Five Signs You've Outgrown Your Cache + Database Architecture

For years using an external caching layer with an RDBMS was the accepted conventional wisdom for attaining performance and reliability. Data was kept in memory by the cache to reduce the amount of time spent querying the underlying data store. As data volumes grew, NoSQL databases were substituted for the RDBMS to provide horizontal scaling to clusters and to keep latency down. Then "Digital Transformation" happened.

In the blink of an eye, simple web applications have become systems of engagement (SoE) that are now serving billions of objects, with millions of contextual data points and enabling rich interactions and engagement - all in a few milliseconds. Data has grown in both velocity and volume - the more data, delivered faster, the richer the engagement and the better the decision. Datasets from 10's of TB up to PBs are now common.

The only way to support modern velocities and simultaneously deal with growing cache data volumes is to subscribe to a never-ending cycle of adding cache nodes and deploying increasingly more complex cache management systems and strategies. This approach ignores the high cost of external DRAM caching layers. As data volumes expand external caching solutions are un-fundable over time and require a significant investment in managing complex data lifecycle issues such as cache consistency and correctness. Translation: cache is unaffordable, un-trustable, and unstable at high volumes.

What are the five signs you may have outgrown your cache + database architecture?

Sign 1	Your caching nodes are growing uncontrollably
Sign 2	Repopulating your cache after a disruption takes too long
Sign 3	You're missing your SLAs
Sign 4	New architecture may not support new or unexpected growth
Sign 5	Repeated cache stampedes

Sign 1: Your caching nodes are growing uncontrollably

As the value of engagement increases, new applications and projects clamor for database access, increasing transaction volumes and cache working set sizes. Server counts need to keep up, despite what has been budgeted. When growth happens beyond expectations it must be addressed, or your System of Engagement won't be able to keep up.

Sign 2: Repopulating your cache after disruption takes too long

Provisioning a new caching server in the cloud and DevOps era is now a matter of minutes for most companies but that doesn't apply to the data in your caching layer. It has to be "rehydrated" to a level where the hit rate is acceptable before it can reduce database load. For most data-heavy companies this process can take hours or even days, forcing them to deal with limited performance, inaccurate data, the unnecessary cost of even greater over-provisioning, and application complexity.

Sign 3: You're missing your SLAs

It's common knowledge that neither an RDBMS nor a first-generation NoSQL database is ever going to be fast enough on its own to meet sub-millisecond response times, which is why you have to put a cache in front of it. But it's rarely as simple as that, and this architecture does nothing to guarantee meeting SLAs in the face of growth.

Sign 4: New architecture may not support new or unexpected growth

You may already be using sharding inside your application to create additional capacity. However, your strategy has problems that get progressively bigger with scale. Advanced management techniques like sharding and cluster management require costly expert services. Still, even such experts can't eliminate the inherent stability, accuracy and management issues of cache-first architecture.

Sign 5: Repeated cache stampedes

All of the methods of dealing with a cache stampede involve code changes at the application level, which then must be "sold" to development groups so they can incorporate them into their code bases and prevent a recurrence. At the end of the day, none of these methods solve the problem permanently. A cache adds needless complexity and risk to a business.

Get ready for the future with Aerospike

Even though it's still common practice today, the harsh reality is that using an external cache layer as a fundamental component of a System of Engagement (SoE), where huge scale, ultrafast response, and rock-solid reliability are critical success factors, is a legacy approach for solving a next-generation problem. Instead of more memory, or a better cache, perhaps what's needed is a better data architecture such as the unique, highly patented Aerospike Hybrid Memory Architecture™ (HMA).

Aerospike HMA combines DRAM's parallelism and SSD's fast, random bulk storage capability to provide near DRAM performance with SSD reliability and lower cost. A hybrid memory database provides automation to simplify the access and processing of data, and support for transactional, operational and analytical workloads. It leverages SSD and flash natively in an optimized and engineered manner.

Fortunately, the Aerospike database, built with hybrid memory architecture, allows you to eliminate that external caching layer while simultaneously handling internet-scale data volumes at sub-millisecond response times and with significantly fewer servers. It is used in production and trusted by industry-leading organizations to power their Systems of Engagement.