

# A Parallel and Distributed Computing Environment for Scientific Applications

## Annual Report 97

K. Maly, M. Zubair, C.M. Overstreet

Department of Computer Science  
Old Dominion University  
Norfolk, VA 23529-0162

email: (maly, zubair, cmo) @cs.odu.edu}  
(757) 683-4817  
Fax: (757) 683-4900

### Submitted to:

Dr. Jaroslaw Sobieski, leader, Computational AeroSciences,  
and Multidisciplinary Research Coordinator,  
Research and Technology Group, MS139  
NASA Langley Research Center  
Hampton, VA 23681-0001  
Tel: 757 864-2799  
Fax: 757 864-9715

E-mail: [j.sobieski@LARC.NASA](mailto:j.sobieski@LARC.NASA).

## Summary of progress

**pPVM:** Under NASA-NAG-1-1550, we have developed the pPVM environment at ICASE to run standard PVM and MPI programs. We have completed the design, implementation, and testing of the environment, running on an ICASE network of 8 workstations connected by dual FDDI rings and an Ethernet. In September 1997, we held a workshop as part of the ICASE Roundtable.

The web site we created: <http://www.cs.odu.edu/~kelkar/ppvm/> provides information about:

About pPVM <http://www.cs.odu.edu/~kelkar/ppvm/ppvm/intro.html>

How to Install pPVM <http://www.cs.odu.edu/~kelkar/ppvm/ppvm/install-help.html>

How to Run pPVM <http://www.cs.odu.edu/~kelkar/ppvm/ppvm/run-help.html>

Testbed at ICASE <http://www.cs.odu.edu/~kelkar/ppvm/ppvm/configuration.html>

*About pPVM:* Parallel PVM (pPVM) is PVM modified to provide high performance communication support in a cost effective way for parallel and distributed computing. pPVM allows PVM applications to communicate concurrently over parallel physical networks to improve the communication performance in a way transparent to the user. For many communication intensive applications the network bandwidth becomes the bottleneck. For these applications cluster workstations need to move data at a rate higher than the bandwidth available to them. The assumption here is that the cluster workstations are capable of sending/receiving the data at this rate. We proposed a cost-effective approach of parallel networking to improve the communication bandwidth of the network and make cluster computing effective for a larger class of applications. Based on this approach we have modified the PVM system to a parallel PVM (pPVM) system. In past, we had experimentally demonstrated the pPVM concept for parallel Ethernets. A substantial reduction was observed in the communication component of a parallel application running over a cluster of workstations with parallel Ethernets. The current pPVM implementation allows the use of parallel FDDIs or heterogeneous networks (Ethernet + FDDI) for transferring application data. It has been developed at the Old Dominion University, for the ICASE, NASA Langley Research Center and is the modified PVM version 3.2. To transfer the application data over the parallel networks, different scheduling algorithms such as round-robin, weighted round-robin and adaptive are designed and implemented into pPVM. A number of experiments were conducted to observe the benefit of pPVM over PVM. In essence, the pPVM adaptive scheduling is beneficial to the applications in most cases and is never worse than PVM.

**JAVADC:** The remainder of the effort was to develop a Web based environment, JAVADC, to run arbitrary distributed programs using web interfaces. <http://shark.icasel.odu.edu:8080/>

**ARCADE:** Finally, we started to work on the future support of distributed computing.

### **A Collaborative Environment to Design, Execute, Monitor and Control Distributed Heterogeneous Computing Applications A White Paper**

Distributed heterogeneous computing is being increasingly applied to a variety of large size computational problems. Such computations, for example, the multidisciplinary design optimization (MDO) of an aircraft, generally consist of multiple heterogeneous modules interacting with each other to solve an overall design problem [2]. Typically these modules are developed in different disciplines and are optimized independently. The traditional way of integrating these modules and optimizing them for the overall design is a long and tedious process (typically taking several weeks). The slowness of this process is

mainly due to the absence of a collaborative environment where (i) different modules and their interaction can be specified, and (ii) testing, monitoring, and steering of the overall design can be done. A system that provides a collaborative environment which is easy-to-use and widely accessible is lacking. Recently, there has been some efforts to design and prototype such systems [1,2,3]. However, these systems lack a Web-based collaborative environment for solving large multidisciplinary problems. The increasing use of Web technology for Internet and Intranet applications is making the World-Wide Web an attractive framework for collaboration.

The focus of our project is to develop a Web-based collaborative environment to design, execute, monitor, and control distributed heterogeneous computing applications. The proposed environment will have the following features: (i) it will allow collaborative design and control, (ii) it will be easy to access, (iii) it will be easy to use, (iv) it will work on heterogeneous platforms, (v) it will be flexible to adapt to different scenarios, (vi) it will be secure, and (vii) it will support parallel computing.

In this proposal, we want to design and prototype a three-tier system, that provides a collaborative environment and reduces significantly the overall time for solving a multidisciplinary optimization problem. The front-end, first tier, of this system is a GUI interface integrated with the Web. It will be implemented by using Netscape Internet Foundation Classes (IFCs). Integrating the interface in a Web browser, e.g., Netscape, will provide users with a familiar interface on desktops ranging from Unix based workstations, to Windows-based PCs, and Macintoshes. The middle tier consists of logic to process the user input and interact with modules running on a heterogeneous set of machines. These modules along with the control processes form the last tier. The overall design is a client-server based architecture. The main advantage of a three-tier system is that the client or the front-end becomes very thin, thus making it feasible to run on low-end machines.

We now give a typical scenario of how the proposed system will be used. A project leader, responsible for the overall design problem, will interact with the front-end to set up the project. As the front-end is integrated with the Web, he can do this from any place where he has access to the Internet. The set-up for the project involves:

- (i) Identification of team members and set up of their privileges to access and modify certain modules during the optimization process.
- (ii) Specifications of modules including their interaction.
- (iii) The set-up of monitoring points and control conditions.

Once the project is set up, team members can complete the specification for the modules for which they are responsible. Note that a module can consist of several sub-modules interacting with each other. The team members also map the module to an available hardware resource. Next the application is executed in a distributed environment using a heterogeneous network of workstations and multiprocessor machines. During the execution, team members from anywhere on the Internet monitor, and control the optimization process using a standard graphical Web browser. The team members can see the currently executing modules at any level of the hierarchy. They can also view the intermediate data flowing between different modules, and in some cases visualize large data sets.

A team member responsible for a particular subsystem can change data values under the control of the subsystem in order to steer the computation in the right direction. The team member would also be allowed to dynamically alter the control flow if necessary. For example, in a design cycle of the optimization process, the responsible team member may decide that a particular module is not affecting the optimization and may bypass the module by using old values in each cycle. Similarly, the team could replace a module with a plug compatible module, for example, use another algorithm.

We will leverage our experience with (i) designing and building, JAVADC, a GUI based environment for PVM and pPVM-based SPMD applications, and (ii) Opus, a coordinated language for multidisciplinary applications [4]. Also, wherever possible we will use existing public domain software and technologies. JAVADC is a Web-Java based environment to enable distributed computing with pPVM or PVM. The interface for JAVADC was implemented using Netscape Internet Foundation Classes. The back-end server for JAVADC was based on JAVA. JAVADC is currently restricted to pPVM or PVM and has no multi-user support. The proposed work can utilize some of the technologies developed for the JAVADC project. The Opus language was motivated by a lack of features in data-parallel languages for heterogeneous multidisciplinary applications. We propose to use Opus-like specifications as the intermediate representation of the multidisciplinary application being addressed. This specification would be interpreted by the logic at the middle tier to control the execution of the complete application.

#### REFERENCES

1. Thomas M. Eidson, Robert P. Weston, "FIDO - Framework for Interdisciplinary Design Optimization", <http://hpccp-www.larc.nasa.gov/~fido/homepage.html>
2. John C. Peterson, "Multidisciplinary Integrated Design Assistant For Spacecraft (MIDAS)", [http://mishkin.jpl.nasa.gov/Midas\\_Page](http://mishkin.jpl.nasa.gov/Midas_Page)
3. Dimple Bhatia, Vanco Burzevski, Maja Camuseva, Geoffrey Fox, Wojtek Furmanski and Girish Premchandran, "WebFlow - a visual programming paradigm for Web/Java based coarse grain distributed computing" <http://www.npac.syr.edu/projects/webbasedhpcc/index.html>
4. B. Chapman, M. Haines, P. Mehrotra, J. Van Rosendale, and H. Zima. Opus: A coordination language for multidisciplinary applications. To appear in Scientific Programming, 1997 (Also available as ICASE Technical Report).

#### Acknowledgments

The initial design of this work was based on discussions between Piyush Mehrotra of ICASE and Jim Rogers and Andrea Salas of the MDOB, NASA Langley.

#### ICASE/ODU -- Arcade Project One Year Milestones:

- Task 1: Visual Specification of the application.
- Task 2: Semi-intelligent resource mapping support.
- Task 3: Support for SPMD modules.
- Task 4: Data monitoring support.
- Task 5: Multidomain execution.
- Task 6: Limited security

#### ODU One Year Budget for Arcade Project

Maly, 1.5 mos	\$19,367
Zubair, 2mos	\$15,267
CMO, 0.1 mos	\$1,000
2 students	\$24,000
system, Ajay Gupta	\$3,500
Subtotal	\$63,134
Fringes	\$4,500
Subtotal	\$67,634
Travel	\$3,000
Computing	\$2,500
Equip., 3PCs	\$6,000
Subtotal	\$79,134
Indirect Cost, 27%	\$19,746
Total	\$98,880