Research Overview

Jeffrey S. Chase
Department of Computer Science
Duke University
Spring 2008

My research focus and methodologies are rooted in the operating systems research community. My research is experimental in nature, complementing the analytical approaches dominant in many areas of computer science. Experimental systems research blends science with the development skills and creativity needed to convert good ideas into functioning, useful software systems. Its purpose is to develop and refine the fundamental techniques relevant to practical needs in computing systems, implement them in a realistic context, design and conduct experiments to evaluate them, and in many cases, to release system prototypes that serve as a basis for continuing research in the broader community. Experimentation is the ultimate test of the validity and practicality of new ideas, and it is an essential element of technology transfer in computer science.

My research since 2001 focuses on managing networked computing and storage "cyberinfrastructure" as a service *utility*, in which shared hardware resources are provisioned or sold according to demand, much as electricity is today. This vision rests on several key premises:

- An effective environment for dependable computing will combine stateless client devices with a distributed service backbone (data storage, Web services, interaction, computation) whose functions are made scalable and reliable through geographic dispersion and replication [20].
- These services are hosted on a server backbone infrastructure residing in data centers and other edge sites throughout the Internet; the edge resources (CPU, storage, memory, etc.) may be shared, like the network itself.
- Flexible service architectures and dynamic resource sharing will enable a fluid mapping of service functions and data onto the infrastructure, so one may instantiate services wherever resources are available and demand exists.
- Critical services must sense-and-respond to adapt automatically to changes in traffic demands or resource conditions, while holding human administrative burdens constant.
- The server network and the hosted services constitute a critical public infrastructure requiring open, flexible, secure, robust, and decentralized control. The control architecture must resolve the "tussle" of contending demands, changing priorities, and rapidly advancing technology, all within the framework of a self-sustaining system. Any such system is inherently economic.

Variations of this vision are shared by many of my colleagues, and the fundamental elements and approaches are evolving under various names including utility or on-demand computing, self-managing systems or autonomic computing, grid computing, service-oriented architectures, federated and incentive-structured systems, market-based systems, overlay networks, and peer-to-peer computing. Each of these names is associated with a community that addresses common core problems from a particular perspective, often under different assumptions about the infrastructure, workload, or architectural principles. I have engaged with each of these communities to varying degrees.

My research has addressed various elements of a computing utility economy: models to predict demand and behavior [23, 57, 58]; fundamental abstractions and scheduling mechanisms to "slice" the resources as a measured and metered quantity [10, 18, 37, 39]; protocols to represent and enforce accountable contracts [27, 68, 69, 66, 34]; control and optimization of resource slices for service quality metrics including dependability [16, 23, 40]; scalable architectures for resource brokering and discovery [10, 27]; flexible and adaptive service architectures for content services and network storage [7, 63, 42, 24, 12, 2]; data center architecture and server energy management [16, 53, 44, 47]; statistical inference of system behavior from instrumentation data [22, 43, 44]; and market-based resource management [36, 34].

Through this program of research, I have focused on a core set of architectural principles as a practical and general foundation for critical utility services. Server functions are distributed across a modest subset of sites according to their identities and attributes, rather than across a massive number of anonymous entities as in peer-to-peer systems [59]. Guest services interact with the utility to obtain leases for virtualized shares of "raw" resources provisioned to meet target levels of service quality. Resource leases may be brokered by third parties, and represent contractual arrangements among self-interested participants, who are held accountable for their contracts. Instrumentation generates a continuous feedback signal to drive policy choices. The participants act based on local information and internal models of the system and applications. These local choices produce an emergent global behavior.

The web sites for the NICL lab and the Orca project provide the best current information about our research toward this vision. In the course of our research, the Orca project has developed open-source software for on-demand virtual computing environments based on a foundational abstraction of resource leasing. Orca project software includes the *Shirako* leasing core [35, 50], the *Automat* control portal and related components, a new implementation of the SHARP framework for accountable lease contracts and brokering [27], and Cluster-on-Demand (COD), a back-end resource manager for shared clusters [18, 48]. Orca is a context for our continuing research in automated resource management and adaptation, which is a primary focus of our ongoing research and development [33]. Two recent doctoral students (Laura Grit and David Irwin) completed their dissertations in the Orca project.

1 Some Projects and Contributions

Managing server resources in hosting centers. Our SOSP 2001 paper established the foundations for adaptive resource provisioning in utility data centers [16]. It described an operating system—called Muse—for a Web hosting center, and it was the first work to envision operating system functions at the scale of an entire building containing multiple servers and other cluster components such as redirecting network switches. Muse continuously monitors incoming request traffic and server load levels for multiple hosted services, and dynamically adjusts the allocated server resources to match the load. The resource allocation policy is based on a simple economic formulation that seeks to maximize global benefit (center profit) as defined by utility functions that capture value for a given level of delivered performance to each sevice. Muse incorporates a simple model-based utility-maximizing optimizer within a global feedback loop. It addressed the core issues of feedback control in this new setting: filtering instrumentation data gathered from servers and switches, calibrating responses to real-world load swings to balance stability and agility.

Muse was also the first system to explore the role of dynamic provisioning for energy management in hosting centers. It could respond to an unexpected power or cooling "browndown" by allocating scarce resources to their highest and best use, or reduce power consumption during periods of light load, holding service quality constant. The issue of server energy and thermal management—now a critical concern—was just beginning to be recognized. The Muse paper demonstrated the potential of coordinated cluster-level energy management without requiring sophisticated energy management features in each server.

Muse and the Slice project (described below) also explored the role of reconfigurable network switches to direct the flow of requests to the server set for each service [12]. We developed this idea further in the Anypoint project [63]) and the dissertation research of doctoral student Ken Yocum.

Networked utilities. We next began to consider how a utility operating system could manage a network of sites hosting widely distributed services. We wrote a concept paper on a hypothetical system (Opus) that exposed core issues for a large-scale network utility [10]: balancing local autonomy with global coordination, optimizing for multiple service quality objectives including consistency and availability, control based on approximate global state, adapting service placement to request locality, and configuring multiple service overlay networks from a pool of shared network resources. Opus set the context for my subsequent work in this area in my collaboration with my colleague Amin Vahdat, who is now at UCSD.

Opus influenced the PlanetLab initiative, which has since created a network utility testbed spanning hundreds of Internet sites. PlanetLab is widely used in the network systems community, and hosts services (such as content distribution) with large user communites. I was engaged with the PlanetLab effort at an early stage. In various research projects I have investigated the relationship of the resource virtualization approach adopted in Planetlab with resource management approaches used in grid computing middleware (e.g., [52, 50]). My research on networked computing utilities continues in the Orca project.

Secure resource peering and accountability. Our 2003 SOSP paper on Secure Highly Available Resource Peering (SHARP) addressed the challenge of managing a PlanetLab-like networked utility as a collection of autonomous sites without central control [27]. SHARP laid the groundwork for an extensible, federated resource economy with accountable contracts. The key to SHARP is a mechanism for cryptographically secure and accountable delegation of control over resources to another actor, such as a broker or a manager for a hosted service, for a bounded period of time (analogous to hierarchical leasing). It addresses key challenges for networked utilities: preserving availability of resources when the actors controlling them fail, decentralized trust management, enforcement of contracts, and reconciling local autonomy with global coordination. SHARP shows how accountable delegation enables a flexible and extensible brokering architecture, making it possible to combine an evolving set of policies for distributed resource management within the utility. We built a SHARP prototype for PlanetLab.

We are currently exploring economic resource management in a broker community in the Cereus project [34]. Cereus shows how to use the SHARP primitives to implement an accountable virtual currency suitable as a generic medium of exchange to allocate community resources to community members using market principles.

Accountability We are also considering the broader role of accountability in distributed systems that cross multiple trust domains, including critical infrastructure control. A system is *accountable* if it provides a means to detect and expose misbehavior by its participants. A system is *strongly accountable* if it provides a means for each participant to determine for itself if others are behaving correctly, without trusting assertions of misbehavior by another participant who may itself be compromised. Accountability provides powerful incentives to promote cooperation and discourage malicious and incorrect behavior [68, 66]. In recent work we developed a toolkit for strongly accountable network services called CATS—Certified Authenticated Tamper-Evident State Store—and showed how to use it to develop a strongly accountable network storage service [67].

Green data centers. The Muse project led to the first investigation of dynamic thermal management for data centers, in collaboration with Partha Ranganathan and others at HP Labs [53, 44, 47]. We observe that power and cooling costs make up a significant share of operating costs for a data center, and that vulnerability to failures of cooling equipment increases with server density. When the center is not saturated, dynamic mapping of workload to servers (as in Muse) can play an important role by balancing thermal load across the data center, and placing load where the cooling system is best able to handle it. We have investigated several approaches to inferring the “thermal topology” of the data center from continuous sensor monitoring [46, 45]. We have developed policies for *temperature-aware workload placement* and evaluated them using computational fluid dynamics (CFD) simulations with fault injection.

Our most sophisticated policies can yield up to a factor of two reduction in annual data center cooling costs for representative workloads. This work on “green” data centers was the dissertation research of doctoral student Justin Moore, and it received notice in MIT’s Technology Review. Partha was honored as a TR-35 Young Innovator for 2007.

With Partha and David Irwin I have also explored approaches to allocate power in dense blade server systems in which the aggregate burst capability of the hardware modules exceeds the power/thermal budget for the combined system enclosure [51].

Self-managing systems. The dynamic feedback-controlled structure of our systems has built awareness of fundamental techniques for self-managing or “autonomic” systems. We are investigating a range of techniques to infer application profiles or application models from passive observations (e.g., of network traffic) and sensor streams, and to use those models as a basis for automated management of storage and CPU resources and multiple tiers; the first example of this approach is [23].

During my 2003-2004 sabbatical at HP Labs I worked with colleagues there on automatic configuration of storage systems to balance of cost and dependability, using an off-the-shelf optimization solver [40]. With another group I investigated statistical induction techniques to recognize system states that correlate with service agreement violations or failures in a multi-tier service [22], using a restricted form of Bayesian network classifier. This approach embodies a rudimentary form of automatic diagnosis: it associates these states with specific combinations of metrics that suggest repair actions.

More recently, I have worked with my Duke colleague Shivnath Babu and our doctoral student Piyush Shivam to induce application performance models from execution histories and instrumentation data. In particular, we have explored proactive approaches to speed learning by perturbing the resources available to each execution and observing the effect on performance. Several publications deal with this research [54, 56, 55].

2 Early Research

I participated in several earlier research projects in operating systems, cluster computing, and distributed computing. They reflect common themes of resource virtualization, uniform storage and sharing, dynamic resource management, and reconfigurable applications and services.

Distributed cluster computing. Dynamically reconfigurable cluster computing has been a theme of my research since the Amber project, which was one of the earliest systems for parallel cluster computing [14, 25]. Amber enabled applications to configure the mapping of data and computation to cluster nodes to balance competing performance objectives of load balancing and locality. Experiments with the prototype demonstrated the potential of reconfiguring this mapping automatically to respond to changes in the set of physical nodes [25]—foreshadowing my current work in adaptive services hosted on dynamic virtual clusters.

Operating systems for wide-address architectures. My dissertation research on the Opal system was the first of several research projects to experiment with a *single address space operating system* approach on emerging 64-bit processors. The appeal of uniform sharing in a protected global address space has produced a long history of systems that approached this objective through varying addressing and protection models. Opal established the basic structure and issues for a protected uniform virtual memory model incorporating shared data and storage, and demonstrated how this structure could enable a continuum of protection and sharing relationships among modular software components in order to balance performance and failure isolation. Elements of the Opal research focused on programming environments for data persistence and protected sharing [11], recovery and consistency for distributed data [26], architectural issues for address translation and protection [41], and operating system abstractions and resource management [19].

Cluster memory and storage. My research in network storage systems began with the Global Memory Service (GMS) project at the University of Washington. GMS was a cooperative network memory page caching service that adapted to changes in cluster load and memory demand. GMS was a response to an order-of-magnitude jump in local network bandwidth, which made it faster to fetch a page from the memory of another node (network memory) than to fetch it from a local disk. We integrated GMS with the Trapeze network I/O system described below, and investigated global prefetching [60, 4] in the context of GMS. A novel prefetch-safe trace reduction algorithm called FastSlim made it possible to evaluate virtual memory caching and prefetching schemes efficiently using trace-driven simulation [38].

After considering various approaches to cluster storage [15, 8], we developed an ensemble storage service architecture called Slice [6, 7, 5]. Slice explored use of content-based request routing to distribute file service traffic across a dynamic ensemble of servers and network-attached block storage devices, which act together as a unified "virtual file storage appliance". Slice benefits from a dynamic mapping of storage objects to servers to balance the load across the ensemble, without imposing new burdens on users or administrators. Slice was the dissertation research of doctoral student Darrell Anderson.

End-system networking and network storage. The Trapeze project—an outgrowth of our work with network memory—investigated network interface (NIC) device techniques and OS structures to harness the performance potential of emerging high-speed networks. We developed a new firmware program for programmable Myrinet NICs, and OS kernel software. Trapeze established new performance standards for page transfer latency and delivered network bandwidth in clusters. It served as a basis for experimentation with novel NIC techniques and related OS structures [65, 62, 4, 8, 15, 64]. We also used Trapeze to study the effects of NIC and OS features for TCP/IP performance on several platforms [32, 17, 13]. The Trapeze research improved understanding of the role of advanced network elements in high-performance services, and was useful to industry in designing systems for high-speed Internet networking, particularly in the network storage arena. This research also led to a brief mention in the New York Times, and a licensed patent for self-tuning NIC features to balance transfer latency and bandwidth.

At that time there was a resurgence of industry interest in direct-access NICs supporting "user-level" networking and features for Remote Direct Memory Access (RDMA), primarily for high-performance network storage in data centers. In a partnership with Harvard and Network Appliance, we developed a reference implementation of a proposed Direct Access File System (DAFS) standard, and investigated operating system structures and application-controlled I/O caching in this context [42]. I also played a role early in an Internet (IETF) standards process to promote understanding of RDMA and its interaction with Internet transport protocols and other approaches to low-overhead networking. More recently, we established analytical bounds on the potential benefit from RDMA and other low-overhead networking schemes as a function of key technology-independent ratios [57].

Web content delivery. My work on techniques for scalable Internet content caching and distribution led to an improved understanding of content-sharing protocols and the scale and performance of Internet information sharing. My early work with Misha Rabinovich on the CRISP project showed that the multicast probes then commonly used in distributed proxy caches were unscalable and increased miss costs. We proposed a directory-based approach to allow content sharing across a group of caching sites operating as a unified Internet object cache. Our research explored a continuum of directory management schemes suitable for collective caches of varying size and geographic scale [28, 29, 31, 49], yielding one patent. Vicinity Cache [49] showed how to build highly scalable caches in which replicated directory information is propagated by gossip and degrades with distance. My subsequent work in this area explored the implications of heavy-tailed popularity distributions for "supply-side" content delivery networks [30] and for request distribution policies in Web server clusters [24].

Data-intensive computing. Trapeze was also the initial basis for productive interdisciplinary collaborations with algorithms researchers (Arge, Vitter, and Agarwal) working on massive-data algorithms and applications, primarily in spatial data domains such as Geographic Information Systems (GIS). These collaborations led to two large interdisciplinary NSF grants of which I was the Principal Investigator, combining experimental systems and algorithm engineering. In addition, I was co-PI (with Vitter) on an NSF ITR grant for research on active storage systems and algorithms. Some papers related to these grants include [42, 9, 61, 2, 1, 3].

Executable code rewriting. Amber was an early system to use automatic executable code rewriting to "glue" applications to a system infrastructure for distributed data sharing. Following this theme, we developed the JOIE rewriting toolkit [21] for Java bytecode. JOIE was used in a number of research projects at Duke and elsewhere.

References

[1] S. Anastasiadis, S. Gadde, and J. S. Chase. Scale and performance in semantic storage management of data grids. *International Journal on Digital Libraries*, 5, April 2005. Special Issue on Semantic Web and Science Data Interoperation.

[2] S. Anastasiadis, R. Wickremisinghe, and J. Chase. Circus: Opportunistic block reordering for scalable content servers. In *Proceedings of the 3rd USENIX Conference on File and Storage Technologies (FAST 04)*, March 2004.

[3] S. Anastasiadis, R. Wickremisinghe, and J. Chase. Lerna: An active storage framework for flexible data acces and management. In *Proceedings of the Fourteenth International Symposium on High Performance Distributed Computing (HPDC-14)*, July 2005.

[4] D. Anderson, J. S. Chase, S. Gadde, A. J. Gallatin, K. G. Yocum, and M. J. Feeley. Cheating the I/O bottleneck: Network storage with Trapeze/Myrinet. In *1998 Usenix Technical Conference*, June 1998.

[5] D. C. Anderson and J. S. Chase. Failure-atomic file access in an interposed network storage system. In *Proceedings of the Ninth IEEE International Symposium on High Performance Distributed Computing (HPDC)*, August 2000. Extended version appears in Cluster Computing.

[6] D. C. Anderson, J. S. Chase, and A. M. Vahdat. Interposed request routing for scalable network storage. In *Proceedings of the Fourth Symposium on Operating System Design and Implementation (OSDI)*, October 2000. Extended version appears in ACM Transactions on Computer Systems (TOCS).

[7] D. C. Anderson, J. S. Chase, and A. M. Vahdat. Interposed request routing for scalable network storage. *ACM Transactions on Computer Systems (TOCS) special issue: selected papers from the Fourth Symposium on Operating System Design and Implementation (OSDI), October 2000*, 20(1), February 2002.

[8] D. C. Anderson, J. S. Chase, and K. G. Yocum. A case for buffer servers. In *Proceedings of the Workshop on Hot Topics in Operating Systems (HOTOS)*, April 1999.

[9] L. Arge, J. S. Chase, L. Toma, J. S. Vitter, R. Wickremesinghe, P. N. Halpin, and D. Urban. Flow computation on massive grids. In *ACM-GIS, ACM Symposium on Advances in Geographic Information Systems*, November 2001.

[10] R. Braynard, D. Kostić, A. Rodriguez, J. Chase, and A. Vahdat. Opus: an Overlay Peer Utility Service. In *Proceedings of the 5th International Conference on Open Architectures and Network Programming (OPENARCH)*, June 2002.

[11] J. Chase, H. Levy, E. Lazowska, and M. Baker-Harvey. Lightweight shared objects in a 64-bit operating system. In *Proc. of the Conference on Object-Oriented Programming Systems, Languages, and Applications*, Oct. 1992.

[12] J. S. Chase. Server switching: Yesterday and tomorrow. In *Proceedings of the Second IEEE Workshop on Internet Applications (WIAPP '01)*, July 2001.

[13] J. S. Chase. *High Performance TCP/IP Networking (Mahbub Hassan and Raj Jain Editors)*, chapter Software Implementation of TCP. Prentice-Hall, 2003.

[14] J. S. Chase, F. G. Amador, E. D. Lazowska, H. M. Levy, and R. J. Littlefield. The Amber system: Parallel programming on a network of multiprocessors. In *Proceedings of the Twelfth ACM Symposium on Operating System Principles (SOSP)*, pages 147–158, December 1989.

[15] J. S. Chase, D. C. Anderson, A. J. Gallatin, A. R. Lebeck, and K. G. Yocum. Network I/O with Trapeze. In *1999 Hot Interconnects Symposium*, August 1999. Invited paper.

[16] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle. Managing Energy and Server Resources in Hosting Centers. In *Proceedings of the 18th ACM Symposium on Operating System Principles (SOSP)*, pages 103–116, October 2001.

[17] J. S. Chase, A. J. Gallatin, and K. G. Yocum. End system optimizations for high-speed TCP. *IEEE Communications, Special Issue on TCP Performance in Future Networking Environments*, 39(4):68–74, April 2001.

[18] J. S. Chase, D. E. Irwin, L. E. Grit, J. D. Moore, and S. E. Sprenkle. Dynamic Virtual Clusters in a Grid Site Manager. In *Proceedings of the Twelfth International Symposium on High Performance Distributed Computing (HPDC)*, June 2003.

[19] J. S. Chase, H. M. Levy, M. J. Feeley, and E. D. Lazowska. Sharing and protection in a single address space operating system. *ACM Transactions on Computer Systems (TOCS)*, 12(4), Nov. 1994.

[20] J. S. Chase, A. Vahdat, and J. Wilkes. Back to the future: dependable computing equals dependable services. In *Proceedings of the 2002 European SIGOPS Workshop*, September 2002.

[21] G. A. Cohen, J. S. Chase, and D. Kaminsky. Automatic program transformation with JOIE. In *1998 Usenix Technical Conference*, June 1998.

[22] I. Cohen, M. Goldszmidt, T. Kelly, J. Symons, and J. Chase. Correlating instrumentation to system states: A building block for automated diagnosis and control. In *Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI)*, December 2004.

[23] R. P. Doyle, O. Asad, W. Jin, J. S. Chase, and A. Vahdat. Model-based resource provisioning in a Web service utility. In *Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems (USITS)*, March 2003.

[24] R. P. Doyle, J. S. Chase, S. Gadde, and A. M. Vahdat. The trickle-down effect: Web caching and server request distribution. *Computer Communications: Selected Papers from the Sixth International Workshop on Web Caching and Content Delivery (WCW)*, 25(4):345–356, March 2002.

[25] M. J. Feeley, B. N. Bershad, J. S. Chase, and H. M. Levy. Dynamic node reconfiguration in a parallel-distributed environment. In *Proceedings of the 1991 ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, April 1991.

[26] M. J. Feeley, J. S. Chase, V. R. Narazayya, and H. M. Levy. Integrating coherency and recoverability in distributed systems. In *Proceedings of the First Symposium on Operating System Design and Implementation*, pages 215–227, Nov. 1994.

[27] Y. Fu, J. Chase, B. Chun, S. Schwab, and A. Vahdat. SHARP: An Architecture for Secure Resource Peering. In *Proceedings of the 19th ACM Symposium on Operating System Principles*, October 2003.

[28] S. Gadde, J. Chase, and M. Rabinovich. Directory structures for scalable Internet caches. Technical Report CS-1997-18, Department of Computer Science, Duke University, November 1997.

[29] S. Gadde, J. Chase, and M. Rabinovich. A taste of Crispy Squid. In *Proceedings of the 1998 Workshop on Internet Server Performance*, pages 129–136, June 1998.

[30] S. Gadde, J. S. Chase, and M. Rabinovich. Web caching and content distribution: A view from the interior. *Computer Communications (Selected Papers from the Fifth International WWW Caching and Content Delivery Workshop)*, 24(2):222–231, February 2001.

[31] S. Gadde, M. Rabinovich, and J. Chase. Reduce, Reuse, Recycle: An approach to building large Internet caches. In *Proceedings of The Sixth Workshop on Hot Topics in Operating Systems (HOTOS-VI)*, pages 93–98, May 1997.

[32] A. J. Gallatin, J. S. Chase, and K. G. Yocum. Trapeze/IP:TCP/IP at near-gigabit speeds. In *1999 USENIX Technical Conference (Freenix track)*, June 1999.

[33] L. Grit, D. Irwin, A. Yumerefendi, and J. Chase. Virtual Machine Hosting for Networked Clusters: Building the Foundations for "Autonomic" Orchestration. In *Proceedings of the First International Workshop on Virtualization Technology in Distributed Computing (VTDC)*, November 2006.

[34] D. Irwin, J. Chase, L. Grit, and A. Yumerefendi. Self-Recharging Virtual Currency. In *Proceedings of the Third Workshop on Economics of Peer-to-Peer Systems (P2P-ECON)*, August 2005.

[35] D. Irwin, J. S. Chase, L. Grit, A. Yumerefendi, D. Becker, and K. G. Yocum. Sharing Networked Resources with Brokered Leases. In *Proceedings of the USENIX Technical Conference*, June 2006.

[36] D. Irwin, L. Grit, and J. Chase. Balancing risk and reward in a market-based task service. In *Proceedings of the Thirteenth International Symposium on High Performance Distributed Computing (HPDC-13)*, June 2004.

[37] W. Jin, J. S. Chase, and J. Kaur. Interposed proportional sharing for a storage service utility. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (ACM SIGMETRICS/Performance)*, June 2004.

[38] W. Jin, X. Sun, and J. S. Chase. FastSlim: Prefetch-safe trace reduction for I/O system simulation. *ACM Transactions on Modeling and Computer Simulation TOMACS*, 11(2), April 2001.

[39] M. Karlsson, C. Karamanolis, and J. Chase. Controllable fair queuing for meeting performance goals. In *Performance 2005*. IFIP/Elsevier, October 2005.

[40] K. Keeton, C. Santos, D. Beyer, J. Chase, and J. Wilkes. Designing for disasters. In *Proceedings of the 3rd USENIX Conference on File and Storage Technologies (FAST 04)*, March 2004.

[41] E. J. Koldinger, J. S. Chase, and S. J. Eggers. Architectural support for single address space operating systems. In *Proceedings of the Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, October 1992.

[42] K. Magoutis, S. Addetia, A. Fedorova, M. Seltzer, J. Chase, R. Kisley, A. Gallatin, R. Wickremisinghe, and E. Gabber. Structure and performance of the Direct Access File System. In *USENIX Technical Conference*, pages 1–14, June 2002.

[43] J. Moore, J. Chase, K. Farkas, and P. Ranganathan. A Sense of Place: Toward a Location-aware Information Plane for Data Centers. In *Hewlett Packard Technical Report TR2004-27*, 2004.

[44] J. Moore, J. Chase, K. Farkas, and P. Ranganathan. Data Center Workload Monitoring, Analysis, and Emulation. In *Eighth Workshop on Computer Architecture Evaluation using Commercial Workloads*, February 2005.

[45] J. Moore, J. Chase, and P. Ranganathan. ConSil: Low-cost Thermal Mapping of Data Centers. In *First Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML)*, June 2006.

[46] J. Moore, J. Chase, and P. Ranganathan. Weatherman: Automated, Online, and Predictive Thermal Mapping and Management for Data Centers. In *Proceedings of the 2006 International Conference on Autonomic Computing (ICAC06)*, June 2006.

[47] J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Making Scheduling "Cool": Temperature-Aware Workload Placement in Data Centers. In *Proceedings of the 2005 USENIX Annual Technical Conference*, pages 61–74, April 2005.

[48] J. Moore, D. Irwin, L. Grit, S. Sprenkle, and J. Chase. Managing mixed-use clusters with Cluster-on-Demand. Technical report, Duke University, Department of Computer Science, November 2002.

[49] M. Rabinovich, J. S. Chase, and S. Gadde. Not all hits are created equal: Cooperative proxy caching over a wide-area network. *Computer Networks and ISDN Systems (Selected Papers from the Third International WWW Caching Workshop)*, 30(22):2253–2260, November 1998.

[50] L. Ramakrishnan, L. Grit, A. Iamnitchi, D. Irwin, A. Yumerefendi, and J. Chase. Toward a Doctrine of Containment: Grid Hosting with Adaptive Resource Control. In *Supercomputing (SC06)*, November 2006.

[51] P. Ranganathan, D. Irwin, and J. Chase. Enclosure-level power management for dense blade servers. In *Proceedings of the International Symposium on Computer Architecture (ISCA)*, June 2006.

[52] M. Ripeanu, M. Bowman, J. Chase, I. Foster, and M. Milenkovic. Globus and PlanetLab Resource Management Solutions Compared. In *Proceedings of the Thirteenth International Symposium on High Performance Distributed Computing (HPDC)*, June 2004.

[53] R. K. Sharma, C. L. Bash, C. D. Patel, R. J. Friedrich, and J. S. Chase. Balance of Power: Dynamic Thermal Management for Internet Data Centers. *IEEE Internet Computing*, 9(1):42–49, January 2005.

[54] P. Shivam, S. Babu, and J. Chase. Active and Accelerated Learning of Cost Models for Optimizing Scientific Applications. In *International Conference on Very Large Data Bases (VLDB)*, September 2006.

[55] P. Shivam, S. Babu, and J. S. Chase. Active sampling for accelerated learning of performance models. In *First Workshop on Tackling Computer Systems Problems with Machine Learning (SysML)*, June 2006.

[56] P. Shivam, S. Babu, and J. S. Chase. Learning performance models in network utilities. In *IEEE International Conference on Autonomic Computing (ICAC)*, June 2006.

[57] P. Shivam and J. S. Chase. On the elusive benefits of protocol offload. In *Proceedings of the SIGCOMM Workshop on Network-I/O Convergence: Experience, Lessons, Implications (NICELI)*, August 2003.

[58] P. Shivam, A. Iamnitchi, A. R. Yumerefendi, and J. S. Chase. Model-Driven Placement of Compute Tasks and Data in a Networked Utility. In *2nd IEEE International Conference on Autonomic Computing (ICAC)*, June 2005. Poster.

[59] A. Vahdat, J. S. Chase, R. Braynard, D. Kostic, P. Reynolds, and A. Rodriguez. Self-organizing subsets: From each according to his abilities, to each according to his needs. In *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS)*, March 2002.

[60] G. M. Voelker, E. J. Anderson, T. Kimbrel, M. J. Feeley, J. S. Chase, A. R. Karlin, and H. M. Levy. Implementing cooperative prefetching and caching in a globally managed memory system. In *Proceedings of the ACM Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '98)*, June 1998.

[61] R. Wickremisinghe, J. S. Chase, and J. S. Vitter. Distributed computing with load-managed active storage. In *The Eleventh IEEE International Symposium on High-Performance Distributed Computing (HPDC-11)*, July 2002.

[62] K. G. Yocum, D. C. Anderson, J. S. Chase, S. Gadde, A. J. Gallatin, and A. R. Lebeck. Adaptive message pipelining for network memory and network storage. Technical Report CS-1998-10, Duke University Department of Computer Science, April 1998.

[63] K. G. Yocum, D. C. Anderson, J. S. Chase, and A. Vahdat. Anypoint: Extensible transport switching on the edge. In *Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems (USITS)*, March 2003.

[64] K. G. Yocum and J. S. Chase. Payload caching: High-speed data forwarding for network intermediaries. In *Proceedings of the USENIX Technical Conference*, June 2001.

[65] K. G. Yocum, J. S. Chase, A. J. Gallatin, and A. R. Lebeck. Cut-through delivery in Trapeze: An exercise in low-latency messaging. In *Sixth IEEE International Symposium on High Performance Distributed Computing (HPDC-6)*, pages 243–252, August 1997.

[66] A. Yumerefendi and J. Chase. The Role of Accountability in Dependable Distributed Systems. In *Proceedings of the First Workshop on Hot Topics in System Dependability*, June 2005.

[67] A. Yumerefendi and J. Chase. Strong Accountability for Network Storage. In *Proceedings of FAST*, February 2007.

[68] A. Yumerefendi and J. S. Chase. Trust but Verify: Accountability for Network Services. In *Proceedings of the 11th ACM SIGOPS European Workshop*, September 2004.

[69] A. R. Yumerefendi and J. S. Chase. CATS: Certified Authenticated Tamper-evident State Store for Network Services. Technical Report CS-2005-5, Computer Science Department, Duke University, January 2005.