Los Alamos Computer Science Institute

Statement of Work for Academic Participants

Part I: Overview and Management

Introduction

The *Los Alamos Computer Science Institute (LACSI)* was created to foster computer science and computational science research efforts at the Los Alamos National Laboratory (LANL) that are both internationally recognized and relevant to the goals of LANL. LACSI is a collaboration between LANL and the Rice University Center for Research on High Performance Software, along with partners at the University of Houston (UH), the University of Illinois at Urbana-Champaign (UIUC), and the University of Tennessee at Knoxville. LACSI has major components on site at LANL and at Rice University.

LACSI personnel include member researchers, called *Fellows of the Institute*, both on and off site. The on-site researchers are called *LANL Fellows*, while the off-site researchers are referred to as *Academic Fellows*. While most of these will be at Rice, Academic Fellows can be at any participating institutions. The initial list of fellows is included below by institution:

Rice University: Ken Kennedy, David Applegate, Bob Bixby, Alan Carle, Alan Cox, Bill Cook, Keith Cooper, John Dennis, Nate Dean, Mike Fagan, Rob Fowler, Richard Hanson, Matthias Heinkenschloss, Charlie Hu, Guohua Jin, Petr Kloucek, John Mellor-Crummey, Dan Sorensen, Bill Symes, Frank Toffoletto, Richard Tapia, Linda Torczon, Joe Warren, Yin Zhang, Willy Zwaenepoel.

University of Houston: Lennart Johnsson, Barbara Chapman, Roland Glowinski, Yuri Kuznetsov, Jaspal Subhlok.

University of Illinois at Urbana-Champaign: Dan Reed, Ruth Aydt.

University of Tennessee at Knoxville: Jack Dongarra, Antoine Petitet.

Background:

As background for the technical statements of work provided later, it is helpful to provide our understanding of the LACSI goals, management structure, and modes of operation. This section is meant for perspective only. Our understandings do not supercede any formal agreements John Reynders, Andy White and others have made with DOE officials at LANL and Washington, D.C.

Goals:

The Los Alamos Computer Science Institute was founded with these goals:

- To build a presence in computer science research at LANL that is commensurate with the strength of the physics community at LANL.

- To achieve a level of prestige in the computer science community that is on a par with the best computer science departments in the nation.
- To pursue computer science research that is relevant to the goals of High Performance Computing (HPC) programs at LANL.
- To ensure that there remains a strong focus on high-performance computing in the academic computer science community.

To achieve these goals, the LACSI was established as a collaborative effort with the Center for Research on High Performance Software (HiPerSoft) at Rice University and some of its partner institutions. The purpose of this collaboration is to support joint research on high-performance scalable computing that is relevant to the overall LANL goals and to foster a strong relationship between LANL and the academic partners, especially Rice University, the University of Houston, the University of Illinois, and the University of Tennessee.

Management:

LACSI is managed by an executive committee consisting of senior Fellows from both on and off site. This committee conducts a regular review process, described below.

Executive Committee: The management of the computer and computational science activities will be the responsibility of the Executive Committee, which consists of senior researchers at LANL and the participating institutions. Current membership of the Executive Committee is as follows:

- John Reynders, Institute Director, Chair, LANL
- Lennart Johnsson, UH
- Ken Kennedy, Vice-Chair, Rice
- Jack Dongarra, Univ. of Tennessee, Knoxville
- John Mellor-Crummey, Rice
- Dan Reed, UIUC
- Andy White, LANL
- Rick Stevens, ANL
- Dan Sorensen, Rice

This group will be expanded in the future as appropriate to meet the management needs of the Institute.

The Institute Director will chair the executive committee and the Rice Project Director (initially Kennedy) will serve as vice-chair.

Annual Symposium and Planning Meeting:

1. The *LACSI Symposium* is both a meeting to showcase results of LACSI research efforts and as a forum for presenting outstanding research results from the national community in areas overlapping the LACSI technical vision. This will be a traditional conference-style meeting with participation by LACSI Fellows, outside scientists from academic partner institutions, and other scientists from the community at large.
2. The *LACSI Planning Meeting* is a smaller internal workshop designed to develop plans for new and continuing research directions to be pursued over the next year.

The observations from these meetings will be incorporated into the planning activities of the Executive Committee, establishing new directions along with new goals and modified milestones. The Executive Committee will evaluate progress according to two criteria:

1. The quality of the research from the perspective of computer and computational science. The principal question to be answered is whether the research is of high quality and is producing results that are innovative, long-term in focus, and of high relevance to the general area of high-performance computing.
2. The relevance of the research to LANL application themes. The key question would be to what extent the work is making a contribution to the application activities on LANL computing platforms and what the prospects are for such contributions in the near or distant future.

In performing this evaluation and planning process, the Executive Committee will be assisted by the LACSI Planning Committee, which consists of all members of the Executive Committee plus additional project leaders from LANL and the Academic Partners. Currently, those additional members are Pete Beckmann and Rod Oldehoeft from LANL, Jack Dongarra from Tennessee, and Keith Cooper, Alan Carle, Alan Cox, Matthias Heinkenschloss, Dan Sorensen, and Linda Torczon from Rice.

The LACSI planning process will be driven by the annual review cycle. Based on the outcomes of its reviews of research, the Executive Committee, assisted by the LACSI Planning Committee, will propose a collection of projects to be undertaken, along with goals for those projects, and identify projects to phase out. Institute Fellows to lead the new efforts will be identified and work will be initiated. The resulting work will be evaluated in subsequent reviews.

Collaboration Management

Rice University will be the lead on the contract for all academic partners. The initial contract will have a one-year term with at least four one-year continuance options. The continuance option will be evaluated yearly by the LANL members of the Executive Committee.

At the end of the first three years of activity, a review committee, consisting of external computer and computational scientists along with LANL staff members not affiliated with LACSI, will conduct an evaluation to determine the extent to which LACSI is meeting its goals. This initial contract will be for five years with a follow-on contract dependent upon the results of the review.

The Academic Partners Management Committee, a committee of senior Academic Fellows drawn from the participating academic institutions, will coordinate activities of the academic partners. Current membership of this committee includes Ken Kennedy (chair), Alan Carle, Keith Cooper, Alan Cox, Jack Dongarra, Matthias Heinkenschloss, Lennart Johnsson, John Mellor-Crummey, Dan Reed, Dan Sorensen, and Linda Torczon.

Modes of Interaction and Collaboration

A principal goal of LACSI is to foster collaborative relationships between LANL participants and the participating academic institutions. These collaborations fall into four broad categories: joint recruiting, on-site visits, technical meetings, and industrial partnerships.

Recruiting

If LACSI is to succeed in its goal of invigorating computer science and computational science research at LANL, it will need to successfully recruit some of the best computer science researchers in the nation. Rice University and the other academic partners can assist in this endeavor in several ways. Academic faculty at Rice can serve on search committees for LACSI, providing an academic perspective and knowledge of the computer science community drawn from years of faculty recruiting.

In addition, LACSI positions can be made more attractive by including a faculty appointment in either the Rice Computer Science Department or the Computational and Applied Mathematics Department. Such faculty appointments are subject to Rice University policy and guidelines and are typically at one of three levels:

1. *Adjunct Professor*, a courtesy appointment typically accorded a distinguished researcher at another institution.

2. *Visiting Professor*, which is used for faculty at other institutions who are on a one- to three-year visit on site at Rice.

3. *Faculty Fellow*, which is used for on-site faculty-level researchers who are supported entirely on grants and contracts. A Faculty Fellow can apply for grants from Rice as a Principal Investigator.

One strategy for recruiting is to have new researchers spend a year or two at Rice as Visiting Faculty or Faculty Fellows before coming to LANL. This would allow time for the clearance process to complete while the researcher begins work on a LACSI effort. Other staff members might return to Rice for extended visits later in their careers. Of course, we will also make ample provision for long-term visits to LANL by faculty from Rice and other academic institutions.

On-Site Visits

Both Rice and LANL will provide space for long-term visitors from other LACSI sites. In addition, LACSI will establish programs for visiting faculty (including sabbaticals), graduate students, and undergraduate students at LANL.

Joint Meetings

In addition to the annual symposium and the annual planning meeting, LACSI will regularly sponsor technical workshops on special topics of interest to the participants. These will typically involve outside researchers and will lead to published proceedings wherever possible.

Industrial Partnerships

LACSI partners will establish joint relationships with selected industrial partners, particularly those critical to the success of ASCI and follow-on programs (e.g., SGI, IBM, The Portland Group).

Computational Resources

The academic partners will be provided with access to ASCI computing platforms at LANL on a predetermined basis for development and testing. The process will make it possible to allocate a small cluster of nodes each week and a larger cluster of nodes once a month. It is understood that dedicated access may be needed for key tests and performance analyses.

In the following research section, all Rice deliverables for individual projects will include an annual report describing the project, any affiliated software, and its application to LANL problems. Source code affiliated with any project will be delivered to LANL upon request. Such software will be available for use or modification by the United States government, LANL, and the University of California without restriction or fee.

Part II: Research

A. Compilation, Systems, and Performance Evaluation of Large Scale Parallel Machines

Investigators: John Mellor-Crummey, Keith Cooper, Alan Cox, Rob Fowler, Richard Hanson, Y. Charlie Hu, Guohua Jin, Ken Kennedy, Dan Reed, Linda Torczon, Willy Zwaenepoel

Compilation strategies

Investigators: John Mellor-Crummey, Rob Fowler, Richard Hanson, Guohua Jin, Ken Kennedy

Project Description: The objective of this project is to develop compiler and run-time technology that will help application developers achieve a high fraction of peak performance on large-scale parallel computing systems.

Achieving this objective on conventional large-scale systems requires eliminating obstacles to efficiency at multiple levels: within a single processor, within a symmetric shared-memory multiprocessor node, within a cluster of nodes sharing memory with non-uniform memory access (NUMA) latency, and between NUMA shared-memory systems coupled with a message passing interconnection network. A major challenge is to improve utilization of multi-level memory

hierarchies within a single thread of control using a combination of data restructuring to improve spatial locality and reduce conflicts, computation restructuring to improve temporal reuse, and software prefetching to improve latency tolerance. One way to achieve this is by adapting the ATLAS technology to ASCI related needs. In addition we will develop portable performance monitoring technology that can be used to detect execution rate, cache behavior, and overall performance bottleneck in user's applications. Additional challenges for achieving high performance include analyzing and transforming programs within and across procedures to expose massive parallelism, partitioning data and computation to exploit multi-level parallelism effectively, and scheduling computation and communication to avoid resource contention and load imbalance. All of these issues need to be addressed in the context of scientific applications that employ sophisticated methods for processing large-scale data sets, including the use of unstructured meshes and dynamic adaptation of data structures. Procedurally, this effort will involve experimentation (including simulation, measurement, and analysis of applications) to identify the most significant performance bottlenecks, developing and prototyping techniques for improving performance in such applications, and developing compiler analyses and transformations to automate application of these techniques to the greatest extent possible.

In addition to technology for conventional large-scale parallel systems, we will begin to explore compilation technology appropriate for promising petaflops architecture designs. The leading design sketch for a future petaflops computing system has a radically new organization, employing processor-in-memory technology in concert with a multi-threaded central processor. Software technology for using such systems effectively is in its infancy.

Milestones: (Year 1) Work with LANL application scientists to analyze representative applications (including the compact apps) to identify the key classes of bottlenecks; develop and test program restructuring strategies for addressing these bottlenecks; and begin assembling compiler analysis and transformation infrastructure to assist in the diagnosis and remediation of performance bottlenecks. A particular focus of this work will be on memory hierarchy optimizations for irregular computations. **(Year 2)** Continue to study applications with LANL scientists and develop compiler and run-time technology to implement restructuring transformations that address bottlenecks uncovered in ongoing experimentation. Research will focus on techniques for improving memory hierarchy performance of irregular and adaptive codes, along with compilation strategies for improving the efficiency of programs for hybrid shared-memory and message-passing systems. Lay foundation for compiler-assisted tools to support massively parallel programming. **(Year 3)** Continue work on compiler and runtime techniques for improving performance on large-scale machines, including research on compiler support for exploiting massive parallelism effectively, analysis and optimization of programs with explicit synchronization, and beginning investigation of compiler technology for radically new machine organizations for petaflop computing. Demonstrate prototype compiler-assisted tools. **(Years 4-5)** Investigate techniques for optimization of object-oriented programs**,** dynamic compilation techniques that can use runtime estimates to optimize code on-the-fly; and generation of configurable code to support runtime optimization. Refine technology for compiler-supported tools that assist development of portable high-performance codes by providing fine-grain control over compiler optimizations to retarget codes for particular architectures.

IA 64 Compilation Issues:

Investigators: Keith D. Cooper, John Mellor-Crummey, Ken Kennedy, Linda Torczon

Project Description: The objective for this investigation is to develop compiler technology to attain a reasonable fraction of the available performance on IA 64 platforms when running code compiled from classical languages, like dialects of Fortran, C, and C++. Many distinct issues contribute to performance on machines like the IA 64. For high performance, a compiler must:

Keep the functional units busy. This requires the compiler to transform the input program so that it has enough IP to sustain the computation rate. It requires instruction-scheduling techniques that can convert available ILP into dense schedules — for simple loops, for loops with control flow, and for straight-line code.

Have operands ready for each instruction. This will involve transforming programs to match their locality to the memory hierarchy of the target system, including real applications of blocking, prefetching, and (perhaps) streaming. Once the data is actually on-chip, the compiler may need to manage instruction and data placement with respect to the clustered register file, along with the classic problems of allocation and scheduling.

Handle predication with a holistic approach. If-conversion is not the whole answer. Open issues include understanding the tradeoff between predication, with its sparser execution pattern, and branching to denser code; predicate register management; the interaction between predicate lifetimes and instruction placement in the scheduler; and minimizing the impact of predicate evaluation on overall application performance.

The IA 64 is complex enough that we will need realistic architectural simulations to understand program behavior, the impact of various transformations on performance, and where real potential for improvement exists.

To address these issues, we have brought together a team that includes expertise in analysis and translation for parallel systems (K. Kennedy, J. Mellor-Crummey), in managing memory hierarchies and locality (K. Cooper, J. Mellor-Crummey, K. Kennedy), and in low-level code generation issues (K. Cooper, L. Torczon).

Milestones: (Year 1) Investigate a broad array of issues that arise in "wide" machines; work with LANL personnel to characterize application performance issues; work with Intel to get necessary non-disclosure presentations; work internally to connect the various tools together into a test bed for our ideas. **(Year 2)** Study application performance on the IA 64; tailor investigations begun in year 1 to IA 64-specific issues including memory hierarchy management, managing predicated execution, and handling the register architecture; pursue direct relationships with IA 64 system vendors to transfer results into their commercially available tools. **(Year 3)** Demonstrate prototype implementations of the transformations developed in year 2; report on the effectiveness of these transformations from simulation and/or hardware performance studies. Work with LANL application developers to improve, directly, the performance of their codes; launch investigations of new problems revealed by studies in previous years. **(Years 4-5)** Work with LANL applications developers and vendor compiler groups to identify new problems exposed by processor and system architecture that are amenable to compiler-based improvement and develop techniques to address them.

Tools

Investigators: Dan Reed, Ruth Aydt, Barbara Chapman, Ken Kennedy, John Mellor-Crummey, Rob Fowler

Project Description: Parallel computing is rapidly evolving to include heterogeneous collections of distributed and parallel systems. Concurrently, applications are becoming increasingly multidisciplinary, with libraries and other application components implemented using diverse programming languages, models, and parallelization strategies. In consequence, it is now extraordinarily difficult to achieve high fractions of peak hardware performance on large-scale parallel systems, emerging networks of workstations, or wide area computational grids.

To optimize the behavior of such complex applications, performance analysis software must also evolve, replacing simple measurement tools with deep integration of compile-time transformations with measurement and analysis. Moreover, time varying resource availability and demands will necessitate increasing use of real-time, adaptive performance optimization and just-in-time compilation. This integration, based on user-specified, compiler-synthesized, and measurement-validated performance contracts, will enable creation of a new generation of nimble, high-performance applications.

Hence, we will develop a new generation of software tools that integrate the compilation research described previously with real-time performance measurement, adaptive control systems, and intelligent data visualization and control. The goal of this work is to create an intelligent performance toolkit that allows software developers to create and runtime systems to negotiate "performance contracts" among software components and hardware/software systems.

Milestones: (Year 1) Identification of collaboration opportunities, preliminary performance scalability estimation, measurement, and visualization, targeting shared memory systems (memory hierarchies), networks of PCs/workstations and wide area networks. **(Year 2)** Performance specification systems based on performance contracts, together with contract validation and constraint-based performance tuning. **(Year 3)** Integrated adaptive performance tuning and just-in-time compilation based on resource availability, together with software distribution and support. **(Year 4)** Validation and extension of performance tuning toolkits based on user experience. **(Year 5)** Multilingual, multi-model performance optimization based on real-time measurement and adaptive control.

Parallel Application Development Tools (Barbara Chapman). *Generation of Fortran programs for SMARTS*: SMARTS (Shared Memory Asynchronous Runtime System) was created in ACL to enable the execution of task and data parallel programs on distributed memory architectures with deep memory hierarchies. It has primarily been used as a runtime environment for code generated in POOMA. It has been used in conjunction with C++ applications.

The SMARTS API permits the specification of parallelism within a C++ code in a form that is suitable for execution in an essentially data flow method by the SMARTS runtime system. We plan to develop a system that will support the creation of parallel Fortran programs for execution by the SMARTS runtime system. The system will interact with the user to obtain a suitable version of the code.

The main tasks in this work during the first two years are:

- Specification of a Fortran API for SMARTS.

- Developing a strategy for decomposition of Fortran programs into SMARTS Iterates for data parallel execution.
- Interactive generation of Fortran program versions using Iterates and annotations to express any user input to the process.
- Generation of representations of dependences between Iterates for use by the SMARTS runtime system.

Future work will refine the strategy for program decomposition. In particular, it must permit the extraction of combinations of task and data parallelism, and facilitate exploitation of a wider range of architectures. It will be necessary to find a means to generate and execute SMARTS applications in order to utilize different mixtures of shared and distributed memory in hardware.

User Guidance for Application Migration to Component Systems: Additional work will complement the interactive system developed for generation of SMARTS code in order to provide guidance for the task of adapting code for insertion in component systems. If existing applications are to be incorporated into component architectures, then they must be thoroughly analyzed in advance. In particular, I/O in the code must be adapted to the requirements of the new system. Changes to the control flow may also be necessary, if there are multiple program exits or unwanted modes of behavior. We plan to design and develop a tool that cooperates with the user to adapt legacy code to the requirements of a component-based system.

In the initial work we will:

- identify the requirements for adaptation of code for component--based systems,
- design a guidance strategy based upon which the user cooperates with a tool.

Future work will build on top of this work by seeking strategies to guide the user in the complex task of selecting and realizing a parallelization strategy for programs that will execute on a system with multiple levels of parallelism in the hardware. The tool will not only aid the detection of problematic code features; it will highlight potential performance problems and may suggest high level directives for insertion.

Milestones: (Year 2) Specification of Fortran API for SMARTS and develop an initial strategy for creation of SMARTS Iterates. (Year 3) Full strategy for decomposition of Fortran programs into SMARTS Iterates for data parallel execution; Generation of representation of dependences between iterates for use by the SMARTS run-time system. (Year 4-5) Interactive generation of Fortran program versions using Iterates, and annotations to express any user input to the process

OpenMP

Investigators: Alan Cox, Y. Charlie Hu, Willy Zwaenepoel,

Project Description: The objective of this project is to develop portable shared memory programming support that scales from small clusters of workstations and SMPs to the ASCI supercomputers. Our approach is to support OpenMP — an emerging shared memory programming API standard — directly on top of the TreadMarks software distributed shared memory (DSM) system, which runs on networks of workstations and PCs as well as on distributed memory systems such as the IBM SP2. We have completed a prototype OpenMP to

TreadMarks translator. To support efficient and scalable OpenMP programming, we plan to conduct research on optimizing the performance of software DSMs in the following two directions (a) integrated compiler-time/run-time optimizations for reducing communication for both data and synchronization, aggregating communication, reducing contention, and moving computation to data, and (b) improving the scalability of TreadMarks. In the first area, we will investigate techniques for aggregating synchronization and data communication, merging synchronization and data, aggregating/pushing data, exploring integrated compiler/runtime approaches for reducing contention through prefetching and multicast, and runtime techniques for migrating computation to data. In the second area, we will investigate methods for improving the scalability of software DSM implementations including using relaxed memory consistency models, exploiting more fine-grained parallelism in applications, using more scalable memory consistency protocols, and avoiding non-scalable implementation techniques.

Milestones: (Year 1) Complete our OpenMP to TreadMarks compiler to support the full set of OpenMP directives. **(Year 2)** Access performance impact of different optimizations and identify key obstacles to the scalability of software DSM systems. **(Years 3-4)** Develop integrated compiler-time/run-time optimizations to support efficient OpenMP programs on medium-scale parallel machines, and improve on the scalability of software DSM systems to sustain high efficiency on large-scale systems. **(Year 5)** Demonstrate the fully integrated compiler/runtime system that supports efficient and portable OpenMP programming.

B. Component Architectures for Distributed Parallel Computing and Rapid Application Development and Composition

Investigators: Ken Kennedy, Alan Carle, Barbara Chapman, Keith Cooper, Jack Dongarra, Richard Hanson, Lennart Johnsson, Dan Reed, Jaspal Subhlok, Joe Warren, Willy Zwaenepoel

The overarching goal of this activity is to develop a component architecture that can be used to support rapid prototyping of portable parallel and distributed applications. This architecture would be the basis for a framework for applying advanced compilation techniques, run-time system elements, and programming tools to the support of applications for distributed heterogeneous grids. To succeed, this effort will need to accomplish three long-term goals.

1. It must develop a *framework for integrating existing components* rapidly and conveniently. The framework must be able to integrate components written in different languages, particularly Fortran and object-oriented languages like C++.
2. It must develop a *strategy for producing reliably efficient code* for both Grid configurations and high performance on scalable parallel machines. This will involve some sort of whole-program analysis and optimization framework. Furthermore the program preparation process must be reorganized to support this activity. In particular the compilation framework must be divided into smaller components that can be invoked at various times in the program preparation process.

3. It must develop a *collection of components for use in science and engineering applications*. This collection should be ideally suited for use in the rapid prototyping framework described above. The algorithms must be general, portable, and usable in a variety of situations.

Component Architectures for Problem Solving Environments

Investigators: Ken Kennedy, Alan Carle, Keith Cooper, Jack Dongarra, Jaspal Subhlok

The goal of problem-solving environment (PSE) research is to produce systems that make it possible for unsophisticated end users to rapidly prototype and experiment with new applications developed from existing libraries of components. The ultimate goal would be to use extensive computation to make the resulting applications as efficient as applications written by professionals. Components would be primarily focused on parallel and distributed computing in science and engineering, but many of the techniques would be applicable to all computation domains.

The primary strategy for this activity is to compose applications through the use of (possibly visual) scripting languages. These languages are in wide use today but suffer because the programs that are composed using them are typically not efficient enough for repetitive use in a production situation. Furthermore, there are critical issues of correctness and reliability that may be too technical for the end-user programmer. These issues must be dealt with automatically in the PSE, possibly through the use of extensive computation.

If this effort is to succeed in the Grid environment, it must take into account two important realities. First, many components will be constructed using object-oriented languages, so techniques for optimizing such languages are critical. Second, the execution environments for the resulting programs are likely to be distributed, so the implementation must take into account the performance implications of distributed systems, even if the applications are compiled together. For these reasons, basing a significant portion of the work on the Java programming language makes sense. Java is portable and includes distributed computing interfaces. However, we must overcome one major drawback of Java if it is to be used in scientific computation, namely its less-than optimal performance. Although we intend to focus on Java, many of the strategies developed for Java will extend to other object-oriented languages such as C++.

This research will focus on four important directions:

Component Architectures for Integration. This effort will focus on the design and specification of components that can be used in a PSE for high-performance computation. Significant issues will be flexibility and adaptability of the components to both the computations in which they are incorporated and the platforms on which they will be executed. In addition these components must have architectures that permit the effective management of numerical accuracy.

Compilation of Object-Oriented Languages. As mentioned above, high-performance compilation strategies must be developed for object-oriented languages such as Java and C++. This should include interprocedural techniques such as inlining driven by global type analysis

and analysis of multithreaded applications. This work would also include new programming support tools for high-performance environments.

Integration of Heterogeneous Components and Models . In the future, components and applications will be written in more than one language. To support this kind of application development we propose to work on a number of critical supporting technologies including:

1. Frameworks for integrating components in diverse languages and models
2. Support for data interchange via a uniform interface to a virtual "dataspace"
3. Runtime support to optimize data transfer and minimize buffering
4. Customization for specific coordination and data exchange patterns (e.g. streaming data, broadcasts, concurrent shared objects), that lead to context specific communication and buffering optimizations.
5. Support for large and persistent data objects.

Toolkits for Building Problem-Solving Systems. The effort will also focus on the production of tools for defining and building new domain specific PSEs, including:

1. Tools for defining and building scripting languages
2. Translation of scripting languages to standard intermediate code
3. Frameworks for interprocedural optimization of standard intermediate code supporting multiple languages
4. Optimizing translation of intermediate language to distributed and parallel target configurations
5. Adapting the NetSolve environment to the needs of the ASCI related users.
6. Demonstrations in specific applications domains

An important goal of this effort is to make it possible to build highly efficient applications from script based integration of pre-defined components. Building on the component architecture efforts described above, we will pursue the novel strategy of "telescoping languages" to make it possible to extend existing languages through the use of software components.

Milestones: (Year 2) Definition of component architecture to foster smooth integration, even for numerical components with variable accuracy. Demonstration of a preliminary global optimizing compiler for Java. Specification of the mechanism of telescoping languages for optimization and preliminary implementation effort in Java. Demonstration of a preliminary framework for integrating components to produce efficient parallel programs. Establishment of an active collaboration with the LANL SILOON project. Adaptation of the NetSolve environment to the needs of ASCI users. **(Years 3-5)** Produce a high performance compilation system for Java and extend the techniques to C++. Produce a complete PSE and toolkit for integrating component into efficient parallel and distributed systems. Produce an interprocedural framework capable of integrating PSE components with high performance.

Application Development Support for Distributed Computing

Investigators: Ken Kennedy, Barbara Chapman, Keith Cooper, Lennart Johnsson, Dan Reed, Jaspal Subhlok

The key challenge in building grid applications is to construct applications that are adaptive to changes in the execution environment and that can detect and correct performance problems automatically.

In this activity, we will explore the meaning of network-aware adaptive applications and what the implementation and optimization challenges are for such applications.

The work will proceed in two major subactivities:

Frameworks for Building Adaptive Network Applications. This research will develop a framework for building "network-aware" applications that can adapt their execution dynamically to meet performance and Quality-of-service goals. The main components of this research are as follows:

- A uniform interface to the network state, including the workloads on the computation nodes and the capacities and traffic on the network links.
- Compiler and runtime support for efficient dynamic migration of applications on a network.
- Algorithms to drive dynamic selection of nodes and migration decisions based on application structure and network state. This is a new and important direction of research made possible by the availability of network information about bandwidth and latencies of links and switches. Good node selection is especially important for data parallel simulations since a single slow congested communication link can become a bottleneck and degrade overall performance dramatically.

Compilation of Reconfigurable Object Programs. This research will explore the challenges in compiling programs to be dynamically reconfigurable in the sense described above. It will explore programming models and their translation to efficient collections of tasks that can be targeted to a variety of Grid computing platforms. The long-term goal is to support component-based implementation of applications for grids using PSEs of the sort described in the previous section. A critical technology will be adaptivity to changing performance characteristics of the components of a distributed computing platform. This adaptivity will require reoptimization and migration of components and data.

Milestones: Work with the compilation group and ASCI representatives to define model applications for evaluation of technology, then develop sample codes for test and evaluation of compiler directives and compilation based support for dynamic load-balancing. **(Year 2**) Define three representative application code samples and implement those for one language and execution environment. Design of a framework for adaptive application development. Implementation and evaluation of prototype application development support for ASCI execution environments. (**Year 3-5)** Refinement and optimization of the development framework. Application to additional sample codes.

Integration of heterogeneous sequential and parallel modules (Lennart Johnsson, Jaspal Subhlok). Complete applications are often composed from diverse modules. For example, a

typical scientific simulation consists of preprocessing, simulation, and visualization components that demand different programming models and different execution platforms. The goal of this project is to simplify and ease the process of integration of such diverse modules into full applications. Proposed research will develop a service that will allow heterogeneous modules to exchange data objects using a uniform interface to a virtual ``dataspace". Application modules will be able to perform all external communication using a single interface and a module need not know the programming framework or the execution platform of other modules with which it communicates and coordinates during execution. Data objects are matched by identification tags that serve as indices into the dataspace. Implementation will include runtime support to optimize data transfer and minimize buffering. This project aims to employ the concept of a virtual shared space, pioneered by the Linda system, to build a powerful framework for efficient integration of independently developed modules in a distributed computing environment. Important innovations in this project include the following:

- Support for heterogeneity in terms of languages, parallelism models, and architecture. Languages that we expect to support include C, C++, Fortran and Java.
- Parallel arrays and other compound objects can be exported or imported as single distributed entities, simplifying usage and allowing the use of optimized group communication libraries.
- Customization for specific coordination and data exchange patterns (e.g. streaming data, broadcasts, concurrent shared objects), that lead to context specific communication and buffering optimizations.
- Support for large and persistent data objects. A module can export a data object and terminate execution and another module can import it later. The runtime system attempts to transfer large "file size" objects directly between modules and uses disk buffering only when necessary.

For this project, we plan to coordinate with the group developing the PAWWS system at LANL, and use that software to bootstrap the research and development process.

Milestones: (**Year 2**) Initiate the design of the dataspace programming model, i.e., the API that modules can use for interaction with other modules in a heterogeneous environment. Develop an implementation plan including the selection of machines to be supported and enabling technologies to be used for the project. We anticipate building on top of the PAWWS system. First prototype implementation of dataspace for communication between separate sequential programs running as concurrent processes on networks of machines and on shared memory multiprocessors. (**Year 3**) Develop support for language heterogeneity. Demonstration of exchange of data objects between FORTRAN, C, and Java programs. Analyze the issues that arise in providing support for multiple languages (e.g., data formats, representation of complex objects, communication mechanisms). Support for architectural heterogeneity. Demonstration of communication of data objects between programs executing on platforms with different architectures. (**Year 4-5**) Add support for persistent data objects and ``file size" objects. Demonstrate that writes to large files can be replaced by calls to the dataspace API and buffering in disk is avoided when concurrent readers exist. Demonstrate persistence, i.e., exchange of data objects between modules that execute at different times. Enhance the runtime support to optimize common communication patterns discovered in the course of experimentation. Some expected patterns are asynchronous data streams, parallel access to shared objects, and

multicasts. Perform detailed experiments with selected applications and make the dataspace interface and implementation available to LANL and other selected sites.

Numerical Component Libraries

Investigator: Lennart Johnsson, Jack Dongarra, Roland Glowinski, Richard Hanson, Yuri Kuznetsov, Antoine Petitet, Joe Warren

The principal goal of this effort is to develop libraries of components that can be effectively used in rapid prototyping systems of the sort this project proposes to produce. Key considerations will be production of scientific computing components that can be used reliably on a variety of computation platforms, especially parallel and distributed systems. This will entail establishing an architecture that provides a high degree of flexibility while maintaining efficiency and robustness across a broad spectrum of computational configurations. Important subactivities include:

1. *Standards*. Initiate and lead community effort towards an interface standard that allow for the creation of efficient implementations of common library routines (e.g., FFT, eigenanalysis, equation solvers, BLAS, etc) on a broad spectrum of architectures and heterogeneous distributed environments.
2. *Adaptable Libraries*. Creation of polyalgorithmic, parameterized library functions that allows for the run-time selection/optimization of algorithm selection and scheduling based on the problem at hand and the execution environment.
3. *Fast Algorithms*. Several key ASCI applications are based on mathematical models requiring evaluation of interparticle interactions. We propose to develop library support for use of multipole and multipole like O(N) methods in ASCI applications.
4. *Algorithms for Graphics and Visualization*. Included are algorithms for the manipulation and visualization of extremely large data sets, generation and refinement of unstructured simplicial grids, along with multiresolution analysis over those grids, data management via wavelets, and feature definition and extraction.
5. *Extending the ATLAS* technology so it can gather key parameters of target hardware platforms and optimize selected kernel or software components across a range of platforms.

Milestones: (Year 2) Development of a draft community standard for common scientific software functions in large-scale simulations through email discussions and a workshop on the topic. Demonstration of efficient FFT routines, including "out-of-core" routines on ASCI platforms. Demonstration of linear time, adaptive, N-body solver. Real-to-complex and complex-to-real FFT for ASCI platforms. Prime factor transforms. Fast Legendre and spherical transforms. (**Year 3)** Polyalgorithmic solvers.

C. Computational Mathematics

Large-Scale Nonlinear Optimization Algorithms and Software

Investigators: Yin Zhang, Richard Tapia

Project Description: The objective of this project is the development of reliable algorithms and parallel software for large-scale nonlinear constrained and unconstrained optimization problems, including systems of nonlinear equations. The emphasis will be on constrained least-squares problems for large-scale data-fitting and parameter identification. The algorithms will be based on interior-point methodology. In some of the algorithms and codes, scalability will be the primary focus; in others, reliability will be the emphasis.

Relevance to LANL: This investigation will support computational infrastructure building for emerging applications. It will provide reliable and enabling tools for important computational tasks, such as large-scale constrained data fitting and solution of nonlinear systems that need to be solved in LANL's applications and predictability studies.

Milestones: (Year 1) Perform algorithmic research. **(Years 2-5)** Algorithm testing and initial software design. Develop prototype software using existing linear algebra libraries. Refine the code and test it on DOE/LANL-related applications. Develop a preliminary parallel version of the code. Document research results in technical reports.

Parallel Tools for the Analysis of Linked Subsystems

Investigators: Matthias Heinkenschloss, John Dennis, Joe Warren

Project Description: The objective of this project is the development of parallel tools for the coupling of simulations of aspects of a problem to full system simulations on which decisions can be based. This requires the development of a mathematical and algorithmic framework that supports the coupling of expensive subsystem simulations with varying accuracy, the identification of parameters in these linked systems, and the error prediction in linked simulations through nonlinear backward and forward analyses (if the outputs are allowed to vary in a certain range, how accurate must the parameters be; if the parameters vary in a certain range, what is the variation in the outputs). The research issues that will be investigated are: (1) Parallel algorithms for the solution of blocked nonlinear systems with very expensive and inexact function evaluations; (2) Parallel algorithms for parameter identification governed by expensive and inaccurate simulations; (3) Sensitivity and adjoint equation approaches for nonlinear error analyses; and (4) Implementation of these algorithms in an object-oriented way in parallel computing environments.

Relevance to LANL: This investigation will support the LANL effort in emerging applications, which require the coupling of subsystem simulations. It will provide mathematical and algorithmic frameworks for error propagation in coupling of simulation codes.

Milestones: (Year 1) Design an object-oriented framework for linked subsystem simulations. **(Years 2-5)** Provide an initial implementation of basic algorithms in this framework. Apply this framework to problems relevant to LANL. Expand and refine the subsystem analysis tools and

their application to emerging applications. Document the results in technical reports and implement prototypes.

Code-Based Sensitivity Analysis Project

Investigator: Alan Carle

Project Description: The goal of this project is to develop source-to-source code transformation tools to convert massively-parallel, high-fidelity, physics-based computer simulation codes into codes that are suitable for use in the context of accurate and efficient sensitivity analysis. The research issues that will be investigated are: (1) Automatic differentiation of Fortran 90/95, including techniques for computing first- and higher-order derivatives of parallel codes using the forward and adjoint modes of automatic differentiation; (2) Automatic extension of simulations "at a point" to simulations "in a neighborhood of a point" for use in validation and verification by replacing computations on floating point values with computations on intervals, wavelets, probability distributions, or other representations of ranges of values (generalized ranges); and (3) Novel ways in which information about code structure can be used to improve/accelerate sensitivity analysis.

Relevance to LANL: Sensitivity analysis plays an important role in developing a deep understanding of the results of computational simulations and the underlying mathematical models that drive the simulation. Sensitivity analysis is a key component in techniques for parameter identification and quantification of uncertainty.

Milestones: (Year 1) Investigate availability, acquire, and evaluate Fortran 90/95 compiler infrastructure. **(Years 2-5)** Investigate computational techniques for sensitivity analysis. Develop prototype code augmentation infrastructure for Fortran 90/95 and explore representations, and local rules, for generalized ranges. Investigate techniques for global propagation. Complete automatic differentiation tool for Fortran 90/95 using code augmentation infrastructure and test. Develop prototype of generalized range propagation tool. Complete generalized range propagation tool and test. Improve automatic differentiation tool and generalized range propagation tool based on user feedback. Document research results in technical reports.

Eigenvalue Methods and Software for ASCI-MPP systems

Investigator: Danny Sorensen

Project Description: This investigation will develop methods and parallel software for large eigenvalue problems and related applications. The project will extend the capabilities of the highly successful P-ARPACK eigenvalue software. This software provides enabling technology in numerous application areas. It also can serve as an excellent test bed and point of interaction for new compiler technology. Two additional related areas which impact other projects in the computational mathematics effort are reduced basis methods for dynamical systems and control of processes governed by PDE and regularization of large discrete ill-posed problems such as those arising from inverse problems. Research goals include: (1) The ability to handle increases of two or more orders of magnitude in problem size (current technology is 10^6 to 10^7 variables); (2) The ability to scalably determine a significant percentage of the eigenpairs (increases from 10% to 50% of the total are needed for selected applications in electronic structure applications);

(3) Demonstration of terascale performance for a set of simulations that are central to understanding material properties; (4) Development of new reduced basis methods for dynamical systems and control of processes governed by PDE and provision of new partial balanced realizations with optimal H-infinity error estimates (there is potential for several orders of magnitude increase in time scale for molecular dynamics); and (5) Application of recently developed methods for regularizing discrete ill posed problems to parameter estimation and other inverse problems.

Relevance to LANL: This research will support the LANL effort in emerging applications, which require eigen-analysis, SVD, or principle component analysis. It will support DOE and LANL efforts such as electronic structure calculations on materials and chemical reaction dynamics. The project will provide enabling technology for simulation of composite materials and their decay modes, control systems arising in linked-subsystems investigation, and parameter estimation and inverse problems.

Milestones: (Year 1) Begin initial design of scalable parallel eigenvalue software for terascale architectures. **(Years 2-5)** Complete design of scalable parallel eigenvalue software for terascale architectures. Design interface for relevant applications. Incorporate available technology from compiler and parallel tools projects. Initiate study of reduced basis methods and applications of regularization. Expand and refine the eigenvalue software. Continue investigation and software design for reduced basis methods, regularization schemes and their application to emerging applications. Document results in technical reports and implement prototypes.

Integer/Mixed Integer Programming and Large Scale Problems

Investigators: Bill Cook, Bob Bixby, David Applegate, Nate Dean

Project Description: Discrete optimization is a critical component of computational engineering. It is used to solve practical problems that involve choosing the best alternative from a field of possibilities. Perhaps the most important class of discrete optimization problems are the mixed integer programming models, or MIPs for short. These problems differ from linear programming problems in that some of the variables are required to take on only integer values. These integer variables allow modeling of decisions that are discrete in nature, but their inclusion makes MIP problems much more difficult to solve than the corresponding linear programming problems.

The inherent difficulty of MIP, and of discrete optimization in general, tests the limits of available computing platforms. Moving discrete optimization onto distributed, parallel systems will be a focus of Rice's research in this area, together with fundamental work on solution techniques for very large systems. We will also explore the use of graph theory in optimization and the analysis of large data sets.

Relevance to LANL: Large-scale simulations in the area of predictability will certainly need to include discrete components. The projects in discrete optimization will make tools available to modelers in this area. Discrete optimization, and MIP in particular, will also be important in the development of high performance compiler tools.

Milestones: (Year 1) Study computational aspects of discrete optimization and MIP. **(Years 2-5)** Develop prototype software for the solution of large-scale discrete models. Document the results in technical reports.

Methods and Tools for the Solution of Non-smooth, Multi-scale, Coupled Models

Investigator: Petr Kloucek, Frank Toffoletto

Project Description: The focus of this research project is on the development of computational methods and tools for a prediction of meso-scale phenomena. The computational techniques and massively parallel algorithms developed for this project will include domain decomposition coupled to operator splitting, nonlinear Galerkin and functional multigrid methods, adaptive mesh refinement of unstructured grids, and a technique for connection of non-matching grids.

Relevance to LANL: The proposed mathematical and computational framework is directly related to the predictability of low-dimensional physical events within systems with continuum multifractal structure. Such phenomena include dynamic fracture, large instabilities in random systems, pattern formation, grain growth, and nucleation.

Milestones (Year 1) Begin design of a massively parallel algorithm for the computation of multi-fractal structures in crystalline materials. **(Years 2-5)** Complete design of a massively parallel algorithm for the computation of multi-fractal structures in crystalline materials. Apply this method to model high-frequency loading of shape-memory alloys. Concentrate research on a development of novel computational techniques for non-smooth non-equilibrium systems. Document the results in technical reports and implement prototypes.

Methods and Software for Inverse and Control Problems

Investigators: Bill Symes

Project Description: Characterization and control of complex dynamical systems, such as wave propagation in heterogeneous materials, requires coupling of very expensive and inexact simulations, often involving subsimulations interacting at vastly different scales, with appropriate optimization and linear algebra algorithms. These algorithms must incorporate features not found in contemporary library code, such as management of the interaction of simulation and optimization errors, scalability with maximal architecture neutrality, and transparent management of data storage amongst devices of varying latency. It is virtually inconceivable that such tools could be built, maintained, and successfully transferred to end users without the employment of object-oriented programming methodology.

Progress towards these objectives will be achieved through extension and enhancement of an existing object-oriented numerical software library that already incorporates many of the desired features. Applications to inverse problems and geophysical signal processing and to control of structures subject to dynamic loads will serve as testbeds. Prior experience in coupling to automatic differentiation, automatic class generation from numeric kernels, and adjoint state approach will be leveraged to provide powerful rapid prototyping tools.

Relevance to LANL: This investigation will support the LANL effort in simulation, identification, and control of complex multiscale phenomena through elaboration of an established object-oriented approach to this class of problems.

Milestones: (Year 1) Demonstrate object-based coupling of automatic differentiation, automated class construction, adjoint state sensitivity, and control of large complex systems. Begin initial design of distributed vector base class. **(Years 2-5)** Complete design of distributed vector base class. Implement Arnoldi-based trust-region regularization of ill-posed inverse problems, subspace methods for multiple scales, and interior-point codes for inequality constraints. Perform a preliminary exploration of Java's suitability as a possible replacement for C++. Perform further investigations of algorithm structure, transparent data and operator distribution, and integration of graphical interfaces. Document the results in technical reports.

Efficient Parallel Solvers for Multiphase Flow in Strongly Heterogeneous Porous Media

Investigators: Roland Glowinski, Yuri Kuznetsov, Lennart Johnsson

The simulation of multiphase flow in porous media involves outstanding issues in Modeling and Scientific Computing. Among the computational issues there is the necessity of solving time-dependent nonlinear diffusion equations with highly discontinuous coefficients and anisotropic behavior. Very fast solvers are required for the following reasons:

- Due to the random nature of some of the data, Monte-Carlo techniques may have to be used, each sampling requiring the solution of a time-dependent problem involving a diffusion operator among others.
- The iterative solution of inverse problems (via adjoint equation techniques, for example) may require a large number of simulations and also the backward solution of adjoint equations.The adjoint equation approach for inverse problems will be effective only if there exist fast solvers for the direct time dependent equations modeling the flow and their adjoints.

As already mentioned, the heterogeneity and anisotropy of the diffusion coefficients make unfeasible the use of uniform (or quasi-uniform) finite element or finite volume meshes. A well-known advantage of uniform meshes is that they allow the use of fast solvers (such as multigrid or cyclic reduction based algorithms, for example) of optimal computational complexity. In order to overcome the above difficulties, and obtain efficient solution methods we intend to investigate a computational methodology based on the following components:

- Time discretization by operator-splitting in order to treat optimally the various operators associated to the physics and numerics of the problem under consideration.
- Space discretization by mixed/hybrid finite element or by finite volume methods in order to better represent the geometry complexities.
- Use of logically rectangular, locally refined meshes fitting the various lines or surfaces of discontinuity encountered in the problems.

- Parallelizable domain decomposition methods with the ability (via Lagrange multiplier based mortar techniques) to take into account possibly non-matching meshes at internal interfaces.
- Efficient preconditioners for the parallel solution of the saddle-point problems associated to mixed/hybrid finite element approximations and mortar techniques.
- Binary tree techniques to minimize the memory storage requirements associated to the implementation of adjoint techniques when solving inverse problems.

The parallelization of the mesh generators and solvers will be done in cooperation with Johnsson, Chapman and Subhlok and collaborators in the Department of Computer Sciences at UH.

Milestones: (**Year 2**) Parallel algorithms and numerical results for 2D/3D diffusion type equations by mixed finite-element and finite volume methods on rectangular grids in strongly heterogeneous media. (**Year** 3) Parallel codes for the preconditioned iterative solution of the 2D/3D diffusion problems discretized by mixed finite element or finite volume methods on logically rectangular, locally refined meshes. (**Year 4-5**). Integration of the diffusion solvers into the flow in porous media simulators. Investigation of the fast solution of inverse problems.

Parallel Multilevel Preconditioners for Large Scale Algebraic Saddlepoint Problems

Investigators: Yuri Kuznetsov, Roland Glowinski, Lennart Johnsson

A new efficient finite-difference method for the discretization of partial differential equations containing the operators "grad", "div" and "curl" has been proposed and investigated recently by researchers from Los Alamos National Laboratory and the University of New Mexico. The main advantage of this method is that in the case of strongly deformed and non-smooth logically rectangular meshes the finite difference operators preserve the fundamental functional properties of the original differential operators such as mutual adjointness, self-adjointness, positive definiteness and semi-definiteness. From these properties a number of conservation laws are naturally satisfied on the mesh level. Another important consequence of these properties is that the resulting system of finite difference equations can be formulated as algebraic systems with symmetric matrices in the saddle point form. These matrices have numerous applications in mixed finite element, finite volume and domain decomposition methods and their nice properties have been systematically explored by researches at UH. Development, investigation and application of efficient preconditioned iterative solvers including iterative analysis of saddle point matrices is one of the major research topics in the Department of Mathematics at UH. Since the early 1990's a number of efficient preconditioners based on coupling domain decomposition, fictitious domain and multilevel/algebraic multigrid methods have been developed and applied to numerical solution of partial differential equations. The application area of this technique covers fluid mechanics, porous media flows and electrodynamics.

This project will draw from the expertise of researchers at both LANL and UH in developing new efficient parallel preconditioned iterative methods for the solution of diffusion and advection--diffusion problems in the case of strongly heterogeneous non--isotropic media on a sequence of logically rectangular moving meshes. Coupling of the domain decomposition technique and algebraic/multilevel multigrid methods will be exploited to design parallel block

diagonal preconditioners for large scale algebraic saddle point matrices. The systems in the classical form with positive definite and semidefinite matrices will be considered as well. In the second stage of the project preconditioned parallel iterative methods for algebraic systems which arise from the mimetic finite-difference discretizations of the Maxwell equations on strongly deformed logically rectangular meshes will be the major point of research within this project.

Milestones: (**Year 2**) Algorithms and parallel code for the iterative solution of the 2D diffusion type equations in heterogeneous anisotropic media by the mimetic finite-difference method on logically rectangular strongly deformed meshes. (**Year 3**) Algorithms and parallel code for the iterative solution of the 3D diffusion type equations in heterogeneous anisotropic media by the mimetic finite--difference method on logically rectangular strongly deformed meshes. (**Year 4 - 5**) Algorithms and parallel code for the iterative solution of the 3D Maxwell and 2D/3D advection--diffusion type equations in heterogeneous anisotropic media by the mimetic finite--difference method on logically rectangular strongly deformed meshes.

13-APR-00