACTS Toolkit is the Aztec library being superseded by AztecOO, which is one of the components of the Trilinos solver framework [8]. An umbrella project like ACTS provides mechanisms to help users with the transition and adoption of the new tools.

- *There is a demand for tool interoperability and more uniformity in the documentation and user interfaces*. Users want to experiment with functionalities available in a

subset of the ACTS tools, and finding similar user interfaces and comparable levels of support and documentation makes this task even simpler and risk free. Furthermore, the computational challenges at hand demand new software developments that interact with legacy code practices, data and computer languages. ACTS provides a natural infrastructure to put in practice all the code, data and language interoperability required by these new challenges.

- *There is a need for an intelligent and dynamic catalog/repository of high performance tools.* The need for a centralized software and reference repository is vital for preventing the duplication of efforts. Currently, the ACTS Information Center provides pointers to tools currently funded under ACTS with the accumulated expertise from tool users and scientific domains that are served by the tools. Additionally, we envision the inclusion of pointers to tools offering functionality not currently available in the ACTS Toolkit as well as fair comparisons with other tools that offer functionality that overlaps with the ACTS tools.

5.- Conclusions

The hard problems at hand for the ACTS Toolkit, and high performance software in general, continue to be language and software interoperability, uniform software distribution and licensing, friendly tool interfaces and tool installation procedures, performance, and tool acceptance. Our future agenda plans to implement an expansion of the ACTS Toolkit based on the valuable lessons learned from the outcomes of the current scope of the project, and work with the institutions outside DOE, high performance computing funding programs, and DOE initiatives and projects to formulate solutions to these hard problems. We plan deliver an expanded ACTS framework to host a solid base of computational tools with the appropriate levels of support and expertise, and to serve as a buffer between tool and application development.

The benefits of our proposed work can be measured in many ways. A wide range of scientific code developers and users will benefit from information and education about state-of-the-art, high performance computational tools. They will benefit from the development and promotion of robust, effective, portable, usable, and durable software. They will benefit from the increased interoperability of tools, which promotes the evolution and adoption of current software development projects into future software technologies. They will benefit from multidisciplinary collaborations and the consequent accumulation of expertise. Perhaps they will benefit most from spending less time on code development and having more time to devote directly to scientific discovery.

Acknowledgements.
ACTS is funded by the US Department of Energy, office of Mathematical, Information and Computation Science.

References.

[1] T. Rescigno, M. Baertschy, W. Isaacs and W. McCurdy, 1999. *Collisional breakup in a quantum system of three charged particles*, Science, 286:2474-2479.

[2] P. de Bernardis, P. A. R. Ade, J. J. Bock, J. R. Bond, J. Borrill, A. Boscaleri, K. Coble, B. P. Crill, G. De Gasperis, P. C. Farese, P. G. Ferreira, K. Ganga, M. Giacometti, E. Hivon, V. V. Hristov, A. Iacoangeli, A. H. Jaffe, A. E. Lange, L. Martinis, S. Masi, P. V. Mason, P. D. Mauskopf, A. Melchiorri, L. Miglio, T. Montroy, C. B. Netterfield, E. Pascale, F. Piacentini, D. Pogosyan, S. Prunet, S. Rao, G. Romeo, J. E. Ruhl, F. Scaramuzzi, D. Sforna and N. Vittorio., 2000. *A flat Universe from high-resolution maps of the cosmic microwave background radiation*, Nature, 404: 955–959.

[3] MADCAP, *http://www.nersc.gov/~borrill/cmb/madcap.html*

[4] ACTS Information Center, *http://acts.nersc.gov*

[5] ACTS Tools, http://acts.nersc.gov/tools.html

[6] J. N. Shadid, R. S. Tuminaro, Iterative Methods for Nonsymmetric Systems on MIMD Machines, in Proceedings of the Fifth SIAM Conference on Parallel Processing for Scientific Applications, Eds. Dongarra, Messina, Sorensen, and Voight, Houston, TX, March 25-17, 1991.

[7] R. S. Tuminaro, M. Heroux, S. A. Hutchinson, and J. N. Shadid, *Official Aztec User's Guide: Version 2.1*, December, 1999, http://www.cs.sandia.gov/CRF/aztec1.html.

[8] Trilinos, *http://www.cs.sandia.gov/Trilinos*

[9] Chow, E., A.J. Cleary, and R.D. Falgout, *Design of the hypre Preconditioner Library,* Proc. of the SIAM Workshop on Object Oriented Methods for Inter-operable Scientific and Engineering Computing, Mike Henderson, Chris Anderson, and Steve Lyons, Eds. (SIAM Press, Philadelphia, PA: 1998). Workshop held at the IBM T.J. Watson Research Center, Yorktown Heights, NY, October 21-23, 1998.

[10] PETSc Website, http://www.mcs.anl.gov/petsc

[11] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. *Efficient Management of Parallelism in Object Oriented Numerical Software Libraries*. In Modern Software Tools in Scientific Computing, E. Arge, A. M. Bruaset , and H. P. Langtangen., pp. 163--202, Birkhauser Press, 1997.

[12] Satish Balay, William Gropp, Lois Curfman McInnes, and Barry F. Smith. *PETSc users Manual*, ANL-95/11 –Revision 2.1.1, Argonne National Laboratory, 2001.

[13] SLEPc, http://acts.nersc.gov/workshop/slides/roman.pdf

[14] ODE, http://www.netlib.org/ode

[15] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide,* SIAM, Third Edition, Aug 1999.

[16] LAPACK, http://www.netlib.org/lapack

[17] Nieplocha, RJ Harrison, and RJ Littlefield, *Global Arrays: A portable `shared-memory' programming model for distributed memory computers*. In Proc. Supercomputing'94, pages 340-349, 1994.

[18] NWChem, *http://www.emsl.pnl.gov:2080/docs/nwchem*

[19] I. Foster and C. Kesselman, Eds, 1999. *The Grid, Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, Inc, San Francisco, CA.

[20] Ian Foster, 2002. The Grid: A New Infrastructure for 21st Century Science, Physics Today, February.

[21] G. A. Geist, J. A. Kohl, P. M. Papadopoulos, *CUMULVS: Providing Fault-Tolerance, Visualization and Steering of Parallel Applications,* International Journal of High Performance Computing Applications, Volume 11, Number 3, August 1997, pp. 224-236.

[22] S. Shende, A. D. Malony, J. Cuny, K. Lindlan, P. Beckman and S. Karmesin, Portable Profiling and Tracing for Parallel Scientific Applications using C++, Appears in: Proceedings of SPDT'98: ACM SIGMETRICS Symposium on Parallel and Distributed Tools, pp. 134-145, Aug. 1998.

[23] CCA-Forum, *http://www.cca-forum.org*

[24] CCTTSS, *http://www.cca-forum.org/ccttss*

[25] SciDAC/DOE, *http://www.er.doe.gov/scidac*

[26] FUN3D, *http://fmad-www.larc.nasa.gov/~wanderso/Fun*

[27] PETSc-FUN3D, *http://www.mcs.anl.gov/petsc-fun3d*

[28] Gropp, Kaushik, Keyes & Smith, 1999, *Toward Realistic Performance Bounds for Implicit CFD Codes*, in Proceedings of Parallel CFD'99, Elsevier

[29] Mark Adams, Parallel Multigrid Solvers for 3D Unstructured Finite Element Problems in Large Deformation Elasticity and Plasticity. International Journal for Numerical Methods in Engineering, 48(8):1241-1262, 2000.

[30] ParMETIS, http://www-users.cs.umn.edu/~karypis/metis/parmetis/