globusWorld 2004, San Francisco CA, January 2004.

A High-Fidelity Software Oscilloscope for Globus*

Mark K. Gardner†, Wei Deng•, T. Stephen Markham†‡,
Celso L. Mendes•, Wu Feng† and Daniel A. Reed•

†Los Alamos National Laboratory ‡Brigham Young University •University of Illinois
Los Alamos, NM 87545 Provo, UT 84602 Urbana, IL 61801
{mkg, feng, stephen}@lanl.gov tsm9@email.byu.edu {weideng, cmendes, reed}@cs.uiuc.edu

In the same way that large-scale integration made the development of complex electronic hardware possible, toolkits, such as Globus [3], have improved our ability to develop grid applications. However, the task of debugging and tuning grid applications is still difficult. Not surprisingly, the causes of problems in grid applications are also distributed and hence extremely difficult to identify.

When a hardware engineer needs to diagnose a problem with a circuit, the tool of choice is often an oscilloscope. By adjusting the settings of the oscilloscope, the proper level of detail for the task at hand is easily obtained. The utility of the oscilloscope lies in its ability to provide high-fidelity information and the ease with which the level of detail can be adjusted. An analogous tool, the "software oscilloscope," is needed to make grid application development easier.

A software oscilloscope for grid applications requires a high-fidelity, low-overhead source of data on each node. It also requires a means of aggregating and collating events from multiple nodes into a global perspective of grid operation. Furthermore, it should provide tools to the developer for analyzing the data.

We use the MAGNET toolkit [2] as the source of high-fidelity events on each node of the grid. It provides a very efficient mechanism for obtaining information about an executing grid application without requiring the application to be modified or relinked.

Just as an oscilloscope can produce information with too much detail, the information MAGNET exports can be too low level. We use MUSE [1], to filter out unnecessary events and to synthesize application-level information from sequences of operating system events. (For example, MUSE synthesizes bandwidth from socket send and receive events or process utilization from context switch events.) Filtering and synthesizing are the "knobs" of the software oscilloscope that allows the level of detail to be tailored to the needs of the developer.

The final required mechanism is a means for aggregating and collating information. We use Autopilot [4] to aggregate events from each node, to analyze application behavior and to produce data that can be conveniently displayed by other tools.

Figure 1 shows a block diagram of the interaction of MAGNET, MUSE and Autopilot in the context of a Globus application. A Globus application consists of processes executing on multiple nodes. Each node is instrumented with MAGNET to collect various events from the operating system kernel, such as network I/O, disk I/O, memory management, inter-process communication and context switch events. The events are synthesized into appropriate application-level information, such as bandwidth or processor utilization, by MUSE. The synthesized events are collected, analyzed and prepared for display by Autopilot.

Figure 2 shows an example of the data that is made available by this approach, using sensors with a sampling period of one second. The graph displays how the observed network traffic in one node changed when the Scalapack application was run across grid nodes at Illinois and San Diego. The Scalapack execution started at T=70 and finished at T=240. In Scalapack, messages exchanged between the processors become progressively smaller as the matrix is reduced along the diagonal, an effect that can be clearly noticed from the displayed data.

We have motivated the need for a software oscilloscope by noting that developing, debugging and tuning grid applications is a difficult task, leaving a software oscilloscope deployed in a production system also makes sense. If high-fidelity information about the grid is continuously available, that information can be used to improve the performance of the application. For example, if the processor of one node is highly utilized while the processor of another is not, the computation may complete more quickly if part of the load is migrated away from the high-utilization processor. Autopilot has the capability to adjust the behavior of a grid application. In future work, we plan to show how the data provided by our software oscilloscope for Globus can be used in performance contracts [5] to dynamically monitor and adjust the operation of grid applications for better performance.

References

[1] M. K. Gardner, M. Broxton, A. Engelhart, and W. Feng. Muse: A software oscilloscope for clusters and grids. In *Procedings of the 17th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2003)*, Apr 2003. (Gives the details of the current implementation of MUSE).

[2] M. K. Gardner, W. Feng, M. Broxton, A. Engelhart, and G. Hurwitz. MAGNET: A tool for debugging, analysis and reflection in computing systems. In *Proceedings of the 3rd IEEE/ACM International International Symposium on Cluster Computing and the Grid (CCGrid'2003)*, May 2003. (Gives details about the current implementation of MAGNET).

[3] The Globus Project. `http://www.globus.org/`.

[4] R. L. Ribler, J. S. Vetter, H. Simitci, and D. A. Reed. Autopilot: Adaptive control of distributed applications. In *Proceedings of the 7th IEEE Symposium on High-Performance Distributed Computing (HPDC-7)*, Jul 1998.

[5] F. Vraalsen, R. A. Aydt, C. L. Mendes, and D. A. Reed. Performance contracts: Predicting and monitoring grid application behavior. *Lecture Notes in Computer Science*, 2242:154–165, Nov 2001.

*Los Alamos Unclassifi ed Report (LA-UR) 03-7681.

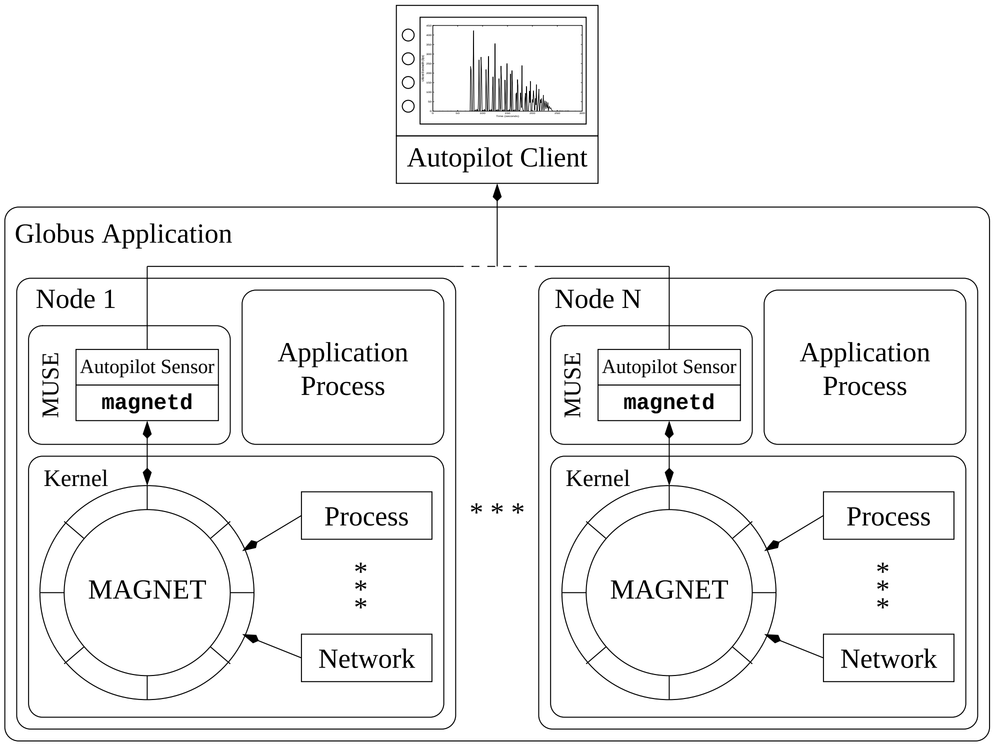

Figure 1. Architecture of a Software Oscilloscope for Globus

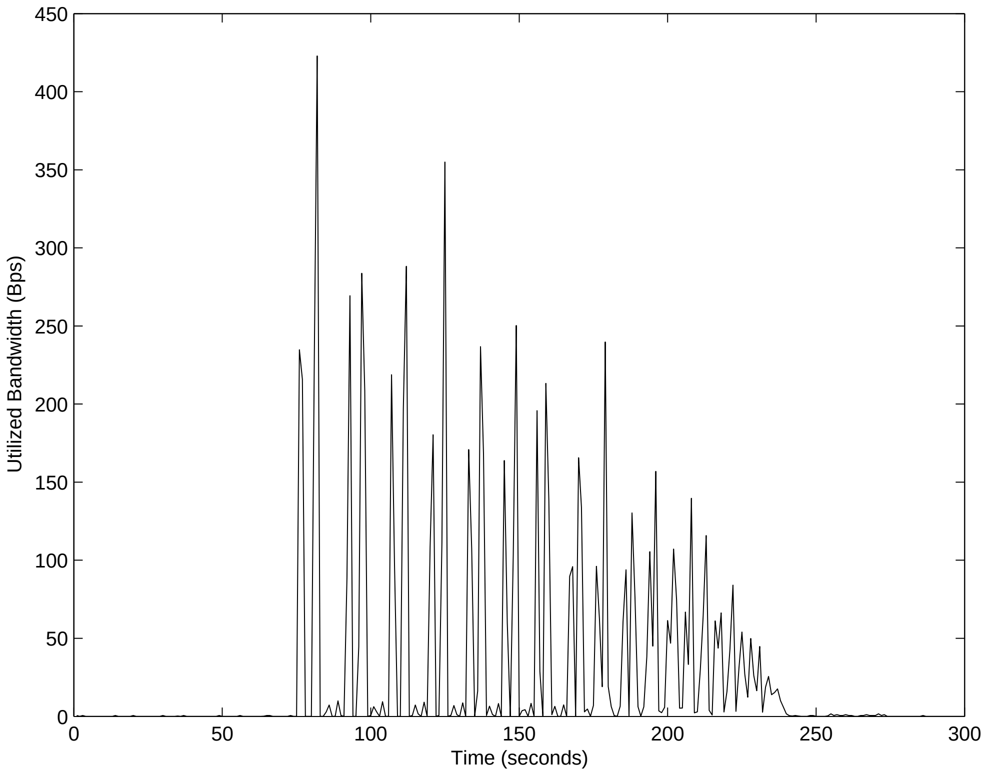

Figure 2. Network Traffic Data for Scalapack, Exported by MUSE via Autopilot Sensors