Low Latency for Cloud Data Management

Felix Gessert

December 18, 2018, Universität Hamburg, DBIS Group

UHH Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Presentation is loading

Web Performance

An Open Challenge With a Huge Impact

amazon **100 ms** faster → **+1%** Revenue = **$1.7** Billion

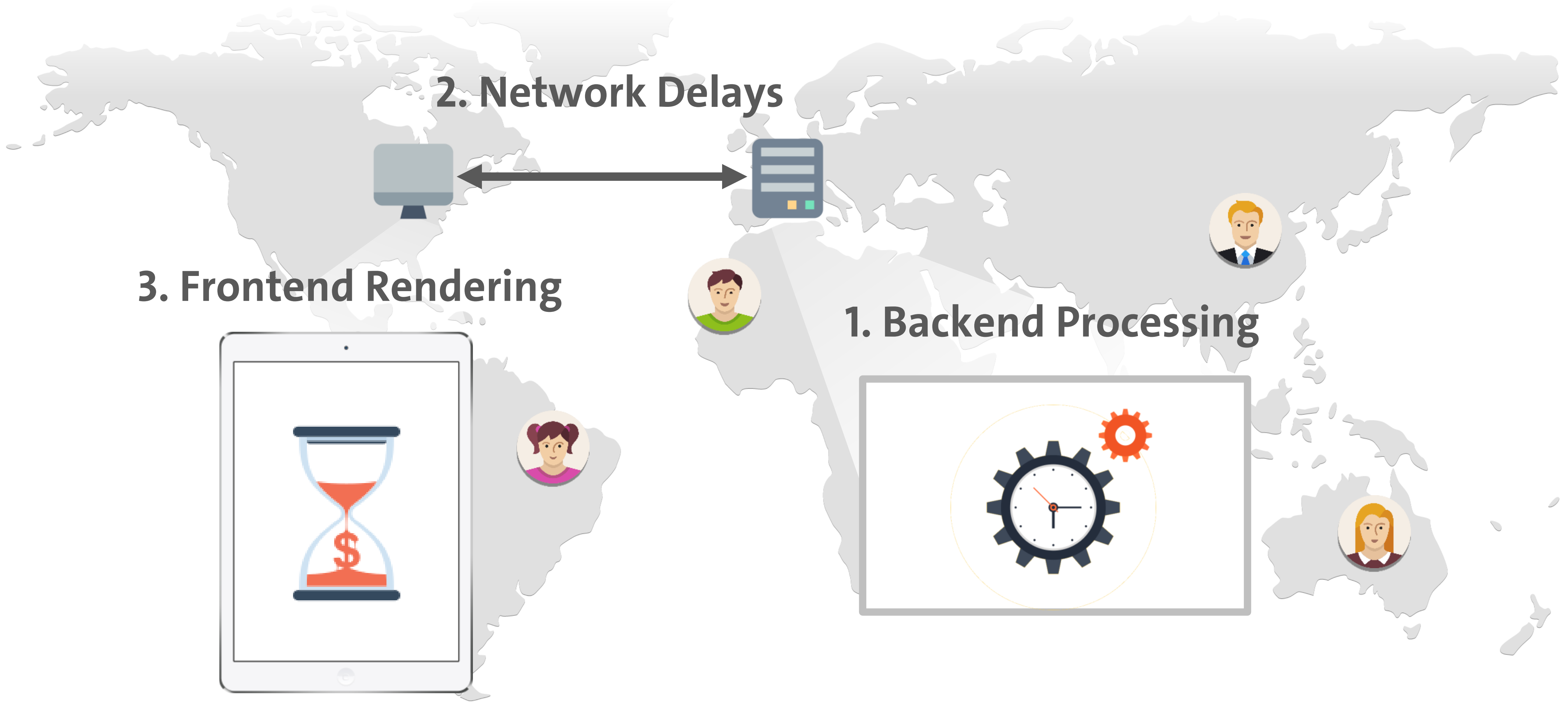
Google **500 ms** faster → **+20%** Ad Sales = **$19** Billion

Greg Linden. Make Data Useful. 2006

Tammy Everts. Time Is Money: The Business Value of Web Performance. O'Reilly Media, 2016.

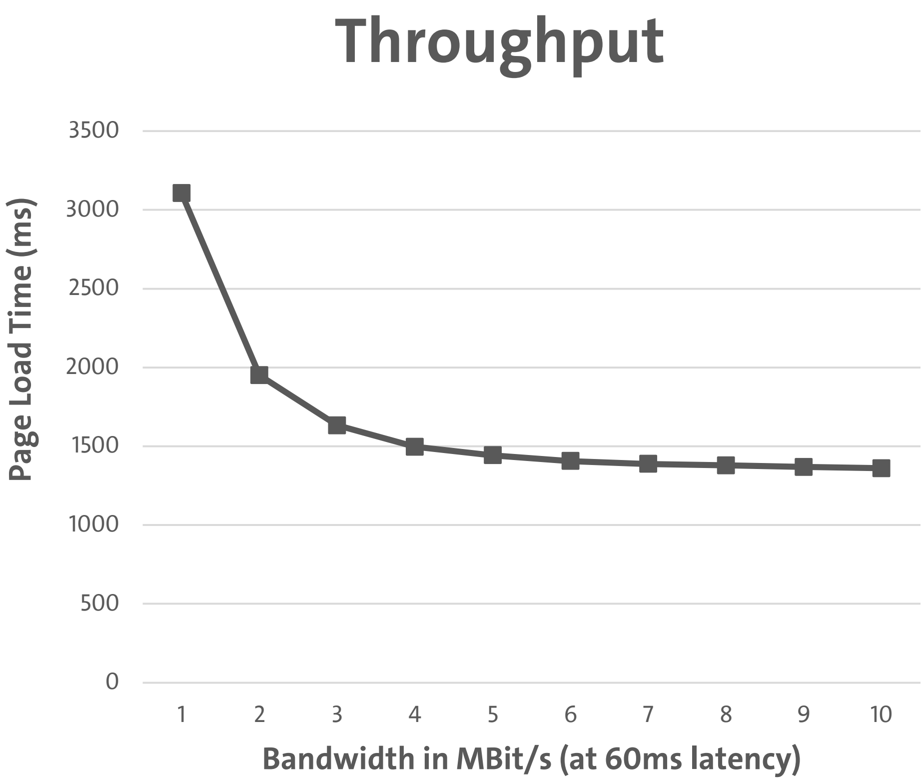
Cloud-Based Web Applications

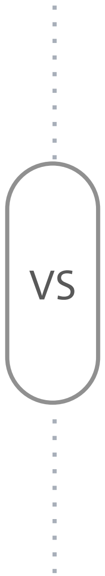
Three Sources of Page Load Time

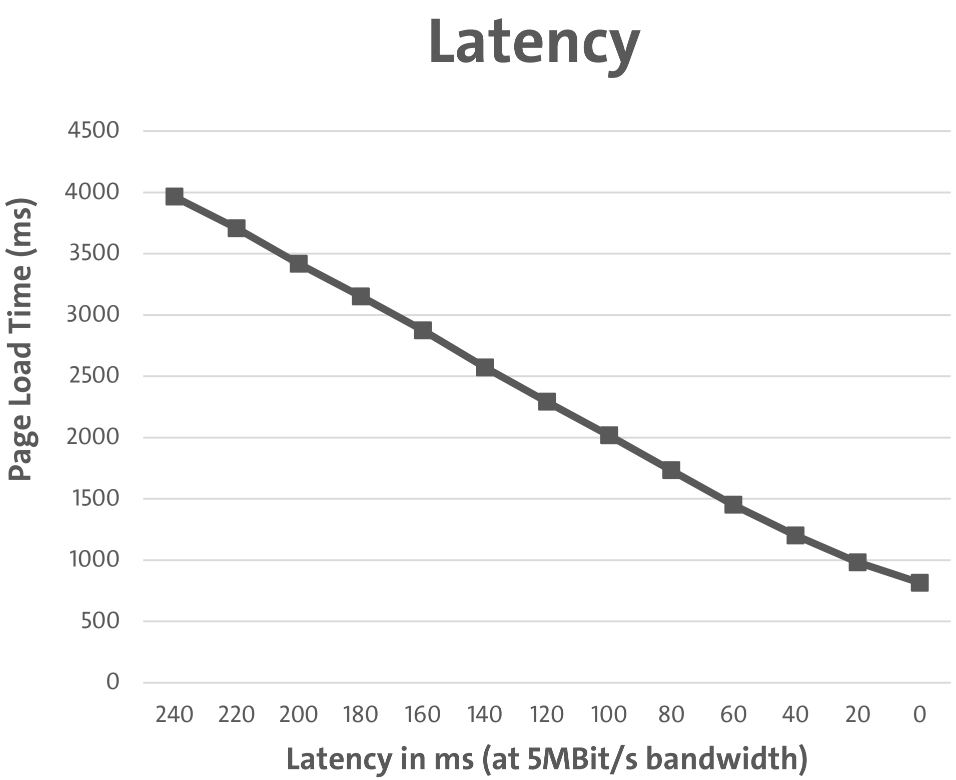


Latency is the Problem

Throughput vs. Latency







Mike Belshe. More Bandwidth Doesn't Matter (much). Technical report, Google Inc., 2010.

Latency is the Problem

Throughput vs. Latency

2× Throughput	VS	½ Latency
=		≈
Same Load Time		½ Load Time

Mike Belshe. More Bandwidth Doesn't Matter (much). Technical report, Google Inc., 2010.

Problem Statement

Four Challenges

Latency of Dynamic Data

Direct Client Access

Transaction Abort Rates

Polyglot Persistence

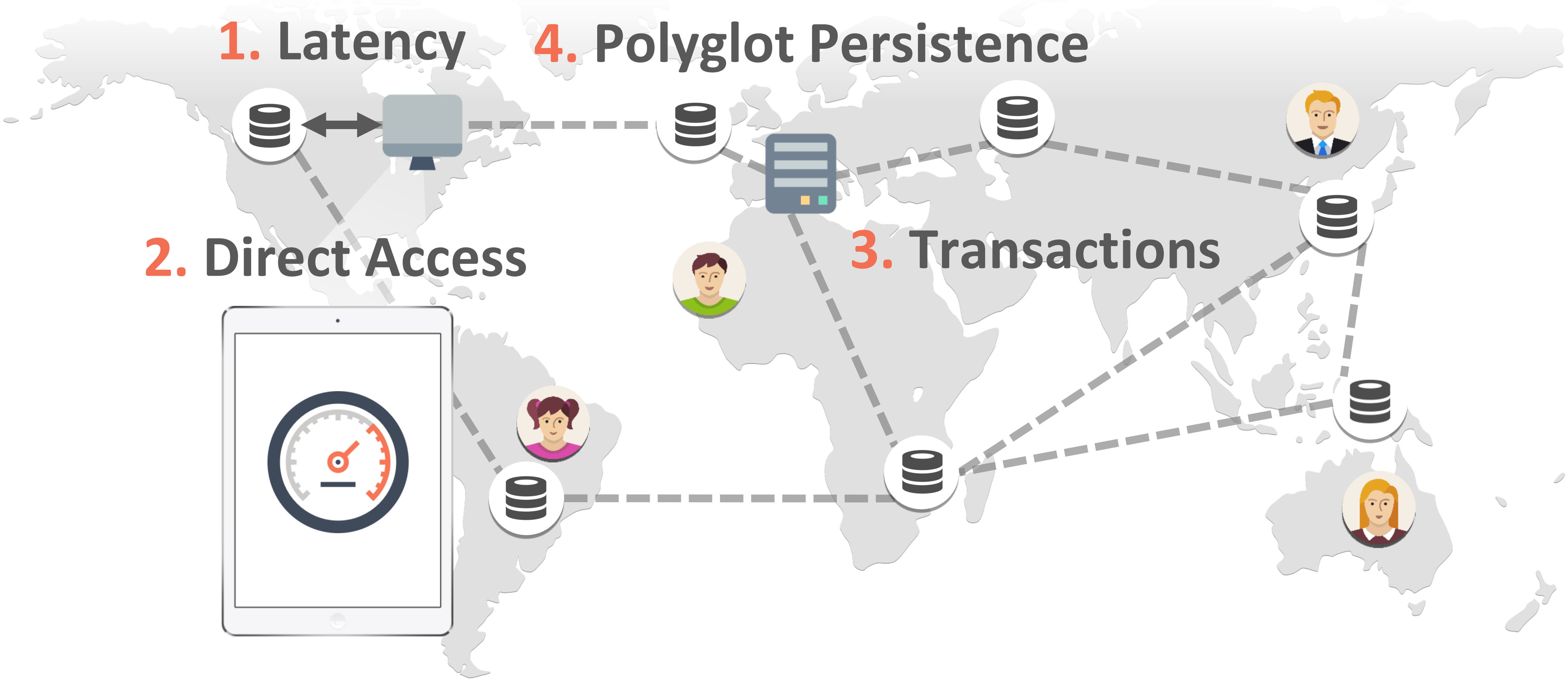
Problem Statement

Research Question

How can the **latency** of retrieving **dynamic data** from **cloud services** be minimized in an application- and **database-independent** way while maintaining strict **consistency** guarantees?

Problem Statement

Four Challenges



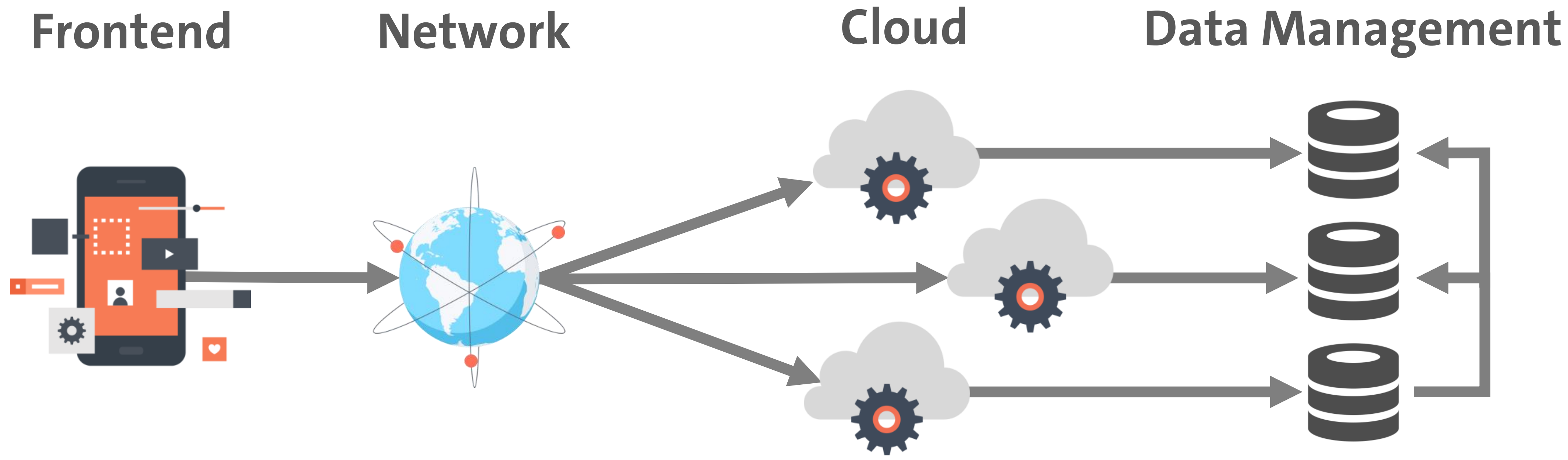
Outline

Background & Motivation → **Cloud Data Management Middleware** → **Caching Dynamic Data** → **Outlook and Summary**

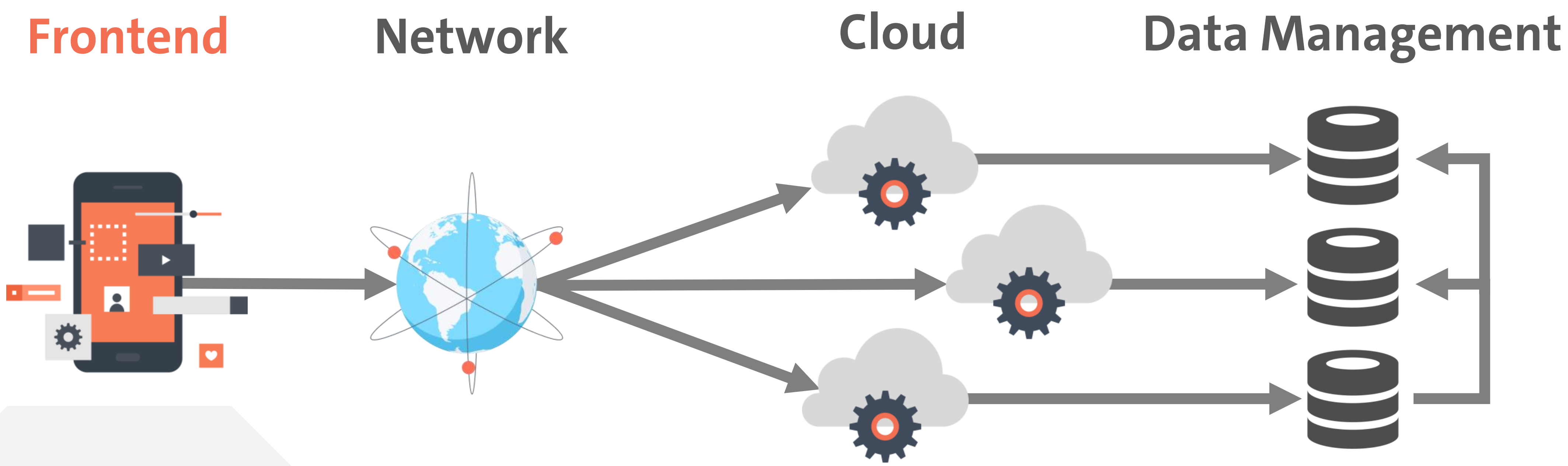
1. End-to-End Latency in Cloud-based Architectures
2. Providing Modern NoSQL Systems as a Low-Latency DBaaS
3. Cache Sketches: Solving Staleness of Reads and Queries
4. The Future of Polyglot Data Management in the Cloud

Why is end-to-end latency an open problem?

Background & Motivation



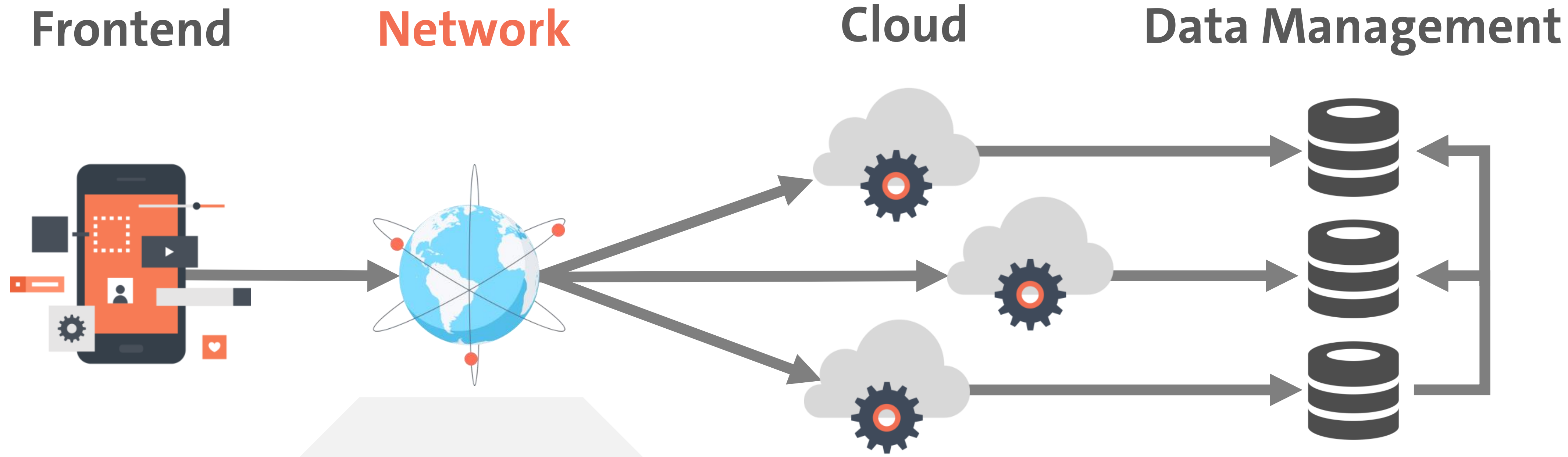
Background & Motivation



State of the Art:
- **Web interface** based on HTML, files, APIs, and application logic
- Performance defined by **critical rendering path**

Problems:
- No direct access or integration to **data management**
- **Storage** maintained manually

Background & Motivation



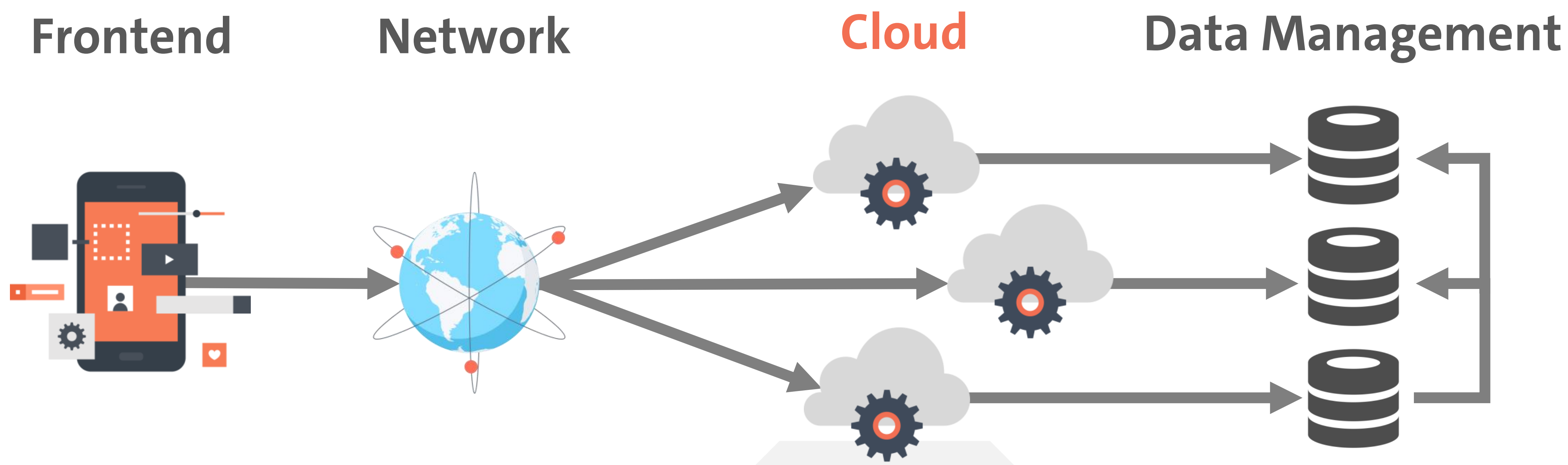
State of the Art:

- Service interfaces use **REST & HTTP**
- **Web caching** can reduce end-to-end latency

Problem:

- Web caching not compatible with consistent **dynamic data**

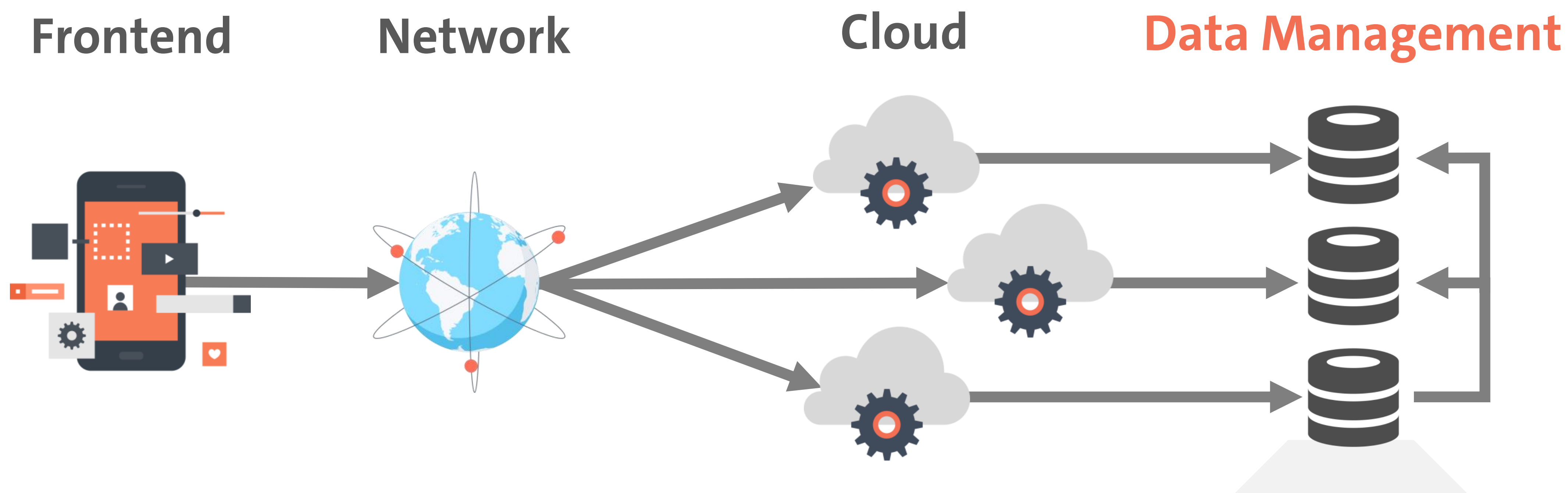
Background & Motivation



State of the Art:
- **SaaS**, **PaaS**, and **IaaS** models
- **Scalability** & multi-tenancy

Problems:
- Combination of cloud services entails **high latency**
- Common application **building blocks** often re-implemented

Background & Motivation



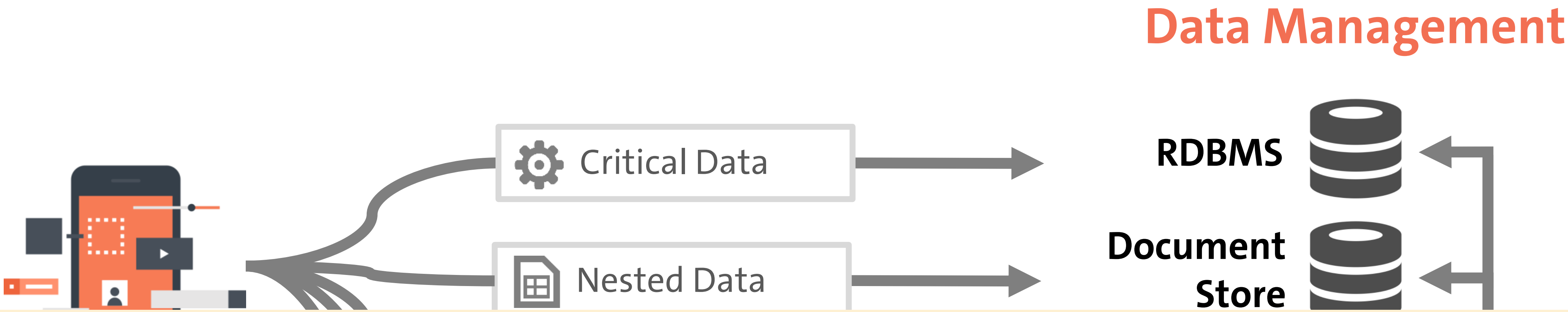
State of the Art:

- **Scalability** & **high availability** through **NoSQL** systems
- **Sharding** & **replication**
- **Database-as-a-Service** (DBaaS) model

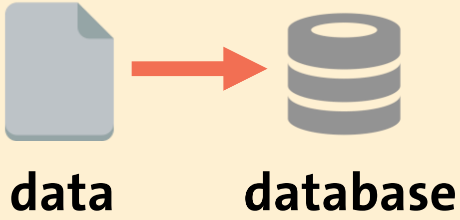
Problems:

- Lack of common **data management abstractions**
- **DBaaS model** not supported
- **Polyglot persistence** manual & error-prone

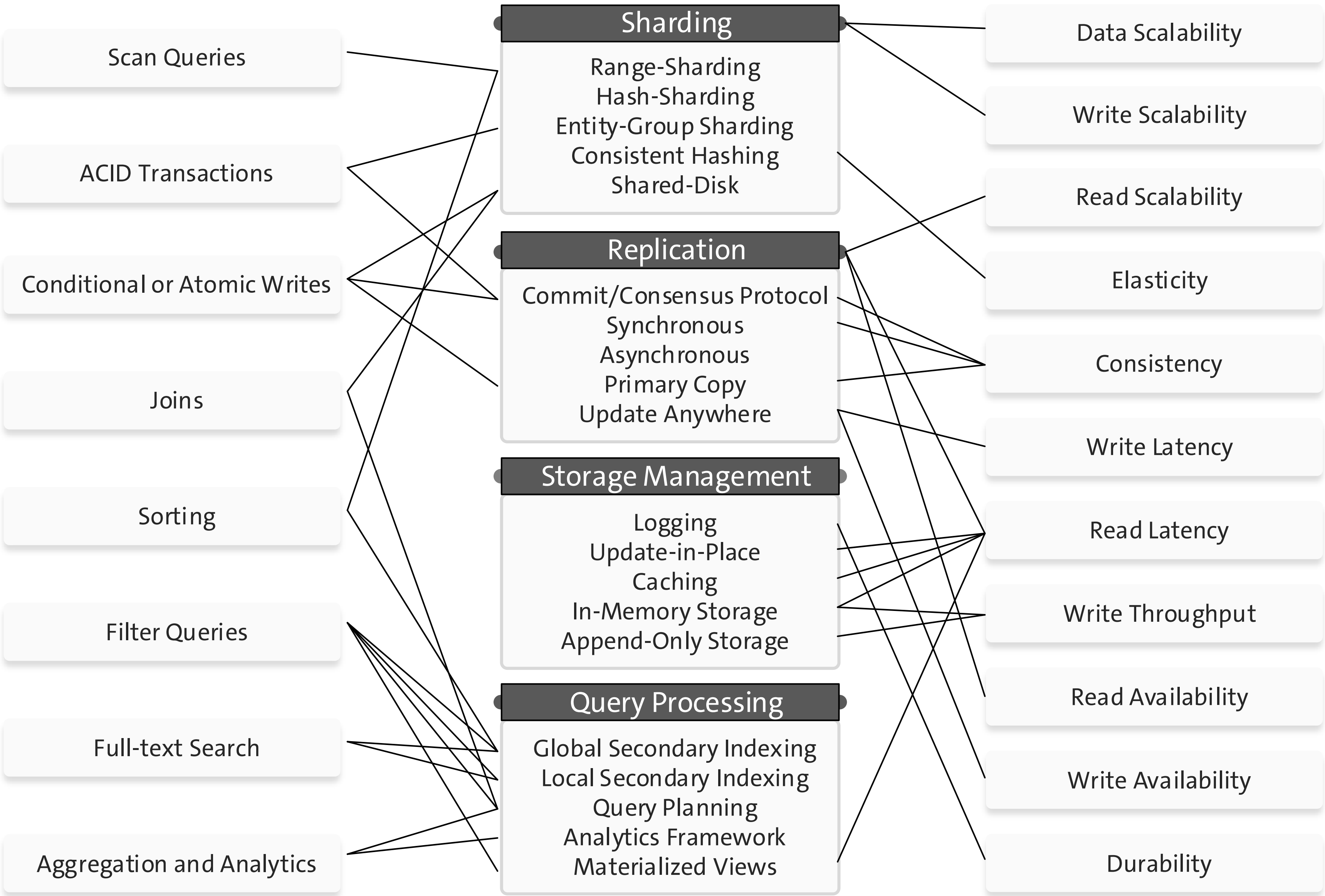
Background & Motivation



Challenge:

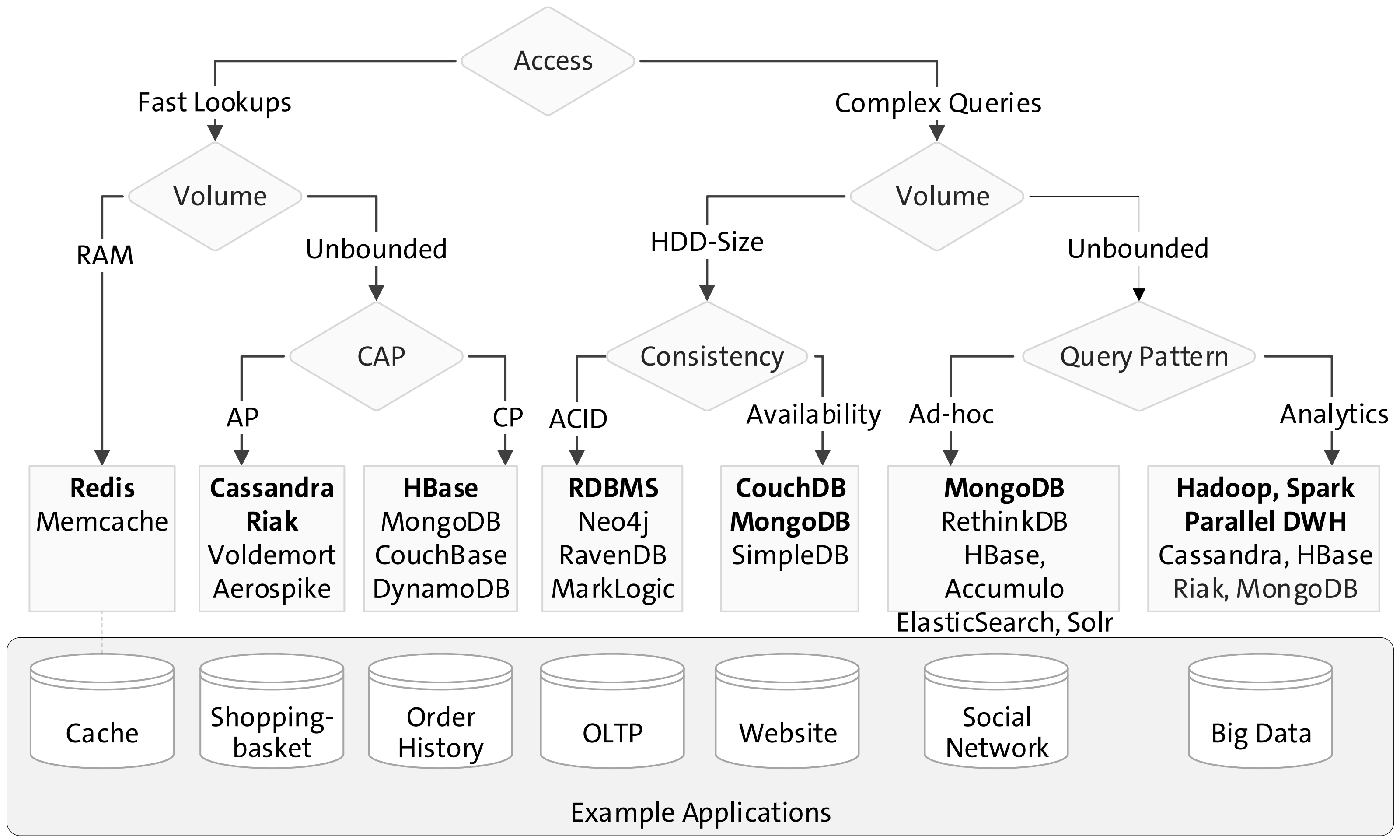
How to tackle the  mapping problem?

Functional Requirements | Data Management Techniques | Non-Functional Requirements



[GWFR16]

NoSQL Toolbox → Decision Tree



[GWFR16]

NoSQL Toolbox → **Decision Tree** → **Unified Data Management API**

Backend-as-a-Service

Database-as-a-Service

- Object Persistence
- Transaction Processing
- Query Support
- Partial Updates
- Schema Management
- Indexing & Configuration

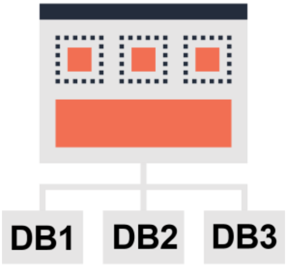
- Data Validation
- Code Execution
- Access Control

[GFW+14]

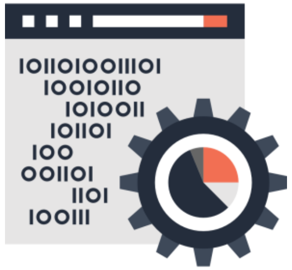
How can cloud data management be unified & combined with low latency?

Orestes: Goals

A Data Management Middleware for Low Latency



Database Independence



DBaaS & BaaS Functionality

Scalable, Available, Multi-Tenant

Low Latency with Tunable Consistency

Orestes Concept

Overview

Unmodified **Database Systems**

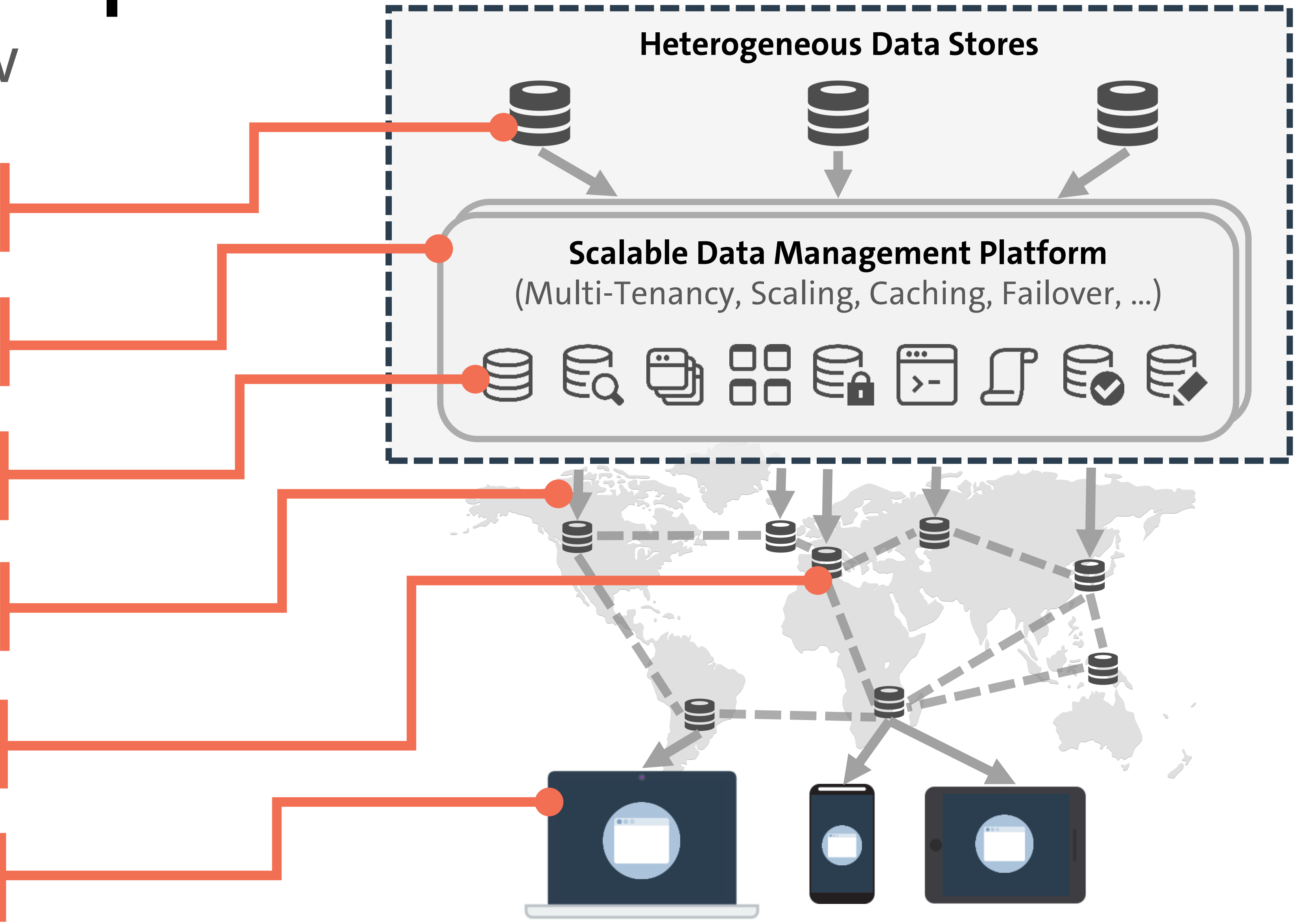
DBaaS/BaaS Middleware

Data and Default **Modules**

Unified REST API

Web Caching for **Low Latency**

Web and Mobile Applications



[GBR14, GB13]

How can dynamic data be accelerated through web caching?

The Web‘s Caching Model

Expiration-Based Caches:

- An object x is considered fresh for TTL_x seconds
- Server assigns TTLs for each object

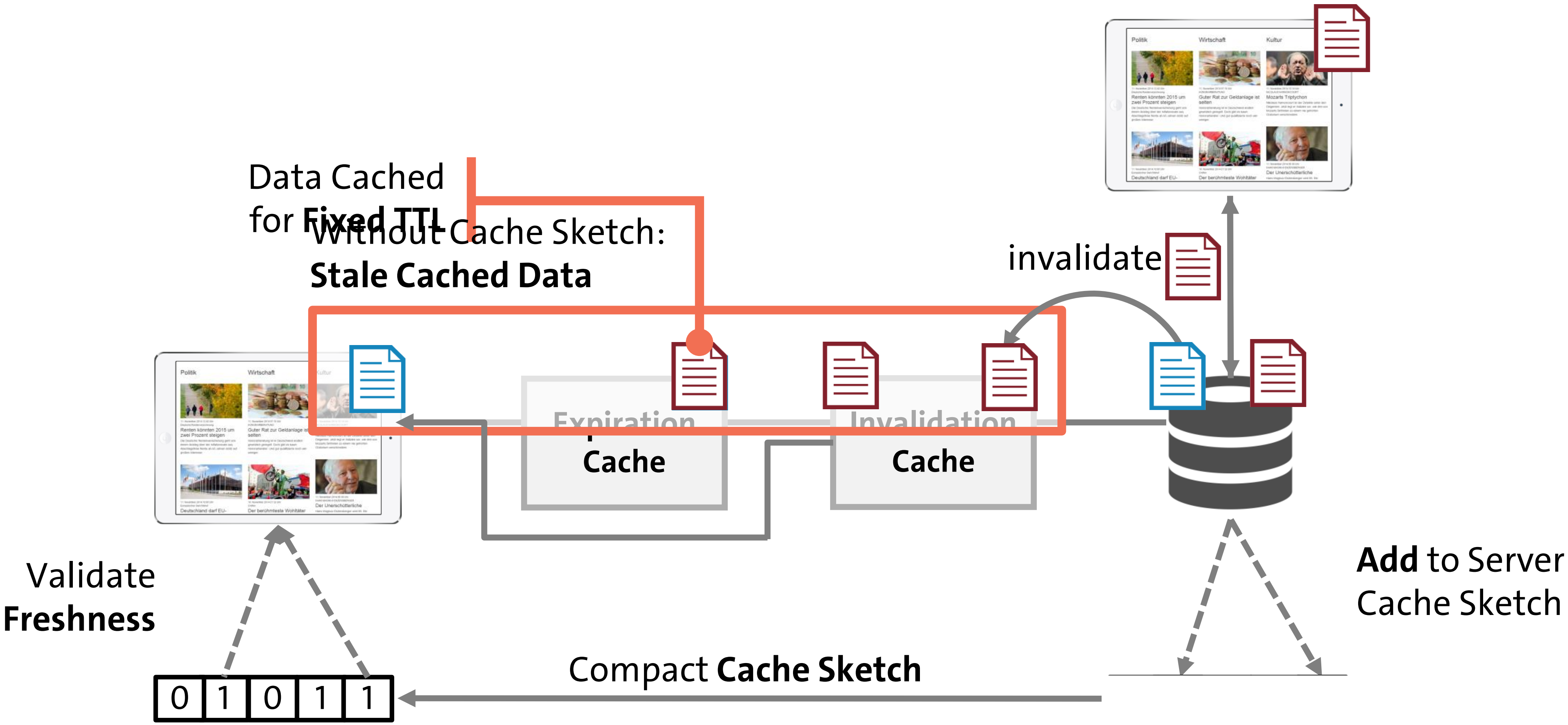
Invalidation-Based Caches:

- Expose object eviction operation to the server

[GSW+15]

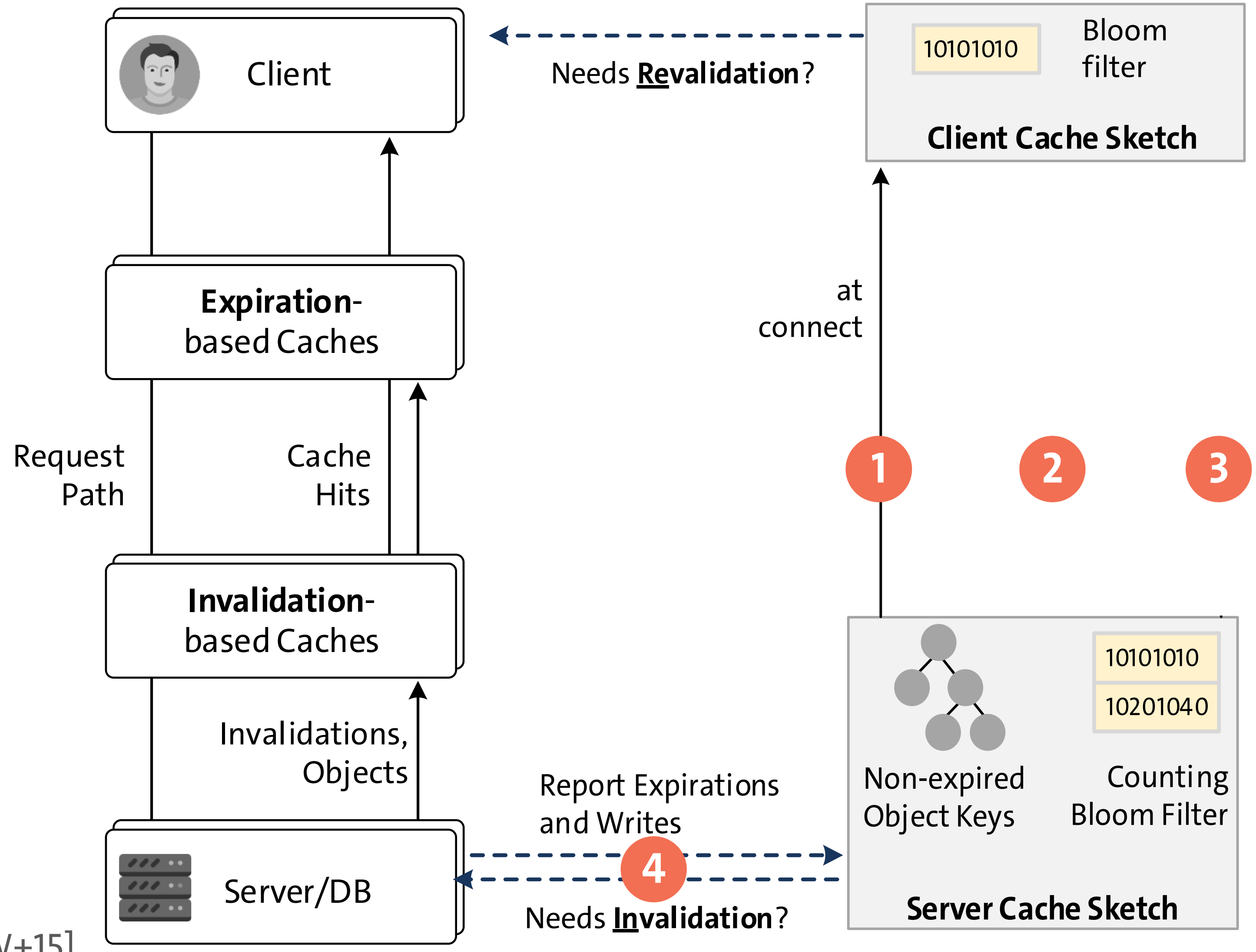
Web Caching for Data Management

Overview of Cache Sketch Method



[GSW+15, GSW+17]

The Cache Sketch Approach



Minimize
Staleness

1 Initialization from Cache

2 Δ-Atomic Consistency

3 Cache-Aware Transactions

Minimize
Invalidations

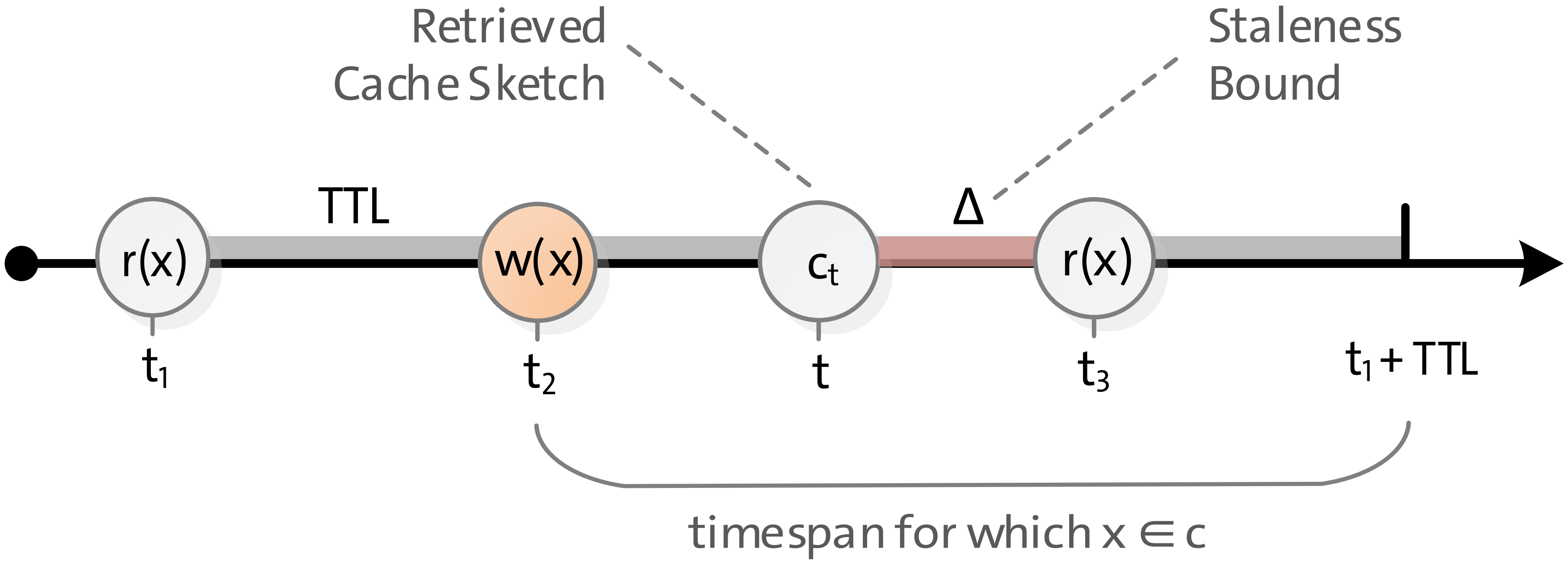
4 Invalidation Minimization

[GSW+15]

Cache Sketch

Main Properties

To ensure **Δ-atomicity** the Cache Sketch at time **t** contains **key(x)** of every object **x** that was <u>written before it expired</u> in all caches.



[GSW+15, GSW+17]

Cache Sketch

Construction

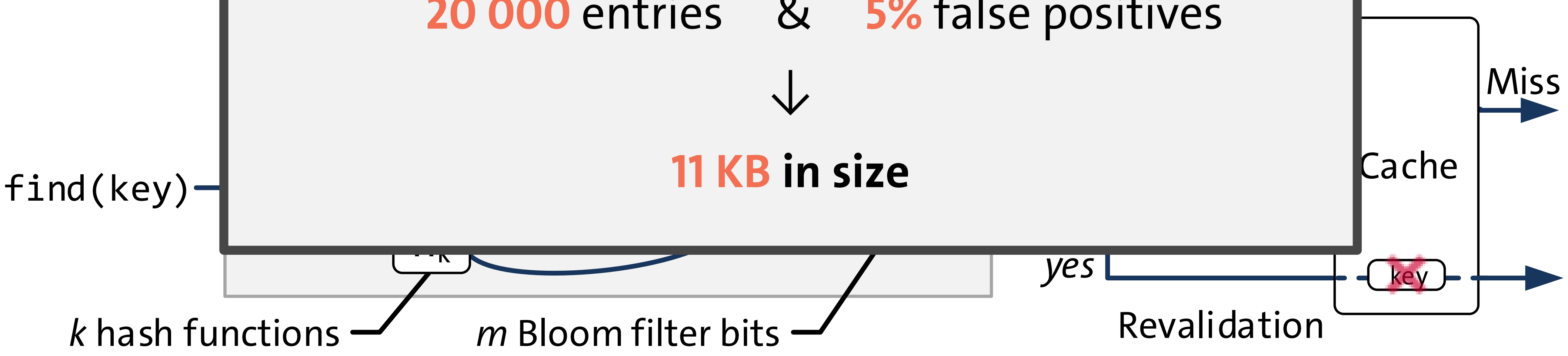
To ensure **compactness** the Cache Sketch stores *n* keys in a **Bloom filter** with *m* bits, *k* hash functions and a … p…

Example

20 000 entries & 5% false positives

↓

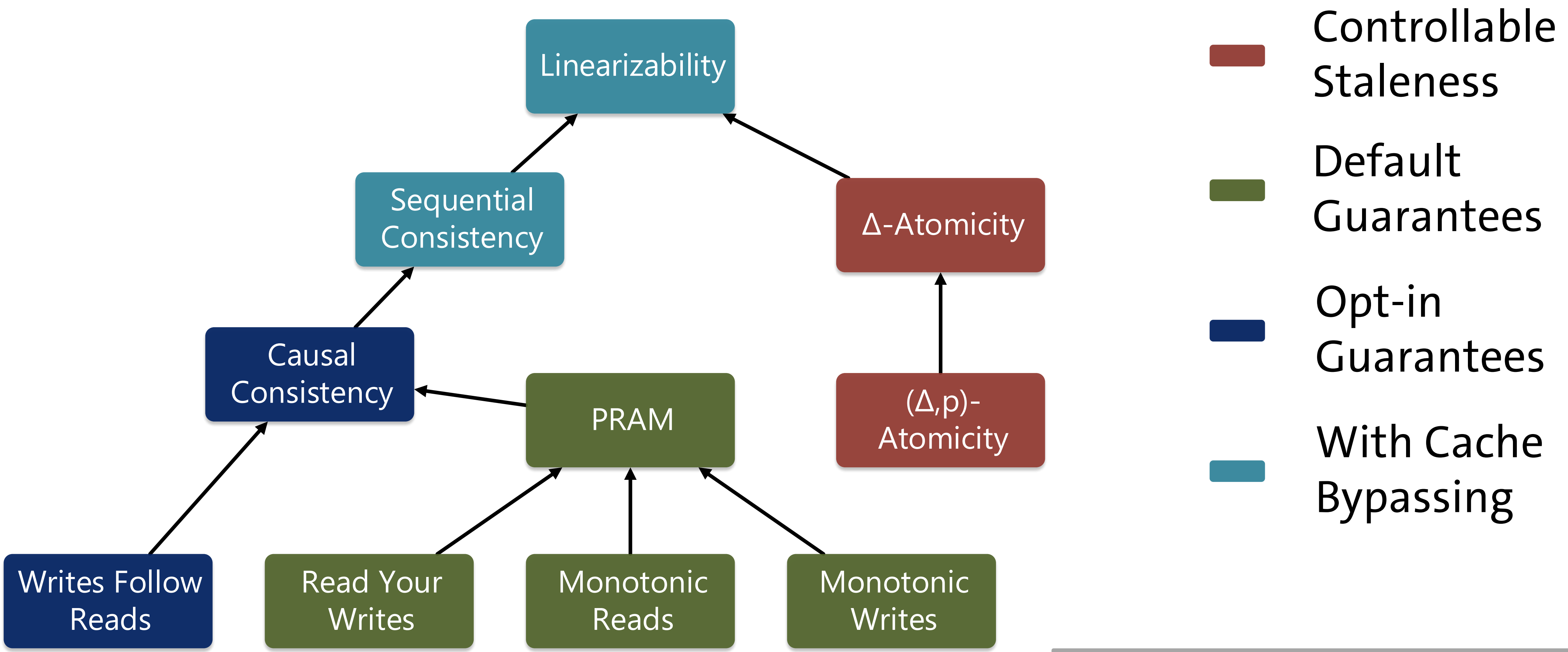
11 KB in size



[GSW+15, GSW+17]

Controllable Consistency Levels

Cache Sketch Guarantees



[WGW+18, GWR17, GWFR16, GR16, FWGR14, FWGR14]

P. Viotti and M. Vukoli´c. Consistency in Non-Transactional Distributed Storage Systems. ACM Computing Surveys, 2016.

Determining TTLs

Trade-Off

Shorter TTLs

⇩ Invalidations
⇩ False Positive Rate

vs

Longer TTLs

⇩ Cache Misses

[GSW+17, GSW+15]

TTL Estimation

Optimizing Expiration & Cacheability

1. Collect **workload statistics** for reads and writes
2. Estimate **time to next write** $E[T_w]$ or mark uncacheable

Constrained Adaptive TTL Estimator

- Ideal for Poisson Processes

C-LM Model

- Quick Adaption to Changes

LWMA Estimator

- Converges for Static Workloads

EWMA Estimator

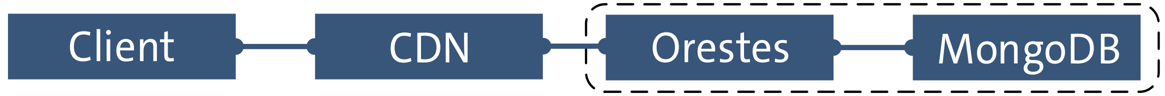
- Highly Space-Efficient

[GSW+15]

Evaluation Results

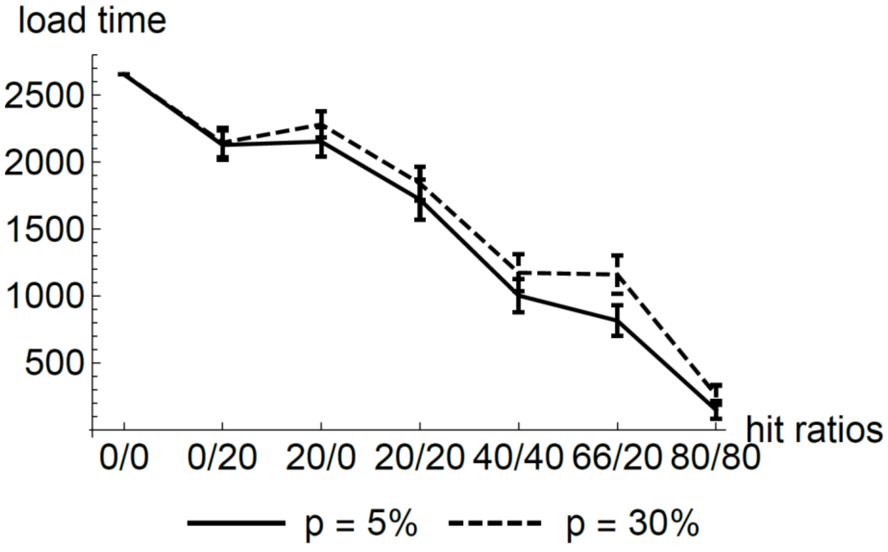
Simulation & Benchmarking

Setup:

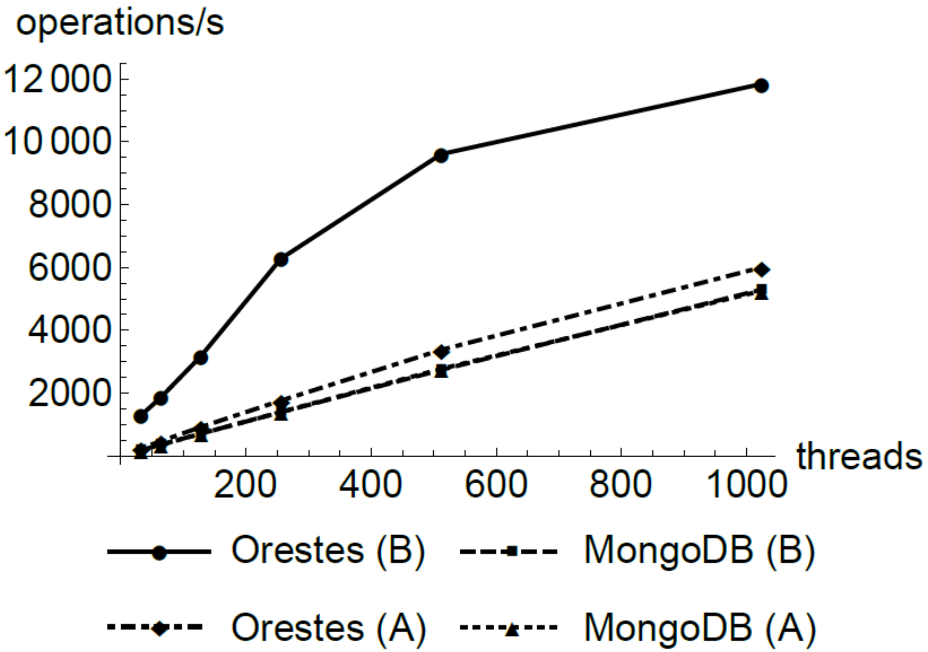


Ireland *California*

Page load times with cached initialization (*Simulation*):



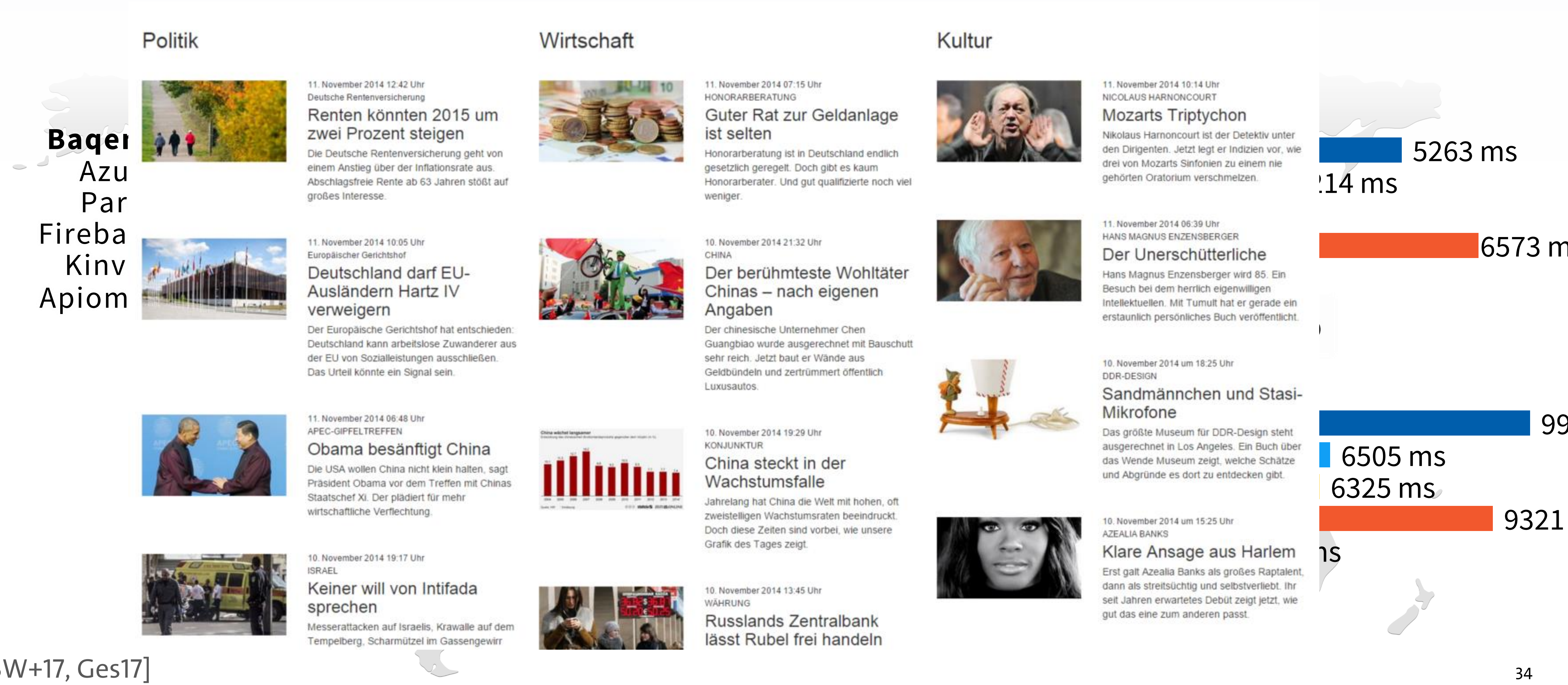
Average **throughput** for YCSB workloads A and B (*YCSB benchmark*):



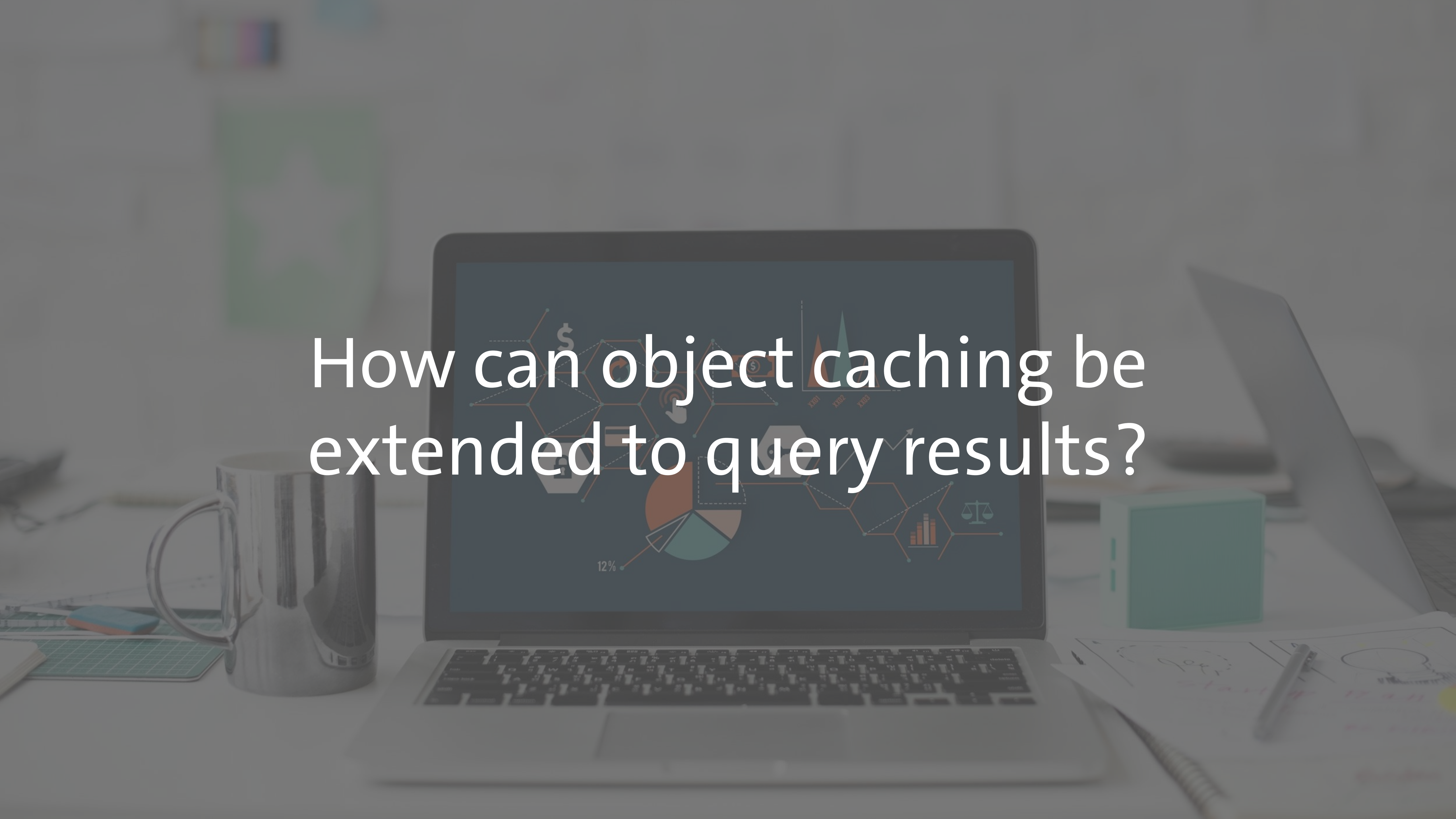
[GSW+15]

Evaluation Results

Industry Evaluation of Commercial Implementation



[GSW+17, Ges17]



How can object caching be extended to query results?

Query Caching

Challenges

Invalidation Detection

When do query results change?

Cache Coherence

How to apply Cache Sketches to queries?

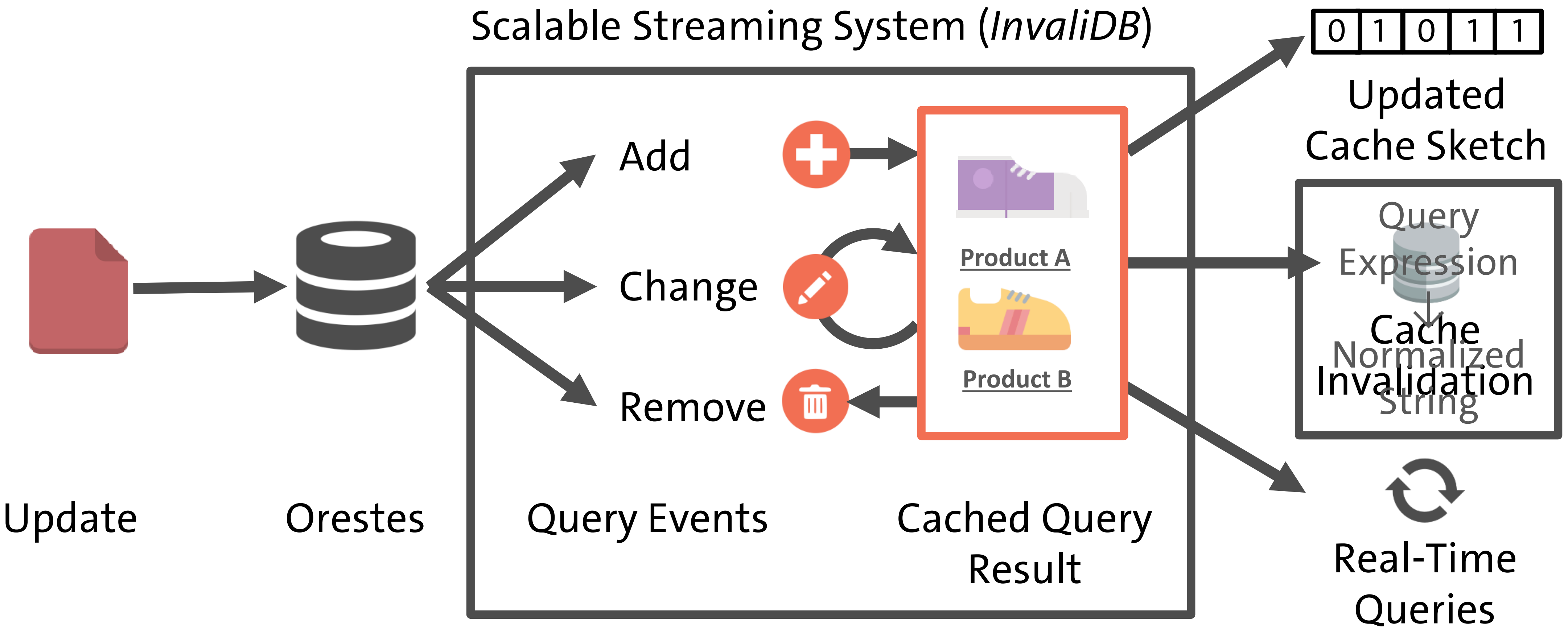
Query Result Representation

What is the best result structure for caching?

[GSW+17]

Invalidation Detection

Cache Coherence for Query Results



[WRG18, WGF+17, GSW+17, WGFR16]

Learning Result Represenations

Handling Changes to Query Results

ID Lists

$$[url_1, url_2, url_3]$$

⇩ Invalidations

VS

Object Lists

$$[\{id\colon\ obj_1, name\colon "alice"\},\ \{id\colon\ obj_2, name\colon "bob"\},\ \{id\colon\ obj_3, name\colon "eve"\}]$$

⇩ Round-Trips

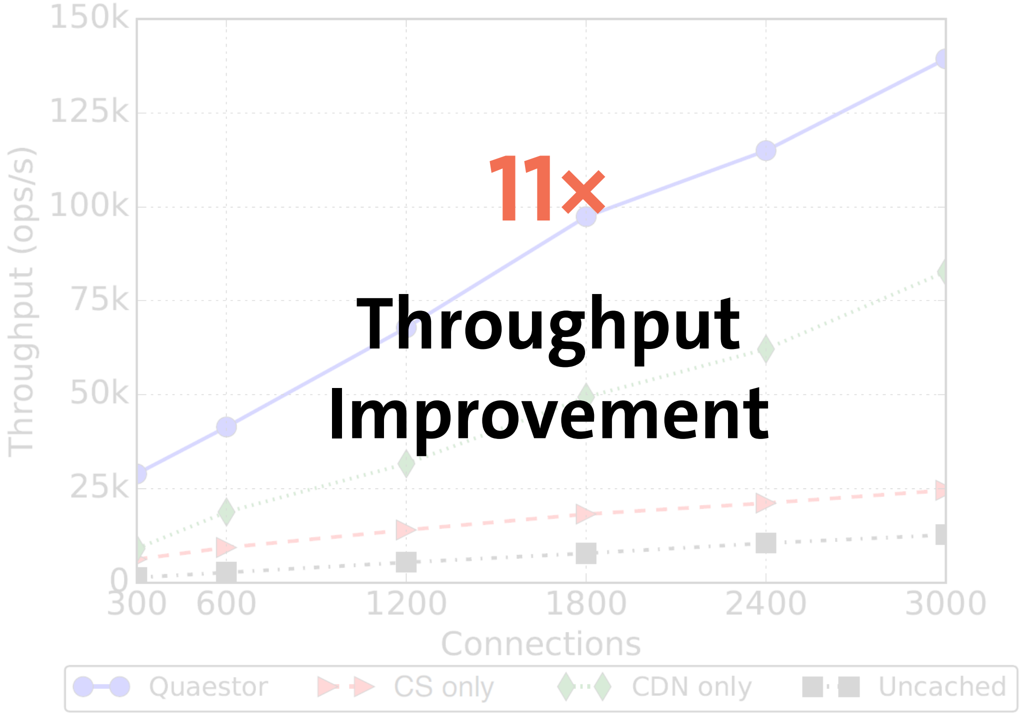
Solution: Cost-based **decision model** weighs expected round-trips vs. invalidations

[GSW+17]

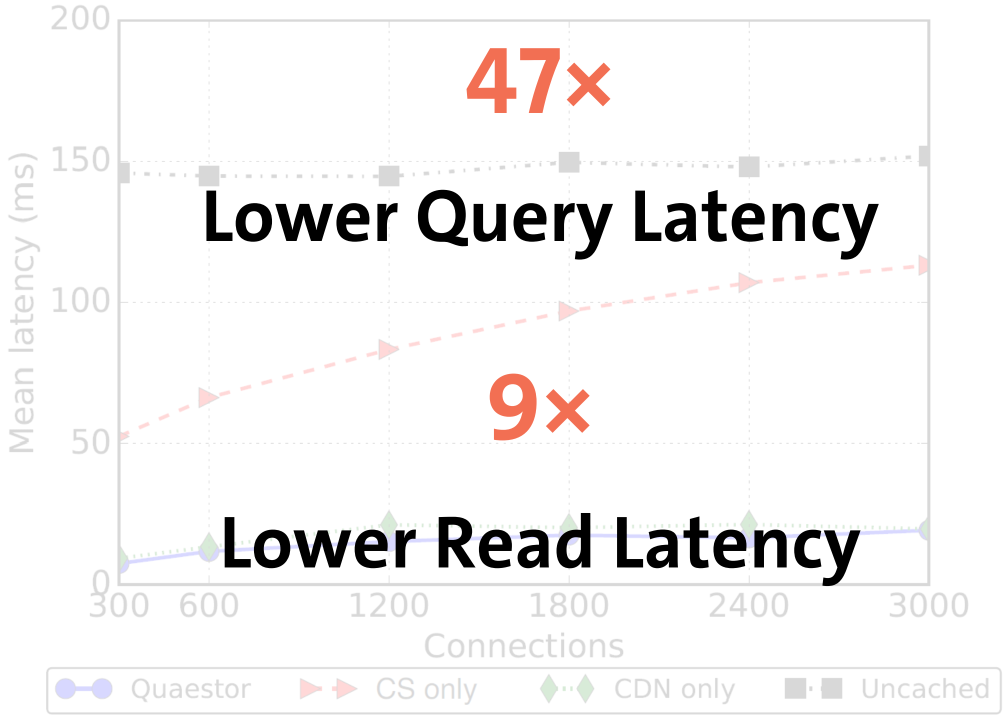
Evaluation Results

Query Caching for YCSB-Based Workloads

Throughput with growing request parallelsim:



Average end-to-end **query** and **read latency**:

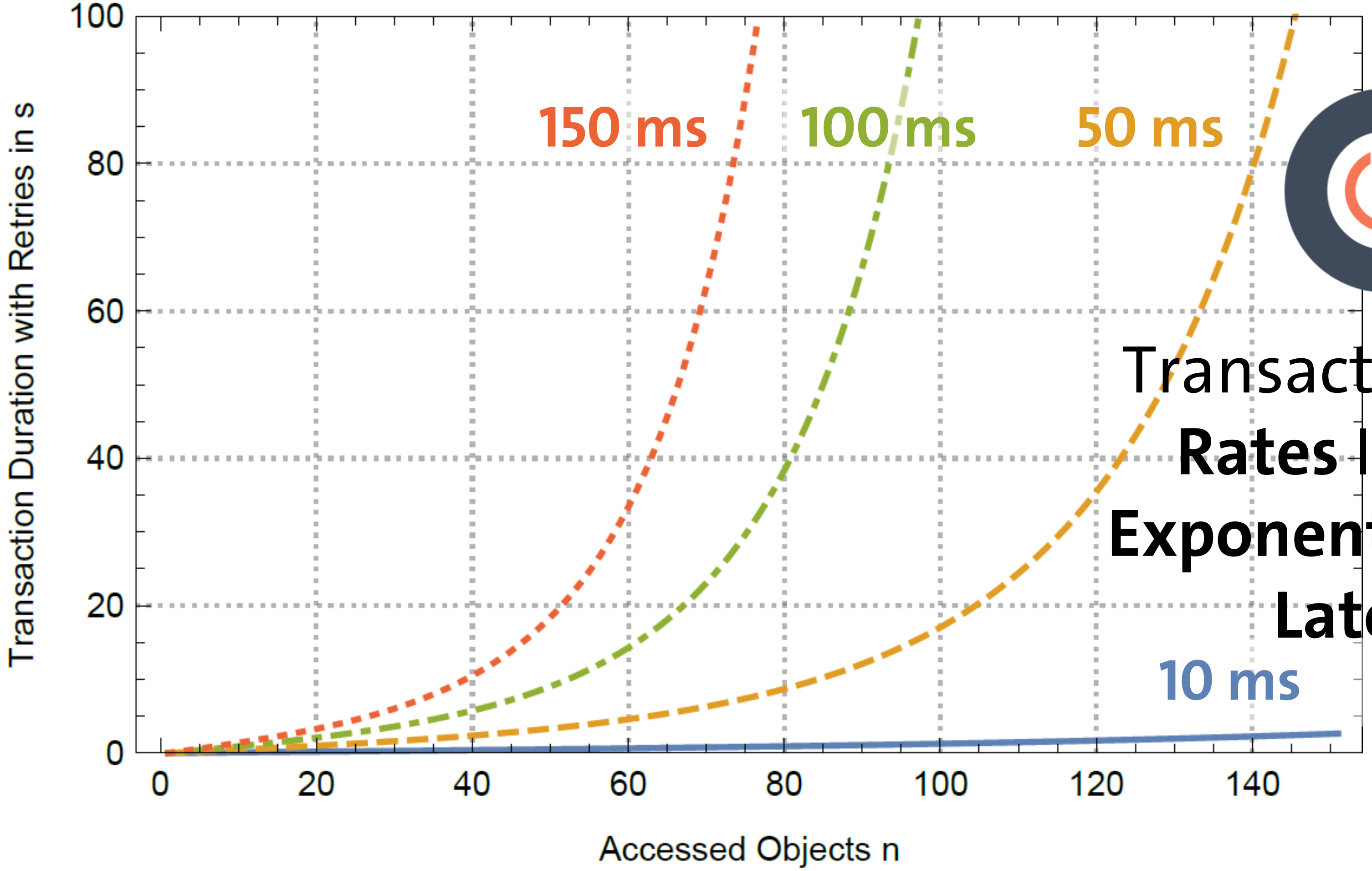


[GSW+17]


Can Cache Sketches improve
transaction performance?

Problem of Optimistic Transactions

Abort Rates Depend on Latency



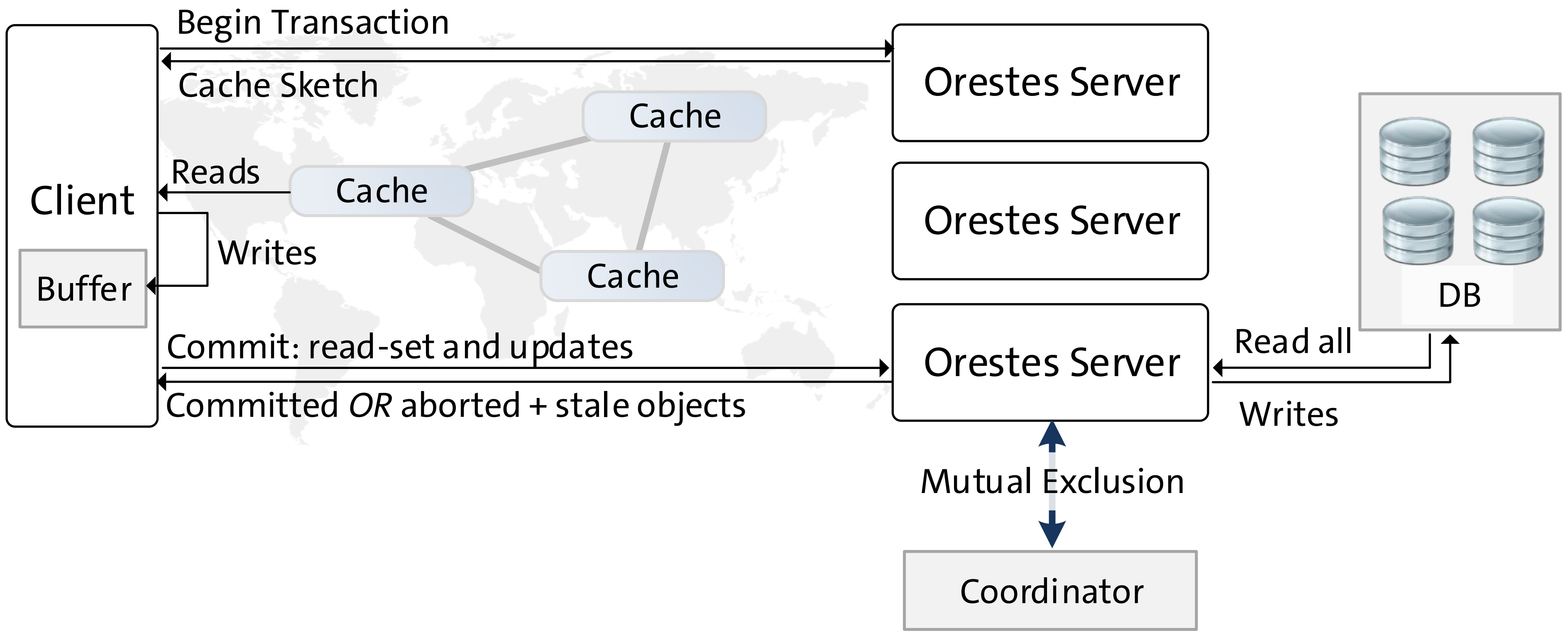
Transaction **Abort Rates** Increase **Exponentially** with **Latency**

[GBR14]

Distributed Cache-Aware Transactions

DCAT Solves Latency Problem

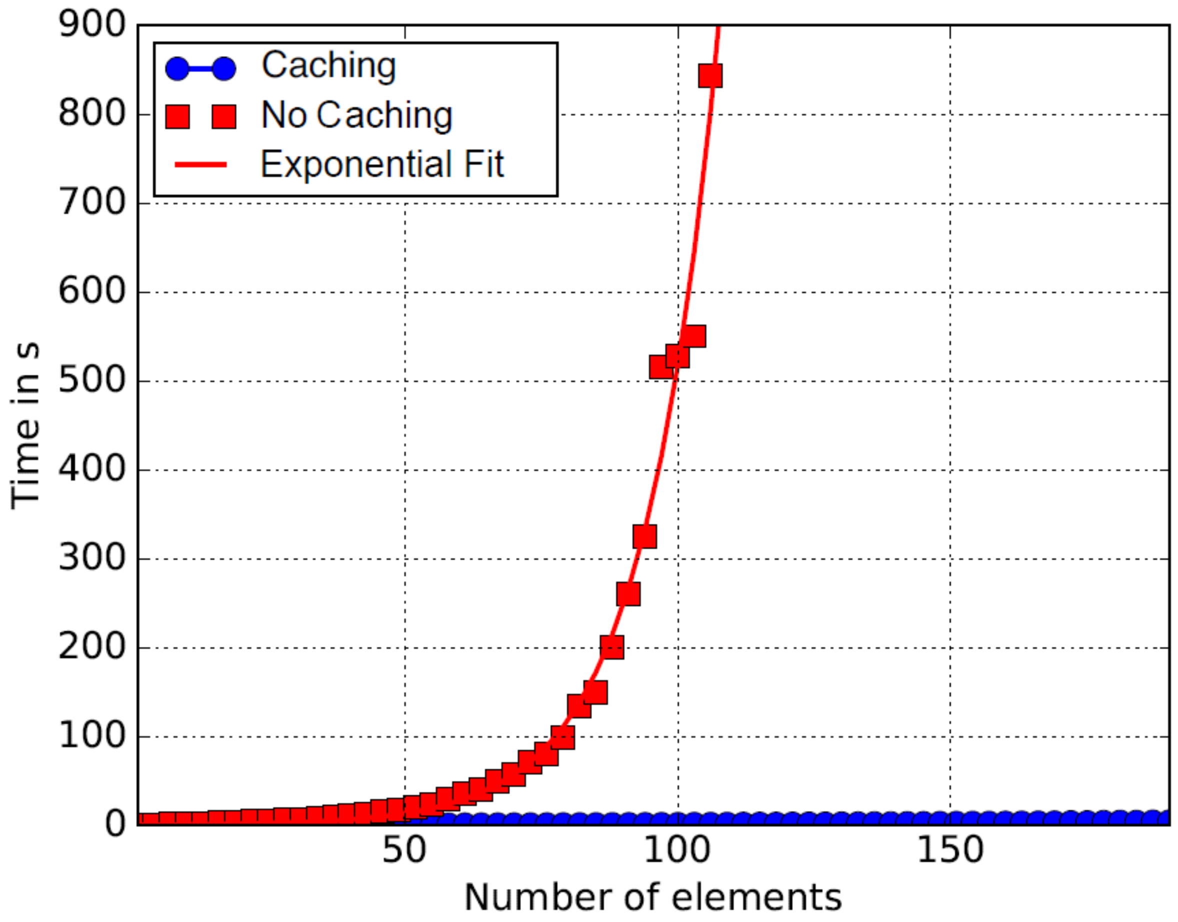
1. Cache Sketch: **staleness barrier** at transaction begin
2. Shorter duration through **cached reads**



[GBR14]

Results

Simulation-Based Abort Analysis



15×

Faster Transactions

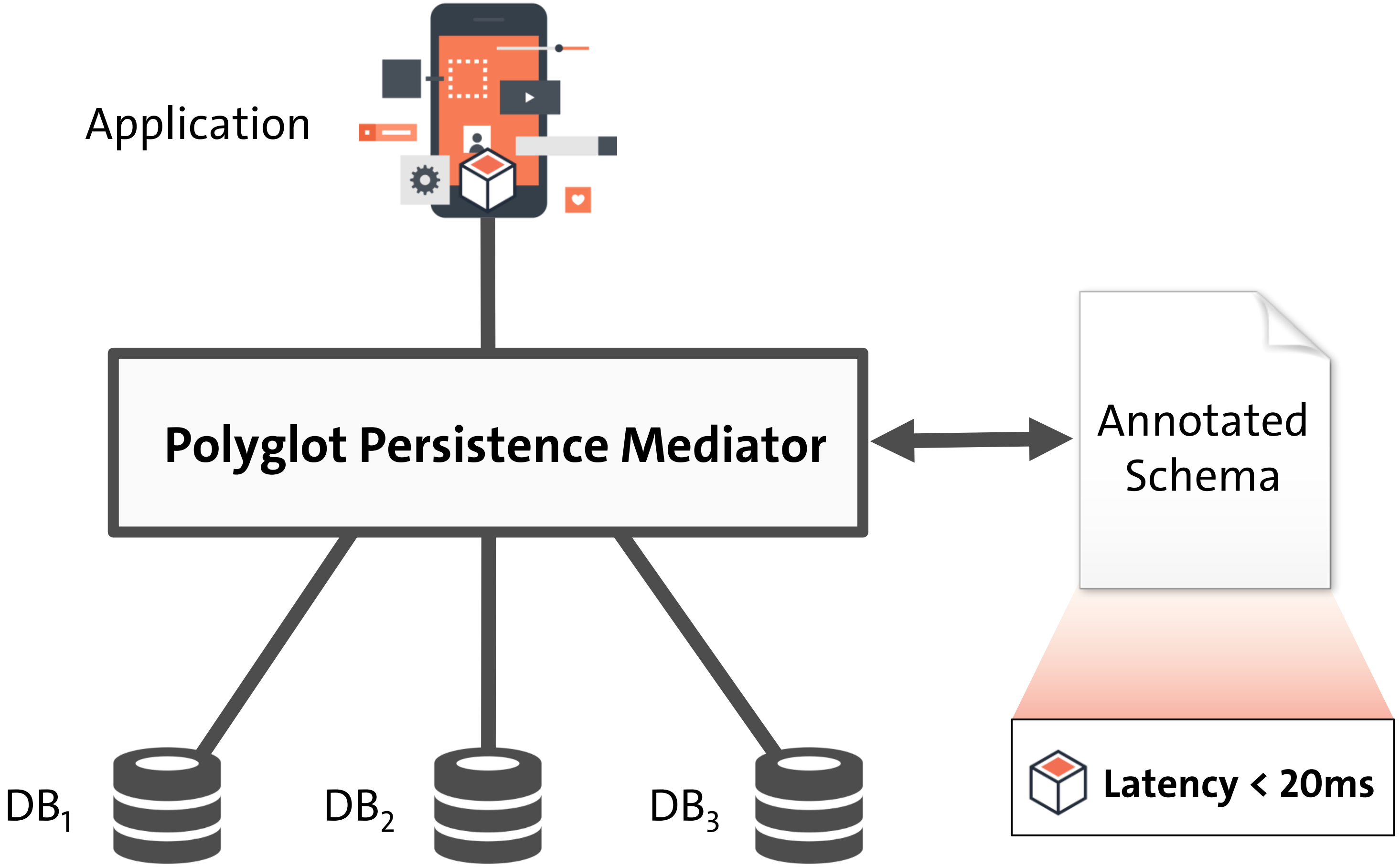
7×

More Objects Before Exceeding 2 Seconds

Can polyglot data management be automated in the future?

Vision

Automated Choice of Databases



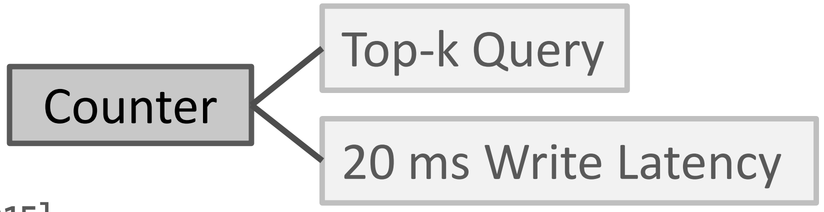
[SGR15]

Towards Automated Polyglot Persistence

Three-Step Process

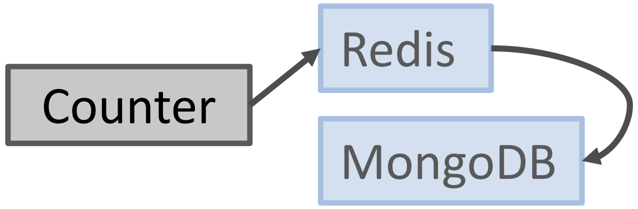
1. Requirements

Requirements specified as **SLA annotations** for schemas (based on *NoSQL Toolbox*)



2. Resolution

Find or provision a suitable combination of databases through **ranking algorithm**



3. Mediation

Mediate **data allocation** and **database operations** between applications and databases

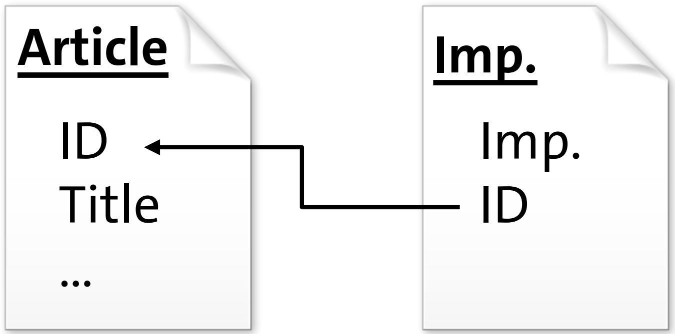
[SGR15]

Evaluation Results

Case Study

Scenario:

News Articles With Impression Counts



Document | Sorted Set

mongoDB | redis

2.1×

Higher Throughput

10 ms

Predictable Write Latency

[SGR15]

Future Work

Three Promising Areas

Proactive SLA Enforcement

- **Monitor** & **predict** database behavior
- **Action**: change routing, live migration, polyglot scaling

Reinforcement Learning of Caching Decisions

- Learn best **TTLs** for any workload
- Applications define **goals**

Polyglot Transaction Processing

- Optimal choice of **concurrency** & **commit protocol** – across DBs

[SKE+18, SG16]

What are the main contributions?

Publications (1/4)

[SKE+18] Michael Schaarschmidt, Alexander Kuhnle, Ben Ellis, Kai Fricke, Felix Gessert, and Eiko Yoneki. LIFT: Reinforcement Learning in Computer Systems by Learning From Demonstrations. arXiv preprint arXiv:1808.07903 (**under submission**), 2018.

[WRG18] Wolfram Wingerath, Norbert Ritter, and Felix Gessert. Real-Time & Stream Data Management: Push-Based Data in Research & Practice. **Springer, book** to be published in late 2018.

[WGW+18] Wolfram Wingerath, Felix Gessert, Erik Witt, Steffen Friedrich, and Norbert Ritter. Real-time Data Management for Big Data. In Proceedings of the 21th International Conference on Extending Database Technology, **EDBT** 2018, Vienna, Austria, March 26-29, 2018. OpenProceedings.org, 2018.

[GSW+17] Felix Gessert, Michael Schaarschmidt, Wolfram Wingerath, Erik Witt, Eiko Yoneki, and Norbert Ritter. Quaestor: Query Web Caching for Database- as-a-Service Providers. Proceedings of the **VLDB** Endowment, 2017.

[GWR17] Felix Gessert, Wolfram Wingerath, and Norbert Ritter. Scalable Data Management: An In-Depth Tutorial on Nosql Data Stores. In **BTW** (Workshops), volume P-266 of LNI, pages 399–402. GI, 2017.

Publications (2/4)

[WGF+17]	Wolfram Wingerath, Felix Gessert, Steffen Friedrich, Erik Witt, and Norbert Ritter. The Case for Change Notifications in Pull-Based Databases. In Datenbanksysteme für Business, Technologie und Web (**BTW** 2017) - Workshopband, 2.-3. März 2017, Stuttgart, Germany, 2017.
[GR17]	Felix Gessert and Norbert Ritter. **SCDM** 2017 - Vorwort. In BTW (Workshops), volume P-266 of LNI, pages 211–213. GI, 2017.
[Ges17]	Felix Gessert. Lessons Learned Building a Backend-as-a-Service. **Baqend Tech Blog**, May 2017. (Accessed on 08/11/2017).
[GWFR16]	Felix Gessert, Wolfram Wingerath, Steffen Friedrich, and Norbert Ritter. NoSQL Database Systems: A Survey and Decision Guidance. **Computer Science - Research and Development**, November 2016.
[GR16]	Felix Gessert and Norbert Ritter. Scalable Data Management: NoSQL Data Stores in Research and Practice. In 32nd IEEE International Conference on Data Engineering, **ICDE**, 2016.
[SG16]	Michael Schaarschmidt and Felix Gessert. Learning Runtime Parameters in Computer Systems with Delayed Experience Injection. In **Deep Reinforcement Learning Workshop, NIPS**, 2016.

Publications (3/4)

[WGFR16]	Wolfram Wingerath, Felix Gessert, Steffen Friedrich, and Norbert Ritter. Real- Time Stream Processing for Big Data. **it - Information Technology**, 58(4), January 2016.
[GSW+15]	Felix Gessert, Michael Schaarschmidt, Wolfram Wingerath, Steffen Friedrich, and Norbert Ritter. The Cache Sketch: Revisiting Expiration-based Caching in the Age of Cloud Data Management. In Datenbanksysteme für Business, Technologie und Web (**BTW**), 16. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme". GI, 2015.
[GR15a]	Felix Gessert and Norbert Ritter. Polyglot Persistence. **Datenbank-Spektrum**, 15(3):229–233, November 2015.
[GR15b]	Felix Gessert and Norbert Ritter. Skalierbare NoSQL- und Cloud-Datenbanken in Forschung und Praxis. In Datenbanksysteme für Business, Technologie und Web (**BTW** 2015) - Workshopband, 2.-3. März 2015, Hamburg, Germany, pages 271–274, 2015.
[Ges15]	Felix Gessert. Low Latency Cloud Data Management through Consistent Caching and Polyglot Persistence. In Proceedings of the 9th Advanced Summer School on Service Oriented Computing, **SummerSOC**, 2015.
[SGR15]	Michael Schaarschmidt, Felix Gessert, and Norbert Ritter. Towards Automated Polyglot Persistence. In Datenbanksysteme für Business, Technologie und Web (**BTW**), 16. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme", 2015.

Publications (4/4)

[WFGR15] Wolfram Wingerath, Steffen Friedrich, Felix Gessert, and Norbert Ritter. Who Watches the Watchmen? On the Lack of Validation in NoSQL Benchmarking. In Datenbanksysteme für Business, Technologie und Web (**BTW**), 16. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme", 2015.

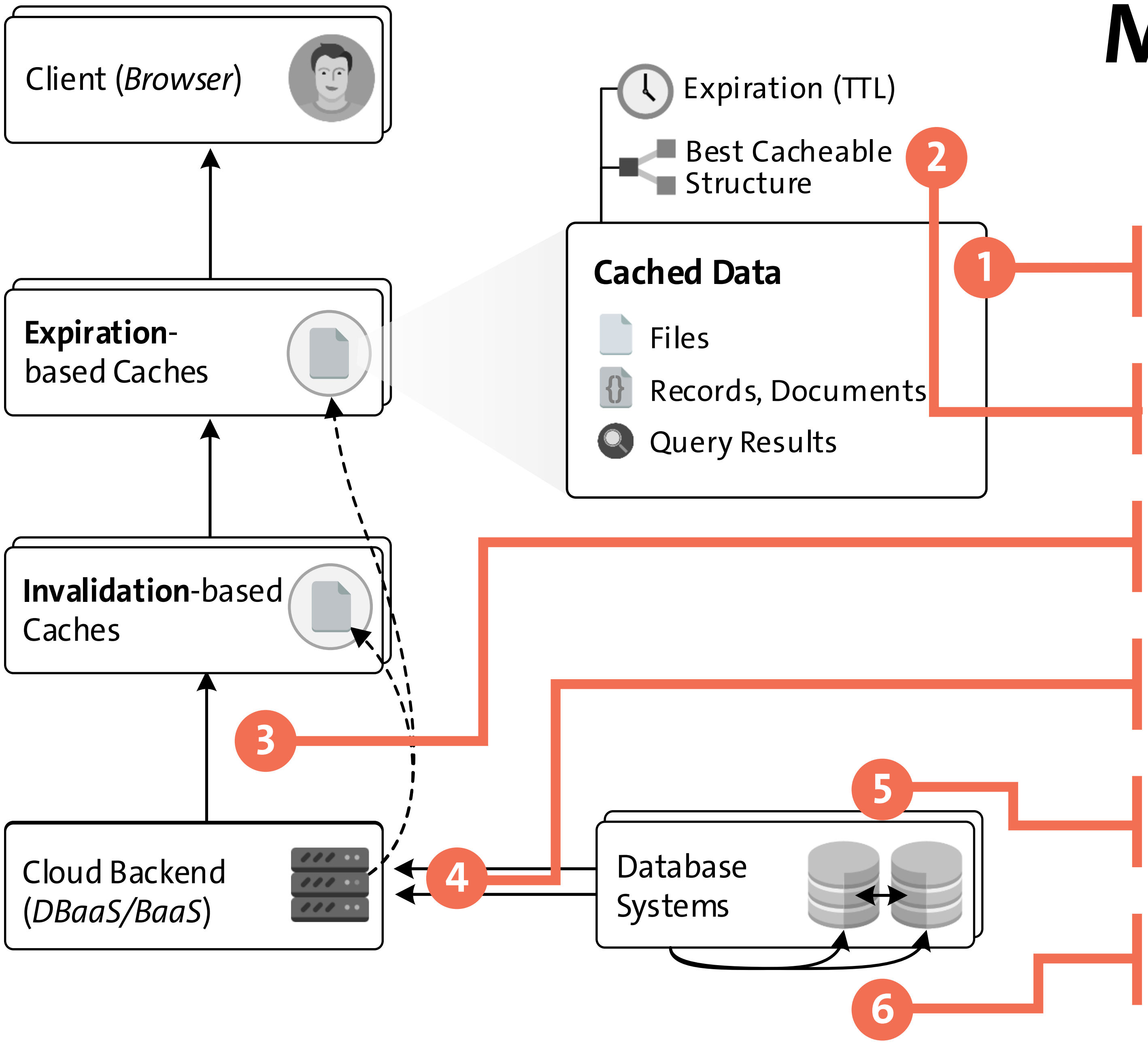
[GBR14] Felix Gessert, Florian Bücklers, and Norbert Ritter. ORESTES: a Scalable Database-as-a-Service Architecture for Low Latency. In **CloudDB**, Data Engineering Workshops (ICDEW), pages 215–222. IEEE, 2014.

[GFW+14] Felix Gessert, Steffen Friedrich, Wolfram Wingerath, Michael Schaarschmidt, and Norbert Ritter. Towards a Scalable and Unified REST API for Cloud Data Stores. In 44. Jahrestagung der Gesellschaft für Informatik, Informatik 2014, **Big Data - Komplexität meistern**, 22.-26. September 2014 in Stuttgart, Deutschland, volume 232 of LNI, pages 723–734. GI, 2014.

[FWGR14] Steffen Friedrich, Wolfram Wingerath, Felix Gessert, and Norbert Ritter. NoSQL OLTP Benchmarking: A Survey. In 44. Jahrestagung der Gesellschaft für Informatik, Informatik 2014, **Big Data - Komplexität meistern**, 22.-26. September 2014 in Stuttgart, Deutschland, volume 232 of LNI, pages 693–704. GI, 2014.

[GB13] Felix Gessert and Florian Bücklers. ORESTES: ein System für horizontal skalierbaren Zugriff auf Cloud-Datenbanken. In **Informatiktage**. GI, March 2013.

Main Contributions
Summary



Cache Coherence for Files, Records & Queries

TTL Estimation & Result Structure

Unified Data Management Interface

Database-Independent **DBaaS** and **BaaS**

Scalable Cache-Aware **Transactions**

Polylgot Persistence Mediation