3 performances improvement in your microservices architecture

With Hazelcast In-Memory Data Grid

@nicolas_frankel

Me, myself and I

- Previously developer, team lead, architect, solutions architect
- Developer Advocate
- Pragmatic but curious

Hazelcast

HAZELCAST IMDG is an **operational, in-memory**, distributed computing platform that manages data using in-memory storage, and performs parallel execution for breakthrough application speed and scale.

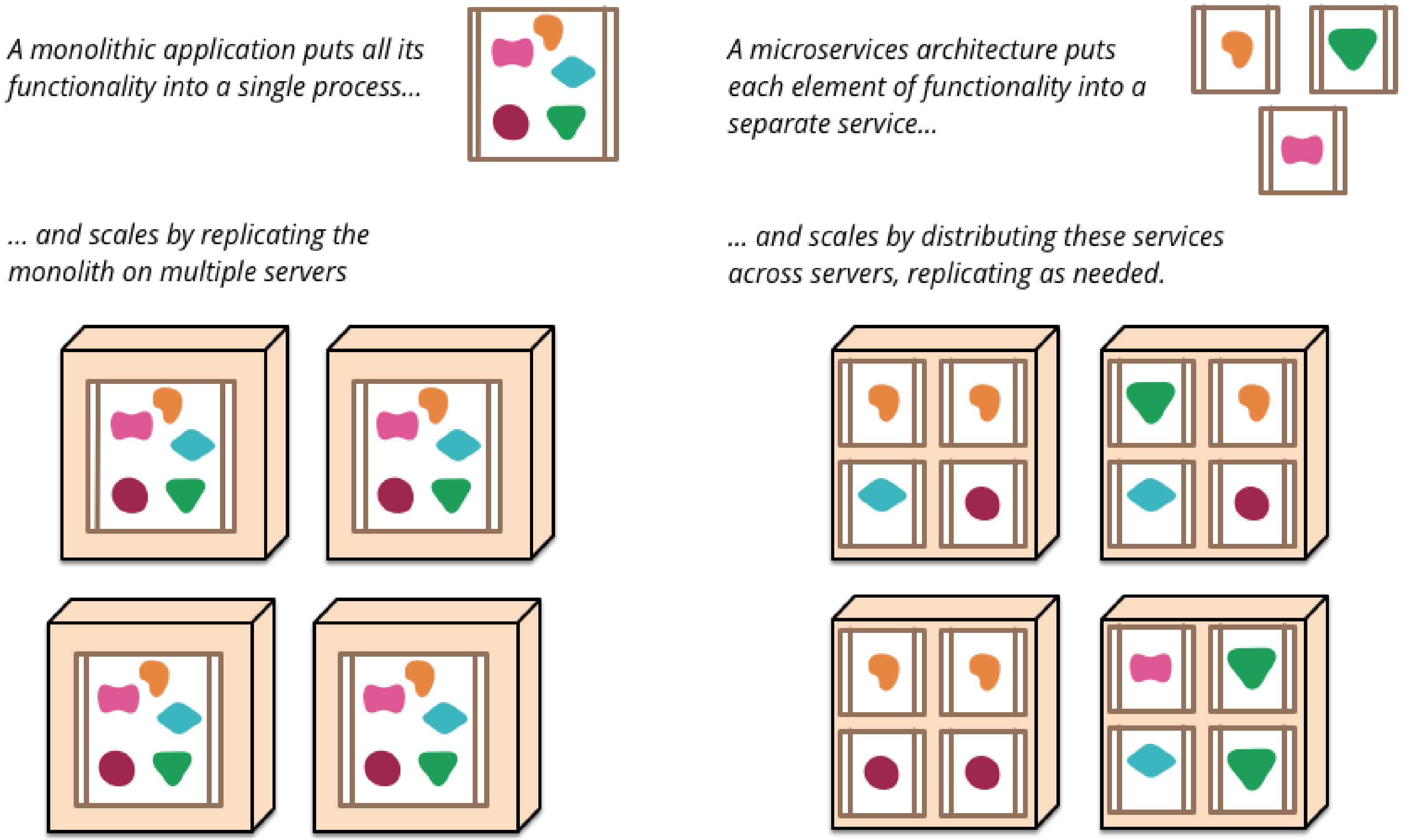
HAZELCAST JET is the ultra fast, application embeddable, 3rd generation stream processing engine for low latency batch and stream processing.

Microservices: a tentative definition

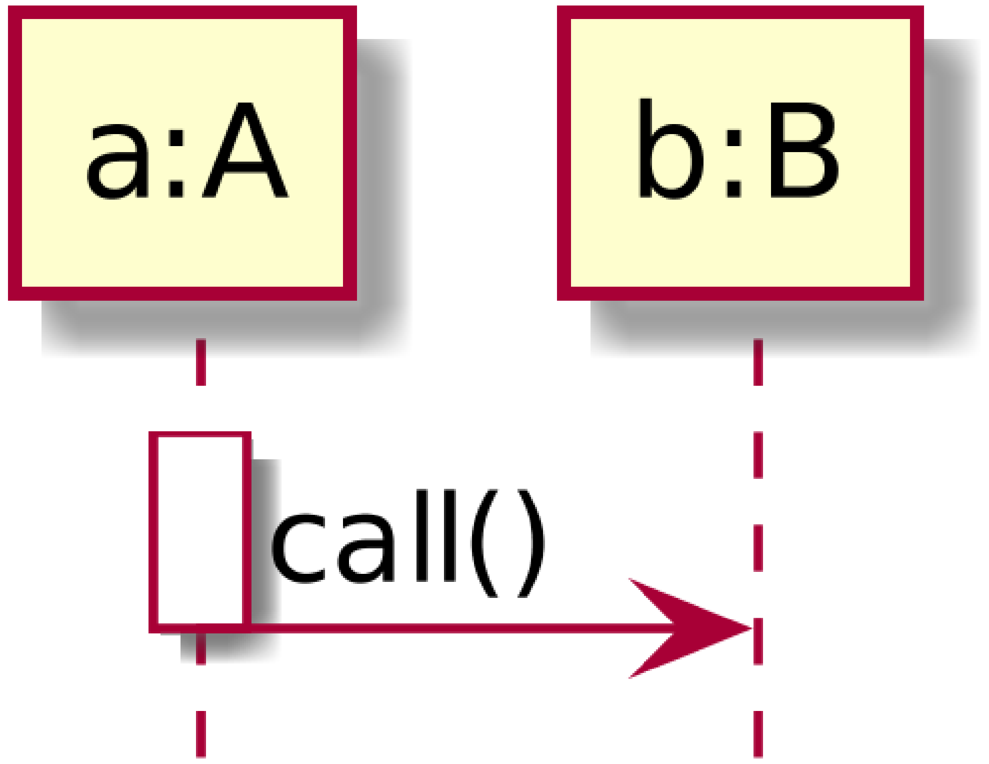
- Componentization via Services
- Smart endpoints and dumb pipes
- Decentralized Governance
- Decentralized Data Management
- Infrastructure Automation
- Design for failure
- Evolutionary Design
- Organized around Business Capabilities
- Products not Projects

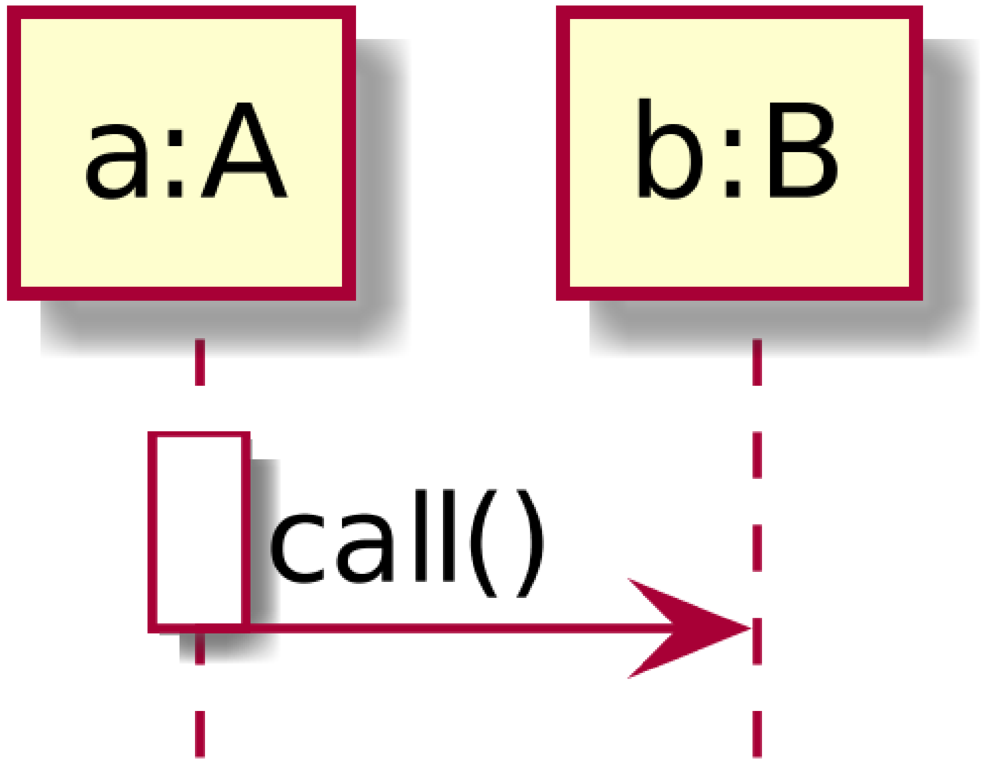
https://martinfowler.com/articles/microservices.html

A benefit: scalability


A monolithic application puts all its functionality into a single process...
... and scales by replicating the monolith on multiple servers
A microservices architecture puts each element of functionality into a separate service...
... and scales by distributing these services across servers, replicating as needed.

Do you spot the difference?





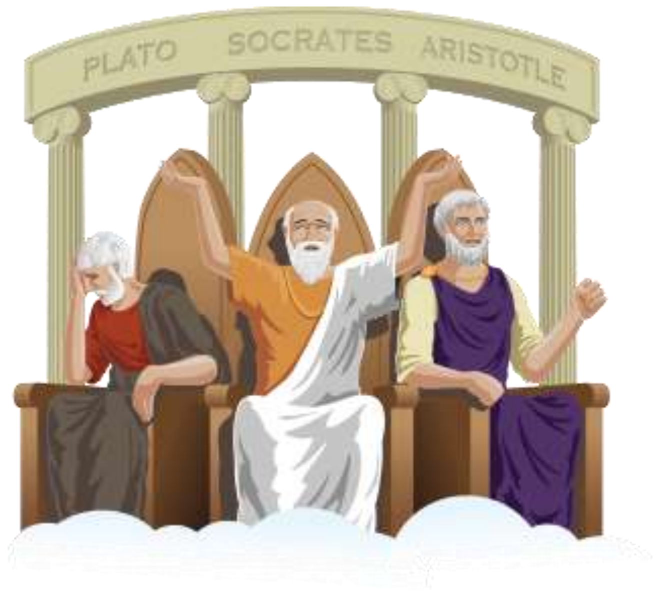
Distributed systems

« You have to be in a really unusual spot to see in-process function calls turn into a performance hot spot these days, but **remote calls are slow**. If your service calls half-a-dozen remote services, each which calls another half-a-dozen remote services, these **response times add up to some horrible latency** characteristics. »

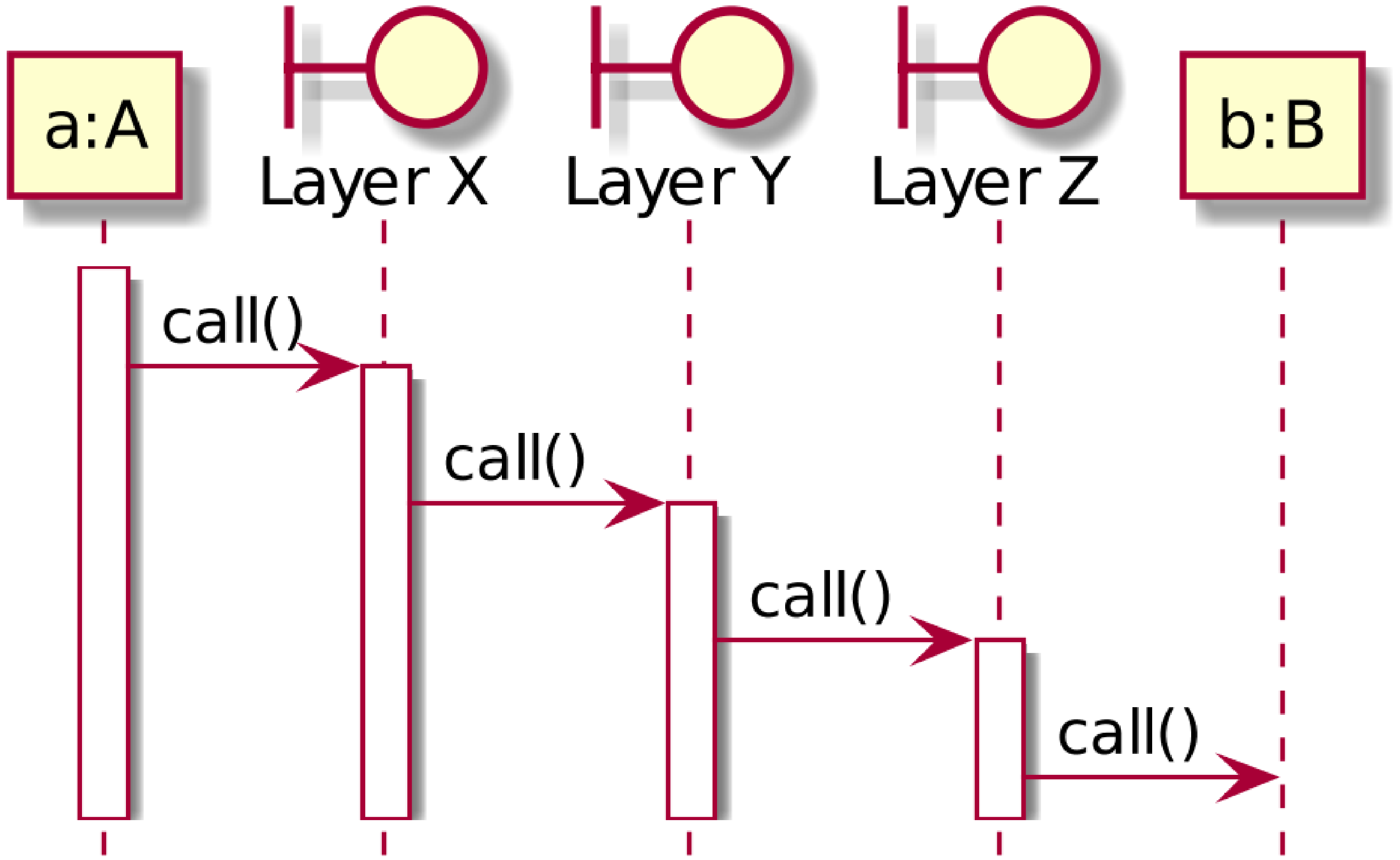
-- https://martinfowler.com/articles/microservice-trade-offs.html

Fallacies of distributed computing

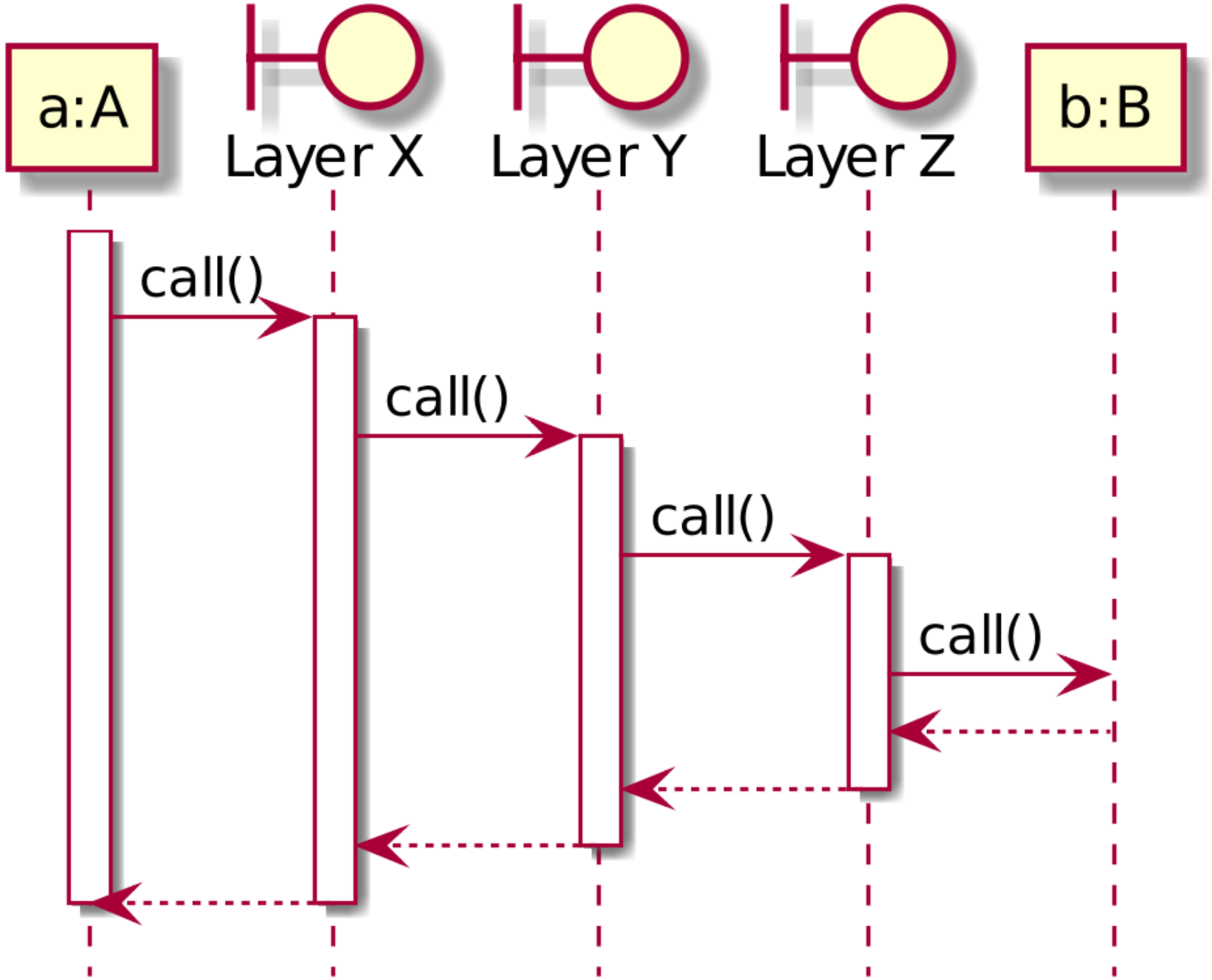
- The network is reliable
- Latency is zero
- Bandwidth is infinite
- The network is secure
- Topology doesn't change
- There is one administrator
- Transport cost is zero
- The network is homogeneous



More like that...

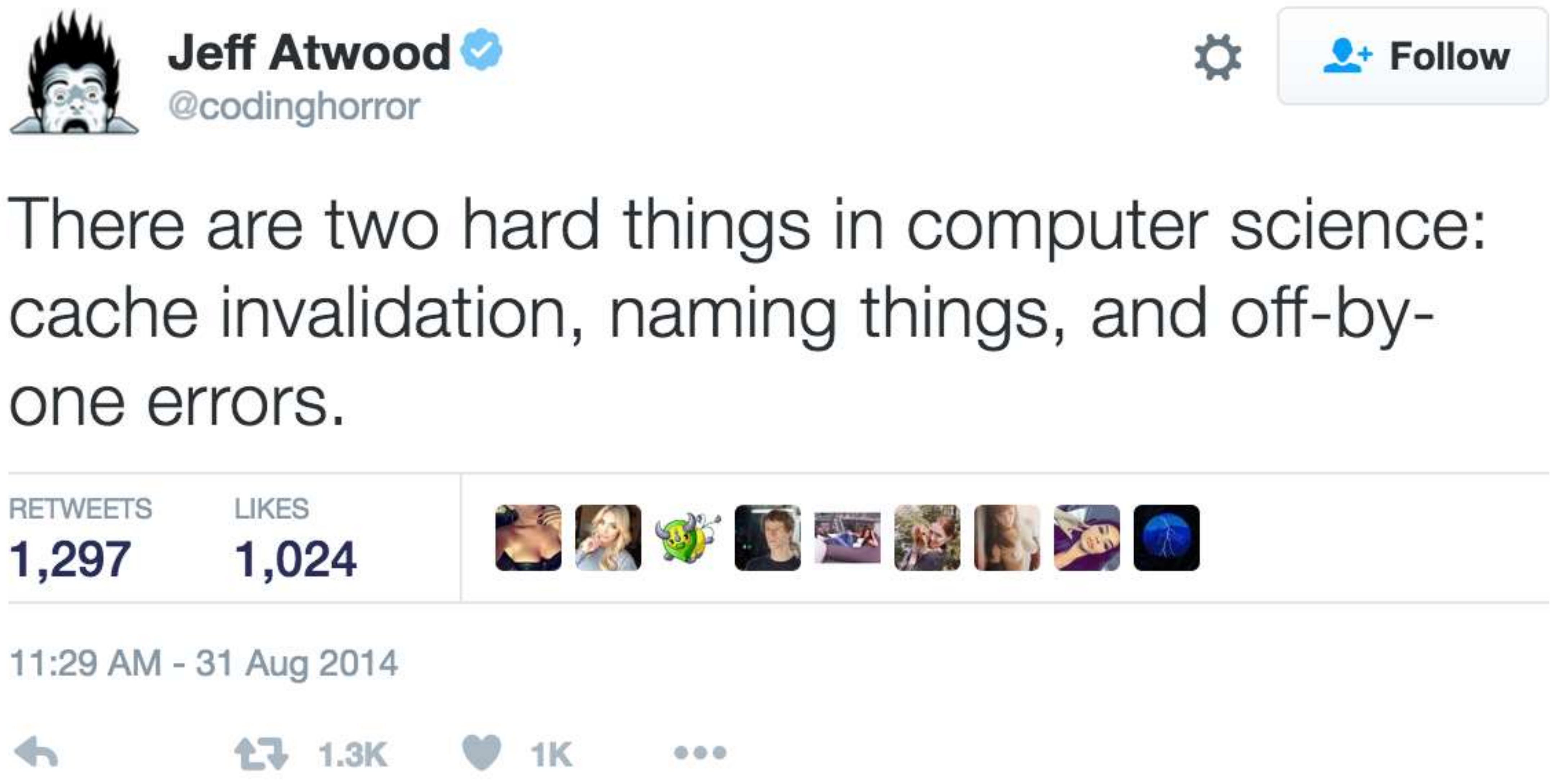


No, like that!



Trade-off

Fast vs. up-to-date



Jeff Atwood
@codinghorror

Follow

There are two hard things in computer science: cache invalidation, naming things, and off-by-one errors.

RETWEETS 1,297 LIKES 1,024

11:29 AM - 31 Aug 2014

1.3K 1K

Caching?

Let’s use a hash map!

- Unbounded
- No eviction strategy
- No TTL
- etc.

In-Memory Data Grid

- A distributed object store
- Think distributed hash map

Caching use-cases in µservices

- Database access
- HTTP call
- Session data

Hibernate

- Object-Relational Mapping framework
- Quite widespread
- JPA implementation

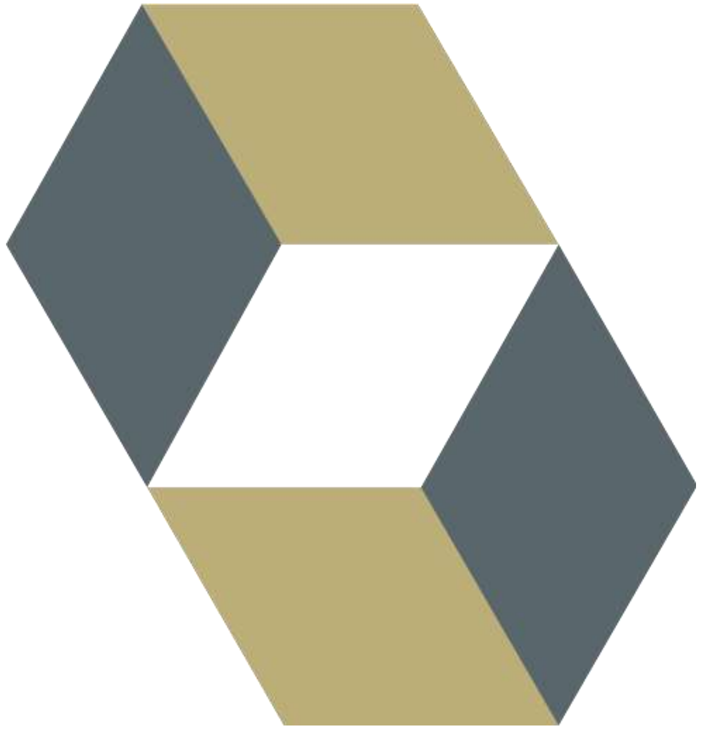
Hibernate

- Level 1 cache
  - Implemented by default
  - Related to the Session object
- Level 2 cache
  - Optional
  - Multiple integrations available

How it reads

- The cache doesn't contain the key
  1. Load from the database
  2. Put it in the cache

- The cache contains the key
  1. Return it

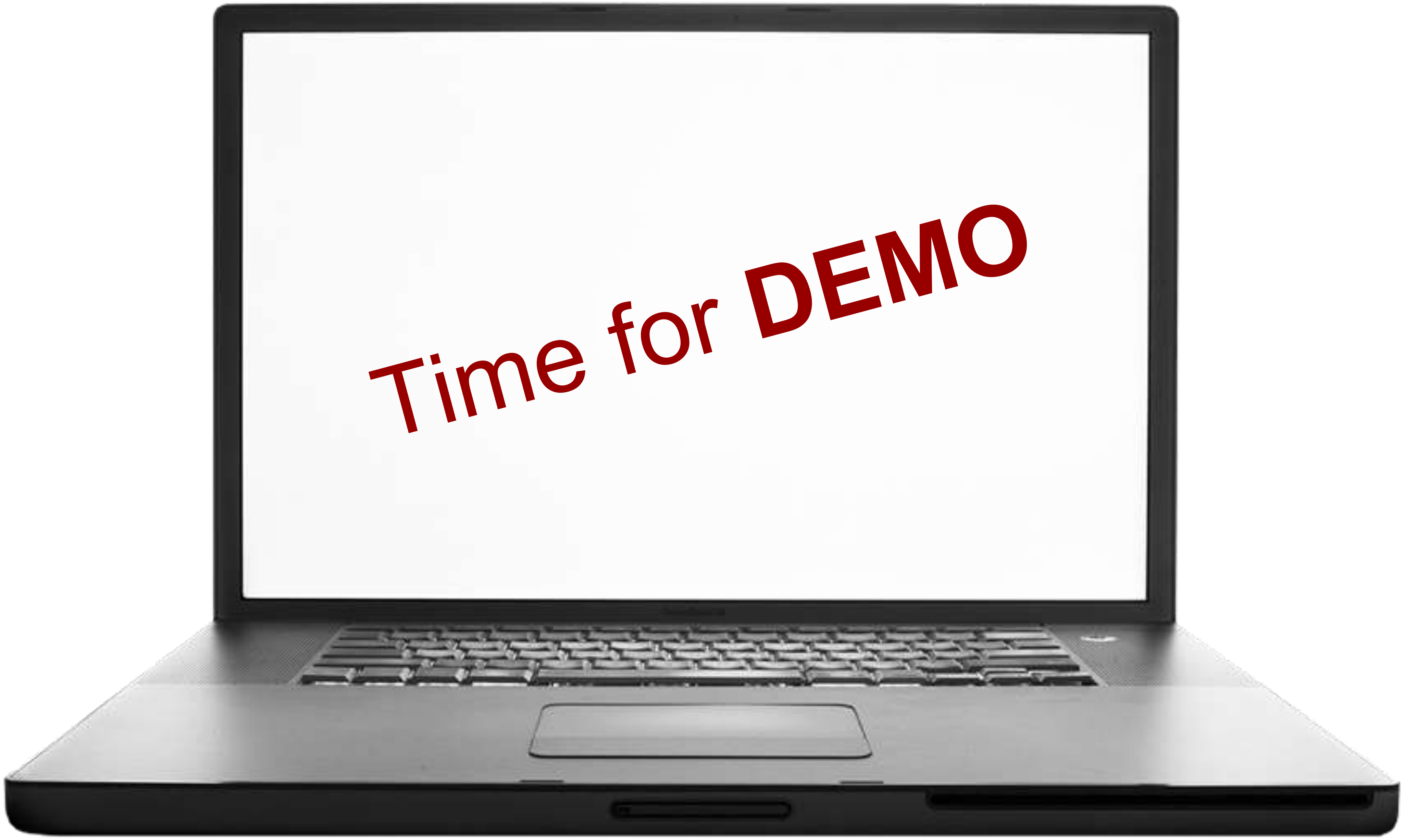


@nicolas_frankel

hazelcast

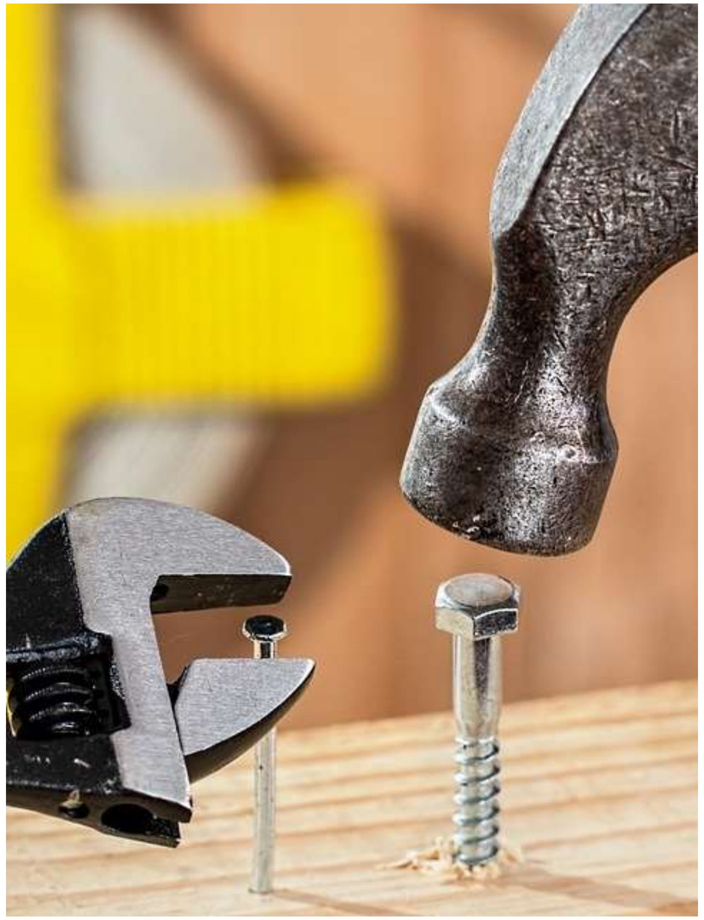
How it writes

Create or update the cached value


Time for DEMO

Alternative

- The code only interacts with Hazelcast
- Registered listeners allow to write to the database
- Sync or async



@nicolas_frankel

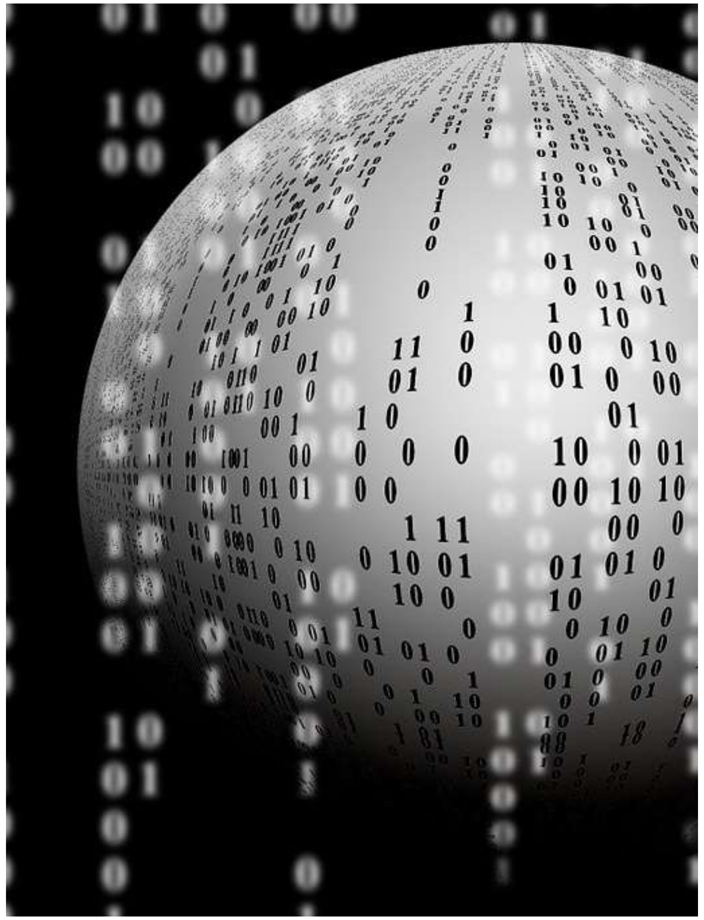
hazelcast

E-commerce architecture

- Catalog service
- Stock service
- Pricing service
- Cart service
- Recommendation service
- Payment service
- etc.

HTTP and cache

- Could be implemented manually
- But there’s a Java API for that!

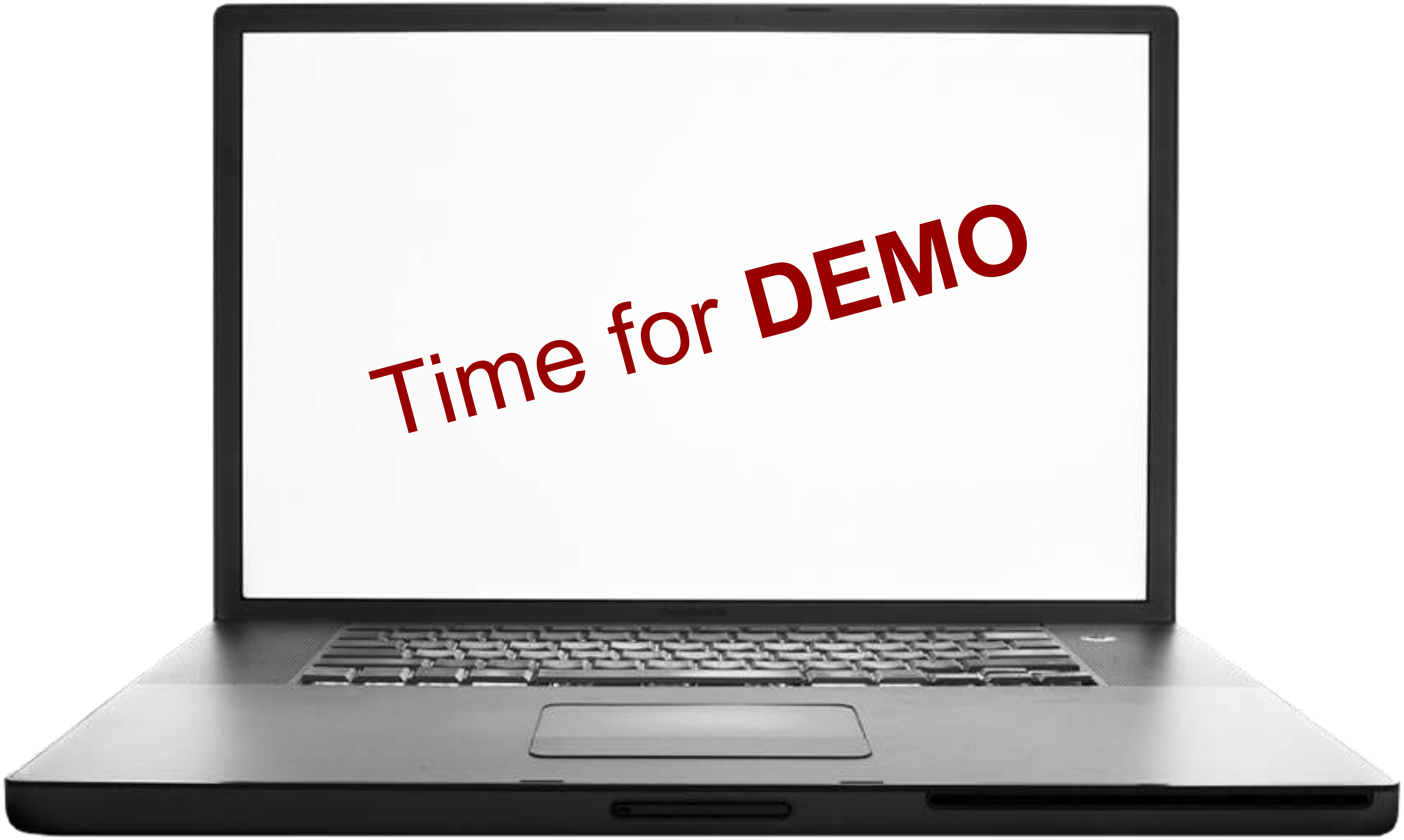


@nicolas_frankel

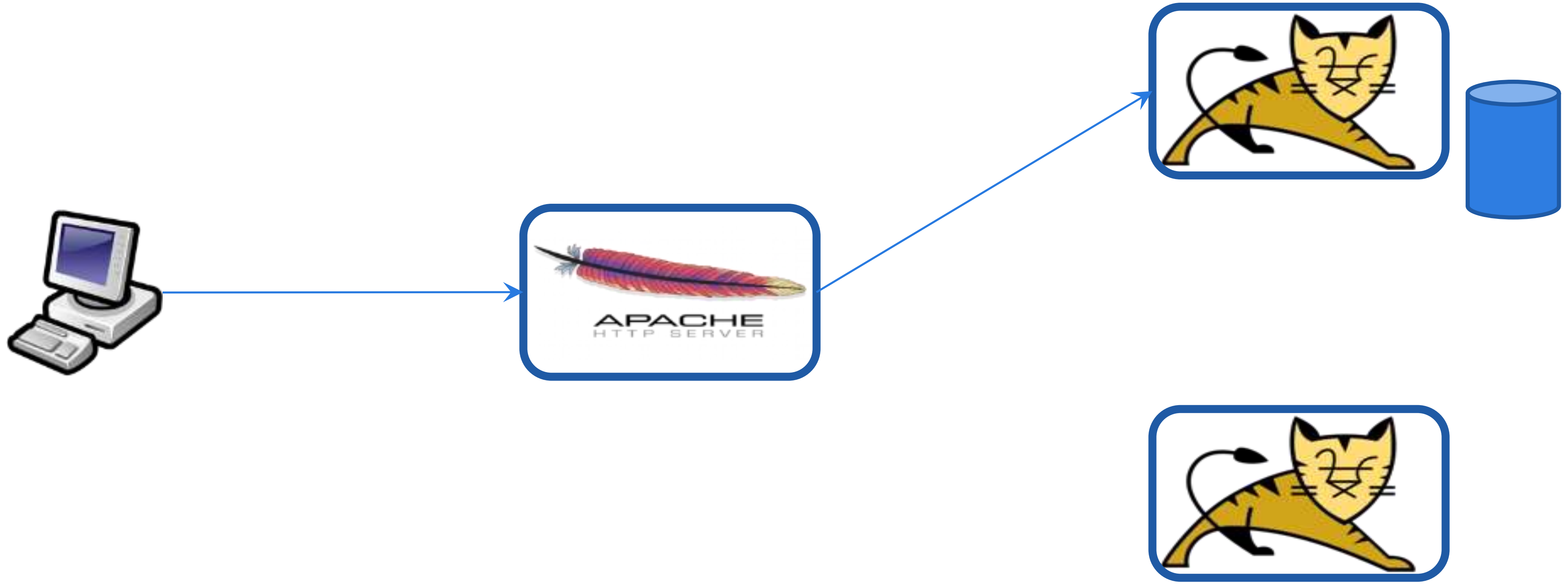
hazelcast

JCache

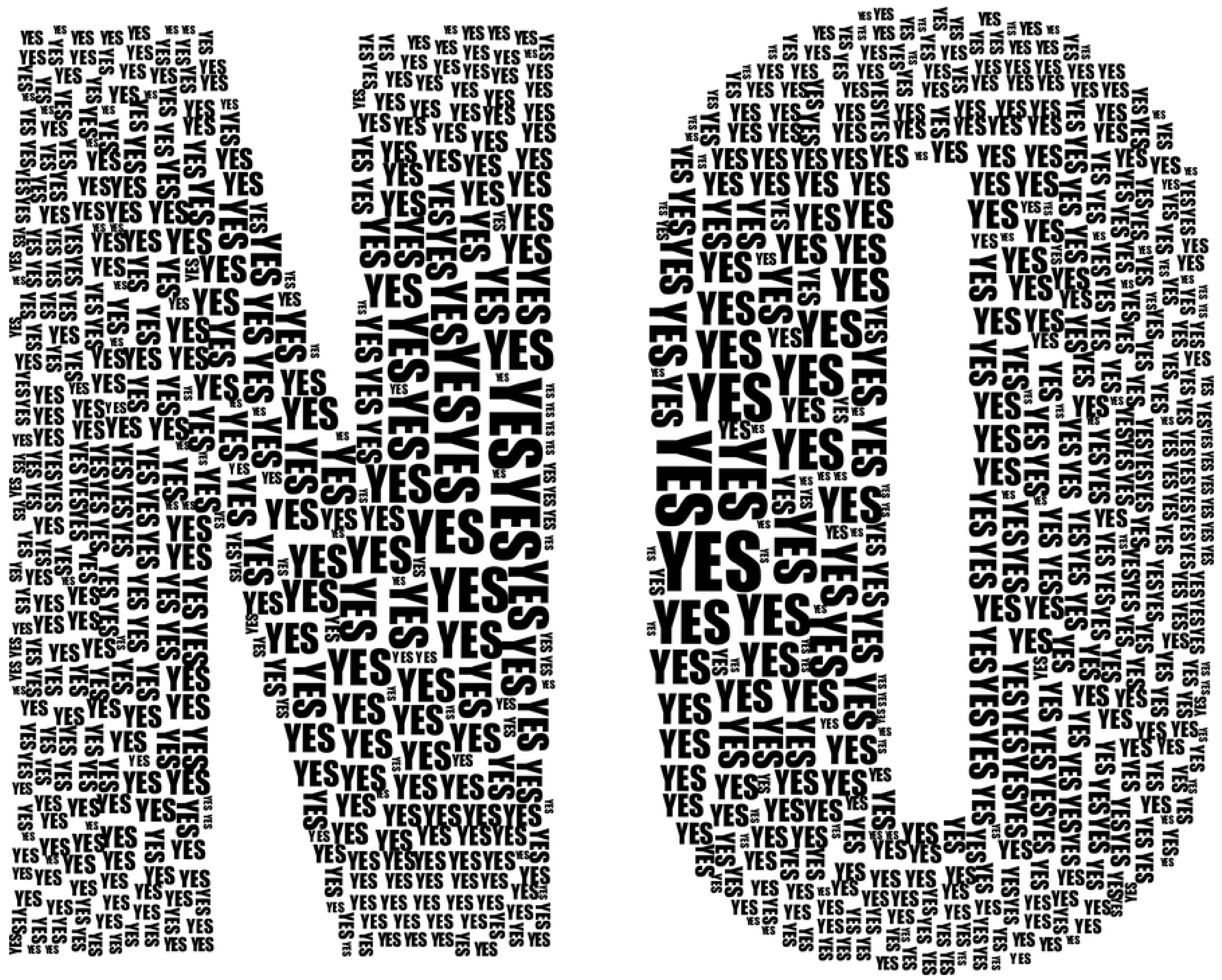
- Specification
- Multiple implementations
- Integrated with Spring


Time for DEMO

Session data in cluster nodes



Standards?

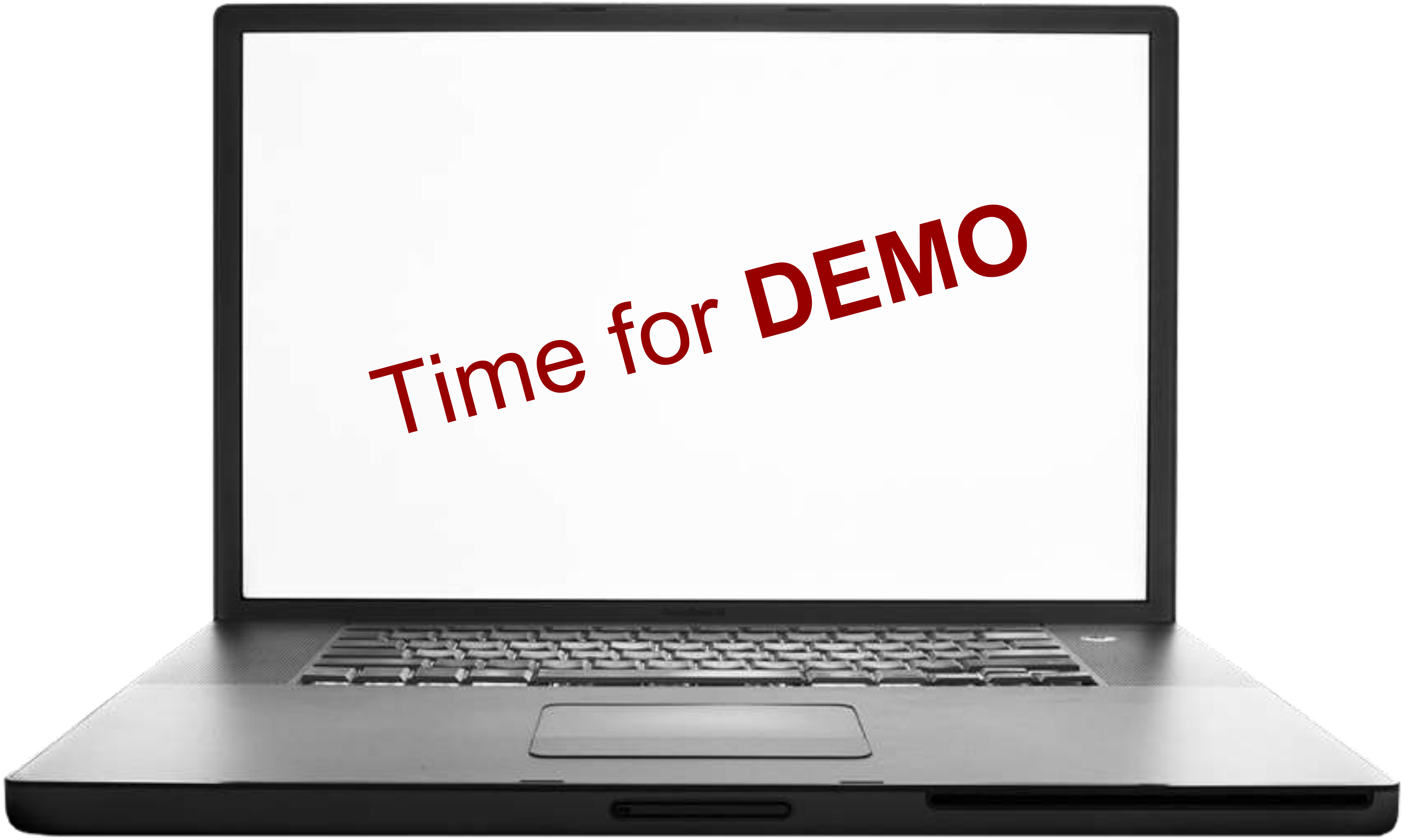


@nicolas_frankel

hazelcast

Hazelcast to the rescue!

1. Filter-based
2. Spring Session integration
3. Direct Tomcat integration
   - Per-node configuration
   - Or through Spring Boot embedded
4. Direct Jetty integration


Time for DEMO

Takeaways

- Scalability and performance are not the same
- Caching helps performance
    - The cost is stale data
- Hazelcast IMDG provides several integration-points for caching across different areas
    - Database access
    - HTTP call
    - Session data

Thanks

- https://blog.frankel.ch/
- @nicolas_frankel
- https://git.io/JenXz