99.99% Uptime at 175 TB of Data Per Day

Ben John

CTO

bjohn@appnexus.com

Matt Moresco

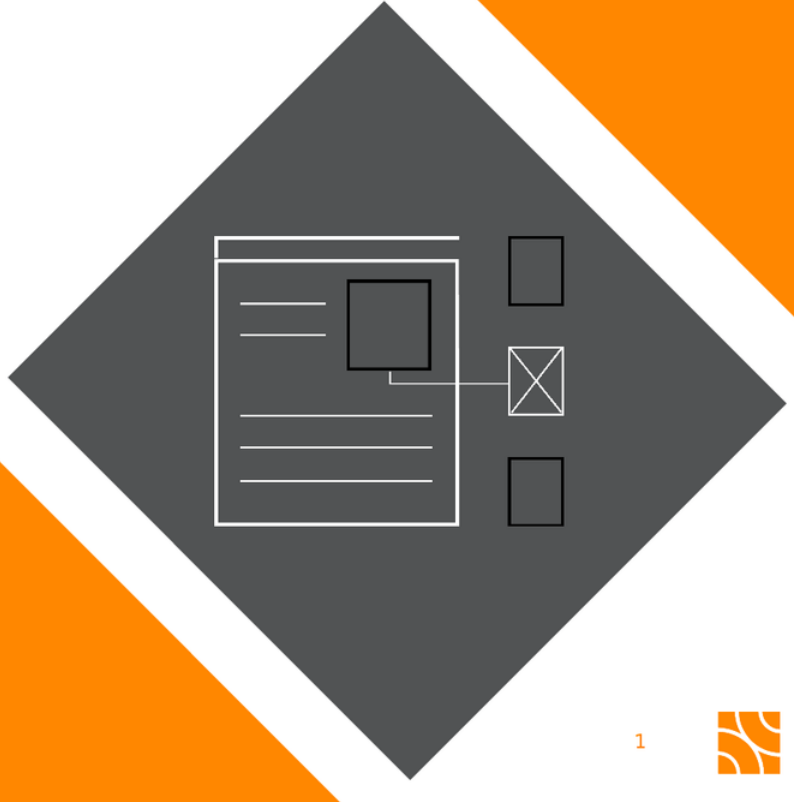
Software Engineer, Real Time Platform

mmoresco@appnexus.com

AppNexus

APPNEXUS

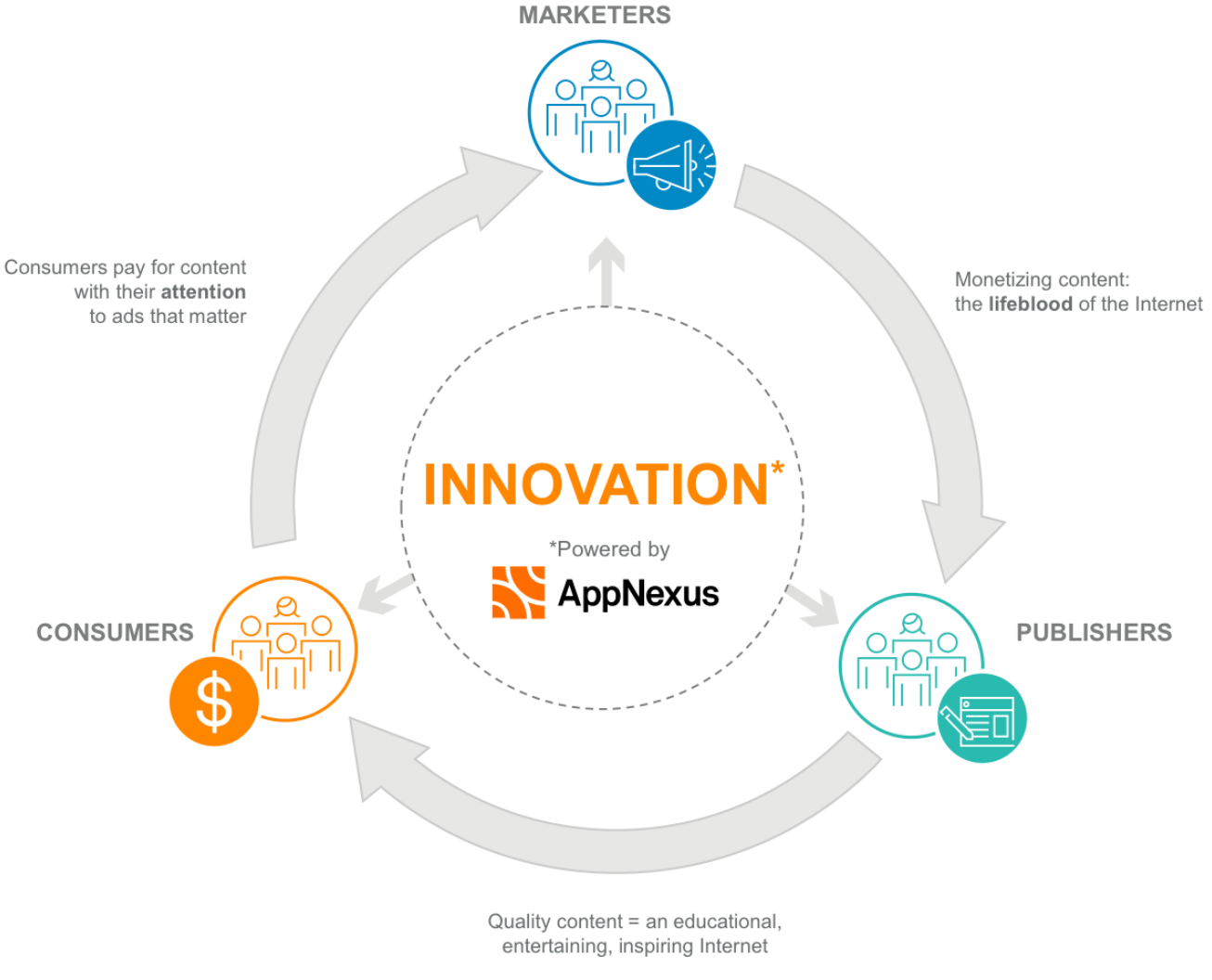
is an internet technology company
that harnesses data and machine learning to power the world's largest digital audience platforms.



What Does A

Better Internet

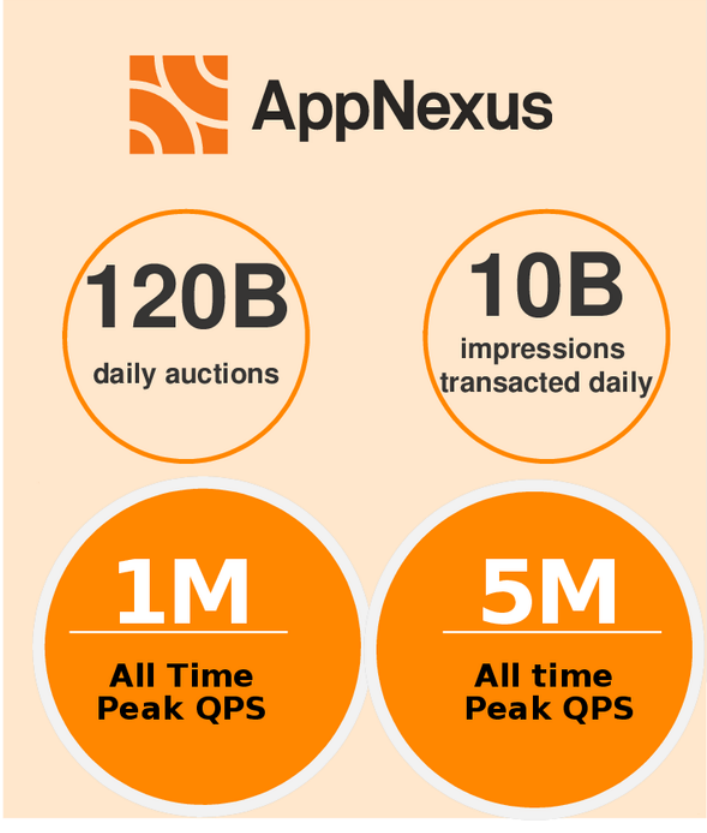
Look Like?

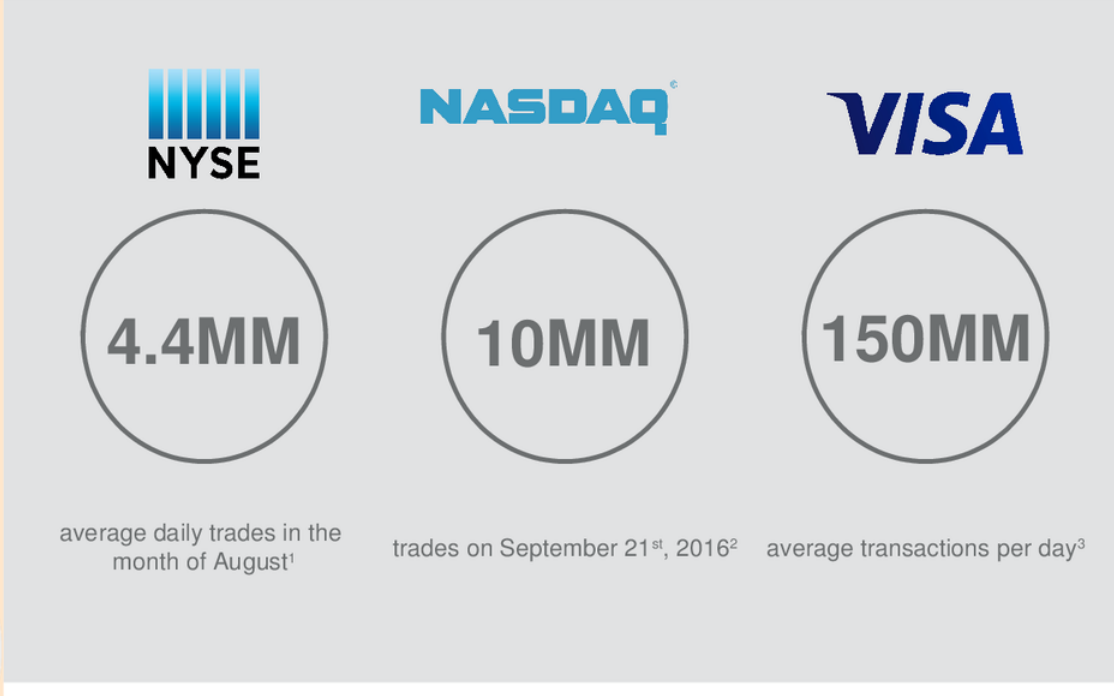


AppNexus

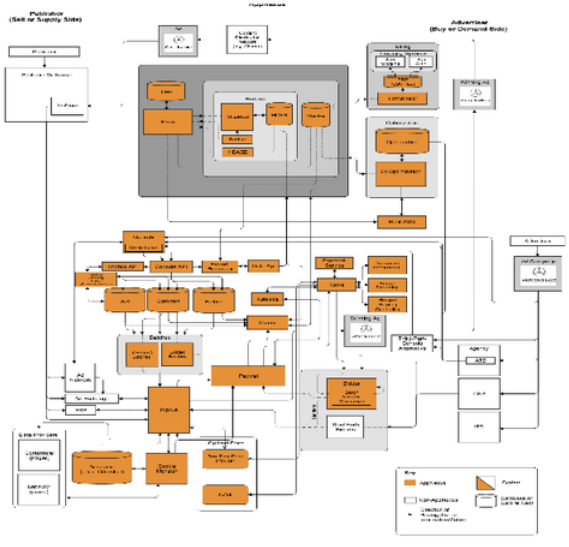
Internet Scale Architecture

Internet Scale





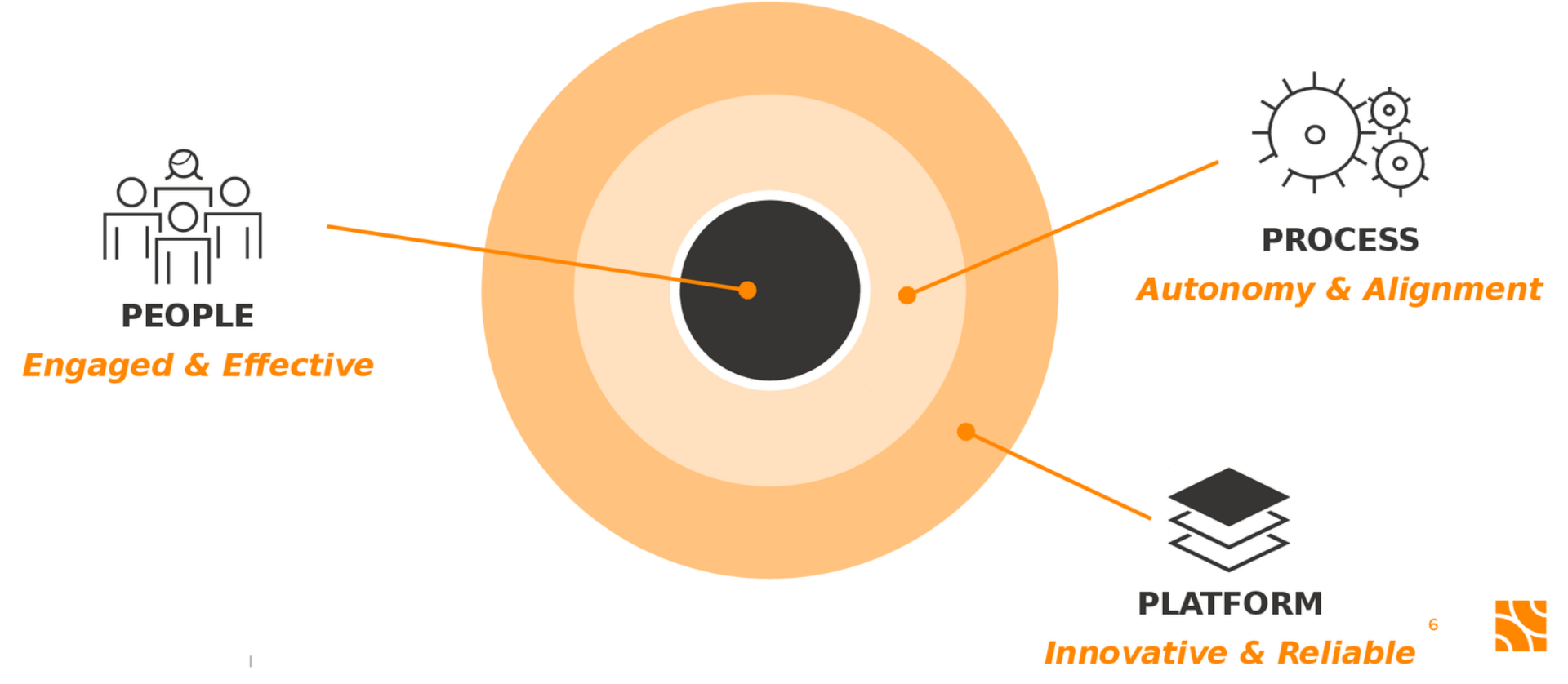
[1] Source: NYSE Market Data
[2] Source: NASDAQ market data
[3] Source: Visa

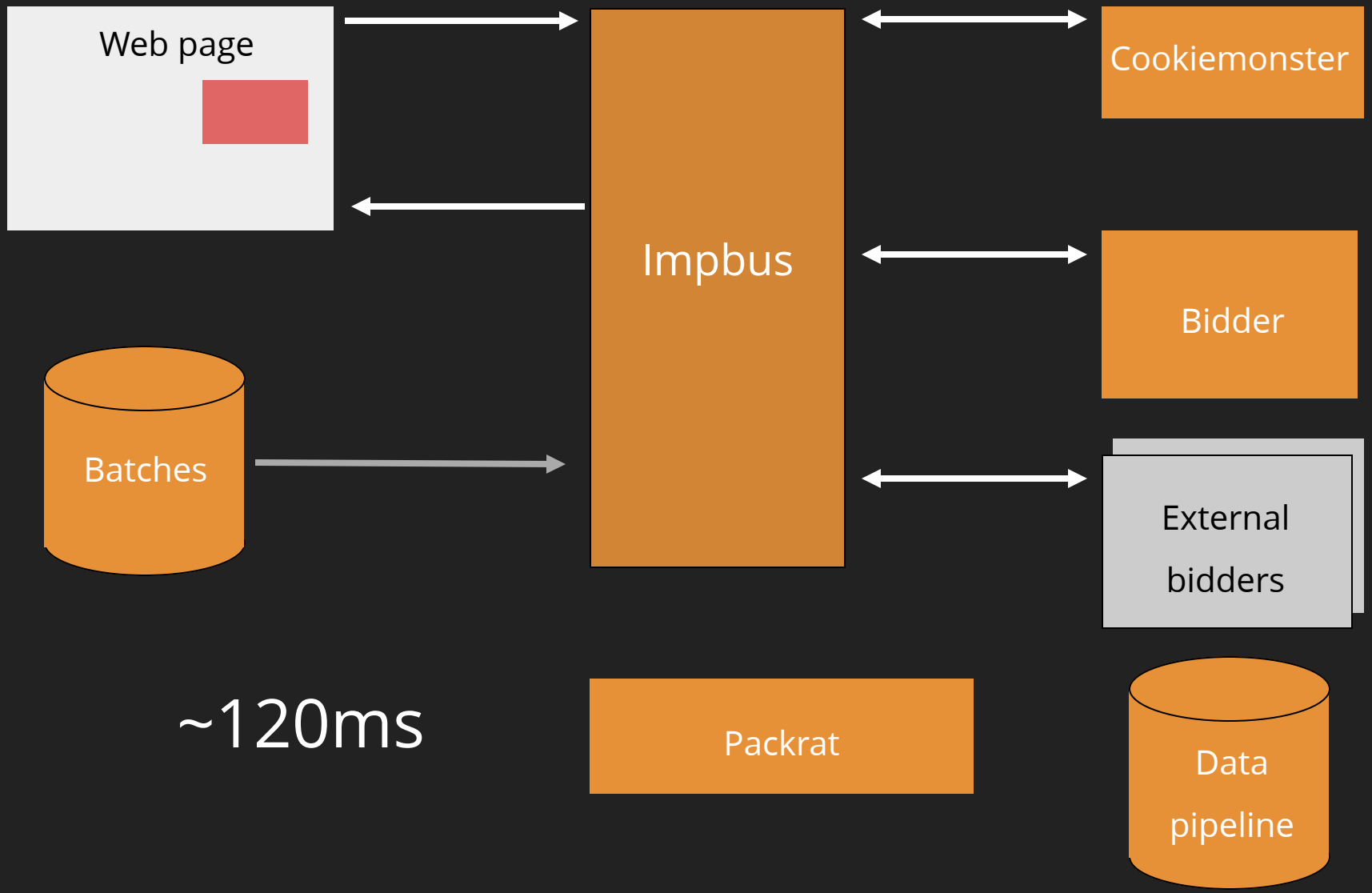


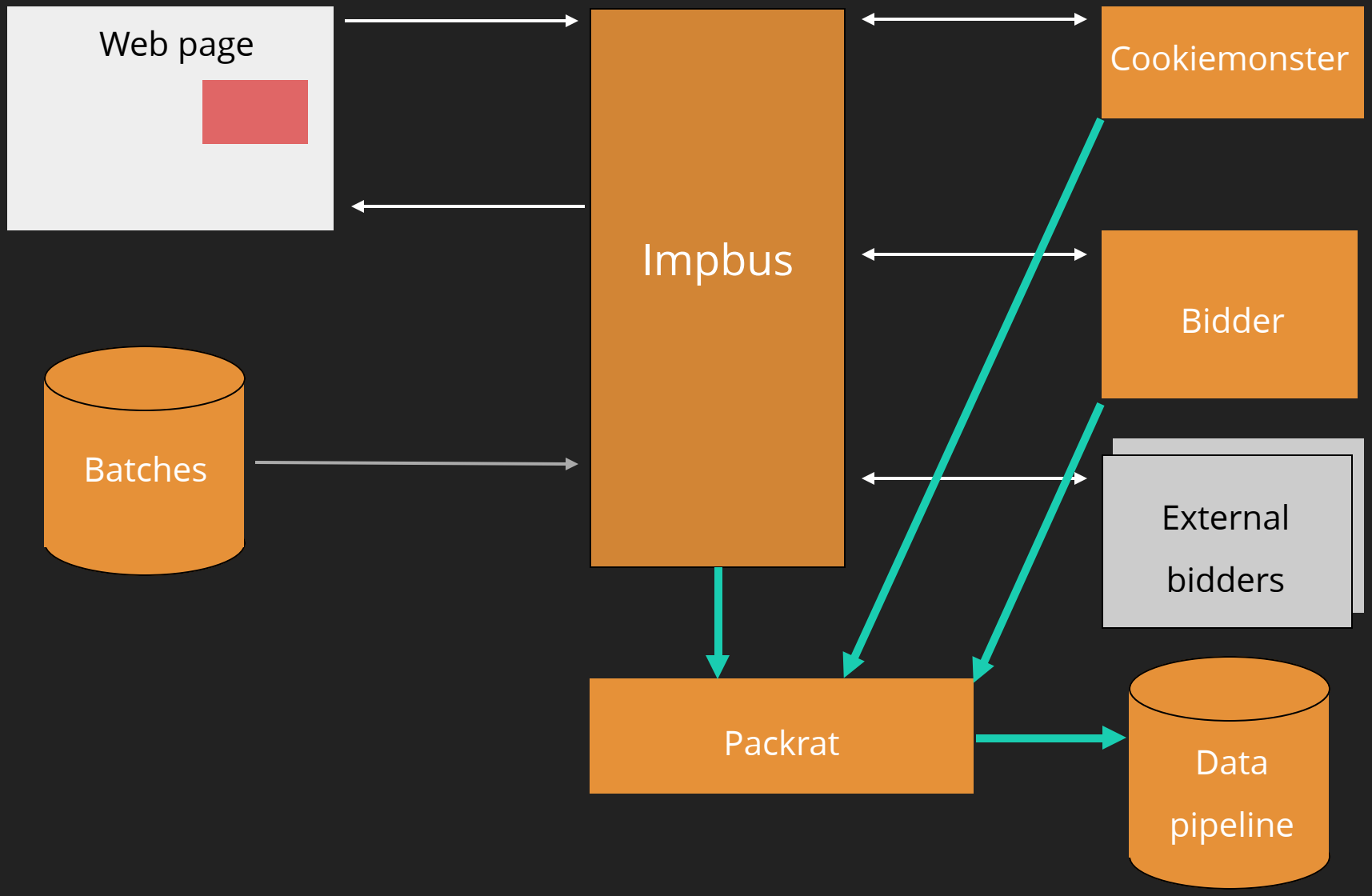
Data processing highlights

- Process 400 trillion audience segments per day
- 5 million records processed concurrently under <10 milliseconds
- Tens of millions of records are processed/Aggregated/priced/algorithms applied in near real-time
- Leverages the best possible data to make the best decision in real-time
- Global scaled to be economical, resilient and expandable

Our Culture


PEOPLE
Engaged & Effective
PROCESS
Autonomy & Alignment
PLATFORM
Innovative & Reliable


Web page
Impbus
Cookiemonster
Bidder
Batches
External bidders
~120ms
Packrat
Data pipeline
AppNexus


Web page
Impbus
Cookiemonster
Bidder
Batches
External bidders
Packrat
Data pipeline

AppNexus

Managing failure

- Prevent it in the first place
  - Unit/Integration tests
  - Canary releases
- When it happens, recover quickly

AppNexus

Ways we fail

- Data distribution unreliability
- C woes
- DDOSing ourselves

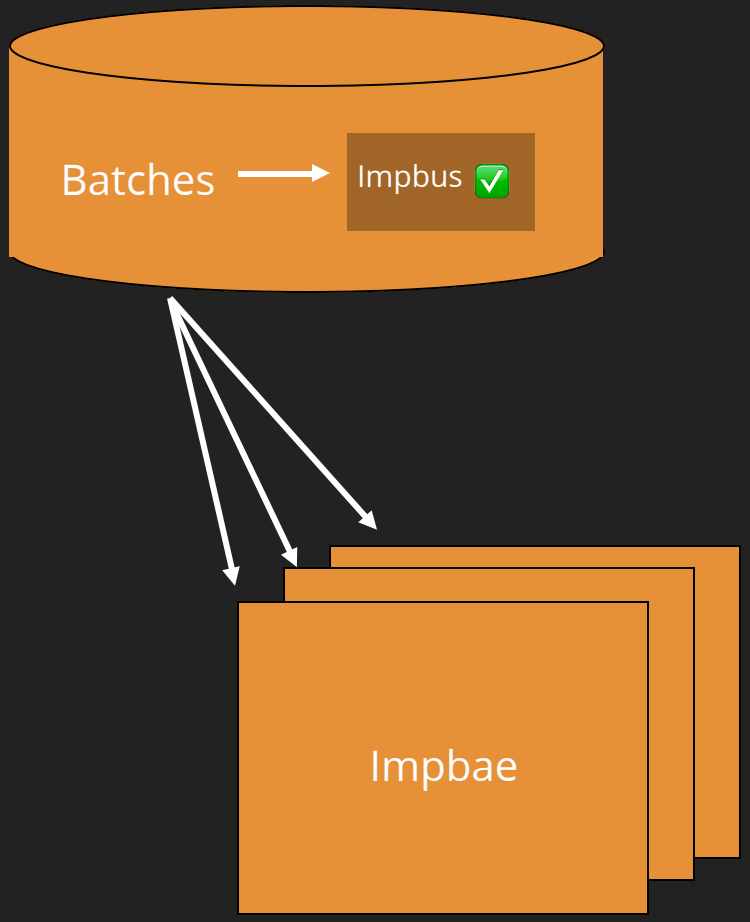
AppNexus

Handling bad data

- **Good news:** our systems deliver object updates to thousands of servers around the world in under two minutes!
- **Bad news:** our systems can deliver crashy data to thousands of servers around the world in under two minutes!

Handling bad data

- **Validation engines:** run a copy of the production app, see if it crashes before distributing data globally
- This can still fail in bad ways:
  - VE version not aligned with production
  - Time-based crashes



AppNexus

Handling bad data

- Feature switches: AN_HOOK
- Roll back time! Prevent distribution past a timestamp

C woes

- No exceptions in C!
- `core_me_maybe`
  - Catch signal, throw out request, return to event loop
  - Flipped off on some instances so we can get a backtrace

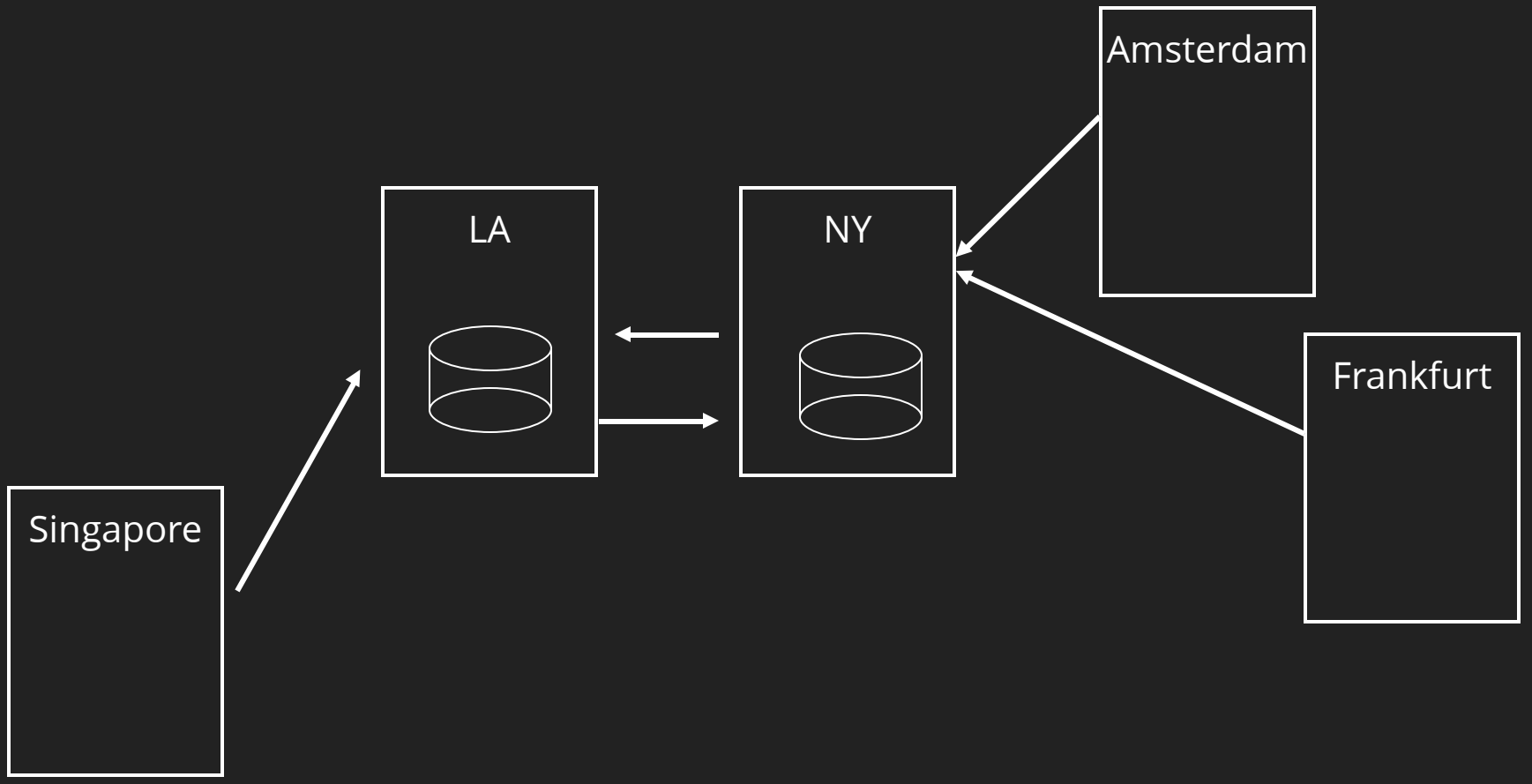
AppNexus

Packrat

- Home grown data router
- Transform, buffer, compress, forward
- Transformations: message format, sharding, sampling, filtering
  - Message formats: protobuf, native x86 format, json
  - (Rolling your own serialization format is *probably* a bad idea)
- High volume disk throughput
- Guaranteed message delivery

AppNexus

Packrat Topology



AppNexus

Packrat protocol

- Group by like type
- HTTP post
- Batch
  - Prefer to send full buffers
  - Fall back to 10s limit
- Snappy compress everything

Packrat failure handling

- Request fails: write it to disk
- `repackd`
  - separate process running on the instance that will continually read failed rows from disk, retry sending them
  - if the retry fails, write to disk, do it all again
- Prone to nasty failure scenarios

AppNexus

Bad data

- If a schema evolution diverges in prod, we will crash
- Because of our failure handling mechanisms, a single bad message can machine gun an entire datacenter

Packrat failure handling

- Because we buffer data in outing requests, we send back a 200 OK before the a message is sent downstream or written to disk
- What about data in memory when packrat crashes?

AppNexus

Packrat failure handling

- Write-ahead log: write every (compressed) incoming request to disk for a 5 minute window
- On startup, replay all traffic (because we don't care about duplicates)

Lessons learned

- If you're going to crash, do everything you can to limit its scope
- Use every possible feature of your environment to your advantage
- Have clear points of responsibility handoff
- Find a way to replicate prod, even if it means testing in prod

AppNexus

System Metrics

Day | Week | **Month**

Impbus Uptime

99.995%

AppNexus