S 7978 C

7

Final Report for
Cooperative Agreement
NASA Ames Research Center
NASA ~~NCC~~ NAG 2-1218
Parallel and Distributed System Simulation

5/1/98 to 9/1/98
$40,000

Principal Investigator
Jack Dongarra
University of Tennessee, Knoxville
Computer Science Department

Technical Officer for the Cooperative Agreement
Subhash Saini
Numerical Aerospace Simulation Systems Branch, T27A-1

Grant Officer
Venoncia Braxton

This exploratory study initiated our research into the software infrastructure necessary to support the modeling and simulation techniques that are most appropriate for the Information Power Grid. Such computational power grids will use high-performance networking to connect hardware, software, instruments, databases, and people into a seamless web that supports a new generation of computation-rich problem solving environments for scientists and engineers. In this context we looked at evaluating the ***NetSolve*** software environment for network computing that leverages the potential of such systems while addressing their complexities. NetSolve's main purpose is to enable the creation of complex applications that harness the immense power of the grid, yet are simple to use and easy to deploy. NetSolve uses a modular, client-agent-server architecture to create a system that is very easy to use. Moreover, it is designed to be highly *composable* in that it readily permits new resources to be added by anyone willing to do so. In these respects NetSolve is to the Grid what the World Wide Web is to the Internet. But like the Web, the design that makes these wonderful features possible can also impose significant limitations on the performance and robustness of a NetSolve system. This project explored the design innovations that push the performance and robustness of the NetSolve paradigm as far as possible without sacrificing the Web-like ease of use and composability that make it so powerful.

The project began exploring a range of advanced techniques for providing fault-tolerance, process migration, data caching, and remote library support. These functionalities enable NetSolve to overcome its early limitations without changing its underlying model. The techniques examined or considered include allowing computations to be moved off servers that fail or are not progressing fast enough, managing user data to eliminate redundant transmissions of large data sets, dynamically loading new software components from network repositories to satisfy user requests, and so on. Success in this research on NetSolve is dramatically enhancing its value to a large user community by improving its reliability and resiliency.

A second, complementary line of inquiry leveraged NetSolve's highly composable architecture and dynamic access to heterogeneous resources to create a testbed that uses real

applications to compare the performance of different techniques for fault-tolerance and load balancing. The availability of such a testbed allows us to develop a better understanding of how to write efficient and fault-tolerant code that can harness the vast, yet dynamically changing processing power of the Grid.

Thus, the design of NetSolve fosters new synergies between the work of computational science researchers and that of computer science researchers: the former can optimize the performance of well-used programs on parallel platforms, while the latter design and implement paradigms for fault-tolerance, load balancing, caching and software repository support. Then NetSolve's modular architecture allows us to make all of these functionalities directly available to users with no additional burden on them. Thus, NetSolve not only functions as a test-bed for research, but as a deployment mechanism so that research may benefit its intended audience immediately.

The research *on and with NetSolve* focused on improving the NetSolve design by exploring several techniques. Research *on* NetSolve examined the following approaches:

- *Fault-tolerance and migration between resources.* This is termed *inter-server fault-tolerance*, in which a computation is moved off a server that fails or is not progressing fast enough.
- *Introduction of storage servers* to store checkpointed state, so that if a given server fails, the NetSolve agent can send its state to a new server.
- *Data logistics for improved performance* through techniques that manage user data, for example to eliminate redundant transmissions of large data sets across the network.
- *Dynamic loading of new software components* from network repositories to satisfy user requests.
- *Integration of NetSolve with different meta-computing paradigms* (e.g. Globus/Nexus), leveraging their technology while complementing their use.

The research *with* NetSolve investigated the following:

- *Test and compare the performance* of a variety of techniques for providing fault-tolerance and load balancing within computational resource pools. This is termed *intra-server fault-tolerance*.
- *Develop non-transparent coordinated checkpointing* as the basic framework for intra and inter-server fault-tolerance.
- *Implement diskless checkpointing* to improve performance and enable experimentation on different mixes of intra- and inter-server fault-tolerance to uncover new paradigms.
- *Develop new techniques for integrated fault-tolerance* by implementing fault-tolerant servers with new programming paradigms, such as shared tuple spaces or distributed objects.

This course of investigation will significantly enhance this easy to use, easy to deploy, highly scalable paradigm for building the computational power grids of the future.