

Robust Large-Scale Distributed Systems

August 15, 2008

Mike Dahlin

Department of Computer Sciences
University of Texas at Austin

My research has focused on constructing robust large-scale distributed systems. The bulk of this work can be understood in the context of two intertwined efforts: constructing *cooperative and peer-to-peer services* and understanding the *fundamental principles of large-scale data replication*.

Cooperative and peer-to-peer services

Cooperative and peer-to-peer services both seek to provide a way to scale services beyond what can be provided by even a high-end server machine. Cooperative services do this by treating a service as a parallel program and then running the program across a cluster of machines. Key challenges include rearchitecting services not only to provide good performance by balancing parallelism and locality but also to ensure good reliability and simple management. Peer-to-peer services go further and enlist machines controlled by different users to collectively provide a service to each other. In addition to the problems of cooperative services, peer-to-peer services have to cope with new issues of trust that arise when a service runs across machines spanning multiple administrative domains with limited trust or competing interests.

Some highlights of this stream of work include

- *Serverless file systems and cooperative caching.* We constructed xFS [ADN⁺96] to explore an extreme point in the design space of constructing file systems as parallel programs: xFS's goals included not just providing high throughput for large files but also providing low-latency access to small files, strong consistency for concurrent access across machines, efficient support for both large and small files, and high reliability by efficiently storing redundant data.

xFS was one of the first examples of the cluster file systems that are now becoming a mainstream approach to file system scalability, and a number of ideas explored by xFS remain important today. For example, one of xFS's central architectural principles, separating data-location metadata from the data itself to allow any data to be stored on any disk, lies at the core of Google's cluster file system [GGL03] and its open-source instantiation HDFS. Another of xFS's key ideas, cooperative caching [DMW⁺94, DWAP94], continues to be an active area of research and development across a wide range of domains including not just cluster file systems [IMO⁺04] but also network attached storage [SMV⁺05], web caching [RLI05], grid file systems [AFM05], mobile data access [YC06], WAN multimedia streaming [NT05], database caching [GMM⁺08], and even hardware caching [CS06].

- *WebOS: Composing WAN services.* The WebOS project [VEY⁺97] explored ways to compose services spanning wide area networks. This effort included measurement studies that sought to characterize the availability properties of the Internet [CDGN01, Cha01], and it also included prototyping efforts targeted at specific design challenges: programming models for composing services [VDAA99], security for applications run by multiple users across resources controlled by multiple owners [BVAD98], and resource management for systems that run untrusted code [CDG⁺01].

One key outcome of this work was the Beyond Hierarchies' web caching system [TDVK98, TDVK99, KD99], which, in 1998 was one of the first systems to be constructed using what have come to be called Distributed Hash Tables (DHTs). This system grew out of our efforts to extend cooperative

services from tightly coupled clusters to wide area networks that must account for network proximity. Although hierarchies provided a way to do that [DMW⁺94], traditional hierarchies introduce performance and reliability bottlenecks near the root of the tree. The Beyond Hierarchies system used PRR trees [PRR97] to construct a scalable, self-healing, distributed directory to track the location of cached objects and to direct read requests to a nearby copy.

- *SDIMS: Scalable cooperative monitoring.* As systems grow to span large numbers of nodes, a common challenge is monitoring, querying, and reacting to changes in distributed state. The goal of the SDIMS project was to define an abstraction for scalable monitoring that could serve as a basic building block for a broad range of applications such as systems management, service placement, data sharing and caching, sensor monitoring, multicast tree formation, and grid scheduling. Our 2004 SIGCOMM [YD04] and 2007 VLDB [JKM⁺07] papers dealt with the problem of making DHT-based monitoring self-tuning so that it could serve as a common building block for both read-dominated and write-dominated workloads and for both systems that required high precision or those that could give up precision to reduce their bandwidth demands.

More recently, our 2008 OSDI paper [JKM⁺08] addresses a problem common to a broad range of hierarchical aggregation systems and sensor networks: safeguarding accuracy despite network failures. The key idea is that the output of a monitoring overlay should not only report answers to queries, but it should also include information about its own stability so that users and applications can assess whether the reported answer is likely to reflect the true state of the system.

- *Byzantine Fault Tolerance and Beyond.* A longstanding vision of distributed systems research has been to construct reliable systems from unreliable components. An enticing formulation of that vision is Byzantine fault tolerant (BFT) replication, in which a collection of machines cooperate to correctly provide a service even if some of the machines misbehave or malfunction in arbitrary (“Byzantine”) ways [LSP82, CL99]. Our work to construct systems that are robust to a broad range of failure modes has yielded two main results.

First, in cooperative environments where a single administrative entity controls all nodes and where it may be reasonable to assume that non-faulty nodes follow a specified protocol, we have driven down the costs and driven up the performance of BFT replication [KD04, MAD02b, MAD02a, YMV⁺03, KAD⁺07b] to the point where we speculate that commercially-viable deployments of Byzantine services can now be constructed for an envelope containing a significant range of interesting services [CMW⁺08]. On the theoretical side, many of our systems have succeeded in driving costs near their theoretical minima [MAD02a, KAD⁺07b, YMV⁺03], in some cases side-stepping and beating what had been viewed as lower bounds [YMV⁺03, MAD02a].

Second, as we consider peer-to-peer environments spanning multiple administrative domains (“MAD systems”), BFT’s dichotomy of correct and faulty nodes becomes less tenable: nodes may deviate from a specified protocol not just because they are “broken” (i.e., due to hardware failure, software bug, misconfiguration, or security compromise) but because they are selfish and alter the protocol to increase their gain or reduce their cost. BFT techniques handle the first class of deviations well, but BFT protocols typically require a bound on the number of faults in the system; assuming such a bound seems dangerous in MAD systems where *all* nodes may benefit from selfish behavior and be motivated to deviate from the protocol. Conversely, economic models that only account for selfish behavior can handle the second class of deviations, but they may be vulnerable to arbitrary disruptions if even a single node is broken and deviates from the expected rational behavior.

To allow construction of services that tolerate both a limited number of Byzantine faults and an unlimited number of rational nodes, we introduced the BAR (Byzantine Altruistic Rational) model [AAC⁺05,

CLN⁺08, CNL⁺07]. In order to demonstrate the utility of this model, we have constructed several demanding BAR services including a peer-to-peer backup system [AAC⁺05] and two peer-to-peer streaming services [LCW⁺06, LCM⁺08]. We are particularly enthusiastic about the robustness exhibited by our most recent streaming service, and we are pushing hard to refine our prototype to make it available for widespread use.

Looking forward, as outsourcing of computer services becomes more widespread, organizations are faced with a middle ground between cooperative services (where all nodes are under the control of the organization) and the wild west of peer-to-peer/BAR services: a service provider may be reputable and contractually bound to attempt to deliver a service correctly, yet the service provider is a black box to the service consumer, so the service consumer may wish to limit its vulnerability to security breaches, network outages, operational blunders, or other failures of the service provider. Our SafeStore system [KAD07a] draws on our BFT and BAR work to explore these new trade-offs in the context of outsourced storage service providers (SSPs). SafeStore uses a “trust but verify” approach: it does not interfere with the policies used within each SSP to maintain data integrity, but it provides an *audit* interface so that the data owner retains end-to-end control over data integrity. This audit protocol tolerates Byzantine faults and it encourages compliance by rational SSPs by producing a cryptographic proof of misbehavior (POM) that serves as a basis for contractual or legal action if an SSP attempts to deceive the audit protocol.

Large-scale data replication

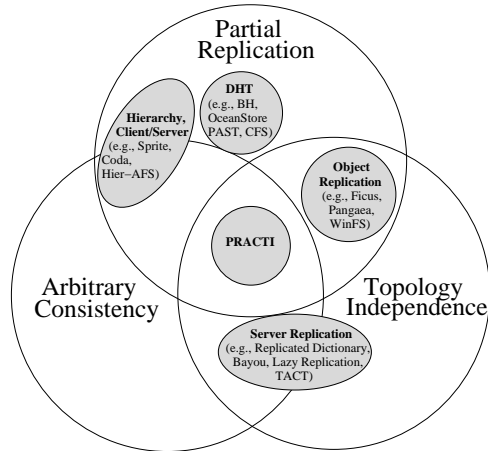
Data replication is a fundamental building block for large-scale services where a key question is how to manage distributed state, and at the core of data replication lie a set of fundamentally hard trade-offs: balancing consistency versus availability and partition-resilience [GL02], balancing consistency versus performance [LS88], balancing resource consumption versus performance, and so on. The second main stream of our work has focused on understanding and improving these fundamental trade-offs so that we can construct better replication systems.

This work began with our efforts on serverless file systems and cooperative caching discussed above. Our experience there led us in several new directions.

- *Scalable consistency.* We have explored ways to improve the consistency v. availability v. performance trade-offs faced by large scale systems. For example, although it was widely believed that providing strong consistency was either impossible or fundamentally expensive for large-scale systems, we developed *volume leases* and demonstrated how to scale callback-based sequential or delta consistency to web-scale services [YADL99b, YADL99a, YADI02]. We also demonstrated that although these tradeoffs are fundamentally hard in the general case, *application-specific* optimizations can exploit knowledge of workloads to get near-ideal trade-offs for important services like large-scale data dissemination [NDI04] and web e-commerce servers [GDN⁺05, GDZ⁺05].
- *Self-tuning speculative replication.* Many systems can make use of speculative replication to improve their performance or consistency, but most existing speculative replication or prefetching algorithms carry with them the risk of catastrophic overload during periods of heavy load, which greatly restricts the circumstances under which sensible system designers deploy prefetching. We developed TCP-Nice, a new network protocol for background transfers that provably limits interference with normal, foreground traffic [VKD02b]. Given this primitive, we can re-think the trade-offs between resource management and performance: prefetching algorithms can now be extremely aggressive because they will only consume spare resources that would otherwise be wasted [VKD02a, NDI04, VYK⁺02, KYVD03].

These techniques have had commercial impact, being used, for example, in IBM/Tivoli’s Provisioning Manager product.

- *PRACTI replication*



Our vision is to develop a *unified data replication architecture* for large-scale distributed services. This architecture would represent a single set of mechanisms on which a broad range of data replication systems can be built for wide area, enterprise, grid, web, and mobile data systems. The core hypothesis is that many existing systems are “special cases” of a more fundamental underlying protocol but that these systems are superficially different because they embed specific policy assumptions in their implementation mechanisms.

The figure above summarizes the state of the art prior to this effort and illustrates a key contribution of the work. We have defined a taxonomy for understanding replication systems based on three vital properties: *partial replication* (PR) means that any data can be replicated on any device (i.e., in order to exploit locality, data placement should be a policy choice); *arbitrary consistency* (AC) means that the system supports a range of consistency guarantees (i.e., applications that need strong guarantees can get them, but applications that require weak guarantees only pay for the guarantees they need); and *topology independence* means that information can flow between any pair of nodes (i.e., the system can use whatever policy it likes for selecting the paths along which information flows in order to adapt to network costs or to reconfigure around node failures.)

Surprisingly, as the figure illustrates for several key families of protocols, existing systems provide at most two of these three desirable PRACTI properties. Conversely, our PRACTI protocol is the first, to our knowledge, to provide all three of these properties.

We believe a PRACTI replication protocol is of both practical and theoretical interest.

On the practical side, we believe that PRACTI represents both a *better way to build replication systems* and a *way to build better replication systems*. We believe PRACTI is a better way to build systems because PRACTI cleanly separates mechanism from policy so that many existing and new replication systems can easily be constructed within a policy layer over a clean set of core mechanisms [BDNZ07]. We believe PRACTI can be a way to build better systems because PRACTI dominates key existing families of architectures by providing order of magnitude performance improvements for some key environments and workloads and matching their performance for most other environments and workloads [BDG⁺06].

More fundamentally, we believe that PRACTI replication is a key step towards a unified replication architecture that can subsume and improve upon existing approaches. A deeper understanding of

replication systems could both simplify teaching the broad design space of replication systems and improve the practice of constructing new replication systems.

References

- [AAC⁺05] A. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J.P. Martin, and C. Porth. BAR fault tolerance for cooperative services. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles*, October 2005.
- [ADN⁺96] T. Anderson, M. Dahlin, J. Neefe, D. Patterson, D. Roselli, and R. Wang. Serverless Network File Systems. *ACM Transactions on Computer Systems*, 14(1):41–79, February 1996.
- [AFM05] S. Annapureddy, M. Freedman, and D. Mazières. Shark: Scaling file servers via cooperative caching. In *Proceedings of the Second USENIX Symposium on Networked Systems Design and Implementation*, May 2005.
- [BDG⁺06] N. Belaramani, M. Dahlin, L. Gao, A. Nayate, A. Venkataramani, P. Yalagandula, and J. Zheng. PRACTI replication. In *Proceedings of the Third USENIX Symposium on Networked Systems Design and Implementation*, May 2006.
- [BDNZ07] N. Belaramani, M. Dahlin, A. Nayate, and J. Zheng. PADRE: A Policy Architecture for building Data REplication systems, October 2007.
- [BVAD98] E. Belani, A. Vahdat, T. Anderson, and M. Dahlin. The CRISIS Wide Area Security Architecture. In *Proceedings of the Seventh USENIX Security Symposium*, January 1998.
- [CDG⁺01] B. Chandra, M. Dahlin, L. Gao, A. Khoja, A. Razzaq, and A. Sewani. Resource management for scalable disconnected access to web services. In *WWW10*, May 2001.
- [CDGN01] B. Chandra, M. Dahlin, L. Gao, and A. Nayate. End-to-end WAN Service Availability. In *Proceedings of the Third USENIX Symposium on Internet Technologies and Systems*, 2001.
- [Cha01] B. Chandra. Web workloads influencing disconnected services access. Master’s thesis, University of Texas at Austin, 2001.
- [CL99] M. Castro and B. Liskov. Practical Byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, February 1999.
- [CLN⁺08] A. Clement, H. Li, J. Napper, JP Martin, L. Alvisi, and M. Dahlin. BAR primer. In *Dependable Systems and Networks*, 2008.
- [CMW⁺08] A. Clement, M. Marchetti, E. Wong, L. Alvisi, and M. Dahlin. BFT: The time is now. In *Large-Scale Distributed Systems and Middleware*, September 2008.
- [CNL⁺07] A. Clement, J. Napper, H. Li, JP Martin, L. Alvisi, and M. Dahlin. Theory of BAR games. In *Brief Announcements: Proceedings of the Symposium on Principles of Distributed Computing (PODC 2007)*, August 2007.
- [CS06] J. Chang and G. Sohi. Cooperative caching for chip multiprocessors. In *Proceedings of the Thirty-Third International Symposium on Computer Architecture*, pages 264–276, New York, NY, USA, 2006. ACM.

- [DMW⁺94] M. Dahlin, C. Mather, R. Wang, T. Anderson, and D. Patterson. A Quantitative Analysis of Cache Policies for Scalable Network File Systems. In *Proceedings of the SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, pages 150–160, May 1994.
- [DWAP94] M. Dahlin, R. Wang, T. Anderson, and D. Patterson. Cooperative Caching: Using Remote Client Memory to Improve File System Performance. In *Proceedings of the First Symposium on Operating Systems Design and Implementation*, pages 267–280, November 1994.
- [GDN⁺05] L. Gao, M. Dahlin, A. Nayate, J. Zheng, and A. Iyengar. Improving availability and performance with application-specific data replication. *IEEE Transactions on Knowledge and Data Engineering*, 17(1), 2005.
- [GDZ⁺05] L. Gao, M. Dahlin, J. Zheng, L. Alvisi, and A. Iyengar. Dual-quorum replication for edge services. In *Proceedings of the ACM/IFIP/USENIX 6th International Middleware Conference*, November 2005.
- [GGL03] S. Ghemawat, H. Gobioff, and S. Leung. The Google file system. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, 2003.
- [GL02] S. Gilbert and N. Lynch. Brewer’s conjecture and the feasibility of Consistent, Available, Partition-tolerant web services. In *ACM SIGACT News*, 33(2), Jun 2002.
- [GMM⁺08] C. Garrod, A. Manjhi, B. Maggs, T. Mowry, A. Tomasic, C. Olston, and A. Ailamaki. Scalable query result caching for web applications. In *VLDB*, August 2008.
- [IMO⁺04] Florin Isaila, Guido Malpohl, Vlad Olaru, Gabor Szeder, and Walter Tichy. Integrating collective i/o and cooperative caching into the ”clusterfile” parallel file system. In *ICS ’04: Proceedings of the 18th annual international conference on Supercomputing*, pages 58–67, New York, NY, USA, 2004. ACM.
- [JKM⁺07] N. Jain, D. Kit, P. Mahajan, P. Yalagandula, M. Dahlin, and Y. Zhang. STAR: Self-tuning aggregation for scalable monitoring. In *International Conference on Very Large Data Bases*, 2007.
- [JKM⁺08] N. Jain, D. Kit, P. Mahajan, P. Yalagandula, M. Dahlin, and Y. Zhang. Network imprecision: A new consistency metric for networked services. In *Proceedings of the Eighth Symposium on Operating Systems Design and Implementation*, December 2008.
- [KAD07a] R. Kotla, L. Alvisi, and M. Dahlin. SafeStore: A durable and practical storage system. In *Proceedings of the 2007 USENIX Annual Technical Conference*, June 2007.
- [KAD⁺07b] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong. Zyzzyva: Speculative Byzantine fault tolerance. In *Proceedings of the 21th ACM Symposium on Operating Systems Principles*, 2007.
- [KD99] M. Korupolu and M. Dahlin. Coordinated Placement and Replacement for Large-Scale Distributed Caches. In *Proceedings of the 1999 IEEE Workshop on Internet Applications*, June 1999.
- [KD04] R. Kotla and M. Dahlin. High-throughput Byzantine fault tolerant services. In *Dependable Systems and Networks*, June 2004.

- [KYVD03] R. Kokku, P. Yalagandula, A. Venkataramani, and M. Dahlin. NPS: A non-interfering deployable web prefetching system. In *Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems*, March 2003.
- [LCM⁺08] H. Li, A. Clement, M. Marchetti, M. Kapritsos, L. Robinson, L. Alvisi, and M. Dahlin. Flight-Path: Obedience vs choice in cooperative services. In *OSDI 2008*, Dec 2008.
- [LCW⁺06] H. Li, A. Clement, E. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin. BAR gossip. In *Proceedings of the Seventh Symposium on Operating Systems Design and Implementation*, 2006.
- [LS88] R. Lipton and J. Sandberg. PRAM: A scalable shared memory. Technical Report CS-TR-180-88, Princeton, 1988.
- [LSP82] L. Lamport, R. Shostack, and M. Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.
- [MAD02a] J.P. Martin, L. Alvisi, and M. Dahlin. Minimal Byzantine quorums. In *16th Symposium on Distributed Computing (DISC)*, October 2002.
- [MAD02b] J.P. Martin, L. Alvisi, and M. Dahlin. Small Byzantine quorum systems. In *Dependable Systems and Networks*, June 2002.
- [NDI04] A. Nayate, M. Dahlin, and A. Iyengar. Transparent information dissemination. In *Proceedings of the ACM/IFIP/USENIX 5th International Middleware Conference*, October 2004.
- [NT05] Jian Ni and D.H.K. Tsang. Large-scale cooperative caching and application-level multicast in multimedia content delivery networks. *IEEE Communications Magazine*, 43(5):98–105, May 2005.
- [PRR97] G. Plaxton, R. Rajaram, and A. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *Proceedings of the Ninth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 311–320, June 1997.
- [RLI05] Lakshmish Ramaswamy, Ling Liu, and Arun Iyengar. Cache clouds: Cooperative caching of dynamic documents in edge networks. *icdc*, 00:229–238, 2005.
- [SMV⁺05] Craig Soules, Arif Merchant, Alistair C. Veitch, Yasushi Saito, and John Wilkes. Method of cooperative caching for distributed storage system. US Patent application US 2006/0174063 A1, February 2005.
- [TDVK98] R. Tewari, M. Dahlin, H. Vin, and J. Kay. Beyond Hierarchies: Design Considerations for Distributed Caching on the Internet. Technical Report TR98-04, University of Texas at Austin Computer Sciences Department, February 1998.
- [TDVK99] R. Tewari, M. Dahlin, H. Vin, and J. Kay. Beyond Hierarchies: Design Considerations for Distributed Caching on the Internet. In *Proceedings of the Nineteenth International Conference on Distributed Computing Systems*, May 1999.
- [VDAA99] A. Vahdat, M. Dahlin, T. Anderson, and A. Aggarwal. Active Naming: Flexible Location and Transport of Wide-Area Resources. In *Proceedings of the Second USENIX Symposium on Internet Technologies and Systems*, October 1999.

- [VEY⁺97] A. Vahdat, P. Eastham, C. Yoshikawa, E. Belani, T. Anderson, D. Culler, and M. Dahlin. WebOS: Operating System Services for Wide Area Applications. Tech Report CSD-97-938, U.C. Berkeley Computer Science Division, March 1997.
- [VKD02a] A. Venkataramani, R. Kokku, and M. Dahlin. Operating system support for massive replication. In *Proceedings of the 10th ACM SIGOPS European Workshop*, September 2002.
- [VKD02b] A. Venkataramani, R. Kokku, and M. Dahlin. TCP Nice: A mechanism for background transfers. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, December 2002.
- [VYK⁺02] A. Venkataramani, P. Yalagandula, R. Kokku, S. Sharif, and M. Dahlin. Potential costs and benefits of long-term prefetching for content-distribution. *Elsevier Computer Communications*, 25(4):367–375, March 2002.
- [YADI02] J. Yin, L. Alvisi, M. Dahlin, and A. Iyengar. Engineering web cache consistency. *ACM Transactions on Internet Technologies*, 2(3), 2002.
- [YADL99a] J. Yin, L. Alvisi, M. Dahlin, and C. Lin. Hierarchical Cache Consistency in a WAN. In *Proceedings of the Second USENIX Symposium on Internet Technologies and Systems*, October 1999.
- [YADL99b] J. Yin, L. Alvisi, M. Dahlin, and C. Lin. Volume Leases to Support Consistency in Large-Scale Systems. *IEEE Transactions on Knowledge and Data Engineering*, February 1999.
- [YC06] L. Yin and G. Cao. Supporting cooperative caching in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(1):77–89, January 2006.
- [YD04] P. Yalagandula and M. Dahlin. A scalable distributed information management system. In *ACM SIGCOMM 2004 Conference*, August 2004.
- [YMV⁺03] J. Yin, J.P. Martin, A. Venkataramani, L. Alvisi, and M. Dahlin. Separating agreement from execution for Byzantine fault tolerant services. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, October 2003.