Your Language Doesn't Scale
A Discussion of the Nature of Scaling

Ikai Lan
Linkedin
Hi! I’m Ikai Lan

Ikai Lan

Loquacious Problem Solver/Causer
San Francisco Bay Area

Current
• Software Engineer at LinkedIn

Past
• Engineering Consultant at CollegeCarrot
• Systems Engineer at Citrix Systems
• Systems Administrator at San Jose State University

Education
• San Jose State University

Recommended
3 people have recommended Ikai
1 manager, 2 co-workers

Connections
172 connections

Industry
Computer Software

Public Profile
http://www.linkedin.com/in/ikailan
This talk is about scaling
Particularly with regards to the web
Attributes of a web app

- Client pull based
- Operations must finish “fast”
- “Fresh” information
The chip on my shoulder
Ruby on Rails can’t scale!

- Ruby is slow
- Default SQL used is not optimal
- Page rendering is slow
All of these are true

Truth
1. Ruby is slow
2. SQL used is not optimal

```
@poll = Poll.find(params[:id])
if @poll.activated?
```

```
SQL (0.000245)  SET NAMES 'utf8'
SQL (0.000119)  SET SQL_AUTO_IS_NULL=0
Poll Load (0.00373)  SELECT * FROM `polls` LIMIT 1
Poll Columns (0.003457)  SHOW FIELDS FROM `polls`
Poll Indexes (0.002590)  SHOW KEYS FROM `polls`
Poll Load (0.000486)  SELECT * FROM `polls` WHERE (`polls` `id` = 22)
```
3. Rendering is slow

I don’t have a clever picture here. Just take my word for it.
Why do these fall flat?
Scaling != efficiency

```ruby
>> 'Scaling' != 'Efficiency'
true
```
Efficiency is

- Minimizing input for output
- Bang for your buck
Scaling != performance

```ruby
>> 'Scaling' != 'Performance'
=> true
```
Performance is

- Maximizing output for some input
- Speed
Scalability is

- Being able to grow capacity with demand
- How well a solution to some problem will work when the size of the problem increases
In an ideal world...

We achieve scalability efficiently while maintaining performance
Myth: Needing to throw hardware to solve a capacity problem means your application is not scalable
Truth: If this works, by the definition, your application IS scalable
... maybe not efficient
So the arguments ...

- Ruby is slow
- Default SQL is not optimal
- Page rendering is slow
Concerning performance ...
How much time?

- Internet
- User
- Rendering: LONG TIME
- Network: ???
- Assets
- Tiny fraction
- LONG TIME
Scaling case study
First let’s define terms
Vertical scaling

- Improving performance in one of our component tiers
- Adding a tier optimized for some portion of the load
Vertical scaling examples

- Upgrading hardware
- Edge caching
- Adding a caching tier for the data store
Horizontal scaling

- Increase capacity by increasing instances of components
- Spreading load across instances
Horizontal scaling examples

- Load balancing (hardware LB or DNS load balancing)
- Data partitioning
Different strokes ...
On to the case study!
These guys made a site

Matt

Ikai
To share pictures!
Our first architecture

- Web Server
- Asset Server
- Database
Awesome! We have users!
We have too many users!
The yelling begins

Dude! Add more hardware!

You don’t think I tried?
The hardware - it does nothing!
Our current architecture
Why isn’t this working?

EXPLAIN PLZ - IM CONFUSD
Bottlenecks! Bottlenecks!
The first bottleneck is always the persistent store
Solution: Replicate data
New problem: replication delay

How come everything I do takes a few minutes to work?
Solution: caching
New problem: stale data

I can’t remove this tag!
Sometimes things aren’t up to date!
So now we’ve spent all that engineering effort and added no new functionality

- (would have been a lot harder if we were also trying to add features to stay competitive)
Are we done? Maybe ...
More growth: all persistence operations are expensive
Have to partition DB

Diagram of system architecture with Load Balancer, Web Server, Asset Server, MemCached, and DB (Master, Read Slave) in clusters.
Our application has to now understand how to speak to a partitioned store
This is really hard!
And I haven’t even gotten to transactional integrity or backups
We’re also introducing many more points of failure = ops burden
Engineering problems

- Cross functional development
- Merging
- Regression testing
Things now cost too much!
Do we even have a business model?
No one is spared!
Questions?
Thanks for coming!

- ilan@linkedin.com
- http://www.linkedin.com/in/ikailan