A Reconfigurable Monitoring System for Large-scale Network Computing

Rajesh Subramanyan[1], José Miguel-Alonso[2], and José A.B Fortes[3]

[1] School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA
[2] Department of Computer Architecture and Technology, The University of the Basque Country UPV/EHU, San Sebastian, Spain
[3] Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA

Abstract. The dynamic nature of large-size Network Computing Systems (NCSs) and the varying monitoring demands from the end-users pose serious challenges for monitoring systems (MSs). A statically configured MS initially adjusted to perform optimally may end performing poorly.
A reconfiguration mechanism for a distributed MS is proposed. It enables the MS to react to changes in the available resources, operating conditions, and monitoring requirements, while maintaining high performance and low monitoring overheads. A localized decision process involving two adjacent intermediate-level managers (ILMs) and values of a local node performance parameter called *temperature* together determine transformations (merge, split, migrate) for each ILM. The reconfiguration mechanisms are derived reusing SNMP primitives. Interactions between MS and NCS are studied by defining a queuing model, and by evaluating different configuration schemes using simulation. Results for the static and reconfigurable schemes indicate that reconfiguration improves performance in terms of lower processing delays at the ILMs.

1 Introduction

A Network Computing System (NCS) is a large collection of heterogeneous resources spread across networks that provides application services to many users. The Grid [1], is an example of such a system. We assume that in NCS, nodes leave and join continually, and are separated by sizeable network distances with unpredictable delays. The performance of the MS is critical to the accuracy and timeliness of monitoring information. A centralized MS cannot scale for large sizes. In [2], we describe a scalable, hierarchical, distributed MS called SIMONE.

Most distributed MS are static, that is, configured manually at start-up time based on the prevalent conditions. The dynamic nature of the NCS environment has been largely overlooked (some exceptions are [3], [4] and [5]). This dynamic nature arises due to (i) changing operating conditions of the NCS, e.g. network load, processor load, varying number of nodes in the NCS pool (alters monitor's

[2] This work has been done with the support of the Ministerio de Ciencia y Tecnologia, Spain, under contract MCYT TIC2001-0591-C02-02.

target size and available resources), and (ii) changing monitoring requirements, e.g. increase parameter update frequency. The application, user, or the system may determine these requirements.

In a dynamic NCS, a distributed configuration initially designed to be optimal may perform poorly over extended periods of time. Monitoring overheads on the NCS may increase due to the skewed usage of resources by the MS.

In this paper, we propose a reconfigurable MS as a solution to the above performance problem. Reconfiguration is an adaptation technique whereby the components of the MS, or the (logical) connections among them, change automatically, adapting themselves to the changing environment with the aim of fulfilling monitoring requirements and reducing overheads. Additional reasons for reconfiguration could include administrative demands, node failures, freeing nodes that are needed for running critical applications.

The remainder of the paper is organized as follows. Section 2 provides some architectural background. Section 3 models the NCS, MS components, and applications running on the NCS, and describes metrics for assessing MS performance. The reconfiguration system and simulation results are discussed in Sections 4 and 5. Related work and conclusions are discussed in Sections 6 and 7 respectively.

2 Towards a Reconfigurable Architecture

In a SNMP-based MS, a manager (client) requests and obtains monitoring information from several agents (server). Agents listen and respond to manager requests (Figure 1). A single centralized manager becomes a bottleneck for agents exceeding 100 to 200 [2]. In the distributed solution [2], several intermediate-level managers (ILMs), in one or more levels, are delegated managerial tasks by the top-level manager (TLM). TLM retains the overall MS control. An ILM is a dual entity: an agent to its manager (a TLM or an upper-level ILM) and a manager to its agents (Figure 1(b)). Its duties are defined as downloadable scripts, providing flexibility, while communication continues to be strictly SNMP. The distributed SIMONE reduces monitoring latency by an order of 2-10 [2]. The configuration is determined at start-up time and remains static.

A reconfiguration mechanism that can be incorporated in SIMONE is proposed. Depending on the temperature (node performance metric) value, a merge, split or a migrate transformation may be performed on the ILM, resulting in the reassignment of managerial tasks or responsibilities (measured in terms of number of managed nodes). The solution must be simple and efficient (imposing low overheads), scalable (we anticipate its use in large-scale NCS), capable of integration with existing monitoring methods and fault-tolerant.

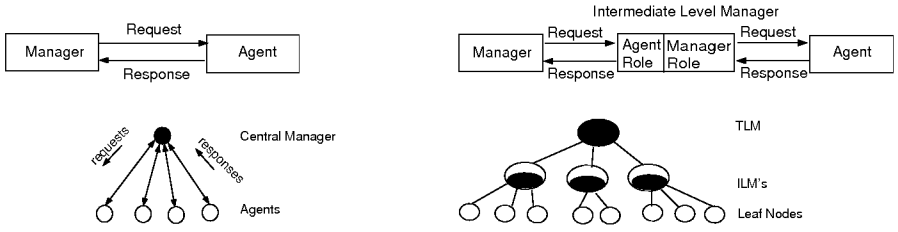

Fig. 1. Model of the client/server paradigm in the centralized (left) and distributed (right) MS

3 Model of the MS in an NCS Environment

A queuing model has been developed to assess the impact of introducing a MS in an NCS and study MS-NCS interactions. Details are described in [6]. Below, is a summary of the most salient aspects of the model (see Figure 2). Nodes and jobs are classified based on their monitoring roles.

1. There are three classes of nodes: leaves (computing nodes), ILMs, and TLM.
2. A node can have an input queue to receive jobs, a CPU queue to process them, and an output queue to send jobs upward in the monitoring tree.
3. A computing node (leaf, ILM) processes NCS jobs (computing jobs), modeling the actual computing tasks assigned by an external scheduler (NCS scheduling is outside MS control). Also, it processes MS-Poll jobs, modeling the monitoring tasks, and generates MS-Report jobs that are sent to their managers.
4. A manager node (ILM, TLM) processes MS-Report jobs. All nodes run agents (MS-agent jobs).
5. The distribution of the inter-arrival time of MS-Poll and MS-agent jobs are obtained from manager polling and agent update periods (MS operational parameters) respectively. Measurements from an actual MS (SIMONE) were used to determine the CPU usage required to process MS-Poll and MS-Report jobs.
6. Analysis of data from an actual Network Computing System (PUNCH [7], developed and operational at Purdue University) allowed us to get good approximations of realistic distributions for inter-arrival time and duration of NCS jobs.

3.1 Performance Metrics

In this paper, latency in measured for a node and not end-to-end. Node latency is the time between a MS-poll job entering a node until it leaves the node (or the last queue). We introduce **temperature** of a node as a local metric, which assesses the performance of an ILM node and the impact of monitoring tasks on NCS applications in the reconfigurable system. It is expressed as $T = aI + bL$, where, "I" is CPU intrusion and "L" is (node) latency, and "a" and "b" are weight factors. Intrusion affects latency as a heavily used node takes longer to process jobs. Since load is subsumed in latency value, we currently assign a weight 1 to latency and 0 to intrusion. Weights, thresholds and temperature expressions are set and can be dynamically changed by the administrator based on user needs. We define two local metrics and a global one.

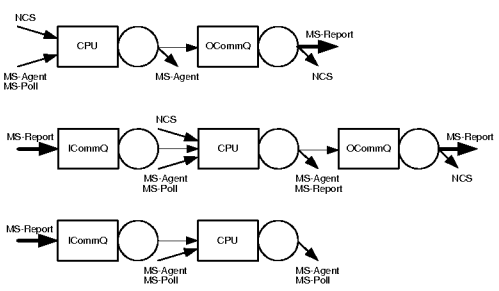

Fig. 2. Models of the components of the monitored NCS: leaf (top), ILM (middle), TLM (bottom). TLM does not process NCS jobs.

- Average node latency: mean delay experienced by poll jobs in an ILM. Each ILM has an average node latency.
- Standard deviation of node latency: measures variations in latency experienced by poll jobs in the ILM node over a period of time.
- Global node latency: mean value of average node latencies of all the ILM nodes. A single value for the whole MS.

4 Reconfiguration

Mechanisms to reduce the overall system temperature fall into two categories,

1. Reducing the activity of the monitoring system [5], e.g. increasing AUP and/or MPP, to reduce the number of parameters measured.
2. Modifying architectural characteristics of the MS to improve its performance without changing MS operational parameters.

Both the approaches are complementary and can be combined. Currently, we assume that the MS operational parameters are already optimally adjusted to satisfy application requirements and further changes compromise monitoring requirements. In this paper we focus only on the second approach.

MS performance can be improved beyond decentralization [2] if the ILM configuration is allowed to change. The arrangement of (logical) connections from leaves to ILMs can be dynamically modified to improve efficiency by moving managerial MS tasks (in terms of number of managed nodes) from heavily loaded ILMs to less loaded ILMs. We call this reconfiguration. Thus, the responsibility of managing node(s) is transferred between ILMs.

The transformations to be applied on ILMs are decided based on current values of temperature which are periodically computed. Keeping the large size of NCS in mind, a localized decision process involving two adjacent levels of nodes is used (see Figure 3). Node A collects temperature and support node lists (SL) for level $I+1$, computes, and then updates the new support list at level $I+1$ thus changing their managerial loads. If Node A detects failure(s) at level $I+1$ (lack of response within timeout), it reassigns the support list of the affected ILM(s) saved from the previous poll, among the remainder level $I+1$ ILMs. ILMs receive a trigger to initiate its monitoring activities [2]; the same is used to trigger transformations. The transformations are now described as,

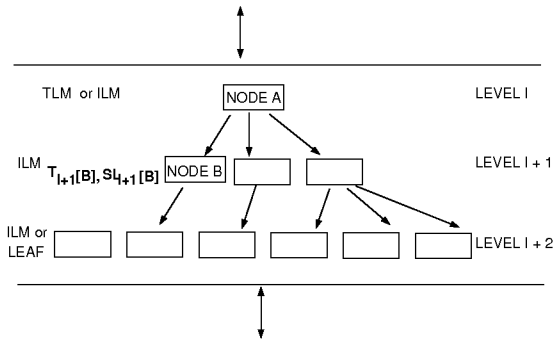

Fig. 3. Segment of the distributed MS showing level I, $I+1$ and $I+2$. Each ILM stores temperature, T, and a list of its supported nodes. Transformations for ILMs at level $I+1$ are decided by level I nodes.

- Migrate- a node stops running an ILM and its managerial tasks are assigned to another, less "hot" node.
- Split- a node does not stop running an ILM but decides to pass part of its managerial tasks to another node.
- Merge- the complement of split. A "cold" ILM assumes the managerial tasks of another cold ILM, which then becomes a regular node.

Migrate and split operations try to reduce the temperature of an ILM. The utility of the merge operation when two nodes are operating within allowable ranges may be debated. The reasons are to free a lightly loaded node that can be fully dedicated to process NCS jobs, experiments show that due to additional communication and synchronization costs, too many ILMs are counterproductive to performance [2], and for symmetry, just as new resources are added when required (split), resources should be freed when not required.

Agents and managers need to be modified to implement these mechanisms. Adhering to SNMP, our design uses an extended MIB to store information needed for reconfiguration. Mechanisms reusing simple SNMP communication primitives are devised for triggers, computation and communication.

The overhead cost, from decision process and transformation enactment, needs to be weighed in before enacting reconfiguration. These are measured in terms of complexity of the algorithm (computational costs and memory), storage costs, communication costs and delays. Decisions and node list computations are performed between two adjacent levels locally. If ILM at level i supports X ILMs, the overheads for ILM level i are: computational complexity of decisions is O(X), storage is $(X + C)$ MIB variables, and communication: $2X$ SNMP requests to send and receive node lists.

5 Simulation, Results and Interpretation

The Ptolemy [8] environment was used to design simulation setup for centralized, static distributed and reconfigurable schemes. Additional components including statistical recorders and processor sharing servers were designed by us. Simulations were conducted for single level 2-64 ILMs, 64-2048 leaf nodes for NCS-P[4] NCS-H (higher load and arrival rate than NCS-P) jobs using job distribution characteristics reported in [6] and data from previous experiments [2]. The simulation setup was validated using experimental data for central and distributed schemes. Both, the experimental and simulation values showed similar trends [6], despite the comparisons being approximate (experiments measured end-to-end delay and used different NCS loads).

Time units were defined as simulated time units (stu), where each stu is a clock tick in the simulation and set to be equal to 100 ms of real time. Simulations were run for identical number of stu's. Statistical components record timing data by observing various stages of the job flow. The following measurements were performed.

[4] P refers to PUNCH, the NCS whose traces where used to model the distribution of jobs

- Global and ILM latencies for different configurations of static and reconfigurable schemes, using NCS-P and NCS-H jobs.
- Impact of thresholds and trigger periods on MS performance.
- MS performance in hot zone (high temperature region) for different schemes.
- Improvements by including past values in temperature computation.

5.1 Global and average latency, deviation in latency

Simulations were conducted using NCS-P jobs for different configurations (64-2048 leaves, single level 2-64 ILMs) for static and reconfigurable schemes. For small (> 32) fanin's, both schemes have low global latencies, with static being slightly lower. Saturation occurs at large fanin's, making comparisons irrelevant. Reconfiguration delays saturation and reduces global latencies as jobs submitted to lighter nodes cause smaller increases in latency. Two ILMs have different average latencies despite reallocation, as a fraction of jobs (MS) alone are rescheduled. Results indicate that the reconfigurable MS performs better than the static; has 10-60% lower global latencies for intermediate fanin's, has smaller and flatter ILM latencies, and smaller latency deviations (sample results are shown in Figure 4).

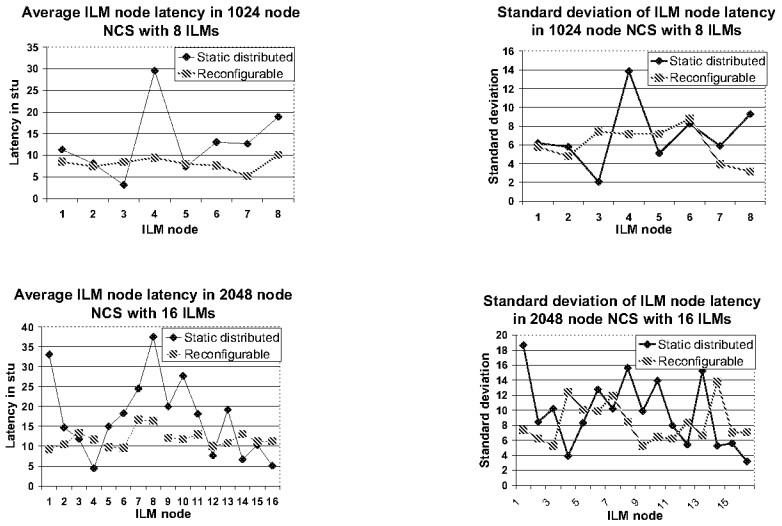

Fig. 4. Comparing average and standard deviation of ILM latencies for static distributed and reconfigurable schemes for 1024 nodes/8 ILMs and 2048 nodes/16 ILMs using NCS-H jobs. All units are in stu.

5.2 Effect of varying thresholds and trigger periods

Threshold values are expressed as [merge, split, migrate], e.g. [0.1-25-500]. Here temperature below 0.1 triggers merge, 0.1 and 25 causes no change, between 25 and 500, a split, and above 500, a migrate. Results of varying one threshold value at a time (Figure 5), increasing one threshold region at a time (Table 1), and

varying the trigger period (Figure 5) using 1024 ILMs with 8 ILMs and NCS-H jobs are shown. The optimal values appear in the middle, a common pattern observed in all cases. At the end regions, either several spikes occur with too few transformations, or frequent transformations lead to poor decisions (an ILM loaded heavily for short intervals need not undergo transformations).

At low split threshold, global latency may even exceed that of static distribution due to very frequent transformations. Large migrate thresholds yield low latencies suggesting sparing use of migrate transformation. Merge frees up ILM nodes, but increases load on remaining ILMs. Low merge thresholds are recommended particularly with few ILMs. Larger region for split (more split transformations) improves performance, provided the split thresholds are not selected too low. Trigger rate improves performance by prompt reaction to spikes, however, very high rates are counterproductive (Figure 5).

Currently, we base threshold selections on user requirements rather than achieving optimality. Results indicate strong effect of threshold and trigger values on MS performance, suggesting automation.

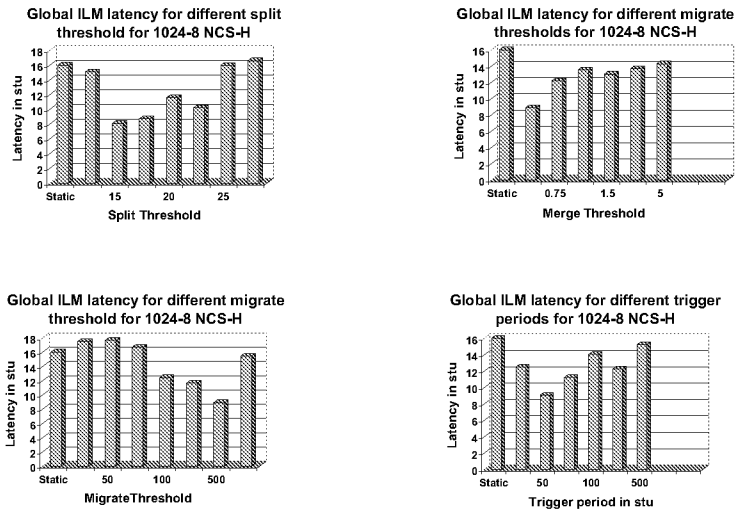

Fig. 5. Effect of varying threshold or trigger for 1024 node NCS/8 ILMs reconfigurable scheme, using NCS-H jobs. The figures above are (i) vary split threshold, merge threshold = 0.1, migrate = 500, (ii) vary migrate, merge = 0.1, split = 17.5, (iii) vary merge, split = 17.5, migrate = 500, (iv) vary trigger period. Global latencies are given in units of stu.

5.3 Operating in Hot Zone

"Hot zone" is a region of high temperature for ILM operation. ILMs with temperature exceeding 1.2 times average global latency are considered to operate in the hot zone. Figure 6(i) indicates that reconfiguration significantly reduces the

Table 1. Effect of thresholds for 1024-8 reconfigurable configuration, using NCS-H jobs. Global latencies are given in units of stu.

Threshold Set	Transformation with increased threshold range	Global latency (stu)
-	Static distribution	15.89
0.1-20-500	Normal	11.6
0.1-10-500	Split	14.27
0.1-20-50	Migrate	17.7
1-20-500	Merge	13.3

percentage of time spent by ILMs in the hot zone. Medium thresholds yield best results. While ILMs in static scheme spent on an average 37% of total time in hot zone, the corresponding values were 23%, 10% and 24% for reconfigurable schemes with low, medium and high thresholds under identical conditions.

ILMs in static scheme spent on an average 19 trigger intervals per visit to the hot zone, while the corresponding values for the reconfigurable scheme with low, medium and high thresholds were 4, 3, and 7 (Figure 6(ii)). The average length of time spent in the hot zone for each visit reflects how prompt and effective the transformations are. High thresholds increases the time spent in the hot zone. In the static scheme, ILMs leave hot zone only due to a fall in NCS activity.

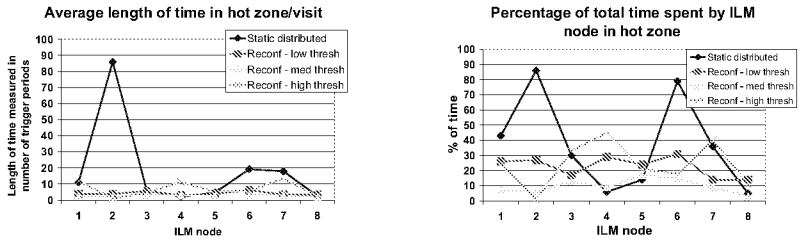

Fig. 6. (i) Percentage of total time spent by ILM in hot zone. Threshold values are low = [0.1-10-500], medium = [0.1-17.5-500], high = [0.1-25-500]. (ii)Average duration (counted in number of trigger cycles) spent by ILM node in the hot zone/visit.

5.4 Temperature Computation with History

Two sets of simulations were carried out under identical conditions, except one uses current temperature (Tc), while the other uses a "weighted" temperature (Tw). Higher weights are accorded to recent polls, and selected as follows: $Tw = 0.57Tc + 0.28Tc_1 + 0.15Tc_2$, where Tc is the temperature during the current poll, Tc_1 the temperature during the previous poll, and so on. Weighted temperature yields better results by helping avoid transformations for short temporary spikes. Using current temperature, global latency values were 14.25, 8.09 and 15.8 for low, medium and high threshold. Corresponding figures with weighted temperature were 13.07, 10.27 and 12.47 respectively.

6 Related Work

Tools designed for network computing environment are: NWS [9] which provides dynamic resource performance forecasts; Globus' Gloperf [10] tool which works like NWS, and periodically schedules end-to-end tests to retrieve latency and

bandwidth information; Netlogger [11], a trace-based system used mainly for diagnosis, and Remos [12] which combines different monitoring techniques such as SNMP and end-to-end tests focusing on providing information for network-aware applications. Performance issues when working in large networks are tackled by using a hierarchy of groups in NWS and Globperf, by stopping and starting logging process in Netlogger, and by implementing a hierarchical monitoring architecture using a data collector in Remos. These solutions are statically configured and use SNMP as an enhancement, thus differing from our solution. The Grid Performance Working Group of the Grid Forum has ongoing work towards defining a monitoring service architecture for the Grid [1]. SIMONE will be made compliant with these specifications to allow interaction with other Grid systems.

While reconfiguration, migration and other concepts have been widely studied in other areas [13], most monitoring systems have static configurations. Mobile code paradigms for network management have suggested taking advantage of migration. Liotta [13] uses a dynamic hierarchical manager model based on delegation using mobile agents, for scalable monitoring. Liotta indicates the complexities and drawbacks of using mobile code. Our solution strictly uses SNMP mechanisms and is applicable to any SNMP-based MS. Lutfiyya's [3] adaptive model, studies the reconfiguration of monitoring elements in CMIP [14] inorder to optimally minimize monitoring overheads in small systems. Our algorithm chooses a simple approximate solution, which may be better suited for large systems. Hollingsworth [5] proposes an adaptive cost system for Paradyn for maintaining perturbation [15] levels in applications by delaying dynamic instrumentation until execution. This allows the flexibility of dynamic insertion and alteration. Performance is gained by reducing instrumentation. Our solution assumes reducing polling frequency would compromise monitoring accuracy requirements. The Quorum [16] program by DARPA has several ongoing initiatives on resource management, scheduling etc.

7 Conclusions

A reconfigurable mechanism is proposed that reassigns monitoring responsibilities (managed nodes) among ILMs in response to changing operating conditions, thus maintaining MS performance. Temperature, threshold, transformations, trigger and other mechanisms are designed fully within the SNMP context and can be integrated with existing SIMONE. Design features allow ILM robustness, local transformation decisions aid scalability, and MIB extensions and SNMP reuse makes the solution simple and interoperable. A queuing model was developed to describe MS-NCS interactions, and distribution of inputs were derived. These were used to design and perform simulations for 64-2048 nodes and 2-64 single level ILMs configurations.

Simulation results indicate 10-60% smaller global latencies and less variation in average ILM latencies in the reconfigurable scheme. Too few or too many transformations reduces performance as observed while varying thresholds or trigger period. The performance is sensitive to threshold selection, suggesting automation. Infrequent merge and migrate is desired combined with medium

occurrence of split transformation. Under identical conditions, static ILMs spent 37% of the time in hot zone compared to 10-24% for reconfigurable schemes. Correspondingly, the average stay/visit in hot zone was 19 for static ILMs as compared to 3-7 in reconfigurable schemes, expressed in units of trigger interval.

The above results indicate that reconfigurable scheme improves performance and significant gains may be achieved with proper selection of thresholds. Adding fork and join transformations, automating threshold selection, and developing backup TLM are future enhancements proposed.

References

1. I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations.", *in Intl. J. Supercomputer Applications*, 2001.
2. R. Subramanyan, J. Miguel-Alonso and J.A.B Fortes, "A Scalable SNMP-based Distributed Monitoring System for Heterogeneous Network Computing", *in Supercomputing*, Nov. 2000.
3. Hasina Abdu, Hanan Lutfiyya and Michael A. Bauer, "A Testbed for Optimizing the Monitoring of Distributed Systems", *in Proceeding of PDCS '98*, 1998.
4. A. Liotta, G. Pavlou, G. Knight, "A Self-adaptable Agent System for Efficient Information Gathering.", *in MATA*, 2001.
5. J.K. Hollingsworth and B.P. Miller, "An Adaptive Cost System for Parallel Program Instrumentation", *in Euro-Par '96*, Aug. 1996.
6. R. Subramanyan, *Scalable SNMP-Based Monitoring Systems for Network Computing*, PhD thesis, Purdue University, Aug. 2002.
7. N.H. Kapadia and J.A.B. Fortes, "PUNCH: An Architecture for Web-Enabled Wide-Area Network-Computing", *Cluster Computing*, Sept. 1999.
8. *Ptolemy 0.7 User's Manual, UC Berkeley, http://ptolemy.eecs.berkeley.edu.*
9. R. Wolski, "Dynamically Forecasting Network Performance using the Network Weather Service", *Cluster Computing*, 1998.
10. C.A. Lee, J. Stepanek, R. Wolski, C. Kesselman and I. Foster, "A Network Performance Tool for Grid Environments", *in HPDC 98*, 1998.
11. B. Tierney, W. Johnston and B. Crowley, "The Netlogger Methodology for High Performance Distributed Systems Performance Analysis", *in HPDC 98*, 1998.
12. Nancy Miller and Peter Steenkiste, "Collecting Network Status Information for Network-Aware Applications", *in Proceeding of Infocom 2000*, 2000.
13. Antonio Liotta, Graham Knight and George Pavlou, "On the Performance and Scalability of Decentralized Monitoring using Mobile Agents", *in DSOM*, 1999.
14. U. Black, *Network Management Standards*, McGraw-Hill, 1995.
15. A.D. Malony, D.A. Reed and H.A.G. Wijshoff, "Performance Measurement Intrusion and Perturbation Analysis", *in IEEE Transactions on Parallel and Distributed Systems*, July 1992.
16. DARPA, *Quorum Project*, http://www.darpa.mil/ito/research/quorum/index.html.
17. G. Goldszmidt, *Distributed Management by Delegation*, PhD thesis, Columbia University, Dec. 1995.
18. A. Waheed , D.T. Rover, M.W. Mutka, H. Smith, and A. Bakic, "Modeling, Evaluation, and Adaptive Control of an Instrumentation System", *in Proc. Real-Time Technology and Applications Symposium (RTAS '97)*, June 1997.
19. M. Siegl, *Design and Realization of a Mid-Level Management System*, PhD thesis, TUW-HDNM, Vienna University of Technology, Nov. 1996.
20. A. Pras, *Network Management Architectures*, PhD thesis, Centre for Telematics and Information Technology, University of Twente, April 1995.