

Harness: Heterogeneous Adaptable Reconfigurable Networked Systems

U.S. Department of Energy Grant DE-FG02-99ER25379
Final Project Report

Vaidy Sunderam
Department of Mathematics and Computer Science
Emory University, Atlanta, GA 30322, USA
vss@mathcs.emory.edu

1 Overview

Issues in reconfigurability and adaptability in heterogeneous distributed systems for high-performance computing are the focus of the work funded by this grant. Our efforts are part of an ongoing research project in metacomputing at Emory University, Oak Ridge National Laboratory, and the University of Tennessee. The project, termed Harness, aims to investigate novel methodologies and tools for distributed metacomputing, focusing on dynamically reconfigurable software frameworks. During the first phase, we defined the metacomputing architecture embodied in Harness and developed prototype subsystems as proof of concept exercises. Subsequently, we designed and developed a complete software framework manifesting the Harness architecture, and also developed several tools and subsystems that demonstrate the viability and effectiveness of our proposed model for next generation metacomputing. In the final extended project period, we completed development and refinement of the research software, emulated multiple programming environments on Harness, and conducted performance evaluation and tuning exercises. This document is the final technical report covering the entire funding and project periods, viz. March 15, 1999 through March 14, 2003.

2 Overall Progress

The Emory portion of Harness research has addressed several areas: definition of an abstract model of the Harness framework, backplane design and implementation, test applications and emulation of programming environments, and support tools for Harness. The Harness

framework is based on the fundamental concept of dynamic reconfiguration, not only in terms of the computers and networks that comprise a distributed virtual machine, but also the capabilities and services offered by the VM itself. In order to realize this infrastructure, a “plug-in” based approach was proposed. A backplane architecture was proposed as an implementation approach for this model. Subsequently, work was undertaken on a Java-based Harness prototype, that eventually grew into a full fledged backplane and kernel environment. Auxiliary subsystems needed for proper operation, including a debugging system and a failure resilient nameserver were also designed and developed. In order to determine the viability of software reconfigurability in Harness, two significantly different programming environments were deployed as Harness plugins. Example applications were developed, both using the native Harness environment as well as by porting applications for the emulated programming systems. Performance tests were conducted, and following detailed analysis, major enhancements to design and implementation aspects were undertaken to optimize and fine tune efficiency. A complete software suite, with example applications, two concurrent programming environments, and various forms of documentation were published.

2.1 Harness Framework and System

The first phase of the project concentrated on the definition of the Harness model, and on the development of a Java-based kernel, or backplane. As defined, the fundamental abstraction in the Harness metacomputing framework is the Distributed Virtual Machine (DVM). DVM’s are abstractions that are converted to metacomputing environments by first adding resources, and then loading appropriate plugins onto those resources. Plugins are elements that provide specific services such as group communication operations or distributed shared memory emulation. Plugins may either directly use resources via native interfaces. Alternatively, they may depend upon either the primitives supported by the Harness kernel (event and message management) or by other plugins.

Based on the above abstraction, a Harness kernel implementation was undertaken. The system is capable of establishing Distributed Virtual Machines (DVMs) with appropriate authentication controls and soft installation of modules from repositories as needed. The DVM can be controlled from graphical panels that allow the addition or deletion of physical resources (machines), the loading and unloading of plugins, and the management of name lookup services. In this scheme, an orthogonal nameserver is assumed that defines naming domains for DVMs (a simple centralized server and a failure resilient one are provided as part of the Harness distribution). Each DVM is defined by the existence of a “status server” that maintains information about the DVM, the loaded plugins, and ownership and authentication data.

The Harness kernel is implemented as a collection of Java classes that include an API for external access. This bootstrap API only includes access methods for a few very fundamental functions such as registration, plugin loading, and event notification. Once kernels are initiated, they communicate with the status server to join the DVM and eventually load

in a set of plugins that brings them up to a baseline, through which services are offered to applications. Plugin loading may occur either on explicit user request, or during the process of upgrading a new kernel to the current baseline of an existing DVM. Once in place, kernels/plugins accept and process requests from applications, and carry out tasks on their behalf. For example, plugins implementing programming environments will present that environment's API to the application. Plugins may also implement parallel computing paradigms, such as the bag-of-tasks model using a master-worker structure, or may directly contain end-application code. System design, refinement of the preliminary prototype, and development and implementation of these infrastructures were completed during the second project period. Several papers, discussing various aspects of Harness were written.

2.2 Auxiliary Subsystems

Given the fragility of wide area metacomputing systems, an overriding design consideration in Harness was to be as unsusceptible to failures as possible. Since the nameserver subsystem is one of the most crucial components of Harness, the development of a failure resilient nameserver subsystem was undertaken in parallel with the Harness kernel development. The Harness Distributed Name Service (HDNS) is intended to be reliable provider of services such as the maintenance of dormant hosts and the identity of DVM's and DVM servers. HDNS was designed as a replicated distributed system organized in a ring topology. A unidirectional ring was used, with updates being injectable at any node. Each node implements a replicated database consistency algorithm where lookup requests are processed locally insofar as possible, and updates are applied synchronously using two-phase commit. Failures of intermediate nodes are detected via keepalive transmissions and timeouts, and reconfiguration protocols to exclude failed nodes are included. The distributed database scheme is deliberately kept simple, and the algorithms and protocols are based on the expected frequency of lookups and updates. An implementation has been deployed across the three project institutions; we have found this to be very robust and tolerant of failures at upto two sites. Furthermore, the distributed nature of the service does not cause any noticeable delay in lookups, and only very minimal delays during updates that occur much less frequently. Details of this subsystem are presented in several publications.

Debugging is another area in which metacomputing systems are notoriously deficient. Having experienced early difficulties in debugging system components, we undertook the development of a distributed debugging system for Harness that has proved very useful. The debugging framework, termed IDVS (Integrated Debugging and Visualization System) is a convenient platform for controlling multiprocess distributed applications with several unique features. A particularly important facility, likely to be useful even outside the Harness context, is transparency in debugging remote method invocations. This feature enables remote method tracing and control in a manner identical to local method invocations, thereby permitting the user to focus on the function being debugged rather than the transport subsystem or other distributed aspects. A second feature is the ability to

switch seamlessly between a user's favorite sequential debugger when analyzing serial code segments, and the IDVS tool when focusing on distributed aspects. The IDVS subsystem provides useful functionality such as debugging multiple Harness plugins simultaneously and managing multiple distributed breakpoints. IDVS also contains an integrated visualization facility that graphically depicts program execution under Harness, indicating plugin activity, computational tasks, and interactions among the distributed components in a DVM. These features have already proven helpful in our own experiments with Harness and IDVS will be a very useful subsystem to external users.

2.3 Programming Environments

The underlying premise of the Harness system is that metacomputing environments can and should be constructed from replaceable components. As such, Harness does not support a specific parallel computing model or framework; plugins implementing different models are loaded into Harness to emulate the desired parallel programming environment. In order to establish the viability of this scheme, we undertook the development of plugins for two different parallel programming environments.

The first programming environment to be emulated under Harness was PVM. A PVM plugin for Harness was undertaken in order to verify the ability of Harness to emulate standard, heterogeneous, message passing, and also to provide a seamless transition path to legacy PVM users. As such, our design constraints mandated that no changes be necessary to allow existing PVM applications to run under Harness. Our proposed solution is to implement the PVM daemon system as a Harness plugin. Application programs would continue to link against existing PVM libraries but run under Harness. The PVM environment is emulated by Harness plugins that can be upgraded to transparently provide enhanced services or interface with other Harness components to extend the functionality available to PVM programs. Our experiences with this exercise have been very satisfactory; most components of the PVM plugin for Harness simply leveraged existing core Harness components, and only a handful of specific classes needed to be written. The PVM emulation software has just been completed and will be included in the next Harness release.

As a second proof of concept exercise, we developed and implemented a complete JavaSpaces programming environment in Harness. As the JavaSpaces technology aims to provide a simple unified mechanism for dynamic communication, coordination, and sharing of objects between Java technology-based network resources, it is a good candidate for a Harness plugin, and likely to be useful in a wide range of applications. JavaSpaces for Harness extends the set of Harness services by a Linda-like coordination space. Instead of providing just an adapter for the existing JavaSpaces implementation, the goals were to implement a distributed space by reusing existing plugins as well as by writing new ones. This exercise too was successful, and a fully functional JavaSpaces plugin has been implemented for Harness. Except for a few specific primitives, our implementation performs as well as the Sun reference implementation. These experiences suggest that the plugin paradigm for

deploying parallel programming environments in Harness works well.

2.4 Performance analysis and optimization

Harness is a software infrastructure that proposed component and service-based frameworks rather than traditional monolithic, platform-specific systems. Contrary to popular belief, such modular software systems are capable of delivering good to excellent performance, support legacy as well as new application programming paradigms, and deliver enhanced functionality. In order to demonstrate this belief, the Harness system was used to emulate the PVM programming environment and was extensively tested with rigorous compute- and communication intensive applications. Numerical kernel benchmarks show that application performance results using the emulator and native versions are within a few percent of each other. Coupled with the ability to leverage pre-existing and specialized modules, our experiences suggest that service-oriented computational grids may be constructed rapidly and effectively via such component-based architectural frameworks that deliver full functionality without compromising efficiency. The results of these benchmarking experiments as well as discussions of the optimizations to the Harness system are detailed in recent publications.

3 Research Output

Our work on the Harness system has resulted in a number of research papers and presentations. In addition, the software itself is also being released to the community for potential use as a distributed high-performance computing platform. The Harness system was demonstrated at several conferences and has evoked considerable interest, with several research groups using the system and its follow on tools.

3.1 Talks and Professional Activities

1. “Standards Based Heterogeneous Metacomputing: The Design of HarnessII”, contributed talk, *International Parallel and Distributed Processing Symposium (IPDPS-HCW) 2002*, Ft. Lauderdale FL, April 2002.
2. “Debugging and Visualization of Metacomputing Applications”, contributed talk, *International Parallel and Distributed Processing Symposium (IPDPS-JPDC) 2002*, Ft. Lauderdale FL, April 2002.
3. “A Comparison of Conventional Distributed Computing Environments and Computational Grids”, contributed talk, *2002 International Conference on Computational Science*, Amsterdam, April 2002.

4. "A Comparative Analysis of PVM/MPI and Computational Grids", invited talk, EuroPVM/MPI '01 Conference, Santorini, Greece, April 2002.
5. "Alternative Approaches to Metacomputing with Harness and IceT", invited talk, The 2001 International Conference on Computational Science", San Francisco, CA, May 2001.
6. "Emulating Parallel Programming Environments in the Harness Metacomputing System", invited talk, Workshop on Clusters and Computational Grids, Lyon, France, September 2000.
7. "Metacomputing Systems and Technologies", invited lecture series, Escola de Computacao 2000, University of Sao Paulo, Sao Paulo, Brazil, July 2000.
8. "A Simple Fault Tolerant Naming Scheme for the Harness Metacomputing System", EuroPVM/MPI Conference, Balaton, Hungary, September 2000.
9. "User Controlled Task Migration in HARNCESS", International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'00), Las Vegas, NV, pp. 1385-1392, June 2000.
10. "The Harness PVM-Proxy: Gluing PVM Applications to Distributed Objects Environments and Applications", IPDPS Heterogeneous Computing Workshop 2000, Cancun, Mexico, May 2000.
11. "Hybrid Simulation in Harness", ASTC High Performance Computing 2000 Conference, Washington, DC, April 2000.
12. "Networking Performance for Metacomputing in Java", 1999 IASTED Conference on Parallel and Distributed Computing and Systems (PDCS'99), Cambridge, MA, November 1999.
13. "Message Passing and Java", invited talk, Java-Grande BOF, Supercomputing '99, Portland, OR, November 1999.
14. "PVM Emulation in Harness: A Plugin Based Approach", contributed talk, EuroPVM/MPI '99 Conference, Barcelona, Spain, September 1999.
15. "Distributed and Collaborative Computing Systems", invited talk, Indian Institute of Technology, Bombay, India, August 1999.
16. "Networking Performance for Metacomputing in Java", 1999 IASTED Conference on Parallel and Distributed Computing and Systems (PDCS'99), Cambridge, MA, November 1999.
17. "Distributed Reconfigurable Simulation in Harness" 1999 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas, NV, pp. 730-736, June 1999.

3.2 Publications:

1. M. Migliardi, V. Sunderam, A. Geist, J. Dongarra, "Reconfiguration and Virtual Machine Management in the Harness Metacomputing System", *Proceedings – ISCOPE 98*, Santa Fe, NM, December 1998.
2. M. Migliardi, V. Sunderam, "The Harness Metacomputing Framework", *Proceedings – Ninth SIAM Conference on Parallel Processing*, San Antonio, TX, March 1999.
3. M. Migliardi, V. Sunderam, "Heterogeneous Distributed Virtual Machines in the Harness Metacomputing Framework" *Proceedings – International Parallel Processing Symposium Heterogeneous Computing Workshop*, San Juan, PR, pp. 60-72, April 1999.
4. M. Migliardi, V. Sunderam, "Behavioral Objects and Layered Services: Application Programming in the Harness Metacomputing System", *Proceedings – High Performance Computing and Networking Europe 1999*, Amsterdam, Netherlands, April 1999.
5. M. Migliardi, V. Sunderam, "Distributed Reconfigurable Simulation in Harness" *Proceedings – 1999 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '99)*, Las Vegas, NV, pp. 730-736, June 1999.
6. M. Beck, J. Dongarra, G. Fagg, A. Geist, P. Gray, J. Kohl, M. Migliardi, K. Moore, T. Moore, P. Papadapoulous, S. Scott, V. Sunderam, "Harness: A Next Generation Distributed Virtual Machine", *Future Generation Computer Systems*, Vol. 15, Nos. 5-6, pp. 571-582, October 1999.
7. V. Getov, P. Gray, V. S. Sunderam, "MPI and Java-MPI: Contrasts and Comparisons of Low-Level Communication Performance" *Proceedings – Supercomputing '99*, Portland, OR, November 1999.
8. M. Migliardi, V. Sunderam, "Networking Performance for Metacomputing in Java", *Proceedings – 1999 IASTED Conference on Parallel and Distributed Computing and Systems (PDCS'99)*, Cambridge, MA, November 1999.
9. M. Migliardi, V. Sunderam, "Hybrid Simulation in Harness", *Proceedings – ASTC High Performance Computing 2000*, Washington, DC, pp. 201-206, April 2000.
10. M. Migliardi, V. Sunderam, "The Harness PVM-Proxy: Gluing PVM Applications to Distributed Objects Environments and Applications", *Proceedings – IPDPS Heterogeneous Computing Workshop 2000*, Cancun, Mexico, pp. 309-322, May 2000.
11. M. Migliardi, V. Sunderam, "User Controlled Task Migration in HARNES", *Proceedings – 2000 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '00)*, Las Vegas, NV, pp. 1385-1392, June 2000.

12. M. Migliardi, V. Sunderam, A. Frisiani, "A Simple Fault Tolerant Naming Scheme for the Harness Metacomputing System", *Proceedings - EuroPVM/MPI Conference*, Balaton, Hungary, Lecture Notes in Computer Science, Vol. 1908, pp. 152-159, Springer-Verlag, September 2000.
13. M. Migliardi, S. Schubiger, and V. Sunderam, "A Distributed JavaSpace Implementation for Harness", *Journal of Parallel and Distributed Computing*, Vol. 60, pp. 1325-1340, November 2000.
14. P. Gray, V. S. Sunderam, and V. Getov, "Aspects of Portability and Distributed Execution for JNI-Wrapped Code", *Journal of Concurrency: Practice and Experience*, December 2000.
15. M. Migliardi and V. Sunderam, "Automatic Reincarnation of Plug-Ins in the Harness Metacomputing System", *Proceedings - IPDPS Heterogeneous Computing Workshop 2001*, San Francisco, CA, April 2001.
16. M. Migliardi and V. Sunderam, "Emulating Parallel Programming Environments in the Harness Metacomputing System", *Parallel Processing Letters*, Vol. 11, No. 2 &3, pp. 281-295, September 2001.
17. D. Kurzyniec, V. Sunderam, "Java Class Loading Techniques in the Harness Metacomputing Framework", *Proc. IFIP Third International Conference on Distributed Applications and Interoperable Systems*, pp. 305-310, Krakow, Poland, September 2001.
18. M. Migliardi and V. Sunderam, "Networking Performance for Distributed Objects in Java", *Proceedings - The 2001 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'2001)*, Las Vegas, NV, June 2001.
19. M. Migliardi, V. Sunderam, "Plug-ins, Layered Services and Behavioral Objects: Application Programming Styles in the Harness Metacomputing System", *Future Generation Computer Systems*, Vol. 17, No. 6, pp. 795-811, April 2001.
20. M. Migliardi and V. Sunderam, "Automatic Reincarnation of Deceased Plug-Ins in the Harness Metacomputing System", *International Parallel and Distributed Processing Symposium (IPDPS-HCW) 2002*, Ft. Lauderdale FL, April 2002.
21. R. Lovas, V. Sunderam, "Debugging and Visualization of Metacomputing Applications", *International Parallel and Distributed Processing Symposium (IPDPS-JPDC) 2002*, Ft. Lauderdale FL, April 2002.
22. M. Migliardi, D. Kurzyniec, V. Sunderam, "Standards Based Heterogeneous Metacomputing: The Design of HarnessII", *International Parallel and Distributed Processing Symposium (IPDPS-HCW) 2002*, Ft. Lauderdale FL, April 2002.

4 Summary

This report has outlined Harness project accomplishments at Emory University under DoE grant DE-FG02-99ER25379 during the project and budget period 3/15/1999 through 3/14/2003. Further details, including copies of publications and software distributions are available through the project website at <http://www.mathcs.emory.edu/harness/>