

**facebook**

# Challenges to Adopting Stronger Consistency at Scale

HotOS 2015  
19 May

Phillipe Ajoux, *Nathan Bronson*, Sanjeev Kumar, Wyatt Lloyd, Kaushik Veeraraghavan

# A user-visible inconsistency

The diagram illustrates a user-visible inconsistency in a social media interface. It shows two overlapping views of the interface:

- Left View (Active):** Shows the top navigation bar with icons for Home, Friends, Messages, Notifications (with a red badge showing '2'), Search, and Menu. Below the navigation bar are tabs for Status, Photo, and Check In. A post from Yosemite National Park is visible.
- Right View (Error State):** Shows the same navigation bar, but the main content area displays an error message: "The page you requested cannot be displayed right now. It may be temporarily unavailable, the link you clicked on may be broken or expired, or you may not have permission to view this page." Below the error message is a "Back to home" link.

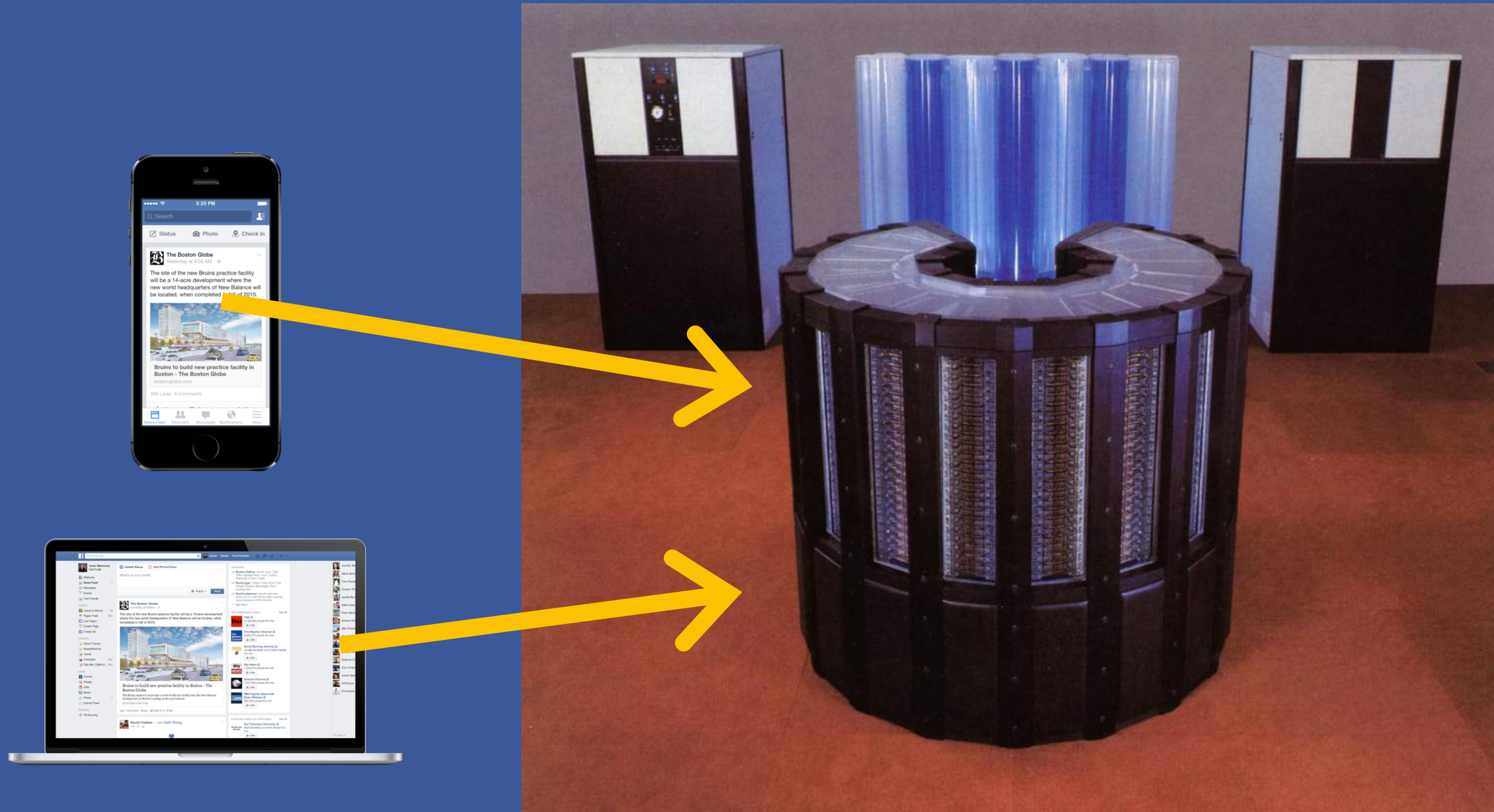
A red arrow points from the Notifications icon in the top navigation bar of the left view to a notification card. The notification card displays:

**Notifications**

Janet Wiener posted in **Consistency Investigations**.  
a few seconds ago

A second red arrow points from the notification card to a large red text block that reads:

**NOT YET ARRIVED**



People should think FB runs on one computer

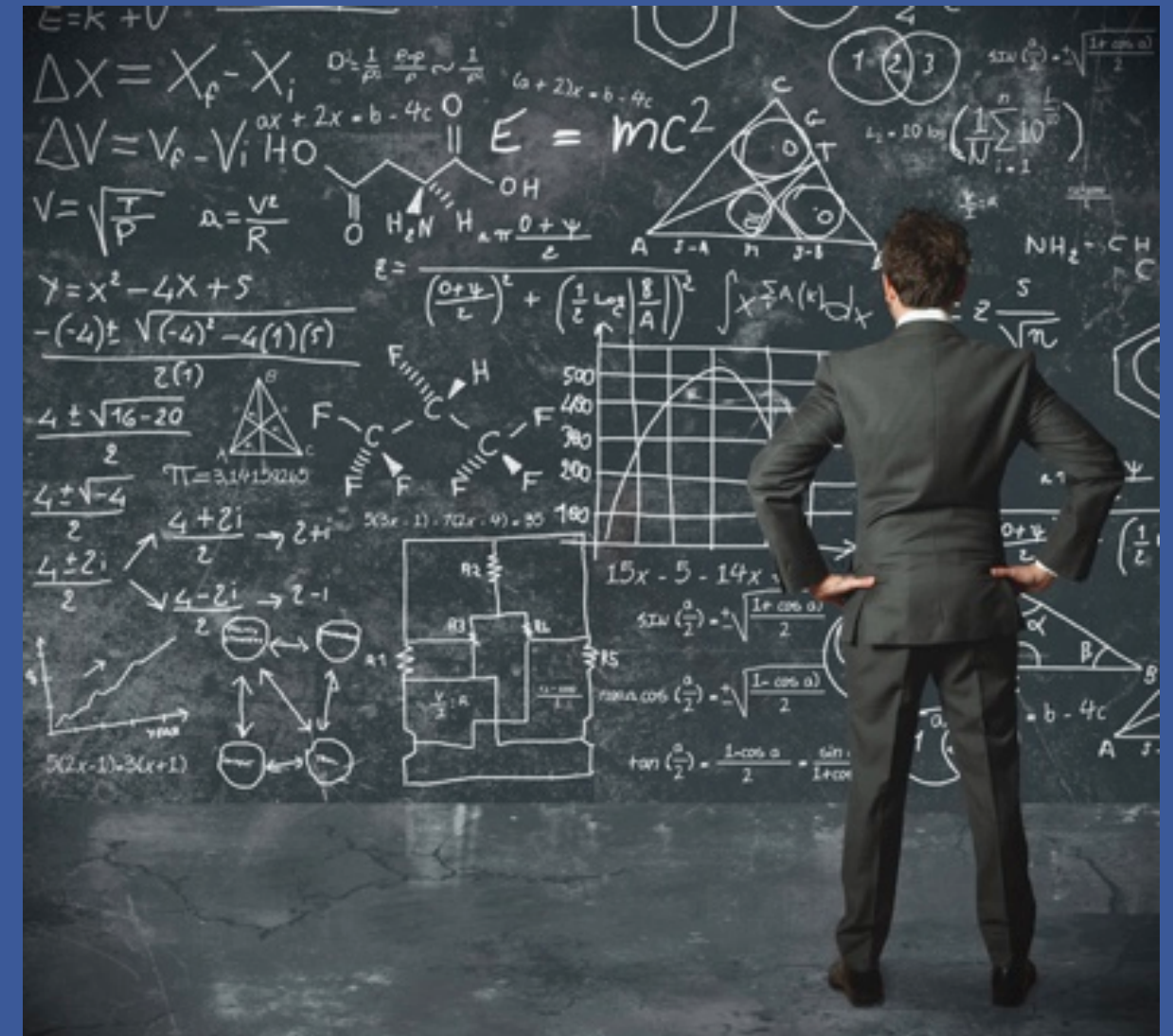


# How can we fix inconsistencies?

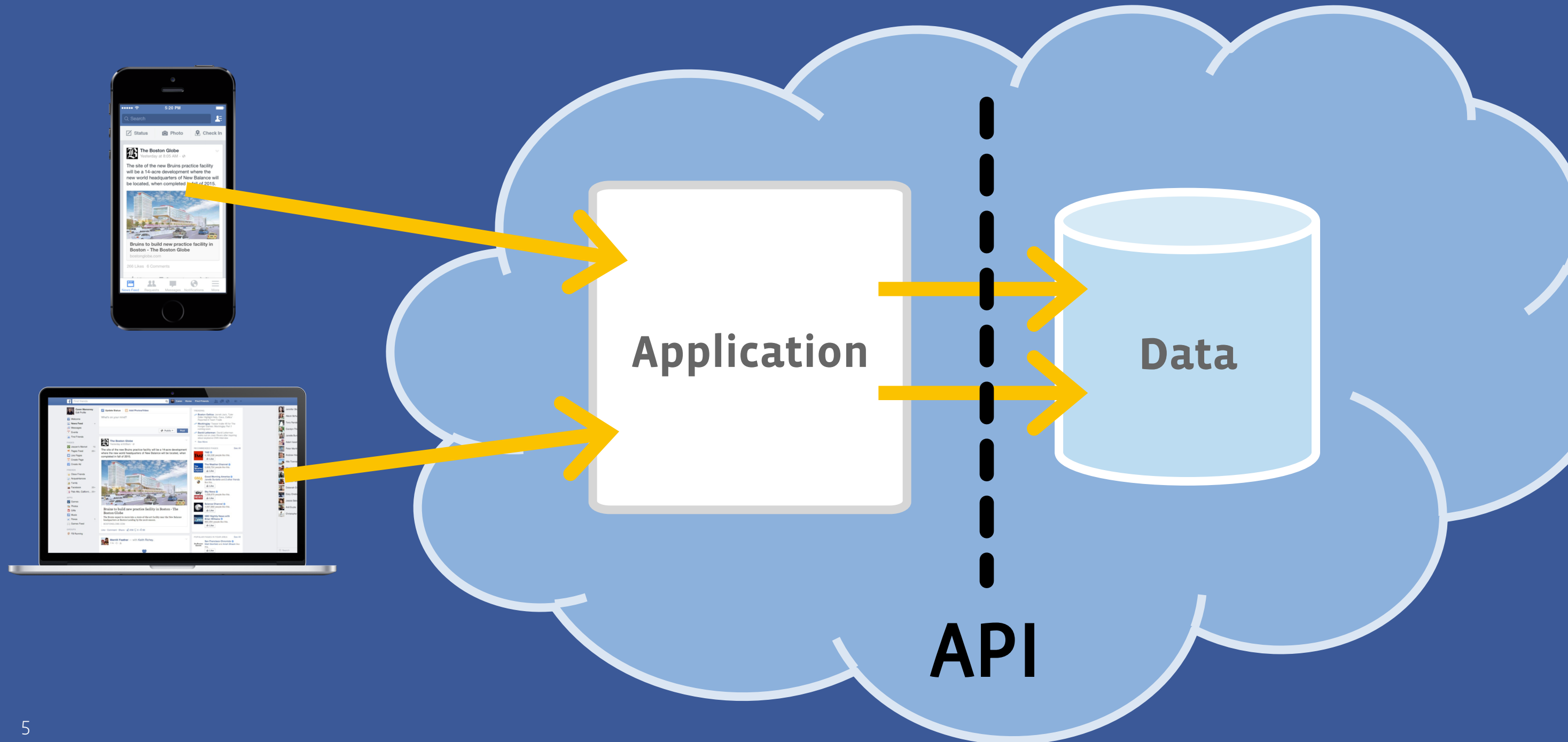


(or both)

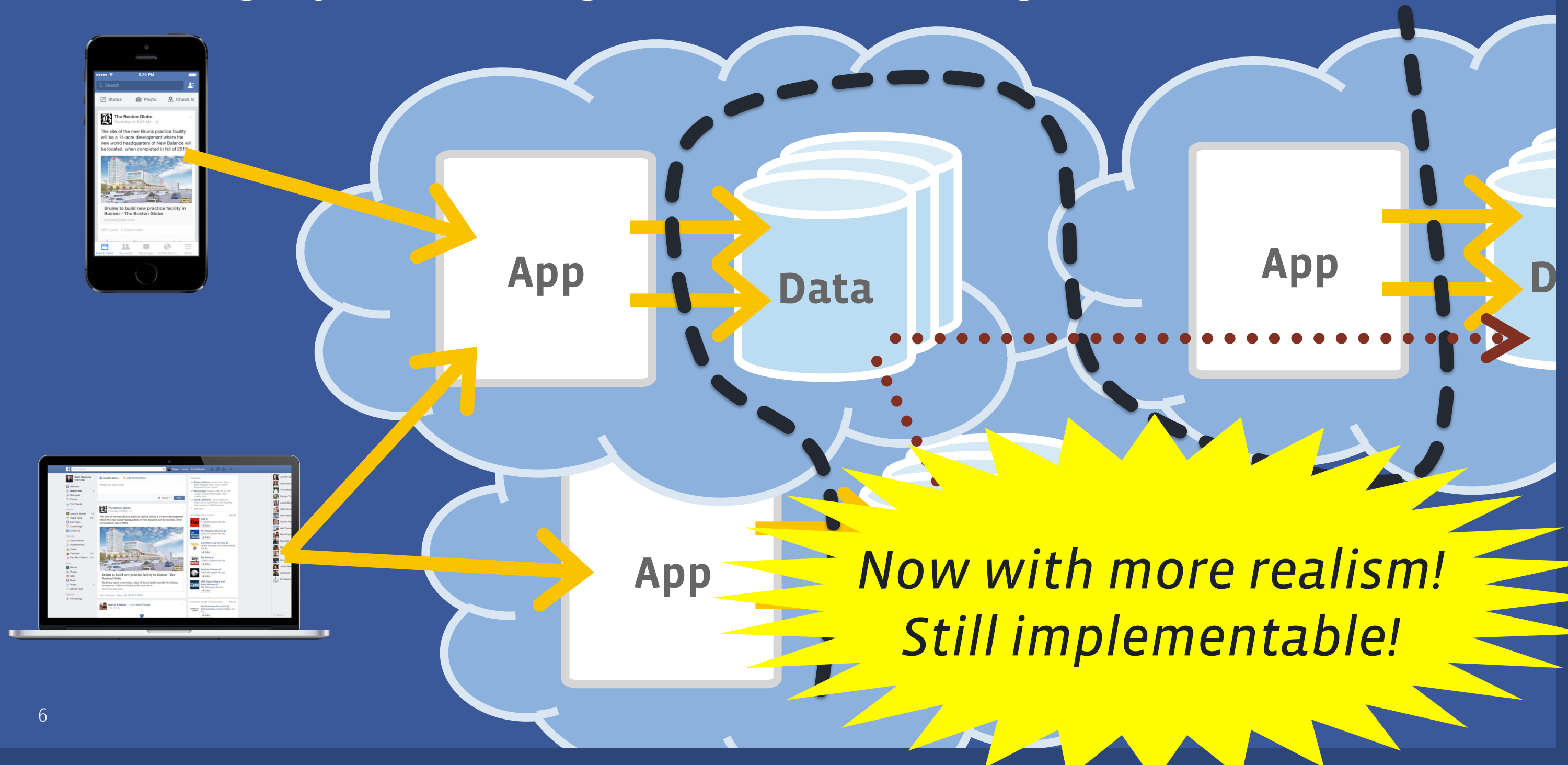
or



# Building a site (software engineering)





# Scaling by sharding and replicating





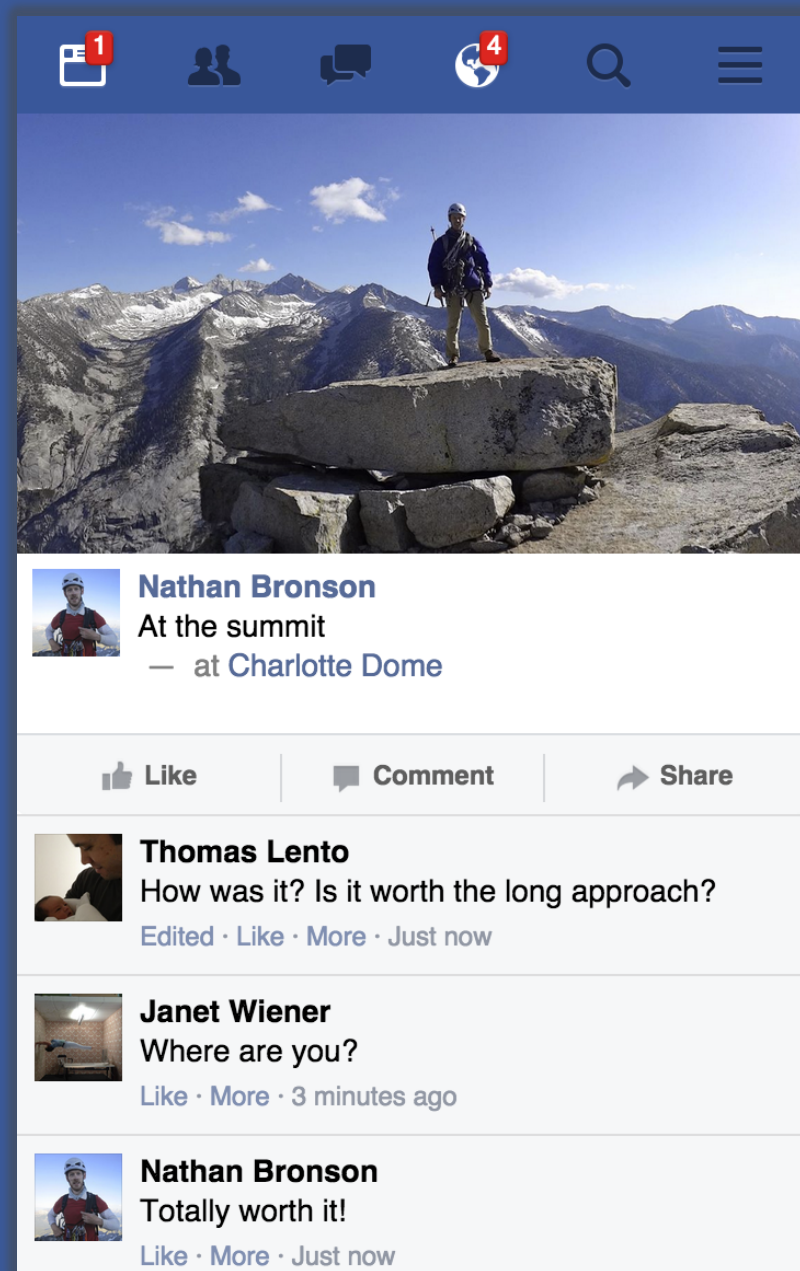
# Why not strengthen the data store?

- Will it meet our requirements?
  - Outlier sensitivity – latency & availability
  - Pathological data access patterns
  - Low average latency needed for efficiency

see the paper
- What about data copies?
  - Lots of systems store ad-hoc data copies
  - Those systems are loosely coupled

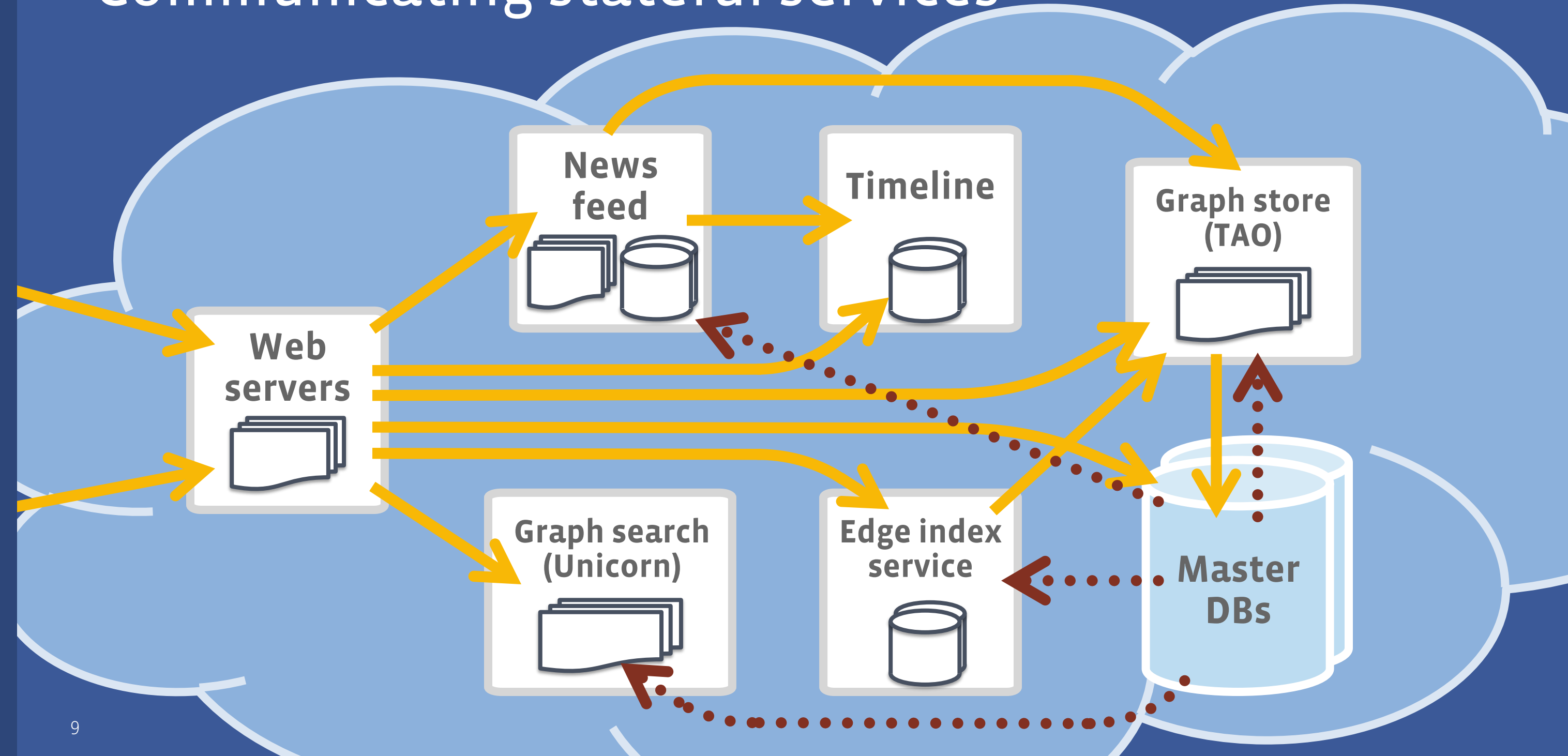
more slides

# Social graph = types nodes + edges



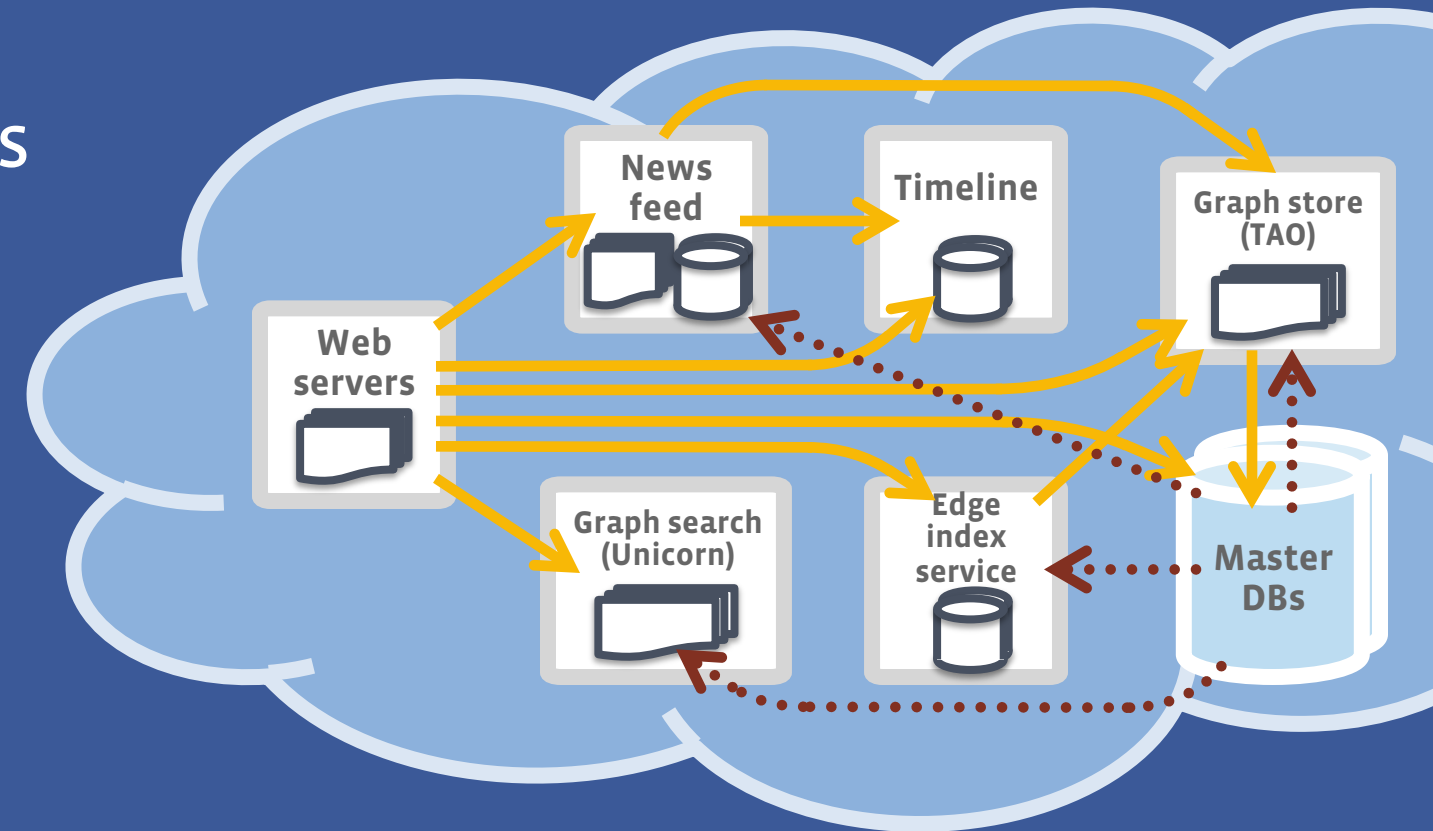


# Communicating stateful services



# Consistency glue challenges

- Multiple copies of the data
  - Copies are materialized query results
- Ad-hoc query languages
  - Service building block is RPC call
  - Didn't design API as a language
- Loose coupling
  - Separate teams, different languages
  - Different deployment schedules, reliability goals



# Techniques from tightly-coupled databases

## *Locality*

- ✓ Denormalization
- ✓ Caching
- ✓ Materialized join views
- ✓ Secondary indices
- ✓ Covering indices
- ✓ Partial indices
- ✓ Stored procedures

## *Consistency glue*

- ✗ Unified analyzable query language
- ✗ Two phase locking
- ✗ Range locks, table locks
- ✗ Predicate locks
- ✗ Totally-ordered sequence numbers
- ✗ Foreign key constraints
- ✗ Linear durable log



# What do I hope?

- Add tools for locality optimization to existing systems
- Make the problem topology more realistic
- Are there tools for end-to-end consistency?

**facebook**