NGI/NW/02

Network Tools: Server Load Service

FY 2003 Proposal to the NOAA HPCC Program

August 19, 2002

| Title Page | Proposed Project | Budget Page |

Principal Investigator:	**Michael T. Knezevich**
Line Organization:	OAR/PMEL
Routing Code:	R/PMEL
Address:	NOAA/PMEL/CNSD 7600 Sand Point Way NE Seattle, WA 98115-0070
Phone:	(206) 526-6766
Fax:	(206) 526-6815
E-mail Address:	Michael.T.Knezevich@noaa.gov

Donald W. Denbo

Donald.W.Denbo@noaa.gov

Proposal Theme:	**Next Generation Internet (NGI) - Innovative infrastructure supporting the NOAA Information Technology (IT) Architecture**
Funding Summary:	FY 2003 $ 25,800

Michael Knezevich	Russell L. Richards	Cynthia L. Loitsch	Eddie Bernard
Computer Specialist	CNSD Division Leader	Program Support Officer	Director
PMEL	PMEL	PMEL	PMEL

Network Tools: Server Load Service

Proposal for FY 2003 HPCC Funding

Prepared by: Michael T. Knezevich

Executive Summary:

Network applications are critically dependent on the availability of adequate server resources. Although tools are available for server load balancing, these tools do not provide feedback to the client application to allow adjustment of server load requirements.

We are proposing to enhance Network Tools (HPCC FY02 funded proposal, NGI/NW/02) to enable application developers to create software that can self-adjust to changing server loads, in order to maximize network throughput and improve application response. With these tools, a client application can manage mirror server connections and dynamic network load adjustments. We will wrap these techniques into the Network Tools Java library that can be easily used by any developer. We are not aware of any alternative Java library that will provide similar functionality. Applications that could benefit from the server tuning tools include HPCC-funded distributed data access applications (such as the Climate Data Portal and Live Access Server) or collaborative tools (such as OceanShare and ImmersaDesk).

We are requesting funding to develop the following Network Tools:

Server Load Service. This service will run on a server and provide the application with important information about the state of the server. Available memory, cpu load, disk I/O, etc., can be used by the application developer to choose between identical mirror servers and/or adjust server resource requirements at run time. Again, these adjustments will enable a client-server application to tune its performance based on the availability of server resources.

Problem Statement:

Problem Statement: As NOAA embraces the Next Generation Internet (NGI) and internet enabled applications, NOAA users will become more and more dependent on remote resources. These resources may be large distributed data sets or remote services like GIS map generation or computational engines. Predictable and consistent access to these resources is important in order to maintain workflow and efficiency. Variations in available memory, cpu load, disk I/O on the server lead to an environment in which static choices of the size of data requests and mirror server selection produce sub-optimal results for data throughput and resource utilization. The resource provider is often unable to keep up with demand during peak usage.

Relationship to NOAA HPCC objectives: This project addresses the two HPCC themes focused on the Next Generation Internet (NGI) and Collaborative, Visualization or Analysis tools. It directly supports the HPCC objective to apply advanced technologies associated with the Internet to enhance access to data holdings and the goal for HPCC networking to improve access in a manner that increases effectiveness.

Proposed Solution:

We propose to address the problem of balancing server load by enhancing a set of Java Network Tools developed in the FY02 HPCC project. These tools will enable developers to adjust the amount of data their applications request from distributed data servers and when mirrored sites are available help balance server load by using the server with the lowest load. The initial implementation of these enhancements will be in Java, a platform independent language, when possible. Platform specific code may be required to access operating system statistics and server load. The platform specific code will be accessed by the enhancements to Java Network Tools using the Java Native Interface (JNI). Initial platform specific implementations will be for Win32 and Solaris 8 environments. The server load services will be implemented as servlets.

Server Load Service. Monitoring server load will require that a server load service run on each server that such information is required. The service will continually monitor various server and application metrics and report these measurements to the client upon request. The metrics to be monitored can include available memory, cpu load, disk I/O, network load, database load, and the load on important server applications. The service can also report to the client application the health of a required server application, whether the application is hung and requires a restart, not running, or alive. The health information could then be used to notify server administrators that an application needs attention.

Load information can be used by the client application to reduce or increase server requests and when services are mirrored to use the server with the lower load. This provides a client side load balancing capability.

Analysis:

Our approach, to use the client application to manage mirror server connections and data request size adjustments, was chosen to reduce server load and allow application developers access to server load information to tune their applications. We will add the server load monitoring capabilities into the Network Tools library, developed as part of a Network Tools project funded in FY02 by HPCC. This Java library can be easily used by any developer whose application could benefit from network and server tuning, for example, distributed data access (Climate Data Portal and LAS Server) or collaborative tools (OceanShare and ImmersaDesk).

We are not aware, at this time, of any alternative Java library that will provide similar functionality. There are tools available for load balancing, for example, IBM's Network Dispatcher, but they do not include feedback to the client application to adjust network data requirements. There are also tools available to help a manager perform network tuning, but these tools are most useful for setting static routing tables for specific network links.

These Network Tools will enable application developers to create software that can adjust to the changing network environment and server load to maximize network throughput and improve application response. For example, the developer of a client-server application that had multiple "mirrored" servers available could use the Network Tools to determine which server should be used to provide maximum data transfer rates. A typical scenario might be: 1) the client application starts up and determines which of its possible servers are currently available, 2) the client then uses the Network Tools to determine the load for each server, 3) the client then chooses to connect to the server with the lowest server load, and finally 4) the client can use the server load to determine how much data should be downloaded from the server (for example, whether to download the fine, medium, or coarse coastline). Steps 2 through 4 might be run only once if a client-server connection is relatively short lived, or run several time during the session in order to adjust to changing server load conditions.

Performance Measures:

At the end of FY2003, we will have designed, implemented, and deployed enhancements to Network Tools to measure server load.

Milestones

Month 5 – Design server load service
Month 6 - Implement server load service
Month 10 - Test server load service
Month 12 - Deploy version 2.0 of Network Tools with server load service.

Deliverables

Network Tools Version 2. Server load service tools will be added to the Network Tools developed in an FY02 HPCC project. Server load service will combine available memory, cpu load, and disk I/O, to determine the current server load.