Proposal Cover Page

Title of Proposed Project

The Net100 Project – Development of Network-Aware Operating Systems

DOE Office of Science Notice 01-01:

Continuing Solicitation for all Office of Science Programs, Published December 7, 2000

Name of University:	Carnegie Mellon University
Principal investigator:	Gwendolyn Huntoon
Position title of PI:	Assistant Director, Pittsburgh Supercomputing Center
Mailing Address:	4400 Fifth Avenue, Pittsburgh, PA
Phone:	412/268-6354
FAX:	412/268-8200
Email:	huntoon@psc.edu

Name of Official signing for University:

Title of official:

Phone:

FAX:

Email:

Use of human subjects in proposed project: No

Use of vertebrate animals in proposed project: No

Signature of PI: Date:

Signature of Official: Date:

1 List of Participants

1.1 Pittsburgh Supercomputing Center

Principal Investigator: Gwendolyn Huntoon
Co-Investigator: Janet Brown
Co-Investigator: Matt Mathis

1.2 Lawrence Berkeley National Laboratory

Principal Investigator: Brian Tierney

1.3 Oak Ridge National Laboratory

Principal Investigator: Thomas Dunigan
Collaborator: William Wing

1.4 University of Tennessee at Knoxville

Principal Investigator: Richard Wolski

1.5 National Center for Atmospheric Research

Collaborator: Basil Irwin

2 Abstract

The well-known difficulties in obtaining good network performance for TCP-based applications without expert tuning or case-by-case application optimization can be overcome by building expertise into the operating system where it will benefit all users and applications.

The Net100 Collaboration (PSC, NCAR, UT, LBL, and ORNL) proposes to develop a model for network-aware operating systems using Web100 as the means for incorporating network information and its analysis into host operating systems to improve performance. To investigate how effective network-aware operating systems can be, we plan a three-phase approach. First, we will use the network-aware; Web100 based operating system that we develop to create a simple, bulk-transport application and demonstrate its use over high performance network links. We will then extend this model to support more advanced and complex applications, moving from point-to-point optimization to optimizations for fully distributed environments. Finally, as proof that a network-aware operating system can tune and optimize performance on behalf of applications, we will also develop application-internal tools (based on NetLogger) to monitor the efficiency of application support, and provide an external monitoring methodology (based on the Network Weather Service) to gauge the impact this system has on the rest of the network.

In addition to serving the needs of high-performance computing and network users, this project will serve as an (open source) showcase for network-aware operating systems, beginning with a Web100-based O/S. The tools, sources, measured results, and methods will be showcased on a "Closing the Wizard Gap" web site and made available to the high performance networking community.

3 Proposal Narrative

3.1 Background and Significance

Backbone speeds on the Research Internets have increased considerably in the last few years as projects like Internet II and NGI have moved forward with aggressive deployment of new technologies. At the same time, projects like NTON [NTON] and SuperNet [SuperNet] are providing a preview of the near future of wide area networks. Unfortunately, applications are rarely able to take full advantage of these new high-speed networks. TCP and applications using it require expert tuning at the end point hosts to achieve reasonable performance. This tuning is either hand optimization of parameters or very sophisticated coding by a network-knowledgeable "Network Wizard" programmer. Frequently network administrators are called on to help (often with erroneous complaints), and optimizations made on one day may be inappropriate the next. This requirement for constant nursing by a Network Wizard is clearly unacceptable. The gap between the network performance achievable at the hands of a Wizard and the performance seen by normal users has become known as the "Wizard Gap." The dearth of wizards or even apprentice-level application programmers will ultimately limit the success of the coming generation of high-performance networks and high-performance distributed applications.

The solution to this problem is to push the network expertise down, either into the operating system itself or into the network. Although someday intelligent networks may become a reality, we believe a more realistic near-term answer is a network-aware operating system. An operating system that is network-aware can dynamically adjust or tune the network services it is negotiating on behalf of applications. These improvements in tuning can transparently improve the efficiency for all hosted applications

Ideally, the underlying network infrastructure should be transparent to the end-user and applications. While there will always be a few applications that may need to have some knowledge of the underlying network conditions, this should be the exception and not the rule. Through the Net100 collaboration, we will begin the migration away from network-aware applications and towards network-aware operating systems. The Web100 instrumented kernel [Web100] provides the perfect mechanism on which to base this migration. It provides a transparent window into the inner workings of TCP, and a bidirectional API for reading and setting TCP's tuning parameters.

3.2 Proposal Overview

3.2.1 Project Summary

The Net100 Collaboration (PSC, NCAR, UT, LBNL, and ORNL) proposes to develop a framework for network-aware operating systems using Web100 as a model for integrating network-based information into host operating systems. The Net100 team is composed of research groups with the right combination of tools and expertise to take the Web100 kernel and refine it to be truly "network-aware", and to verify that the approach being taken will be valuable to a number of types of DOE applications. Net100 will bring together the following technologies and expertise:

- Web100 (PSC/NCAR): Web100 provides a novel architecture within which a variety of different performance sensing and control strategies can be implemented. This group provides the essential TCP expertise required.

- Network Weather Service (NWS) (UTK): NWS is a scalable, end-to-end network performance monitoring and analysis system. This group provides monitoring infrastructure and statistical analysis expertise. NWS will be hosted on systems, which are external to the

kernel-modified systems. NWS will be used to measure impact and "fair use" of applications running on the developing network-aware OS.

- NetLogger (LBNL): NetLogger is a toolkit for detailed distributed application instrumentation and analysis. This group provides application tuning expertise, and expertise on analyzing the complex interaction between the network, network protocols, and the applications. It will be used to instrument and analyze applications running on the modified systems.

- Probe (ORNL): The Probe project is working on optimizing bulk data transfers in a production supercomputer environment. This group provides alternative protocol expertise, a working testbed, and DOE mission-critical application expertise.

The core component of the Net100 project is a network aware operating system (NAOS). The NAOS collectively refers to kernel level modifications as well as API's, libraries, and daemons developed as part of the project to support the network-aware functions. In many cases actual NAOS components may depend on the base operating system and/or the underlying network technologies.

Net100 will use the above technologies and expertise as mechanisms for testing and refining Web100. however, there are some critical missing components which must be developed to allow a thorough analysis. These will also be provided by Net100, and include:

- The Network Tools Analysis Framework (NTAF), a framework for triggering NWS network measurement tools as well as other network monitoring tools, such as pathrate, pipechar, and Iperf. The NTAF will be provided by LBNL.

- The Network Analysis Information Base (NAIB), a monitoring data archive database that provides sophisticated relational queries and analysis. The NWS team at UTK will provide the NAIB.

- Instrumented applications and tools. This involves using the Web100 TCP instrumentation interface to collect TCP data from a variety of applications and monitoring tools, and use NetLogger to format the data send it to the NAIB. LBNL, ORNL, and the NWS team will do this.

The results of this monitoring and analysis will then be fed back to the Web100 team for further refinement of the Web100 kernel.

The goal of Net100 is to eliminate what has been called the "*wizard gap*". Through the integration of end-to-end and application-level monitoring capabilities with the tuning and diagnostic capabilities provided by Web100, we will develop a unique and general-purpose system for optimizing and understanding end-to-end network and application performance. This will allow us to create a network-aware operating system that will be able to maximize network utilization for a wide variety of applications and, without help from the wizards, eliminate the wizard gap.

3.2.2 Approach

In the short term, Net100 will integrate the network-aware, Web100-based operating system with simple, bulk-transport applications and demonstrate its use over point-to-point, high-performance network links. Long term, we will extend this model to more advanced and complex applications with a variety of bandwidth and latency requirements.

As part of Net100, we will also create a testing and analysis framework, NTAF, for combining network test and monitoring tools in a standardized manner. Initially this framework will be used to verify the network characteristics inferred by Web100 and the auto-tuned network-aware O/S. Two

well-known packages, the Network Weather Service (NWS) and NetLogger will form the core of this framework. This framework will be used to measure the impact the bulk transport utility has on the network, in order to guarantee optimal, yet fair, use of the network. The framework will be designed in such a way that other tools, in particular, new and emerging network diagnostic and monitoring tools, can be integrated into the framework in a standardized manner.

During the first year of the project, we will focus on using Web100 to build an auto-tuned bulk-transport utility, developing the NTAF, and using it to validate the results. The goal for the first year is to fully utilize high-performance links for transferring large data sets while also demonstrating the validity of network-aware operating systems. During Year 2 we will integrate this package into distributed application environments, such as high-energy physics Data Grid applications. We will continue to use the NTAF along with the NAIB for validating our results, while also identifying and integrating new active and passive monitoring tools into the framework.

By year 3 we plan to have a fully realized net-aware open source operating system such as Linux, with the additional goal of working with the IETF and vendors to encourage the use of these techniques in vendor supplied operating systems as well. We will also investigate the integration of active and passive monitoring tools into the kernel itself. As in Year 1 and 2, we will use DOE specific applications to test the validity of these modifications, verifying them with the NTAF and NAIB tools.

Beyond this project we imagine a “network instrumentation and control plane” to provide network managers with unified access to the information and controls within the network-aware operating systems. Such a control plane would give network engineers very strong tools to manage their networks, including an accurate picture of how the network and applications interact.

In addition to serving the needs of high-performance computing and network users, this project will serve as an (open source) showcase for integrating the network-aware operating systems, beginning with a Web100-based O/S, with real applications. The tools, sources, measured results, and methods will be showcased on a "Closing the Wizard Gap" web site and made available to the high performance networking community.

3.2.3 Experience and Competance

3.2.3.1 Web100 Project

PSC and NCAR have worked on both individually and together on a number of projects relevant to the Net100 project and are summarized below.

Web100 is collaboration between the Pittsburgh Supercomputing Center (PSC), the National Center for Atmospheric Research (NCAR) and the National Center for Scientific Applications (NCSA). Initial development of Web100 is funded by a three-year grant from the National Science Foundation (NSF) through grant [ANI-0083285]. Work under this grant includes the initial development, deployment and documentation of core Web100 instruments and library functions for the Linux Operating System.

PSC and NCAR have also collaborated over the past five years on the NLANR project. Gwendolyn Huntoon and Matt Mathis from PSC are co-principal investigators on the NLANR Engineering Services (NLANR ES) project, funded through Cooperative Agreement ANI-9720674 with the National Science Foundation. NLANR ES staff at PSC and NCAR provide expert consulting to high performance network engineers, develop documentation on new and emerging high performance networking technologies, and provide tools for diagnosing and understanding network performance. The NLANR ES performance tuning [Perf_Tune] is an authoritative source for information on how to hand-tune a wide variety of operating systems for use over the national high-performance network backbones. NLANR ES co-sponsors the NLANR/I2 Joint Techs Workshops with Internet2, which has become the primary forum for disseminating and exchanging information in the high-performance network community.

Matt Mathis was also the principal investigator on the NSF-funded TCP Enhancements project (ANI-9870758). Under this project, PSC has made a number of contributions to TCP technology, including being the first authors on RFC2018 [RFC2018], "TCP Selective Acknowledgement Options". The SACK RFC has attained wide deployment in the network community. Ongoing research on TCP congestion control has resulted in the development of several algorithms that improves TCP performance in high-performance environments [MM96, MSJ99]. PSC also did groundbreaking work in TCP autotuning [SMM98].

3.2.3.2 Lawrence Berkeley National Laboratory

The LBNL Distributed Systems Department has a great deal of experience in building and tuning data-intensive, wide-area distributed applications. The NetLogger Toolkit, described below, was developed in order to tune and debug distributed applications, has been used extensively in several high-speed networking applications, including the DPSS[DPSS], Radiance, BaBAR[BaBar], Visapult[Visapult], and GridFTP[GridFTP]. This experience has made us highly aware of functionality gaps in the currently available monitoring facilities, and the difficulties of getting the applications and the operating systems to fully utilize high-bandwidth network links.

At the Supercomputing 2000 conference LBNL received the "Fastest and Fattest" Network Challenge award for the application that best utilized the wide-area network. The application was a remote data visualization application called Visapult, which used a peak rate of 1.5 Gbits/second, and a sustained data rate of 680Mbps. This performance was the result of a great deal of hand tuning of the SC2000 network. Visapult transforms data from a simulation using parallel compute nodes and transmits the transformed data in parallel over the network for rendering. The dataset, 80 GB in size, was stored at LBNL and the compute cluster, an 8-processor SGI Origin with 4 Gigabit Ethernet interfaces, was in Dallas, TX. Parallel reading of large datasets stored at a distant location is common in high-energy physics (HEP) applications.

LBNL has also been heavily involved in the Global Grid Forum (GGF) [GGF] and, of particular interest here, the GF Performance Working Group. LBNL co-authored the white paper for a monitoring data architecture called the Grid Monitoring Architecture (GMA)[GMA]. Through our GGF work we have developed a thorough knowledge of the issues involved in designing and a distributed monitoring system. The input into the GMA was drawn from experiences developing a system at LBNL called Java Agents for Monitoring and Management (JAMM).

3.2.3.3 Oak Ridge National Laboratory

ORNL has been active in applied network research since the mid 70's. ORNL developed protocols and applications as part of DOE's early fusion energy research network (MFEnet) in the 70's. The Computer Science division developed protocols, software libraries, and applications as part of ORNL's leading research into distributed, parallel computers (hypercubes, mesh) in the mid 80's. ORNL's research and development in distributed computing is highlighted by the development of the widely used Parallel Virtual Machine (PVM). PVM-related developments at ORNL included adding IP multicast and authentication/encryption to PVM. ORNL deployed one of DOE's first ATM test beds, and PVM was extended to utilize AAL5 over ATM. ORNL extended PVM and modified applications to interconnect Intel Paragon supercomputers at ORNL and Sandia over ATM on DOE's ESnet. In recent years, ORNL has applied its network expertise to intrusion detection systems based on characterizing IP flows and to detecting hidden information in IP packet headers (IP steganography). ORNL operates a DOE-funded ATM testbed with ATM analyzers tightly coupled to GPS time sources that presently provide insights into the behavior of DOE's ATM-based ESnet. Ongoing research includes work on alternate and multi-path routing using application-based routing daemons to improve both latency and throughput. ORNL also has active research in storage-area networks (Probe/HPSS) and in improving bulk data transfers across high bandwidth, high delay networks.

3.2.3.4 University of Tennessee at Knoxville

Researchers at the University of Tennessee, Knoxville (UTK) working on the NWS project have developed and deployed several software products in research and production computing settings.

Dr. Rich Wolski is current using the system to conduct scalable network performance monitoring and forecasting research and development in several different settings.

As an NPACI partner, Dr. Wolski and his group working with NPACI programming support maintain the NWS as a nationally available network performance monitoring resource. NPACI distributed computational science codes such as MCell (a molecular docking application) and The Wide-Area Electron Microscopy project use NWS real-time performance forecasts to determine how they will load the networking resources at their disposal. These codes are hardened, production-quality programs that are gaining an ever-wider user community through NPACI. The persistent NWS installation maintained by UTK and NPACI is an integral part of their success.

In addition to supporting NSF's computational science mission, Dr. Wolski's group is part of the NSF funded "Grid Applications Development Software" (GrADS) project, headed by Dr. Ken Kennedy, and funded under the NGS program (EIA-9972889). GrADS is a comprehensive Grid research project that attempts to codify the software technologies -- from the application level to the infrastructure level – that are necessary to support large-scale, wide-area high-performance computing. In this context, the NWS provides near-real time forecasts of TCP/IP network performance (end-to-end latency and bandwidth), available real (non-paged) memory, and CPU availability. GrADSoft (the software infrastructure being developed by GrADS) is a large-scale software system, within which the NWS is being fully integrated. Our experience with GrADS in coordinating NWS research and development in the context of a collaborative research project will certainly prove beneficial to this work.

The NWS current has also developed an international user community that is currently being supported by Dr. Wolski and his group. In addition to the PACI commitments, the Asia-Pacific Grid project (APGrid – funded by the Japanese government through ETL and TIT), and the Globus DataGrid Projects have all adopted the NWS infrastructure to provide near-real time measurement and forecasting of network, memory, and CPU performance response. In addition to these advanced-development and production computing customers, Dr. Wolski's group also supports several research groups at UCSD, the Free University at Amsterdam, the Netherlands, and ENS-Lyon, France

3.3 Preliminary Studies

3.3.1 Web 100

The goal of the Web100 project is to create a software suite that enables a host-software environment that will run common TCP applications at 100% of the available bandwidth, regardless of the magnitude of a network's capability. The project is based on the following observations: existing host system software is often optimized for low bandwidth environments; it requires a network wizard to optimize the host-software environment in order to improve host TCP performance; and, there is a general lack of host-based TCP instrumentation and tools needed to diagnose end-host (and end-to-end) performance problems.

The Web100 software suite is a set of instruments, referred to as the kernel instrument set (KIS) or derived instrument set (DIS), that collects as much TCP specific information as possible to enable a user to isolate performance problems associated with specific TCP connections. Kernel instruments are based on values or variables that can be directly collected within the host. Examples of Web100 KIS variables include:

- TCP State: A read only integer value representing the connection state from the TCP State Transition Diagram. The reference for this variable is RFC2012 (TCP-MIB v2) Transmission Control Protocol using SMIv2. It is patterned after the Connection Table in RFC2012.
- SACK Enabled: A read-only variable that indicates if SACK has been negotiated ON by both ends of the connection. The reference for this variable is RFC2018 TCP Selective Acknowledgement Options.

- Send Buffer Max: A read-write variable that sets the maximum buffer memory. This variable is specifically used for auto-tuning.
- Send Buffer Min: A read-write variable that sets the minimum buffer memory. As with Send Buffer Max it is specifically used for auto-tuning.

Derived Instruments are those that are directly computed from KIS variables. For example, the Web100 KIS does not include a direct instrument of loss probability, but it does provide direct instruments on the numbers of packets lost and transmitted. The ratio of these two KIS instruments is a derived estimate of the loss probability. The library also provides an extensible set of common derived instruments, however, many Web100 tools will also define their own (non-common) derived instruments.

The Web100 software includes an API and library that provides access to the underlying KIS and DIS information. Specifically, the library includes a TCP auto-tuning tool that provides a mechanism for tuning each TCP connection within the host. While this is not the main focus of the project, it begins to address the wizard gap by optimizing network performance without requiring intervention by a network wizard.

The first stable test version of Web100, referred to as Web100 Alpha0, was released to a group of early adopters in mid-March 2001. This release, as with most releases in the near future, is based on a version of the Linux operating system. The Alpha0 release contains test applications including a basic graphical interface for examining and monitoring variables, as well as scripting primitives and sample code to quick-start additional developers. The Alpha0 KIS has 75 total instruments, of which seven are read/write. All the collaborators on Web100 are among the 9 groups identified as early adopters for the Web100 code thus are in a position to provide early feedback to the Web100 developers.

3.3.2 Probe

The Center for Computational Science (CCS) at ORNL has been designated as a Prototype Topical Center, providing massive computing resources in support of a selected set of problems in computational science. Examples include climate modeling, computational chemistry, and computational biology or bio-informatics. In practice, this means that large data sets, move between DOE's flagship supercomputer center, the National Energy Research Scientific Computing Center (NERSC), and the CCS with some frequency.

ORNL's Computer Science and Mathematics Division have established the Probe testbed to study challenging storage-related and network-related problems [Probe1]. NERSC is a partner in Probe, providing the second node in a wide-area distributed-storage testbed utilizing DOE's ESnet network. Probe is involved in testing new equipment (servers, network interfaces and switches, storage devices), new protocols (scheduled transfer, for instance), and software (the High Performance Storage System, database products, etc.). Probe will be one of the drivers for SurgeNet (another SciDAC proposal), the proposed bandwidth-on-demand OC48 connection between ORNL and NERSC

During the past year we have sought to optimize bulk transfers between the Probe nodes. Large data set transfers of Global Climate Modeling data sets by naïve users between NERSC and ORNL were frequently able to realize throughput of only 250 Kbytes/sec over DOE's ATM/OC3 ESnet network. The NERSC-ORNL link provides high bandwidth (150 Mbs) and high latency (60 ms round-trip time, RTT). FTP was modified to request larger TCP buffer sizes and the maximum buffer sizes were increased in the AIX TCP stacks on the Probe notes. This improved typical data rates to 2 MBytes/sec.

We modified various TCP options in the AIX TCP stack to try to improve performance further and uncovered two bugs in AIX's NewReno and SACK TCP modes that were limiting performance. To better understand the ORNL-NERSC network performance, we experimented with modifying other TCP stack options both through network simulations ns [Ns2] and by developing an instrumented,

TCP-like stack at the application-layer over UDP [Dunigan]. This UDP test harness is similar to PSC's *TReno* [Mathis96]. Our objective was to adjust TCP parameters to avoid packet loss and to speed startup and recovery, but yet remain TCP-friendly. We also did some experiments with parallel TCP streams between ORNL and NERSC using iperf [iperf] and rate-based UDP flows.

3.3.3 Netlogger

The LBNL *NetLogger Toolkit* [NetLogger] is designed to monitor, under actual operating conditions, the behavior of all the elements of the application-to-application communication path in order to determine exactly where time is spent within a complex system.

Our experience has shown that often distributed application developers blame the network for their performance problems, when in fact the problem is application design. Often better pipelining of I/O and computation by the application will lead to dramatically increased performance.

The NetLogger Toolkit is designed to solve the problem of determining whether the bottleneck is the network or the application. Using NetLogger, distributed application components are modified to produce timestamped logs of "interesting" events at all the critical points of the distributed system. The events are correlated with the system's behavior in order to characterize the performance of all aspects of the system and network in detail.

All the tools in the NetLogger Toolkit share a common log format, and assume the existence of accurate and synchronized system clocks. The NetLogger Toolkit itself consists of three components: an API and library of functions to simplify the generation of application-level event logs, a set of tools for collecting and sorting log files, and a tool for visualization and analysis of the log files. In order to instrument an application to produce event logs, the application developer inserts calls to the NetLogger API at all the critical points in the code, then links the application with the NetLogger library.

We have found exploratory, visual analysis of the log event data to be the most useful means of determining the causes of performance anomalies. The NetLogger Visualization tool, nlv, has been developed to provide a flexible and interactive graphical representation of system-level and application-level events.

3.3.4 Network Weather Service

The Network Weather Service (NWS) [NWS] uses up to date performance histories gathered from network and computational resources and statistical forecasting techniques to make short-term performance predictions for those resources.
To provide this functionality NWS:

- Operates a distributed set of performance sensors, from which it periodically collects performance measurements.
- Applies a set of statistical forecasting techniques to individual performance histories
- Generates forecast reports for the resources being monitored, which it disseminates via a number of different APIs.
- The goal of the system is to apply forecasting methodologies to fresh performance monitoring data gathered from Network and Computational Resources resources to make forecasts of available performance in near-real time.

The NWS was developed primarily to support application level scheduling. Automatic scheduling agents, such as AppLeS. [AppLeS] Can consult the forecasts it makes to decide, dynamically, which resources to use when a program is executed. More recently, we have been using the NWS in an operational setting to detect network performance faults such as those caused by routing table misconfiguration [CCGrid]. By comparing NWS forecasts to the expected performance between two end-points, we have been able to detect when the route between those endpoints changes due to a misconfiguration.

3.3.5 Other Related Projects

Many tools already exist to help diagnose, tune and understand end-to-end network performance. However, in most cases, these tools are stand-alone tools, providing insight into one aspect of network performance, not an integrated view of network performance. Brief descriptions of existing tools, including ones that will be incorporated into the Net100 toolkit are described below.

One of the most useful tools for understanding end-to-end performance is ***Iperf***. Iperf, developed at NCSA by NLANR DAST, is a TCP and UDP bandwidth-testing tool similar to but less limited than ttcp and nettest. It consists of a server and client to measure network characteristics between two points. This tool has become one of the standard testing tools within the high performance networking community [IPERF].

Nettest is a framework application produced by Lawrence Berkeley National Laboratory which makes use of SSL to provide a secure measurement infrastructure. A variety of existing and new network tools can be easily incorporated into it. Currently nettest includes iperf, a simple ping tool and a TCP based tool to determine optimal buffer sizes to meet specified bandwidth requirements [Nettest].

Another tool of interest is ***pchar*** - an end-to-end path characterization tool developed by Bruce Mah of Sandia National Laboratories. This tool is originally based on algorithms developed by Vern Paxon for use in pathchar; it can be used to determine bandwidth, latency, and loss on a per link basis. Using UDP packets of varying TTL the metrics are computed on the basis of ICMP reply responses. While similar in many respects to pathchar this is an independent implementation that has been improved over several versions [pchar].

Pathrate is a new tool from the University of Delaware (with CAIDA) that is designed to measure both the capacity and the available bandwidth of a network path. Measuring link capacity is crucial for debugging, calibrating, and managing a path. Measuring the available bandwidth, on the other hand, is of great importance for predicting the end-to-end performance of applications, for dynamic path selection and traffic engineering, and for selecting between a number of differentiated classes of service. pathrate is designed to be robust to cross traffic effects, meaning that it can measure the path capacity even when the path is significantly loaded. Pathrate uses UDP-based measurements with a TCP control channel, and unlike pchar and pipechar, requires both a source and a sink [CAIDA].

The tool ***pipechar*** was developed by Jin Goujun at the Lawrence Berkeley National Laboratory and is used for initial characterization of network problems via bandwidth measurement. Pipechar is different that the other tools in that it is a sender only routine, thus do not need a paired receiver in order to locate unresponsive routers and hosts. Pipechar makes use of UDP and ICMP responses in very sensitive manner to make its analysis and only one pipechar may be run at a time on a host. While a standalone tool, it was designed to be used in conjunction with LBNL's nettest [pipechar].

The most popular passive monitoring package is ***CoralReef***. CoralReef is a trace collection and analysis tool developed at CAIDA. It passively monitors OC3 and OC12 connections via dedicated hardware (PCs) and will capture, analyze and generate reports based on aggregate traffic across an interface. This package includes all necessary tools, drivers, examples and documentation. [CR]

While many of these tools do measure similar network characteristics, their results cannot be collected and compared in a standard, useful manner. Through the NTAF, Net100 will develop a framework for launching many of these existing tools along with NWS to measure and predict network characteristics. Most of these tools are active network probes; by contrast, Web100 passively collects network performance for a specific application as it is running. The Web100 data can be used to modify the behavior of the TCP stack to improve performance of the application as it runs. The Web100 data will also be collected by the NAIB and archived. This data can then be used to verify the predictive information provided by the other tools. OS tools such as *netstat* can provide some information on TCP performance, but the statistics apply to all activity on the system, whereas the Web100 data is specific to a specific TCP flow. There are other passive tools like

tcpdump/tcptrace/xplot that can be used to analyze a specific application flow but these tools do not provide real-time feedback to the running application.

3.4 Research Design and Methods

This section provides a detailed description of the design and development of the four main components of the Net100 project.

3.4.1 Network-aware Operating System

We will use Web100 as the basic building block for developing a network-aware operating system. As indicated above, the Web100 software suite consists of a number of components including: instrumentation in the kernel itself (KIS), derived instruments (DIS) based on the kernel instruments, an API (Application Programming Interface) to access the instruments, and a library of functions. We will enhance and expand these components to provide detailed networking information to the host operating system. The Web100 software suite is currently based on the Linux operating system. Our work will also focus on Linux. However, as with the Web100 project, our instrumentation and library functions will be documented in a standardized manner so that they can easily be incorporated into other operating systems. At all times during the project, we will utilize the most recent and stable version of the Web100 code. The current version Alpha0, provides a core set of KIS and DIS variables, a basic API and a small set of library functions.

In working towards a network-aware operating system, we will start by understanding existing Web100 instrumentation with respect to the Net100 bulk transport application. Specifically, we will analyze the relationship between the current collected information and what information is used or needed by the host operating system. We will consider the implications for both optimizing network and application performance. In an iterative manner, we will add enhancements to the core instrumentation based on our observations and analysis. We will work towards the specific goal of using a network-aware operating system to optimize bulk transfers over a high performance link with the ability to achieve close to full utilization of the link itself.

Once point-to-point, bulk transport issues are understood, we will expand and repeat the process, this time focusing on multi-site distributed applications. Once again, in an iterative manner, we will refine the instrumentation set, API and libraries to support the application.

At the same time, we will begin to develop the mechanisms needed to align and compare the Web100 instruments with existing monitoring and diagnostic tools. We will begin by using the Network Tool Analysis Framework as a way to standardize the collection and archival of the data. We will modify and enhance the instruments as necessary to facilitate the comparison with results from other tools. This aspect of the project requires close collaboration between all four groups on the project.

As discussed below, the NTAF will be developed to verify and validate the results from both the underlying network-aware operating system as well as the application itself. During Year 2 of the project, we will develop methods and protocols, if necessary, to provide remote access to the expanded NAOS instrument set. Currently, Web100 instruments are accessed only from the host using the API and library functions to collect and utilize the data. We will investigate ways of accessing this information, from using existing SNMP mechanisms to developing a new protocol if necessary.

By Year 3 of the project, we will have begun to unify performance measurements across all layers in the network stack. Currently Web100 is focused on instrumenting only host TCP characteristics. We will begin to investigate what other - both network and application specific - information should be collected and how it can be used to provide better, more efficient, network performance.

There are several other research issues that we will address as part of Net100. The first is the frequency at which monitoring should be performed. Running the network tests more often will

provide more accurate results, but tools like *iperf* and NWS put a load on the network, and therefore should be run as little as possible. We will use the NAOS and other Net100 tools to try to determine the optimal test frequency.

Another interesting research issue that we will address is the use of parallel streams by applications. Many people have observed that the use of parallel TCP streams can improve overall TCP throughput [psockets]. However there are several members of the IETF that worry about the widespread use of this technique, and what the impact of parallel streams will be on both TCP congestion control algorithms and the Internet in general [RFC2914]. Net100 will provide the perfect set of tools to research this topic. We can use Net100 to determine what the performance issues on a given link are, to monitor how TCP is behaving, and to determine whether or not parallel streams are appropriate. We can also use the NAOS to monitor the effects of the parallel streams on each other. Net100 will make it easy to compare results using a variable number of streams, and allow us to correlate data from active monitors, passive monitors, and applications.

3.4.2 Applications

Oak Ridge National Laboratory's contribution to the proposed research consists of three activities. First, ORNL would provide a production environment in which to test and evaluate the Net100 tools. ORNL has applications in global climate modeling, genomics, and atmospheric radiation monitoring that require transferring massive amounts of data over wide area networks. ORNL in conjunction with NERSC has already been investigating optimizing bulk transfers over DOE's ESnet as part of the probe [Probe1] and HPSS [HPSS] project. ORNL will use the Net100 tools and API to modify file transfer programs to evaluate their ability to optimize bulk transfer over high bandwidth/delay networks. ORNL's present ESnet connection is OC3 migrating to OC12 this spring. It is anticipated that an "on-demand" OC48 link will be available that will interconnect the Bay Area ESnet sites, ORNL, and possibly Argonne National Laboratory as part of another SciDAC proposal. These high-speed nets and distributed applications will provide quick feedback as to the effectiveness of the Net100 toolset. ORNL will work to integrate the Net100 tools and API into HSI, a widely used interface to HPSS.

A second ORNL research activity will be to look at other extensions to the Net100 toolkit and TCP stack that would specifically benefit DOE applications and fully utilize DOE's high bandwidth ESnet. ORNL will investigate other TCP parameters that might be monitored or tuned by the operating system with the Net100 tools. Possible parameters would include:

- Altering initial segment size and/or using a bigger MTU/MSS to speed TCP startup and recovery after loss [Allman, Floyd1, RFC2414]
- Altering the AIMD (additive increase, multiplicative decrease) parameters to speed recovery after packet loss. Over high delay networks, TCP favors "closer" flows [Hend]
- Doing "tentative" cwnd/ssthresh adjustments on timeouts like FreeBSD
- Avoiding loss by sensing congesting and slowing transmission [Ahn], or utilizing ECN from routers [RFC2481], and/or limiting transmission bursts.

We will study link characteristics that could be advantageously retained between runs of a network application. Retained characteristics might include RTT, RTTVAR, ssthresh, loss rate, and alternate routes. We will investigate what information routers might provide to the Net100 tools for characterizing links.

We will also investigate alternate protocols like SCTP [SCTP] that support out-of-order packet delivery. Bulk file transfers could take packets out of order and do scatter/gather reads and writes to keep the receiver's window "open" and thus possibly perform better than TCP. Out-of-order packets occur because of packet loss and retransmission, but also may occur if the transport protocol or application supports parallel streams and/or multiple paths.

Evaluation of these parameters and tools will be done over DOE's ESnet, and by emulation (to have repeatable test cases), and by simulation with ns [Ns2]. It will be necessary to insure that optimizations are still "fair" to competing network flows [RFC2914].

The third research activity will be to work with other vendors in incorporating the Net100 tools and kernel modifications in their products. The current Web100 tookit uses Linux, and ORNL will deploy Linux Web100 nodes at ORNL and NERSC for testing, but DOE and ORNL have large computer resources based on AIX, Solaris, IRIX, and Compaq's OSF. As part of the Probe network research, ORNL has worked with IBM to optimize their TCP stack for DOE's high bandwidth/delay networks. ORNL will continue to work with the vendors to understand how to tune the operating system and DOE/ORNL applications to provide best network performance.

3.4.3 Network Tool Analysis Framework

Another new component that will be developed by Net100 will be a "Network Tool Analysis Framework" (NTAF). This framework will allow the comparison of data collected from various active and passive monitoring tools, along with application instrumentation, providing the necessary input to evaluate which data are the most beneficial to enable network-aware operating systems.

The starting point for the NTAF will be an existing LBNL developed service called Enable [Enable], which provides the ability to configure and schedule the execution of a variety of network testing tools, and save the results in a database. The design of Enable will be enhanced to make it easy to incorporate new monitoring tools, and to send the results to the NWS NAIB database. We will use the Web100 TCP-KIS interface to instrument a number of state-of-the-art network monitoring tools, and then use NetLogger to format and send this data to the NWS NAIB database.

The NTAF will provide a flexible layer for configuring and triggering a wide variety of network test tools, and aggregating and transforming their results. The NTAF will be configurable, able to combine monitoring data in various ways, thus allowing the design of experiments that compare changes in network performance using various combinations of monitoring data.

The ability to run several different types of monitoring tools and collect the results in one place will allow us to "monitor the monitors". For example, we could run a passive *Berkeley Packet Filter* (BPF)-based tool on the same network as NWS sensors. This tool could measure the load that NWS is placing on the network, and thus allow us to determine the extent to which NWS is interfering with other traffic on that network. In this manner, we can easily measure the network overhead of many types of active monitoring. NTAF will provide fine-grained control over the execution frequency of network monitoring tools; this is particularly important for active monitoring tools, which may otherwise perturb the network.

The results from several different tools that measure the same basic metric, such as bandwidth, will be used as input to the NTAF, so that their usefulness in determining tuning parameters can be compared side-by-side. Statistical analysis methods built into the NWS NAIB database will be used to determine which tools are the most consistent predictors of application performance.

We will also develop an "application friendly" client API to the NTAF, and work with several Data Grid applications, such as the PPDG project, and with Grid middleware service developers, such as Globus, to test what types of information are most valuable to the applications.

We now present a sample usage scenario on how the NTAF might be used. All tools will use the Web100 TCP-KIS interface to collect TCP information from the Web100 kernel, and then use NetLogger to format and send this data to the NWS NAIB database. The NTAF could be configured to run the following network tests every few hours over a period of several days:

- ping -- measure network delay
- pipechar and pathrate -- actively measure speed of the bottleneck link

- iperf -- actively measure TCP throughput. Multiple *iperf* tests could be run with different parameters for the number of parallel streams {e.g.: 1,2,4} and the method of tuning the TCP buffers {Web100 auto-tuned, hand-tuned}
- NWS -- measure and predict network delay and bandwidth using NWS' own sensors

Using this configuration, there is a tremendous amount of analysis that may be performed, including:

- The ability to compare Web100 tuned throughput to hand-tuned throughput.
- The ability to compare NWS predicted bandwidth with application and *iperf* bandwidth.
- The ability to determine the advantage, if any, of parallel data streams, using both hand-tuned and autotuned (Web100-tuned) TCP.
- The ability to see the variability of the results over time.
- The ability to compare pipechar and pathrate to see which is most accurate.
- The ability to measure the impact of tuned TCP streams on non-tuned streams.

This level of analysis requires the ability to easily configure and trigger the desired set of monitoring tools on the desired set of hosts, and then to collect all the information in a common format and in a central location. This would be almost impossible without the framework provided by NTAF and the NWS NAIB.

3.4.4 Network Analysis Information Base

A network-aware operating system must be able to provide its users with a "picture" of how system level performance response is impacting application execution. The system may report packet-traffic along a network route between two communication machines, but that information does the user no good unless it can be used to estimate end-to-end network performance at the application level.

To be truly effective, then, a network-aware operating system must be able to provide predictions of future performance levels to support resource allocation, and to correlate system-level performance data with observed application performance. Both of these requirements can be met by using Net100 as a framework for integrating the Network Weather Service (NWS) and Netlogger to create a new abstraction performance analysis abstraction. Our goal is to provide a time-sensitive performance analysis detailing the various performance metrics available from different network components and to correlate that information with the performance a user experiences when using the network.

Our plan is to develop a "Network Analysis Information Base" (NAIB), based on the NWS and Netlogger, that uses NWS forecasting techniques to construct, dynamically, a picture of the performance that is available from a set of network resources. By comparing the information gathered by the NWS to the performance observed by an application loading the network, we will be able to validate the efficacy of the Net100 toolkit.

Information will be provided to the NAIB from a number of different measurement sources. We will develop an input interface so that NetLogger data can be incorporated. NetLogger provides a common data format for many different resource and application performance measurements. NetLogger formatted monitoring data collected via the NTAF, in addition to data from the standard NWS sensors, will be stored in the NAIB, along with network configuration information collected by traceroute and SNMP. Then, using the NWS statistical models, we will construct an aggregate "picture" of the performance an application can expect when it chooses to load a given set of network resources. At the same time, since application instrumentation data taken from NetLogger will be available, the resulting service will be able to provide real-time measurements of how well observed application performance matches predicted performance. The NAIB therefore constitutes provides a validation facility for Net100 research.

To develop NAIB we will focus on three goals.

- We will develop an extensible infrastructure for management of performance data that is capable of incorporating measurement data from any available measurement facility, using Netlogger as the primary data-delivery mechanism, and Net100 as the integration technology.

- We will investigate forecasting models that are able to combine performance measurement data originating from different sources to form a composite forecast topology of future conditions.

- We will deploy the NAIB as a Net100 tool over ESnet both to verify the efficacy of our techniques and to provide topology service to the wider DOE research community.

3.4.4.1 Developing the The Network Analysis Information Base

Currently, there are a variety of dynamic performance measurement systems deployed for the Internet, but from the perspective of dynamic analysis, they suffer from two common limitations. The first is that they capture measurement data reflecting what conditions were, and not how they will be, and secondly they usually make results available graphically or via HTML. We propose to address both of these issues by developing the NAIB so that it

- can gather and serve data via programatic as well as graphical interfaces,
- incorporates low-level resource measurement, high-level application instrumentation, and forecast data, and
- provides the performance response necessary to make near real-time analysis and validation of user performance useful.

One of the primary concerns in the architecture of the NAIB is scalability. Our initial efforts have been to employ Oracle's commercial RDBMS system, utilizing the Partitioning and Parallel options to achieve storage of and rapid access to this vast volume of data.

Further, it is quite clear that only a certain amount of this data can be archived in the central data repositories. The balance will need to transit the network from producer to consumer as appropriate to minimize wasted effort. To this end, we will leverage our relationship with the Logistical Computation and Internetworking (LoCI) effort on-going at the University of Tennessee. One of the goals of LoCI is to enable the efficient management of shared but distributed state in the network. Results from this NSF-funded and previously DOE-funded effort will be highly relevant to our work with the NAIB.

We will interface the NAIB to both "active" and "passive" measurements sources via Net100. In particular, we wish to incorporate active data gathered using both hard and soft collaboration probing methodologies as well as path-oriented probing methods such as those employed by traceroute, pathchar, clink, and pchar. Examples of passive measurements include periodic snapshots of routing table state, network device utilization, packet traffic counts and Cisco Netflow-style flow data. To the extent that this data is available in Netlogger format, we will choose this method of data incorporation above all others.

To achieve a quality deployment, the staffing requirements for the NAIB component of Net100 cannot be met with graduate students alone. Clearly, we will need to achieve a new set of research results in order to develop performance topologies. To make those results available to DOE users via Net100, however, will require the skills and professional commitment of a full-time scientific staff member.

3.4.4.2 Application Performance Verification

Once the NAIB is in place, we will provide an application performance interface that will be compatible with Netlogger. The goal of this software component to allow the application user to query the NAIB for an estimate of network performance between two end-points, and the compare the performance prediction to the actual performance that the application achieves. By building both the prediction mechanisms and the means to observe the effectiveness of the predictions into Net100, the resulting functionality will offer DOE high-performance network users a uniquely transparent system. All performance data will be available as well as measurement of the error associated with using it to predict application performance. We believe that this level of instrumentation and analysis sophistication will be invaluable as DOE users tune their network applications.

3.5 Outreach

3.5.1 Dissemination and Technology Transfer

The "Closing the Gap" WWW site, distributed between and supported by each of the collaborators on the project, will be used as a mechanism for disseminating information associated with the Net100 project. This web site will highlight current Net100 research activities at each site as well as provide a mechanism for code distribution, documentation, test results and Frequently Asked Questions (FAQ).

A primary goal of the Net100 project will be the dissemination and use of the Net100 components within the community. By using Web100 as a basic building block, we can work with existing DoE Web100 users to test and deploy the NAOS. We will work with the Web100 project, as well as the broader DoE and high performance networking community to provide a mechanism for easily distributing any software developed within the Net100 project. Through ORNL and LBNL we will develop core Web100 and NAOS expertise within the DoE community, expanding it to other DoE laboratories, applications and projects. While not the focus of this project, feedback from this broader community will be used when possible to influence the design and development of NAOS functionality.

Technology Transfer is critical to the long term adoption of network-aware operating systems. To this extent, we will provide standardized documentation of the NAOS so that the vendor community can easily adopt it. Also, by the end of the project, we will make all operational software components of the Net100 project publicly available. Results collected and archived within the NAIB will also be available on a widespread basis. Where possible, we will attempt to establish a collaborative partnership with network backbone provides, such as Qwest and the ESnet administration, to make our findings available to them in support of their on-going diagnostic and capacity-planning activities

A key advantage of the Net100 approach is that it will result in an implementation prototype with real operational utility. We believe, based on our previous experience with other Net100 related projects the core Net100 components, from the NAOS itself to the NAIB, will enable other computational and computer science efforts at DOE that focus on large-scale distributed computing.

The "Closing the Gap" www site will be used as a mechanism for disseminating information.

3.5.2 Relationship to other projects

LBNL has submitted a SCIDAC proposal titled “Self-Configuring Network Monitoring” that would create an infrastructure of passive and active monitoring on ESNet, with results stored in a network assessable monitoring data archive. The proposal will provide an infrastructure where monitoring sensors can be triggered on the fly based on some predefined criteria. For example, if active monitoring shows that the speed of a given link is substantially less than is was in the past, passive

monitors will be started to analyze the cause of the performance decrease. If this proposal is funded, we will interface Net100 with data from that archive.

We plan to leverage our ties to several large "Data Grid" [DataGrid] projects to provide a realistic test environment for the tools and techniques developed in this proposal. These projects include he Particle Physics Data Grid [PPDG], GriPhyN [GriPhyN], the EU DataGrid [EUDG], and the Earth Systems Grid [ESG]. These projects all require the efficient transfer of very large scientific data files across the network, and would all benefit from the work described here.

We will also work closely with the SciDAC proposed "DOE Science Grid" project, led by William Johnston, LBNL. This project proposes creating a multi-laboratory Collaboratory Pilot aimed at integrating, deploying, and supporting the persistent services needed for a scalable, robust, high-performance DOE Science Grid, thus creating the underpinnings for a DOE Science Grid Collaboratory Software Environment (DSG-CSE).

We will use our experience and expertise in tuning high-performance distributed applications to validate the techniques from this proposal. We also have close ties to ESNet, NTON [NTON], and SuperNet [SuperNet] networks, and will work with each of them to deploy these new network tools and services in each of these environments. This will allow us to validate the utility of these tools and services in a real network environment.

3.6 Timeline in Milestones

3.6.1 Pittsburgh Supercomputing Center

Year 1:

- Work with ORNL to understand the instrumentation requirements from the bulk-tramport application and integrate existing Web100 kernel into the host operating system and application itself.
- Based on observations and analysis of the results from using the initial application, modify and enhance Web100 instrument sets, API and libraries.
- Document instrumentation and library functions in a standardized manner. Work with the Web100 developers to incorporate this into a document (i.e, IETF MIB) that can be submitted to the standards track at IETF.
- Begin development of mechanisms for using NOAS generated information with the Network Tool Analysis Framework.

Year 2:

- Work with ORNL to integrate NOAS into multi-site distributed application.
- Identify and implement NOAS instruments and enhancements necessary to support the distributed applications.
- Continue documentation and standardization process.
- Identify key operating systems NOAS should be expanded to. Work with Net100 collaborators to get vendors to adopt the NOAS and Web100 standards.
- Start the proccess to develop remote access mechanisms.

Year 3:

- Continue to expand and enhance NOAS instrumentation and functions based feedback from applications, primarily focusing on integrating NOAS into a range of DoE specific distributed applications.
- Continue process of developing remote access mechanims, including testing these mechanims within the NTAF as well as with other projects.
- Begin the process of unifying performance measurements across all layers in the network. Add instrumentation and functions to the NAOS as necessary.
- Document and standardize all enhancements and provide mechanism for distributing NOAS code to DoE users.

3.6.2 Oak Ridge National Laboratory

Year 1

- Deploy Web100 at ORNL and NBERSC nodes to develop Net100 expertise.
- Develop and demonstrate Web100-aware bulk data transfer application for Probe/HPSS testing between NERSC and ORNL.
- Contribute to test and evaluation of existing end-to-end tools.
- Get access to ESnet ORNL and NERSC routers and investigate possible realtime feedback to application (e.g.: using SNMP).
- Explore transport optimizations for single TCP flows.
- Develop file transfer application/protocol to support out-of-order packet arrivals.
- Deploy a small emulator testbed to test transport protocols/applications.
- Explore turning the IBM/AIX 5.1 TCP stack and investigate extending it with Net100 mods.
- Test Net100 tools on ESnet's OC48 testbed.
- Publish tools and tips on web page and in formal publications and presentations.

Year 2

- Integrate Net100 tools into HPSS transfer protocols (HSI).
- Modify kernels to support ECN and use Net100 tools to tune DOE TCP applications with ECN feedback (assumes DOE's ESnet routers will be ECN-capable).
- Continue Net100 integration with other vendor TCP stacks (Solaris, Irix, Compaq).
- Integrate ORNL's network-daemon statistical sampling research with NWS/Net100.
- Extend Net100 tools to include adaptive source-based routing and multipath routing via application network daemons.

Year 3

- Continue Net100 integration with other vendor TCP stacks (Irix, FreeBSD).
- Extend Net100 tuning to include DOE's IPv6 network (jumbograms).

3.6.3 Lawrence Berkeley National Laboratory

Year 1:

- generalize existing LBNL tools to create the core NTAF framework (6 person-months)
- interface NTAF with NWS (2 person-months)
- instrument *iperf*, *pipechar*, *pathrate* and GridFTP using Web100 TCP-KIS and NetLogger
  - 1person-months
- analyze Web100 autotuning of GridFTP using NTAF and NWS
  - with ORNL, 2 person-months
- help design and test NWS NAB analysis modules (with UT: 1 person-months)
- begin analysis of which monitoring tools will be required for network-aware operating systems (joint effort with all participants: (1 person-months)
  - total effort for year 1: 1 FTE

Year 2:

- continue to improve NTAF framework (2 person-months)
- continue to instrument and integrate new network tools as they become available (1 person-months)
- analyze Web100 autotuning with Grid applications (2 person-months)
- analyze impact of tuned applications on non-tuned applications (with PSC: 2 person-months)
- analyze impact of parallel streams (with PSC: 1 person-months)
- analyze alternative protocols (with ORNL: 1 person-months)
- continue to help design and test NWS NAB analysis modules (with UT: 1 person-months)
- integrate with other monitoring projects (2 person-months)

- total effort for year 2: 1 FTE

Year 3:

- continue to improve NTAF framework (1 person-months)
- continue to instrument and integrate new network tools as they become available (1 person-months)
- help write IETF documents on network-aware operating system techniques (1 person-months)
- continue to analyze Web100 autotuning with Grid applications (3 person-months)
- continue to analyze impact of tuned applications on non-tuned applications (with PSC: 2 person-months)
- continue to analyze alternative protocols (with ORNL: 1 person-months)
- continue to help design and test NWS NAB analysis modules (with UT: 1 person-months)
- continue to integrate with other monitoring projects (2 person-months)
- total effort for year 3: 1 FTE

3.6.4 University of Tennessee at Knoxville

Year 1:

- Generalize NWS interface to accept Web100 measurement information (2 person months)
- interface NTAF with NWS (2 person-months)
- incorporate *iperf*, *pipechar*, *pathrate* and GridFTP using web100 TCP-KIS from NetLogger into NWS (2 person-months)
- analyze Web100 autotuning of GridFTP using NTAF and NWS (with ORNL, 2 person-months)
- design and begin to prototype NWS NAIB analysis modules (6 person-months)
- total effort for year 1: 1 FTE

Year 2:

- continue to design and prototype NAIB (6 person-months)
- continue integrate new network tools as they become available (1 person-months)
- begin integration of NAIB within NTAF (with LBNL 2 person months)
- analyze Web100 autotuning with Grid applications (2 person-months)
- begin development of multivariate analysis models that can incorporate data from multiple measurement sourcdes (1 person month)
- total effort for year 2: 1 FTE

Year 3:

- help to continue to improve NTAF framework (with LBNL 1 person-months)
- complete NAIB prototype (2 person months)
- deploy integrated NTAF and NAIB (with ORNL, PSC, LBNL 2 person months)
- help write IETF documents on network-aware operating system techniques (1 person-months)
- help to enhance Web100 autotuning with Grid applications (with PSC 3 person-months)
- continue to analyze impact of tuned applications on non-tuned applications (with PSC: 2 person-months)
- incorporate and report on multivariate analysis models (1 person month)
- total effort for year 3: 1 FTE