Self-* Storage

Andrew J. Klosterman, Brandon Salmon, John D. Strunk, Gregory R. Ganger
Carnegie Mellon University

We are exploring the design and implementation of self-$*$ storage systems: self-organizing, self-configuring, self-tuning, self-healing, self-managing systems of storage bricks. Borrowing organizational ideas from corporate structure and automation technologies from AI and control systems, we hope to dramatically reduce the administrative burden faced by data center administrators.

As computer complexity has grown and system costs have shrunk, system administration has become a dominant factor in ownership cost and user dissatisfaction. Storage represents 40–60% of hardware costs in modern data centers, and 60–80% of the total cost of ownership. Storage administration (including capacity planning, backup, and load balancing) is where much of the administrative effort lies; Gartner and others have estimated the need for one administrator per 1-10 terabytes, which is a scary ratio with multi-petabyte data centers on the horizon.

The self-$*$ storage project [1] is exploring ways to automate management of the storage system. The high-level system architecture borrows organizational concepts from corporate structure (Figure 1). Workers are storage bricks that adaptively tune themselves, routers are logical entities that deliver requests to the right workers, and supervisors plan system-wide and orchestrate from out-of-band.

Human administrators will still be needed to provide goals and equipment. The administrative interface must help the administrator specify goals and make tradeoffs related to procurement. The administrator needs to be made aware of tradeoffs (e.g., among performance, reliability, and cost) when they expect performance beyond the point of the current system's capabilities.

Workers service requests for, and store, assigned data. We expect them to have the computation and memory resources needed to internally adapt to their observed workloads by reorganizing on-disk placements and specializing cache policies. Workers also handle storage allocation internally, both to decouple external naming from internal placements and to allow support for internal versioning. Workers keep historical versions of all data for recovery from dataset corruption.

Routers deliver client requests to the appropriate workers. Doing so requires metadata for tracking current storage assignments, consistency protocols for accessing redundant data, and choices of where to route particular requests (notably, READs to replicated data). We do not necessarily envision the routing functionality in hardware routers. It could be software running on each client, software running on each worker, or functionality embedded in interposed nodes.

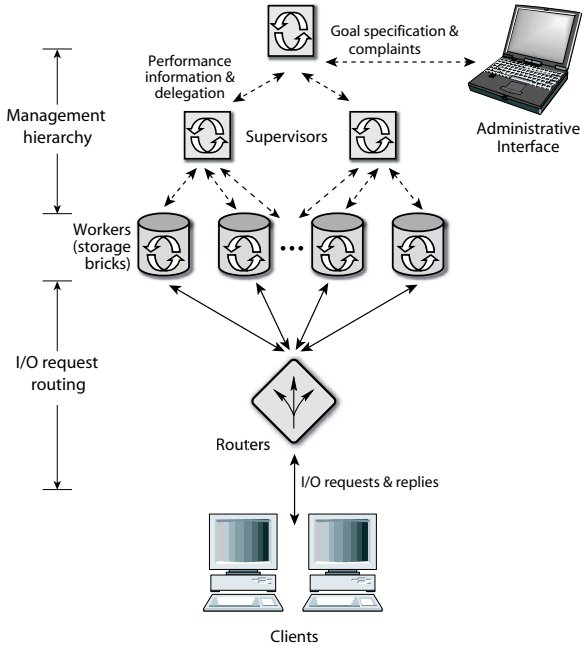

Figure 1: **Architecture of self-$*$ storage.** The top of the diagram is the management hierarchy, concerned with the distribution of goals and the delegation of storage responsibilities from the system administrator down to individual worker devices. The bottom of the diagram depicts the path of I/O requests from clients, through routing nodes, to workers for service. Note that the management infrastructure is logically independent of the I/O request path, and that routing is logically independent of clients and workers.

The supervisor nodes, arranged into a hierarchy, control how data is partitioned among workers. A supervisor's objective is to partition data and goals among its subordinates (workers or lower-level supervisors) such that, if its children meet their assigned goals, the goals for the entire subtree will be met. Prior to partitioning the workload, the supervisor needs to gain some understanding of the capabilities of each of its workers. Of course, this information will be imperfect, resulting in some trial-and-error and observation-based categorization.

To gain practical experience with the management challenges, we are building and deploying a large prototype system. This will allow us to test and refine our ideas in a real environment.

References

[1] G. R. Ganger, J. D. Strunk, and A. J. Klosterman. *Self-* Storage: brick-based storage with automated administration*. Technical report CMU–CS–03–178. Carnegie Mellon, August 2003.