Petabytes and Nanoseconds

Distributed Data Storage and the CAP Theorem

FIN talk

Robert Greiner
Nathan Murray

August 21, 2014

PARIVEDA SOLUTIONS

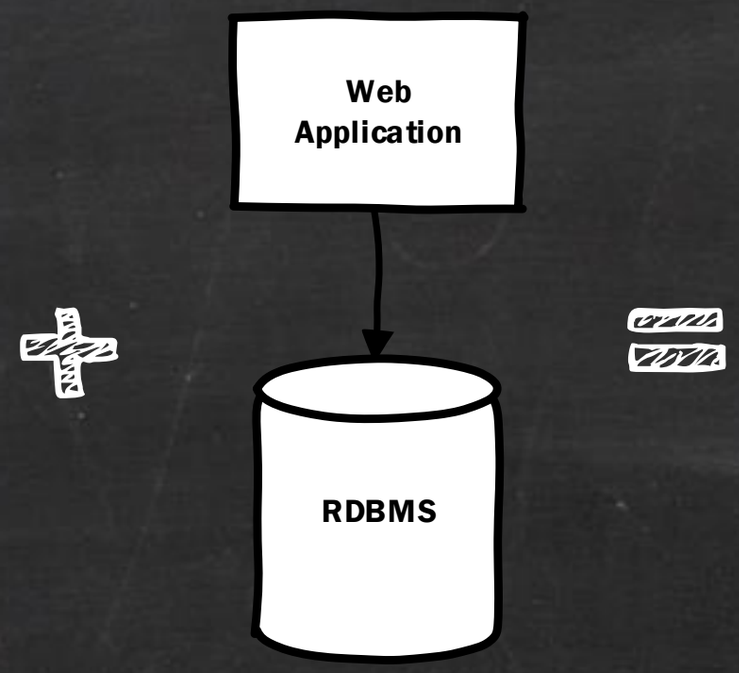
All high frequency trading servers are connected to the NASDAQ network with the same length of cable, so that no party has a speed advantage

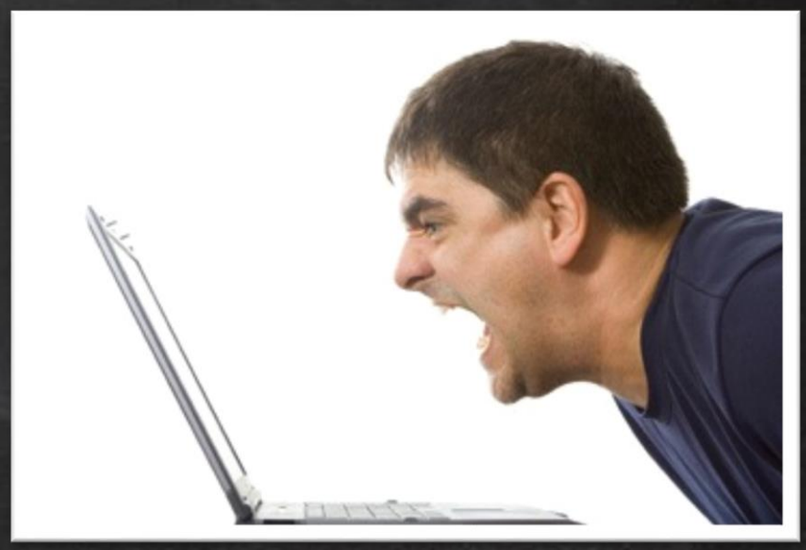
CHAPTER 1

The Problems

Your phone can add two numbers in the same time it takes light to travel one foot

A Common Scenario





The Solution: Scale All the Things!!1

Why should we scale?

Throughput

Latency

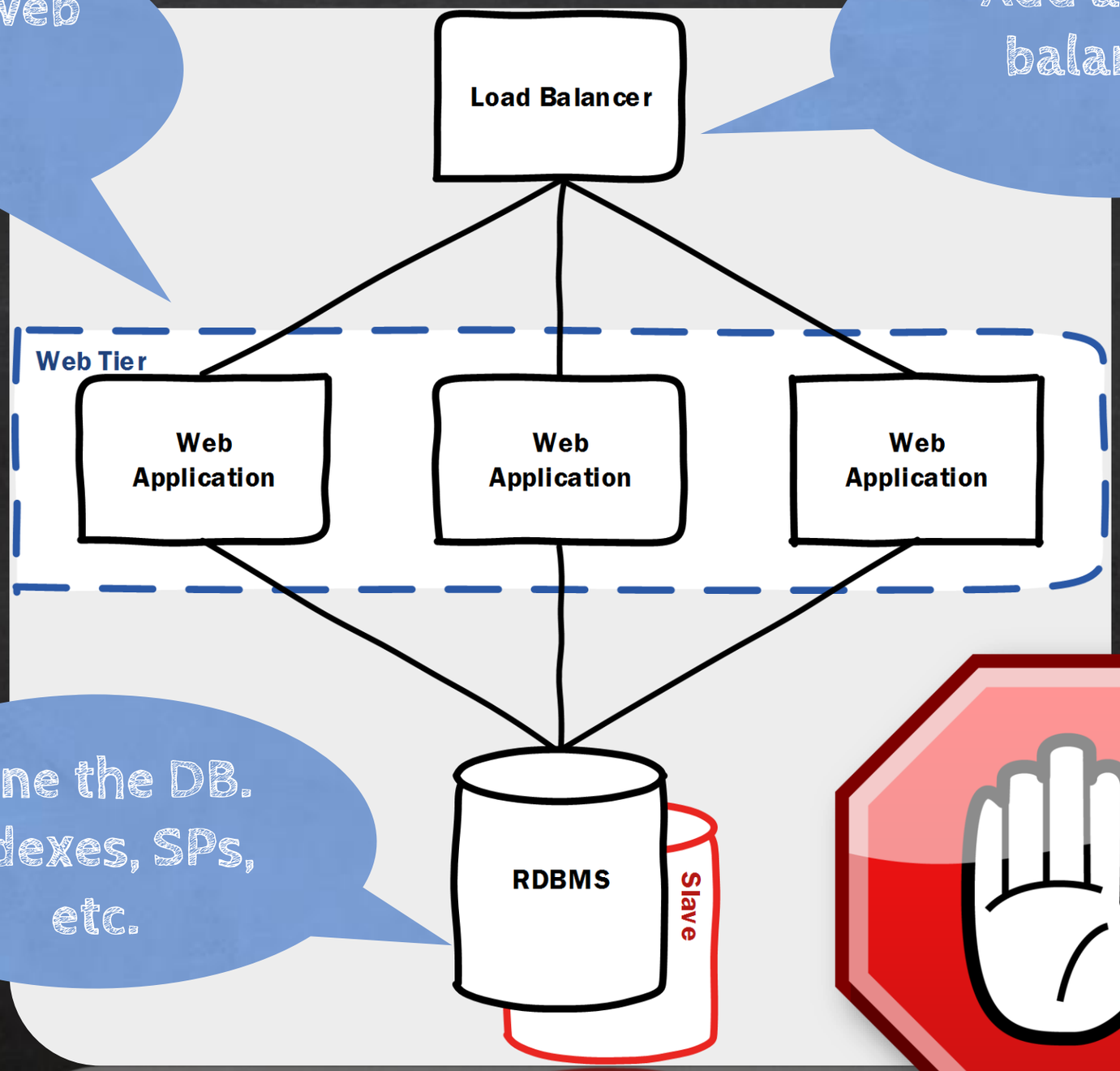
Storage

Reliability

The Solution?

Add more web servers

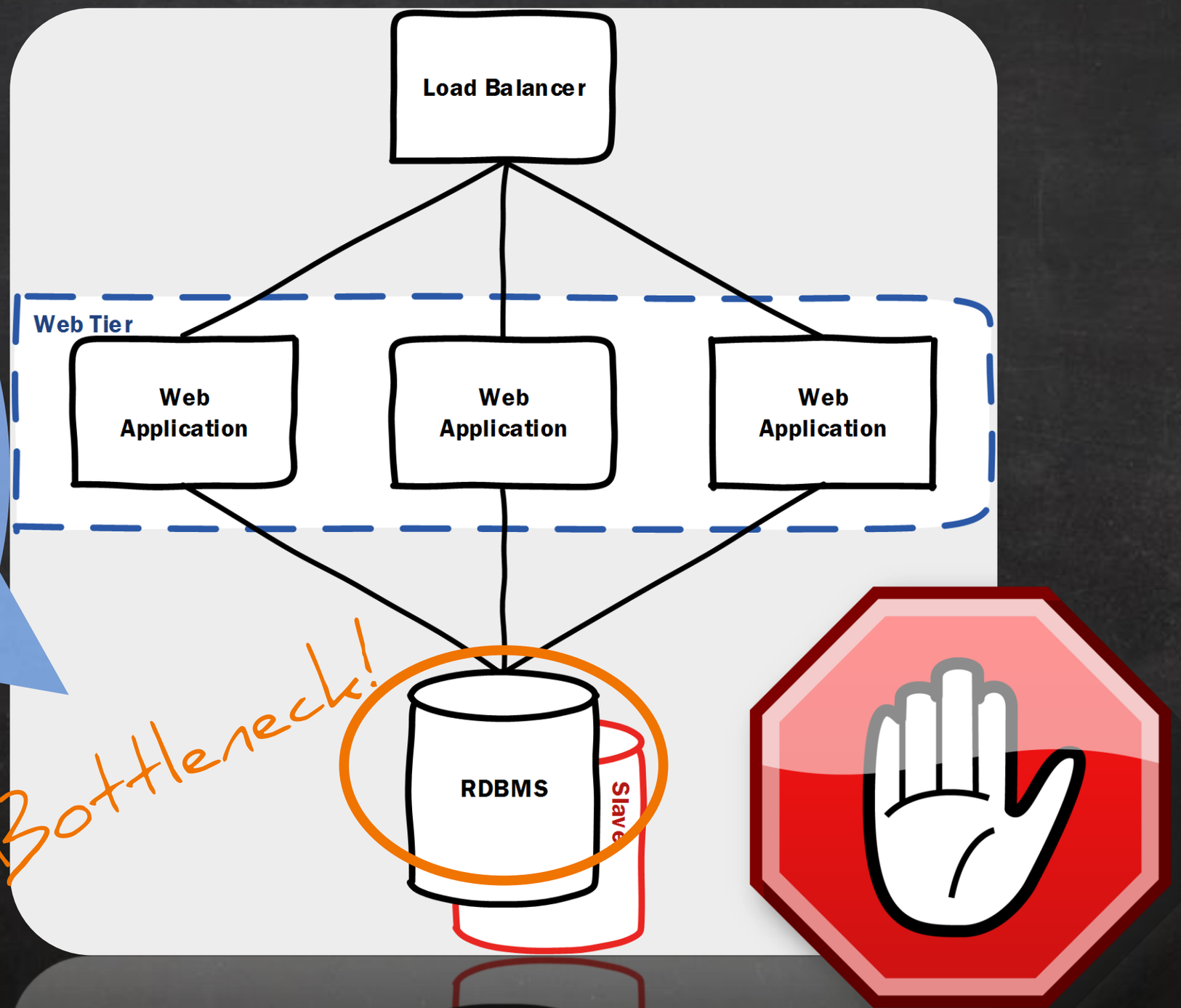
Add a load balancer



Tune the DB. Indexes, SPs, etc.

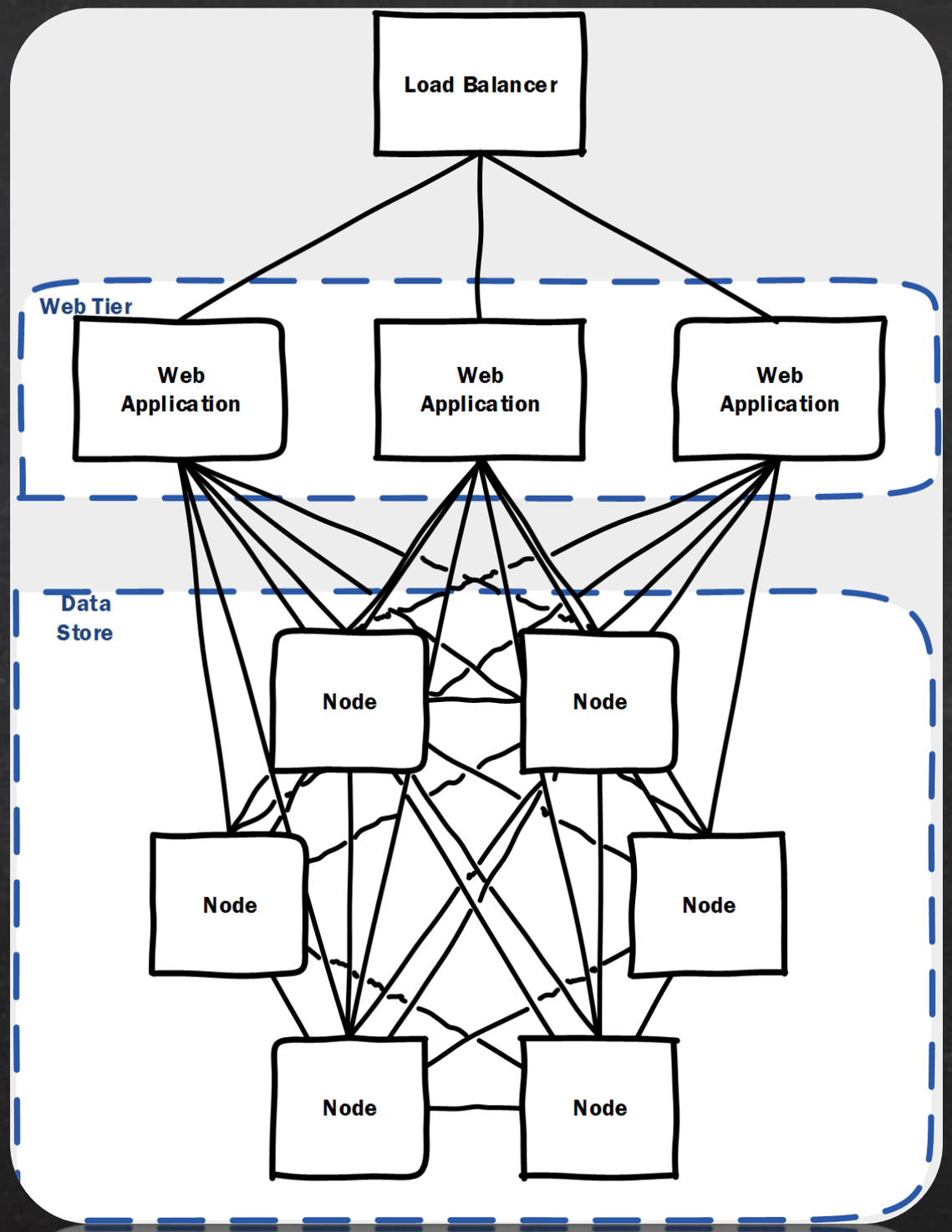
There's a new bottleneck

Generally an RDBMS can become a bottleneck around 10K transactions per second



Next Step... Distribute Your Data

Nodes distribute queries and replicate data - lots more complexity!



Each web server can talk to any data storage node

Cluster = Additional Complexity

If a node permanently fails, how do you recover?

How do you add a node to the cluster?

How do you upgrade the cluster and keep it available?

If a node can't talk to another node, what action can you take?

How do the web servers know which node to use? Do they retry if one fails to respond? If so how long to they wait?

We'll focus on this

Enter the CAP Theorem!

This guy created the CAP Theorem

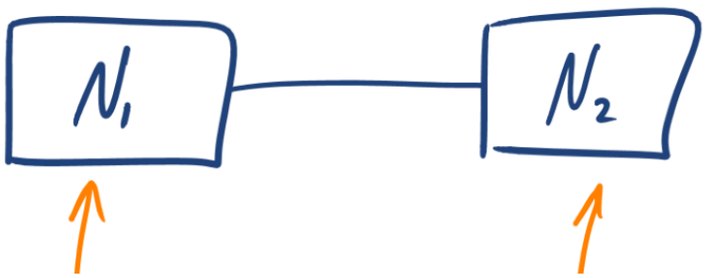
This guy's VP Invented the internet

CAP Theorem: Defined

Within a distributed system, you can only make two of the following three guarantees across a write/read pair

Consistency

Availability



Partition Tolerance

Guarantee 1: Consistency

Note: not the same as the C in ACID!

If a value is written, and then fetched, I will **always** get back the new value

Guarantee 2: Availability

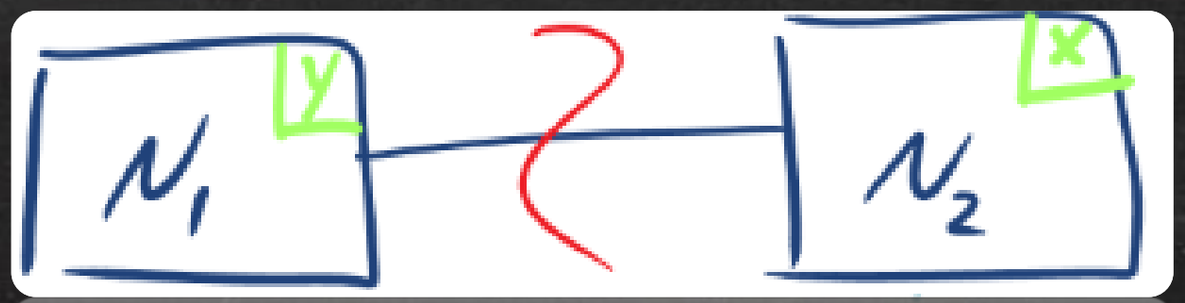
Note: not the same as the A in HA!

If a value is written, a **success message** should always be returned. If a subsequent read returns a stale value, or something reasonable, it's OK.

Guarantee 3: Partition Tolerance

Note: nothing to do with BAC!

The system will continue to function when network partitions occur - OOP != NP.

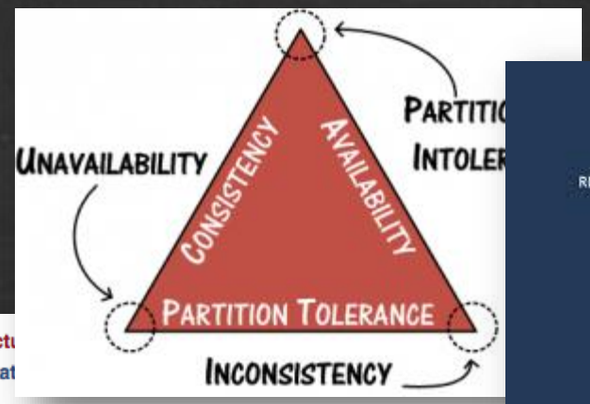


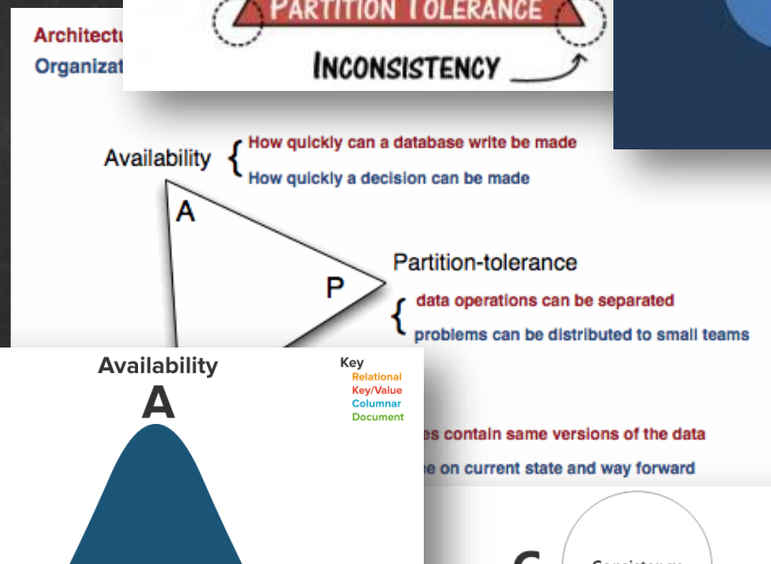
CAP Triangle

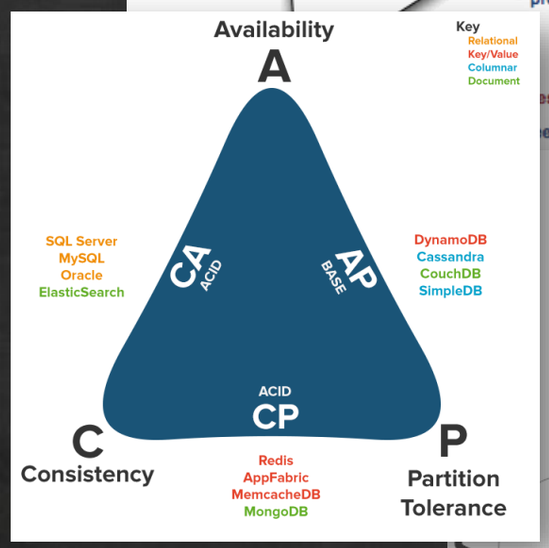
The CAP Theorem is explained as a triangle

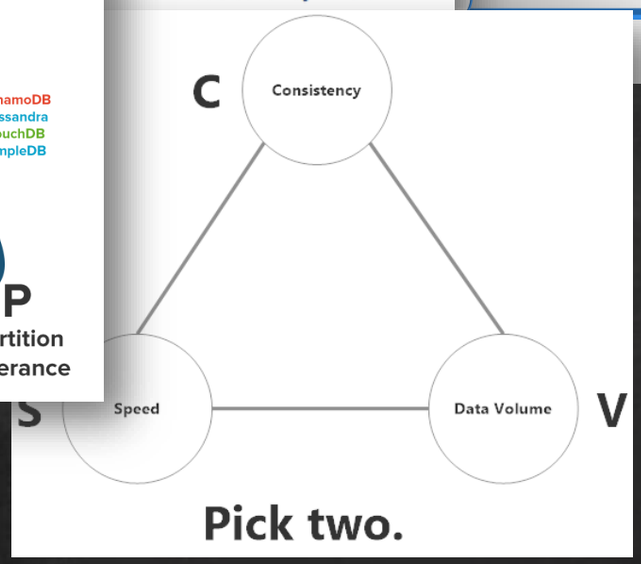
C, A or P: Pick two

This is true in practice, except…









When choosing a distributed system...

VS.

... You Can't Sacrifice Partition Tolerance!

Distributed
(a.k.a. Partition Tolerant)

Available

OR

Consistent

NOT Distributed
(a.k.a. NOT Partition Tolerant)

Available

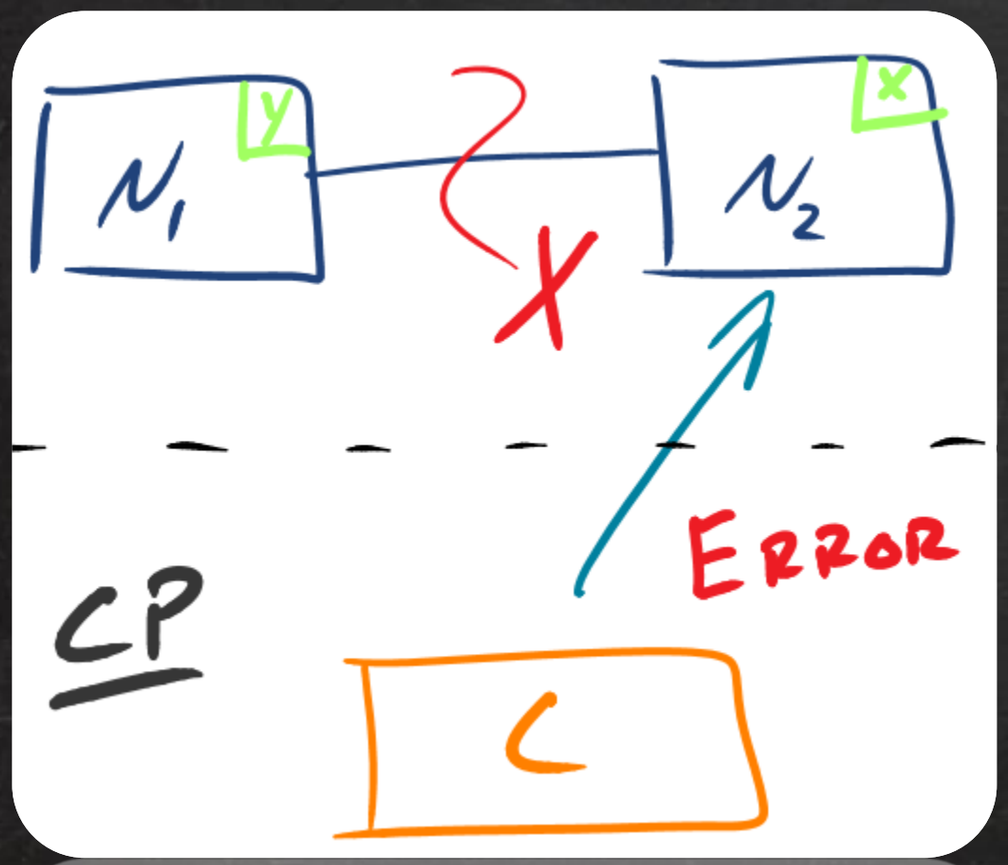
AND

Consistent

CP vs. AP

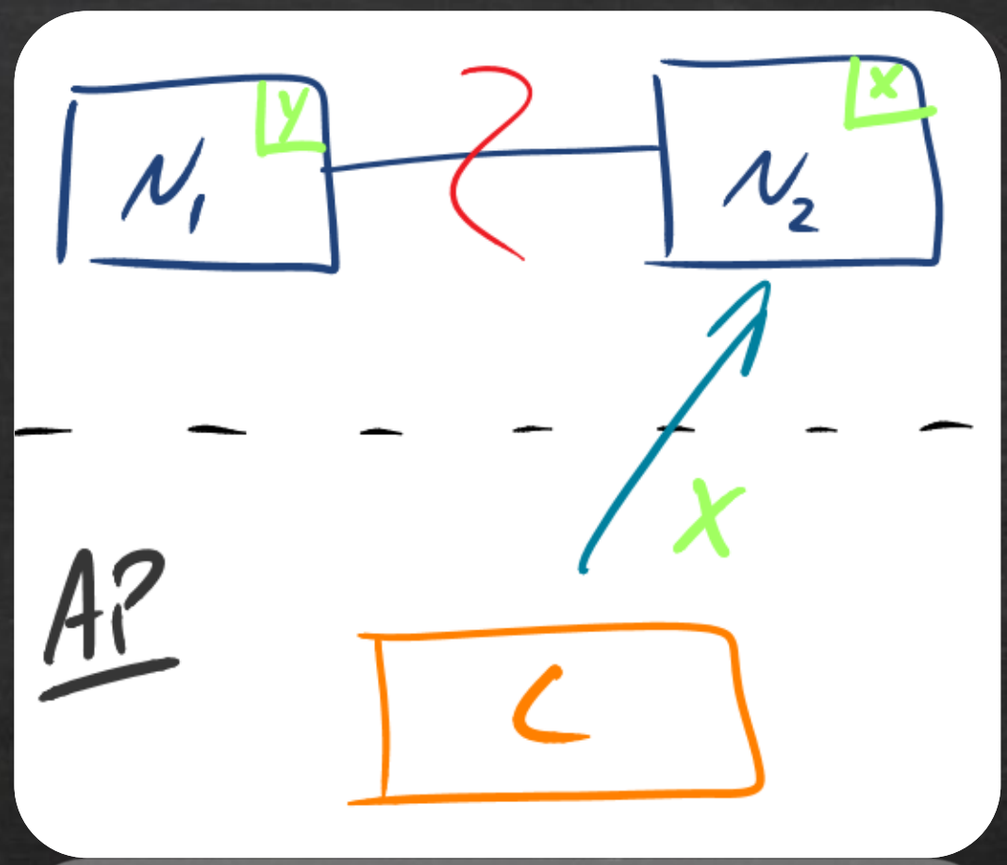
Synchronous.

Waits until partition heals or times out.



Asynchronous.

Returns a reasonable response always.



CP vs. AP

Synchronous.

Waits until partition heals or times out.

Asynchronous.

Returns a reasonable response always.

At a bank, you get a deposit receipt after the work is complete

At a coffee shop, you get a receipt before the work is complete

CHAPTER 2

When do companies care?

Companies care about internet scale

dropcam

amazon

Google

facebook

YAHOO!

Distributed Storage Past

2004
Google's Map Reduce paper published

2006
Google's Big Table paper published

2007
Amazon's Dynamo paper published

2008
Yahoo runs search on Hadoop

2010
Azure Table Storage released

2009
Yahoo open sources Hadoop

2008
Bitcoin paper published

2008
Facebook open sources Cassandra

2012
Google's Spanner and F1 papers

2013
Amazon releases DynamoDB inside AWS

2014
Google's Mesa paper published

2015
????

Looking forward

- Open source implementations of more sophisticated storage systems
- Managed services with more advanced capabilities
  - Google Cloud versions of F1, Spanner, or Mesa?
- NoSQL + SQL
- Distributed data storage in untrusted environments

CHAPTER 3

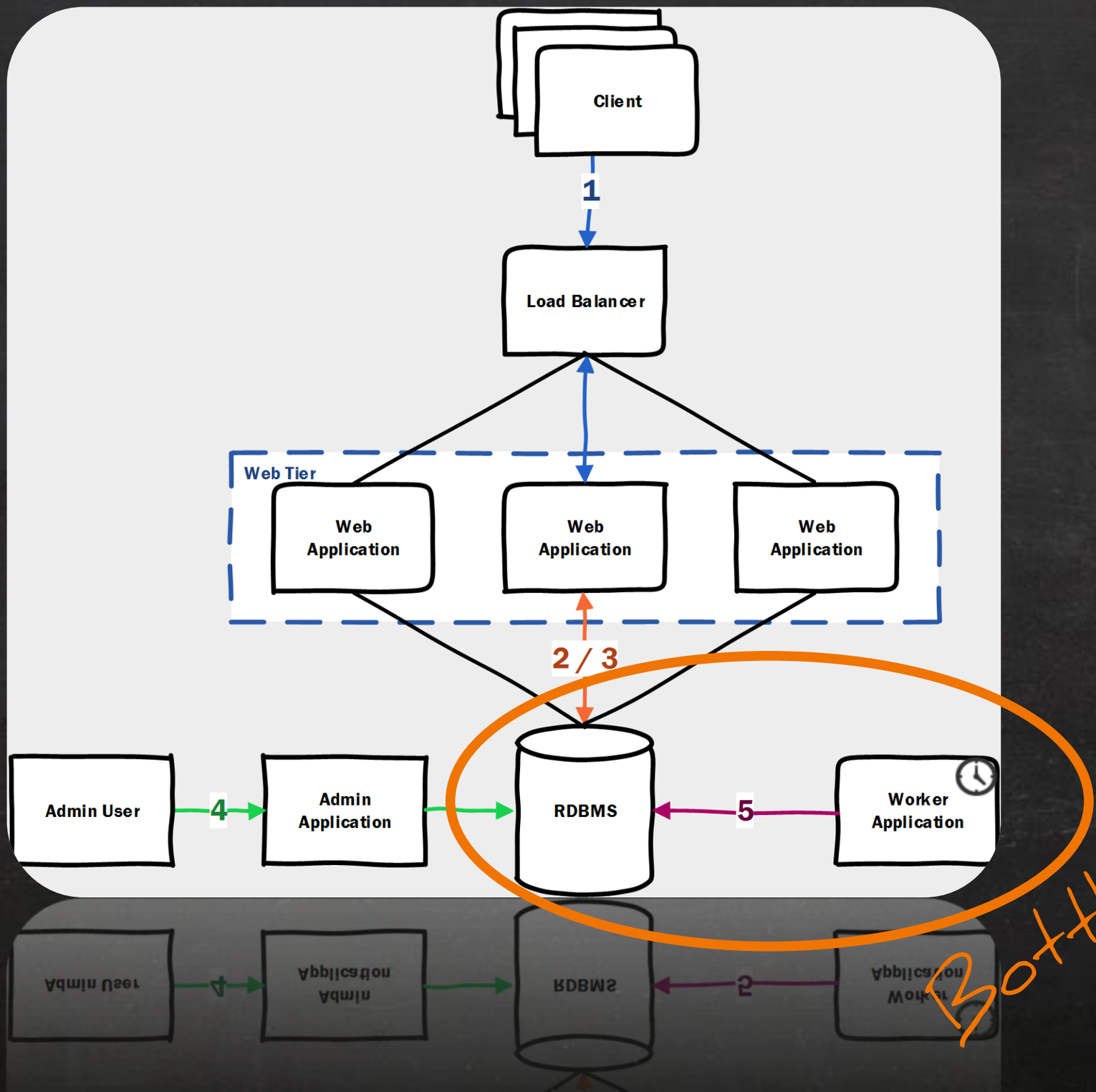
How does this affect me

Even our most “legacy” clients are already starting to care about internet scale:

Scenario

- Client - Energy Retailer (Independent Sales Force)
- Sales Agent captures info about potential customer
- Price generated on-demand based on daily rate curve
- Quote no longer valid at midnight
- Each night, rates are updated based on new rate-curve
- Used to take 4 hours
- Now takes › 24 hours (Due to increased demand)

Current State



Solution Strategy

Assess

- Analyze business performance needs
- Select non-performing work streams
- Filter – (Could/Should)
- Prioritize
- Performance Baseline / Load Test

Strategize

- Identify Bottlenecks (CPU/RAM/Network)
- Optimization strategy
- Technology Selection

Implement

- POC
- Load Test
- Optimize
- Build

Optimize Code

Scale Up

Scale Out

Managed Service

Optimize Code

Level 1

- Least organizational impact
- No architecture changes required
- Use existing development processes

- Risky - Code may be fine
- Expensive - Dev Resources
- Time Consuming - Dev + Deploy

Scale Up

Level 2

C/A

- Easiest solution
- Utilize existing infrastructure
- Little/no architecture changes
- Low probability of network partitions

- May not solve the problem long-term
- Hardware limitations
- Non-linear improvement (2x RAM != 2x Performance)

Scale Out

Level 3

A/C

- Highest throughput
- Improved system up-time
- No single point of failure
- Linear performance increases
- Use commodity hardware - Hard to scale-up CPU

- Increased infrastructure / system complexity
- Increased probability of network partitions
- Automation complexity

Managed Service

Level 4

C/A
A/C

- Low barrier to entry
- No additional hardware investment required
- Treat as extension of existing data center

- Appliance configuration
- Globally redundant (cloud)

- Most organizational change
- Less control and customization
- Built-in redundancy and innovation

Pick One (Or More!)

Optimize Code (Level 1)

- Least organizational impact
- No architecture changes required
- Use existing development processes
- Risky – Code may be fine
- Expensive – Dev Resources
- Time Consuming – Dev + Deploy

Scale Up (Level 2)

- Easiest solution
- Utilize existing infrastructure
- Little/no architecture changes
- Reduce probability of network partitions
- May not solve the problem long-term
- Hardware limitations
- Non-linear improvement

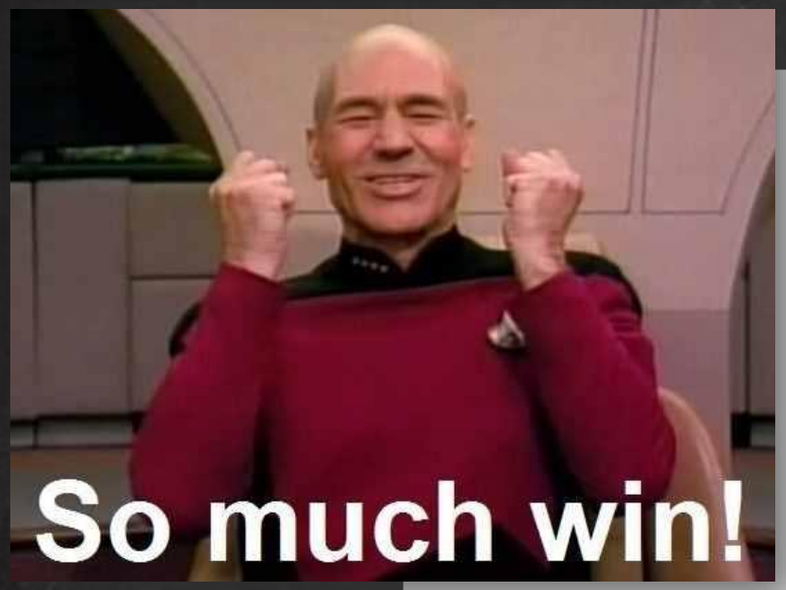
Scale Out (Level 3)

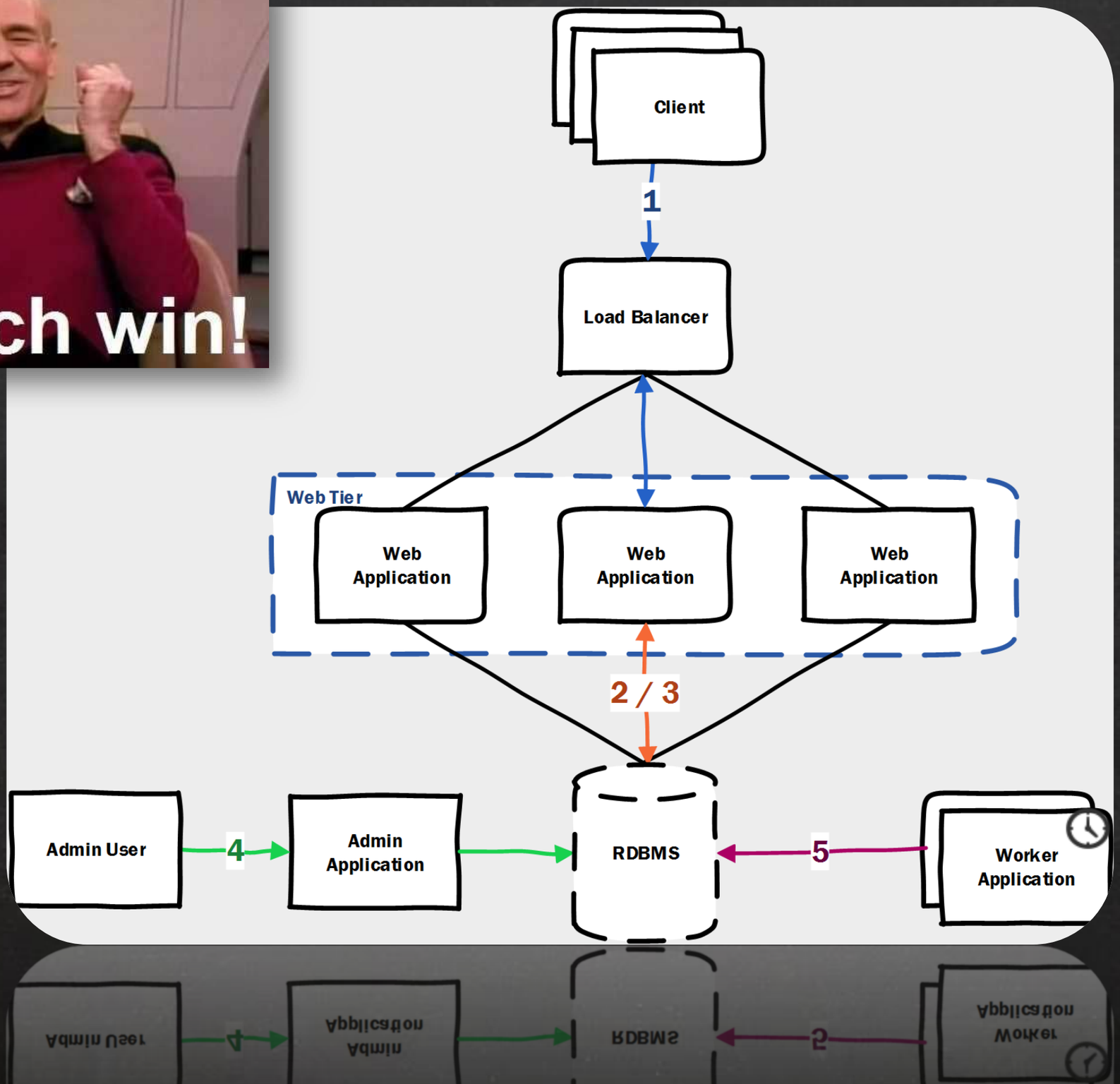
- Highest throughput
- Improved system up-time
- No single point of failure
- Linear performance inc.
- Use commodity hardware
- Increased infrastructure / system complexity
- Increased probability of network partitions
- Automation complexity

Managed Service (Level 4)

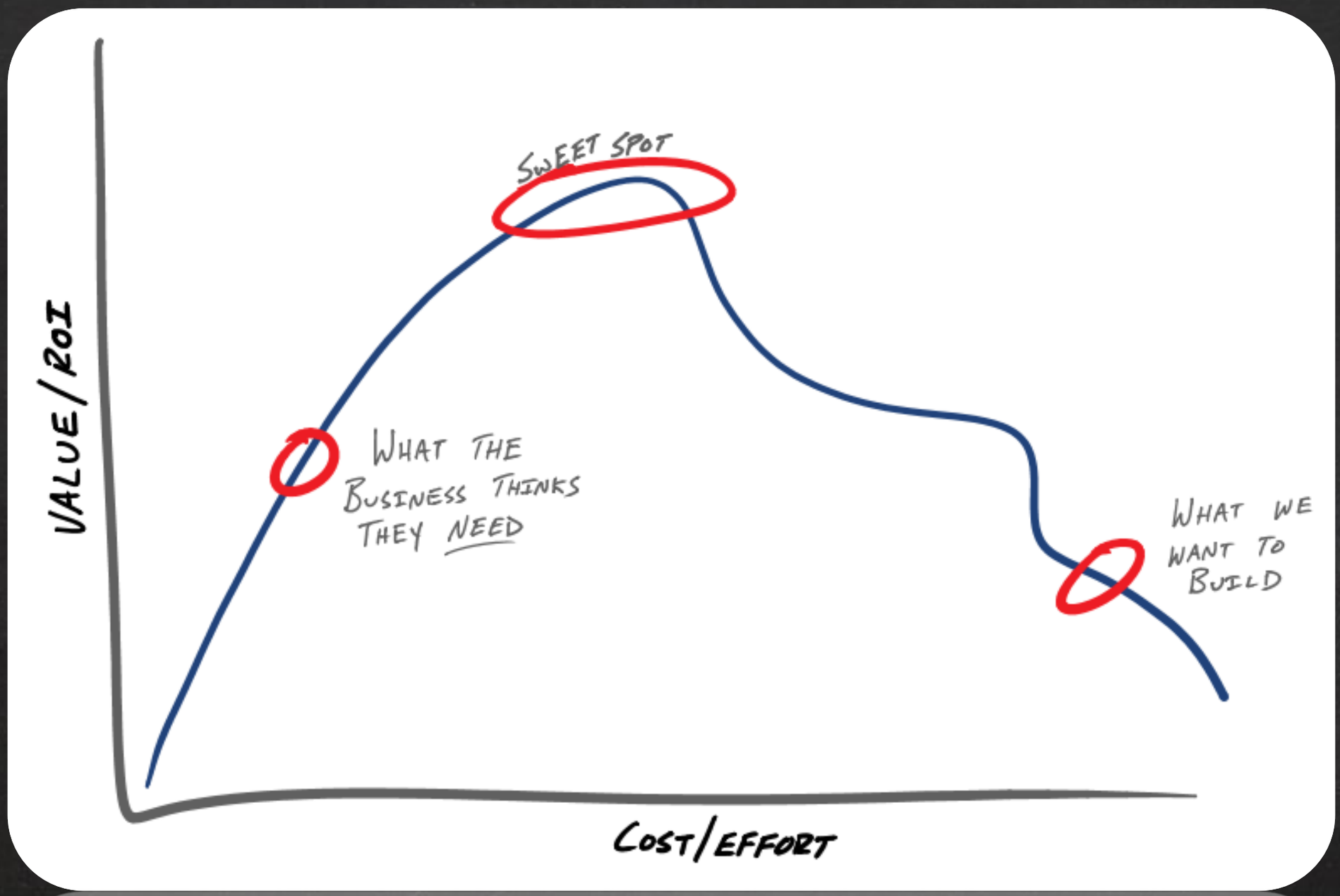
- Low barrier to entry
- No additional hardware investment required
- Treat as extension of existing data center
- Appliance configuration
- Globally redundant (cloud)
- Most organizational change
- Less control and customization
- High innovation

First Attempt

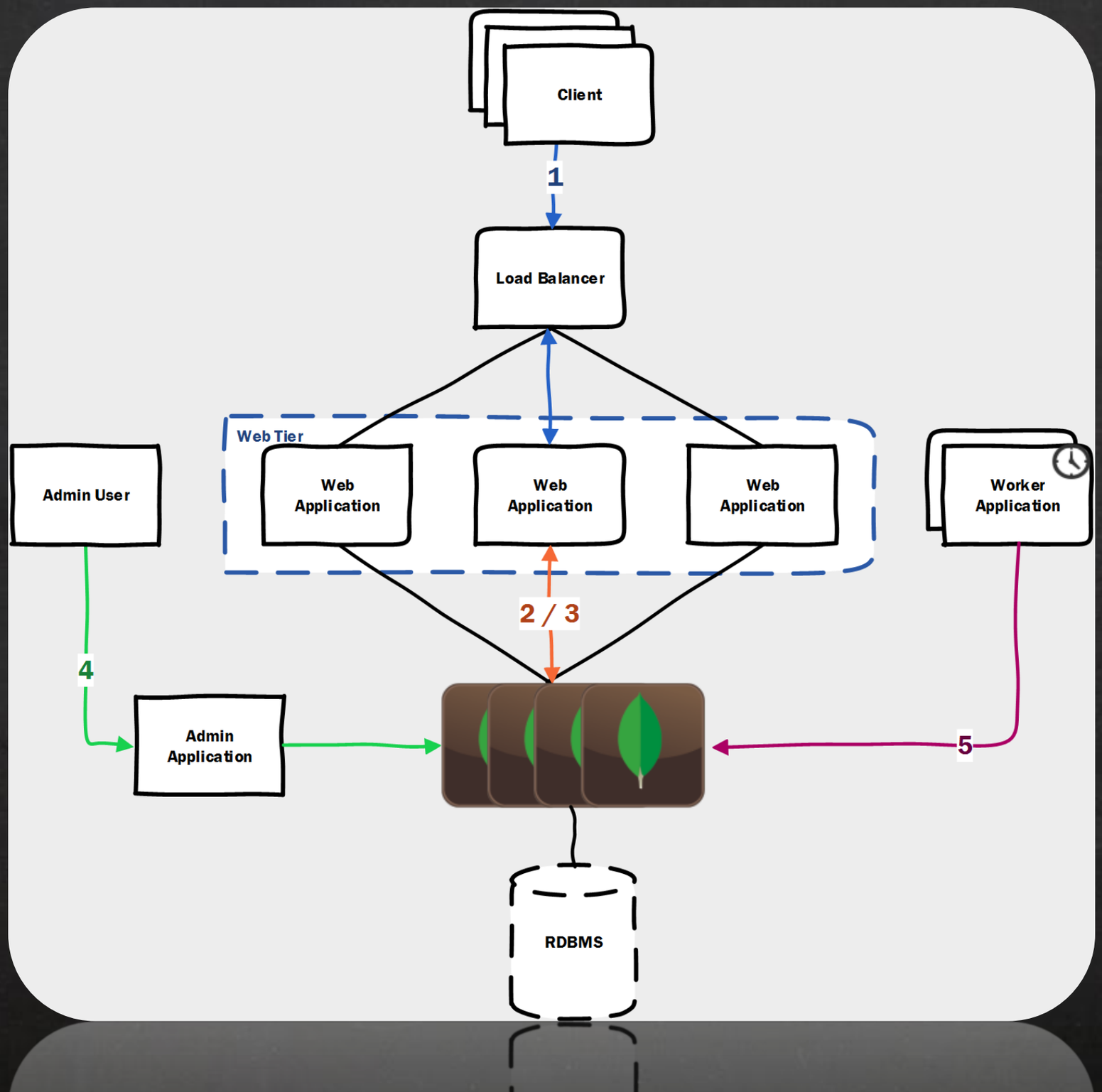




Good Enough?

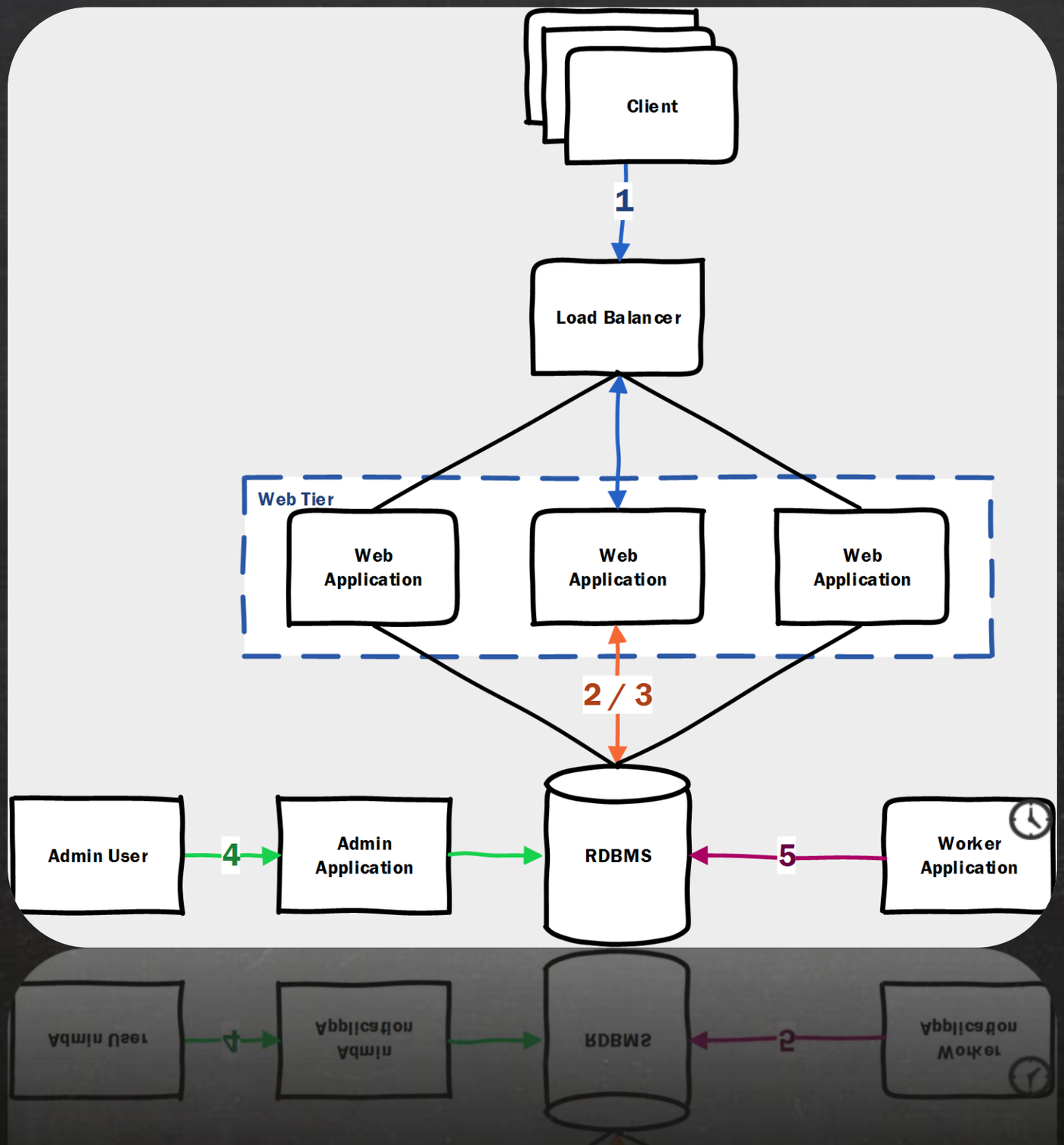

SWEET SPOT
WHAT THE BUSINESS THINKS THEY NEED
WHAT WE WANT TO BUILD
VALUE/IMPACT
COST/EFFORT

Taking It to the Next Level



The Best Solution?

What Would YOU Do?



Fin'

robert.greiner@parivedasolutions.com

nathan.murray@parivedasolutions.com