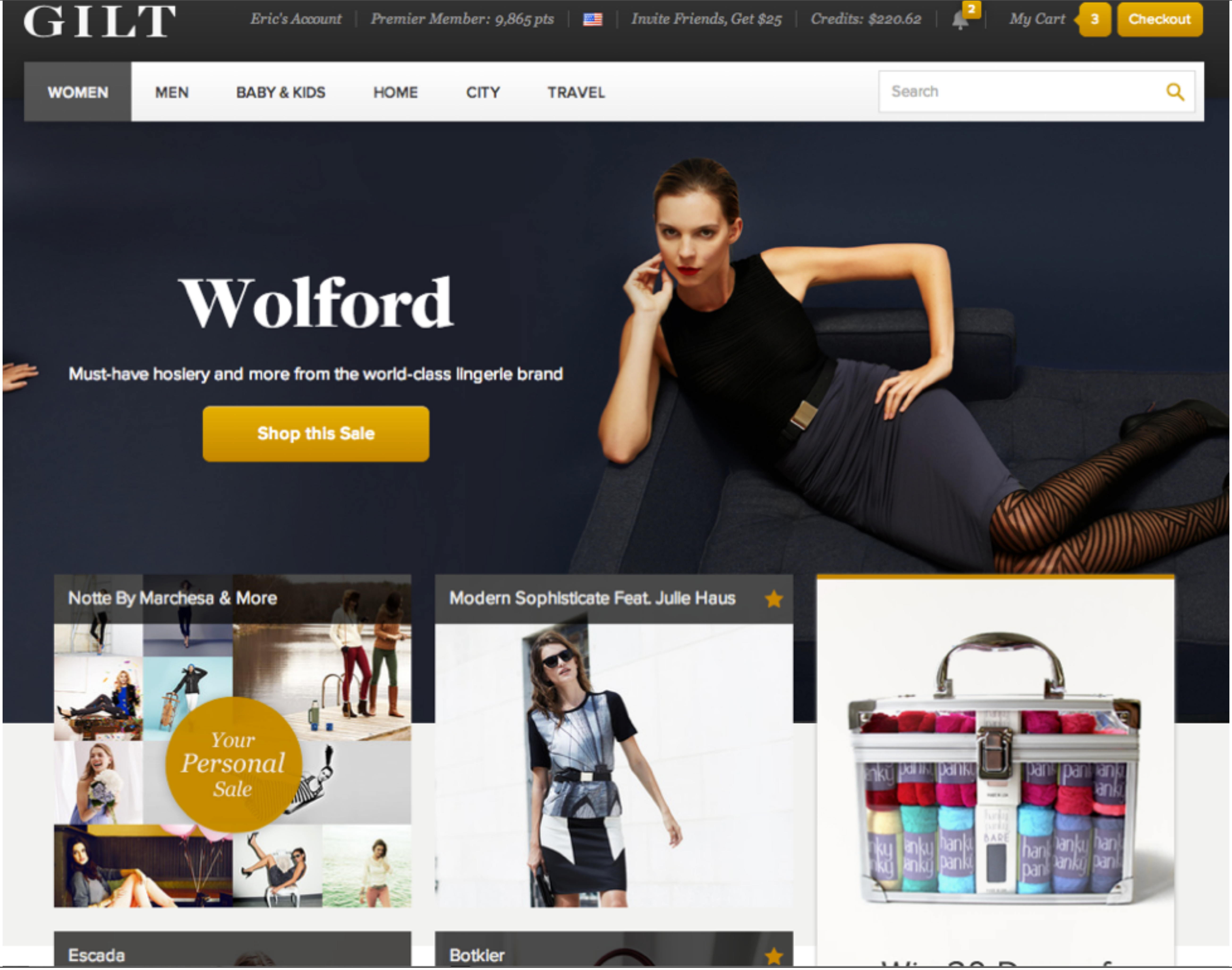
Fast but Not Loose:

Typesafe Clients in a Distributed Service Architecture, a retrospective

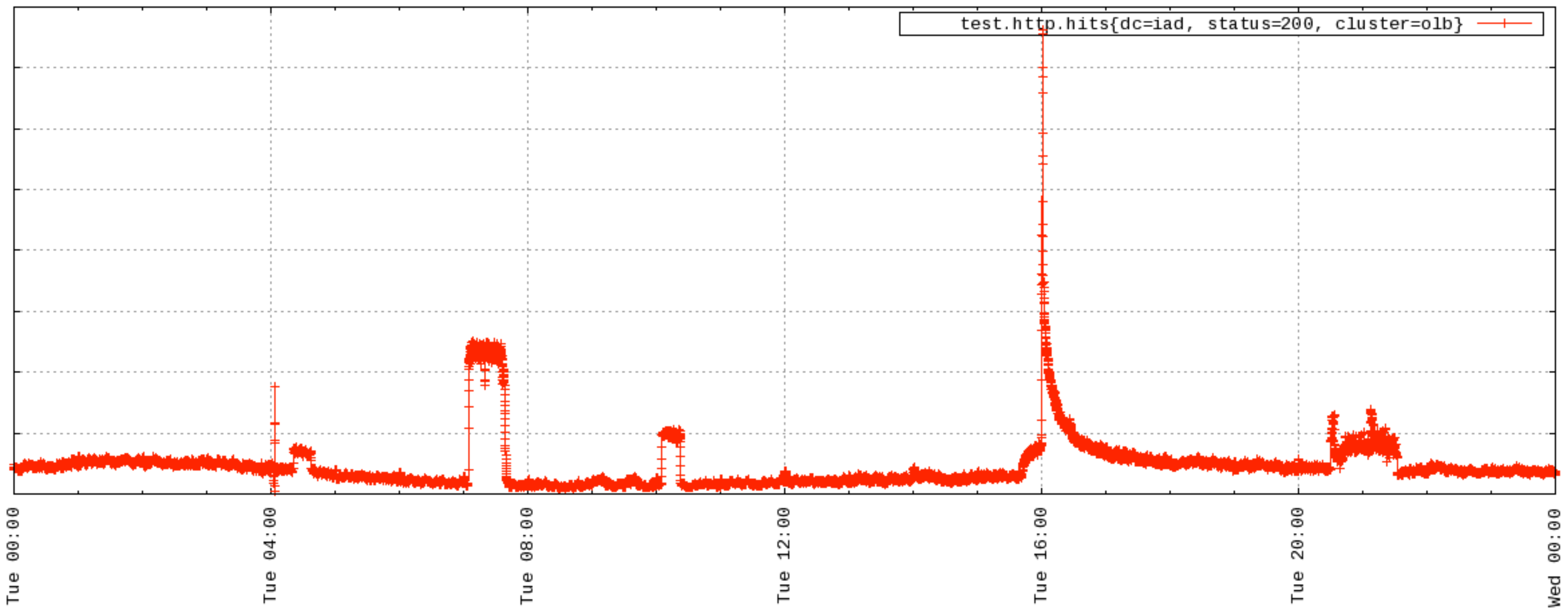
#gotocon #gotoaar #gilttech

Eric Bowman
VP Architecture @ Gilt Groupe

@ebowman
ebowman@gilt.com



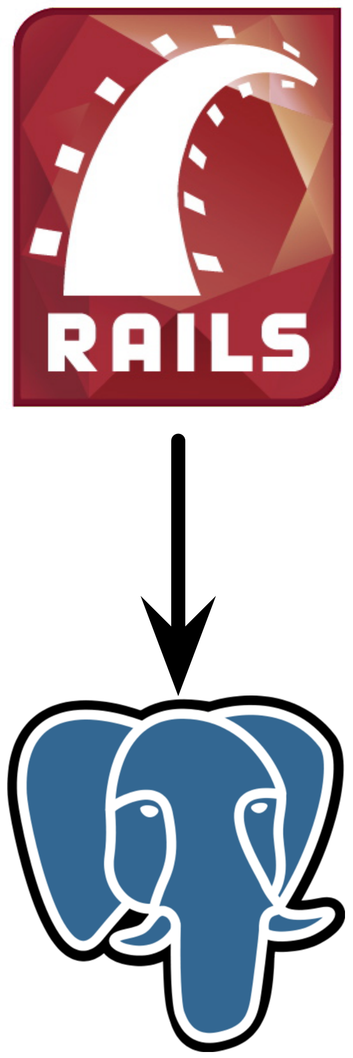
Monday, September 30, 13


test.http.hits{dc=iad, status=200, cluster=olb}
Tue 00:00
Tue 04:00
Tue 08:00
Tue 12:00
Tue 16:00
Tue 20:00
Wed 00:00

- Scala
- Play
- PostgreSQL
- MongoDB
- Voldemort
- Kafka

- Aster Data
- Mahout
- Jersey
- SBT
- Docker
- Continuous Delivery

Microservices



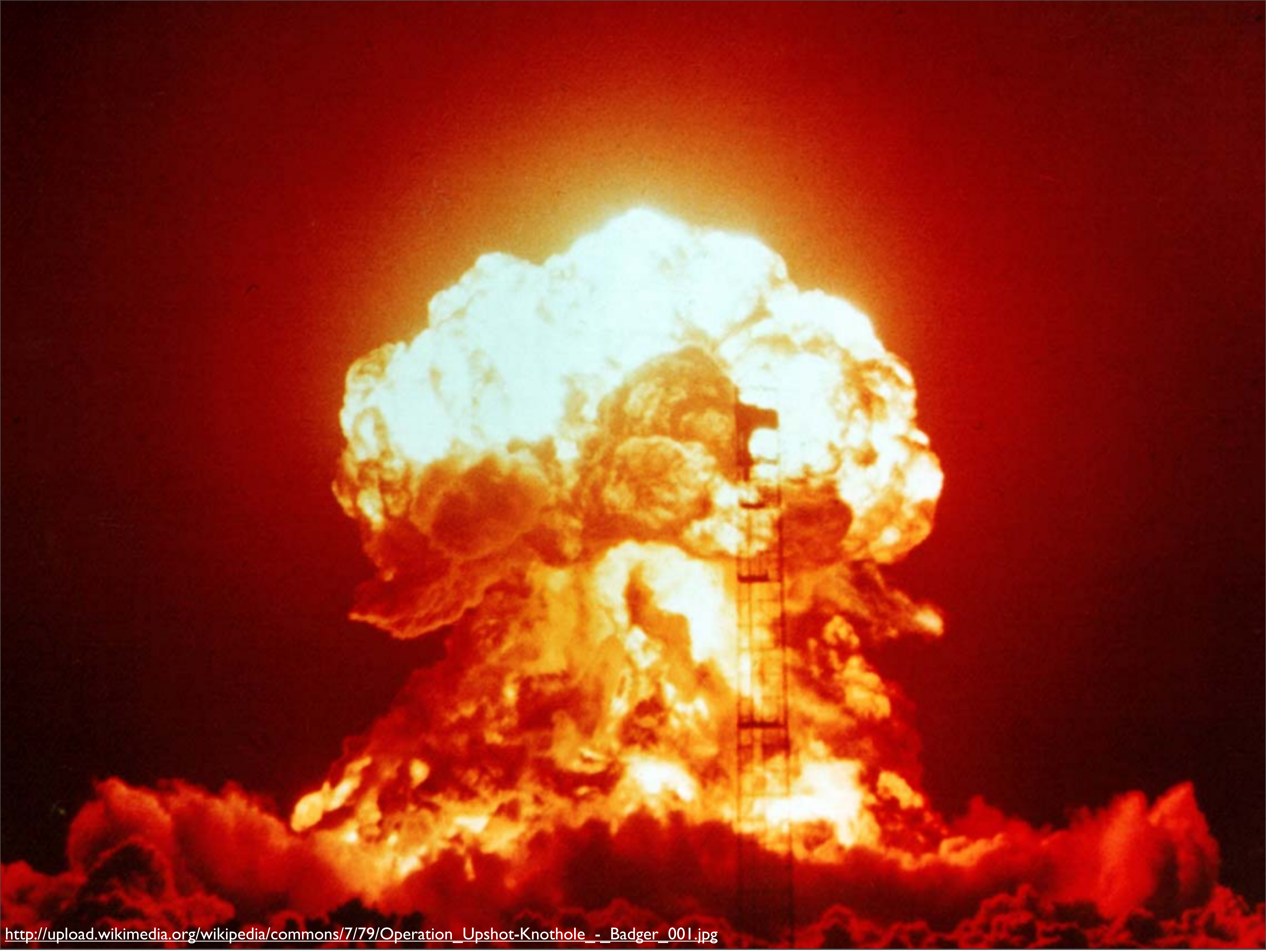
http://upload.wikimedia.org/wikipedia/commons/1/16/Ruby_on_Rails-logo.png

http://wiki.postgresql.org/wiki/File:PostgreSQL_logo.3colors.svg

Christian Louboutin

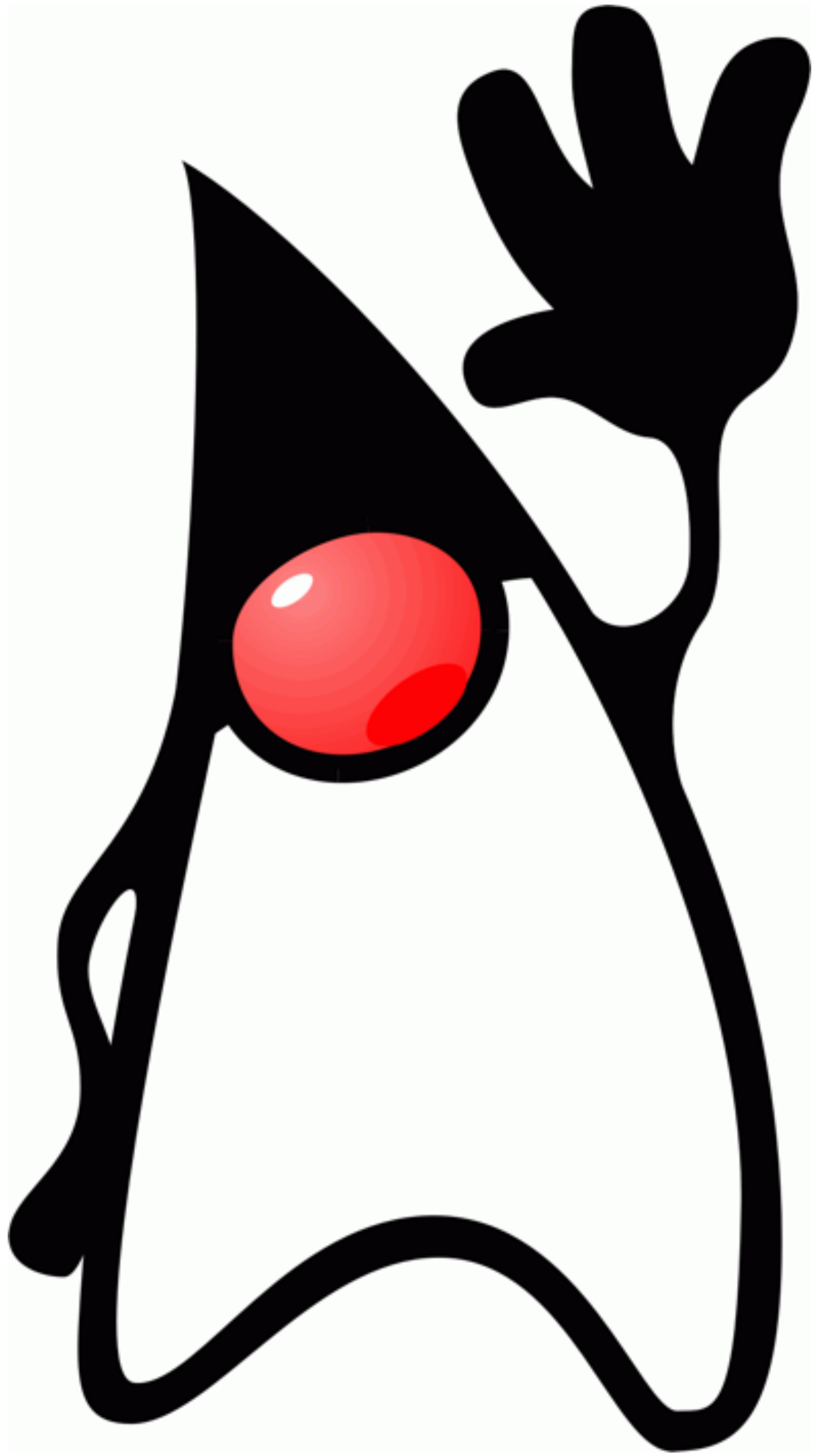
http://logonoid.com/images/thumbs/christian-louboutin-logo.jpg

http://data.iluxdb.com/data/christian-louboutin-daffodile-160mm-python-masai-1130127cm09_001.jpg?dd80c0

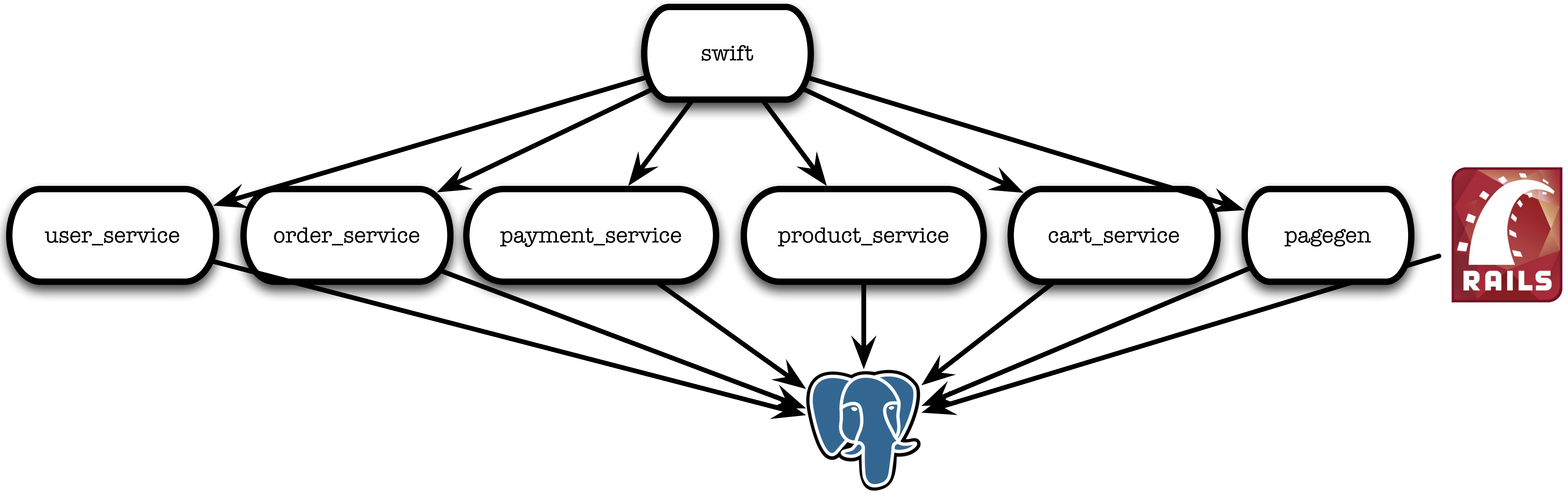


http://upload.wikimedia.org/wikipedia/commons/7/79/Operation_Upshot-Knothole_-_Badger_001.jpg

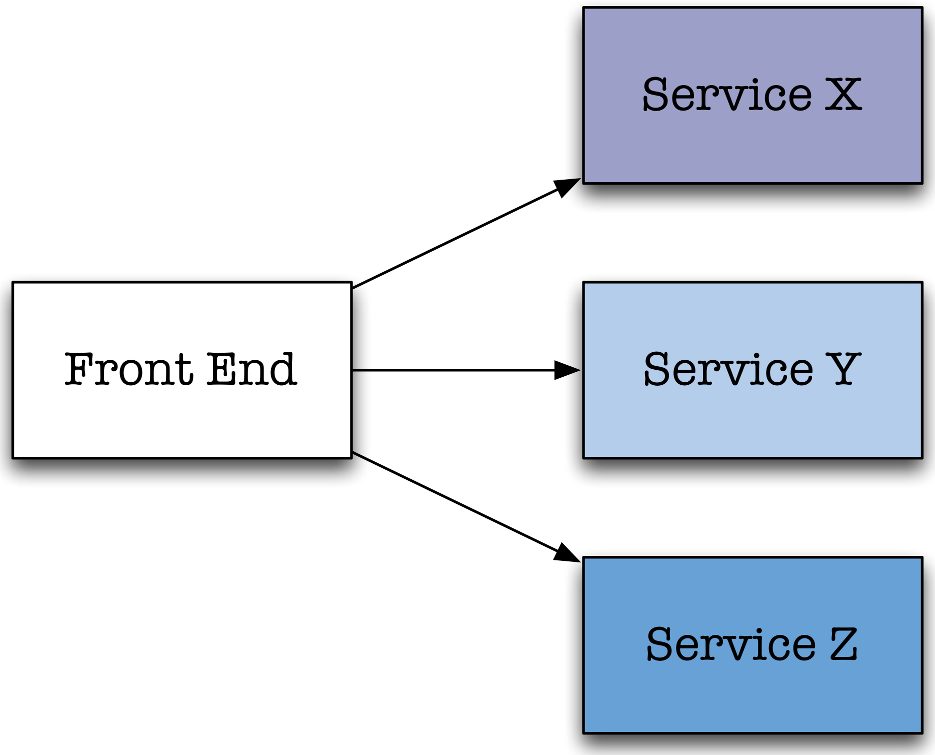
RAILS

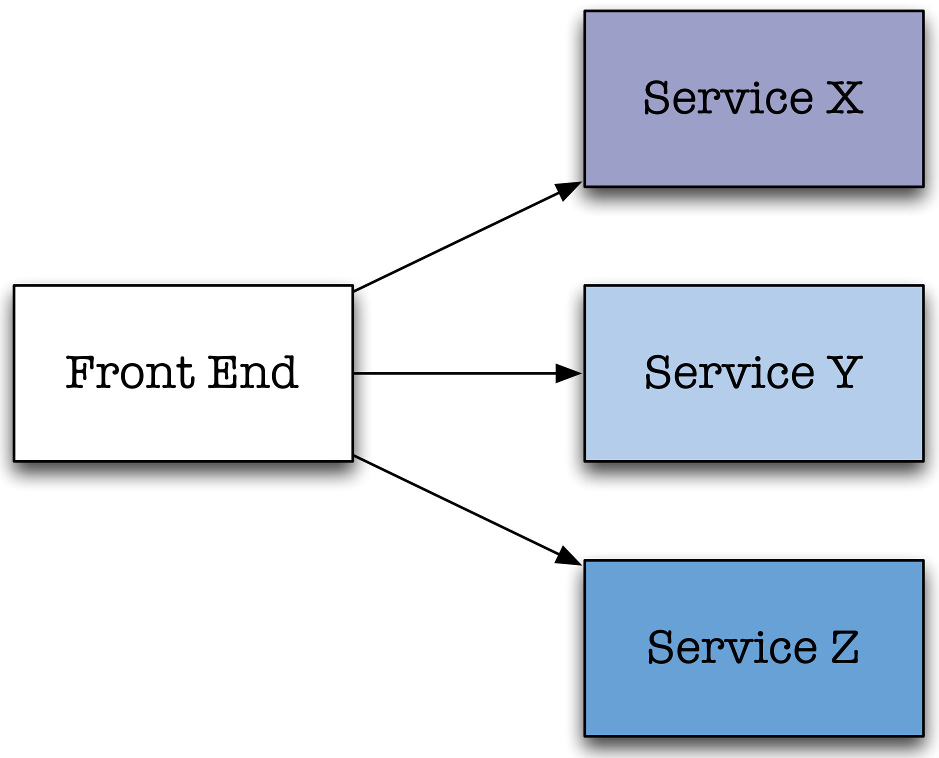


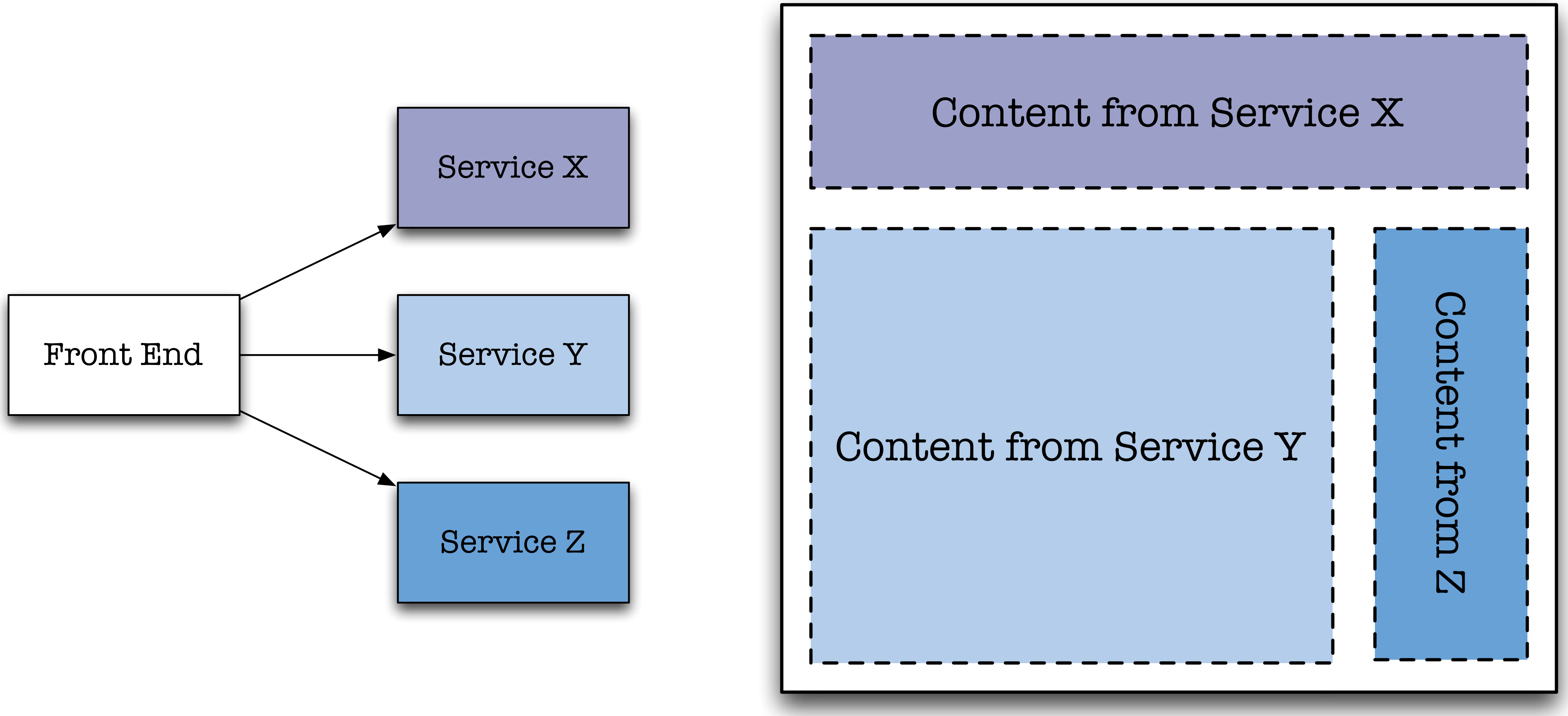
http://blog.verwilst.be/wp-content/uploads/2008/12/java.gif

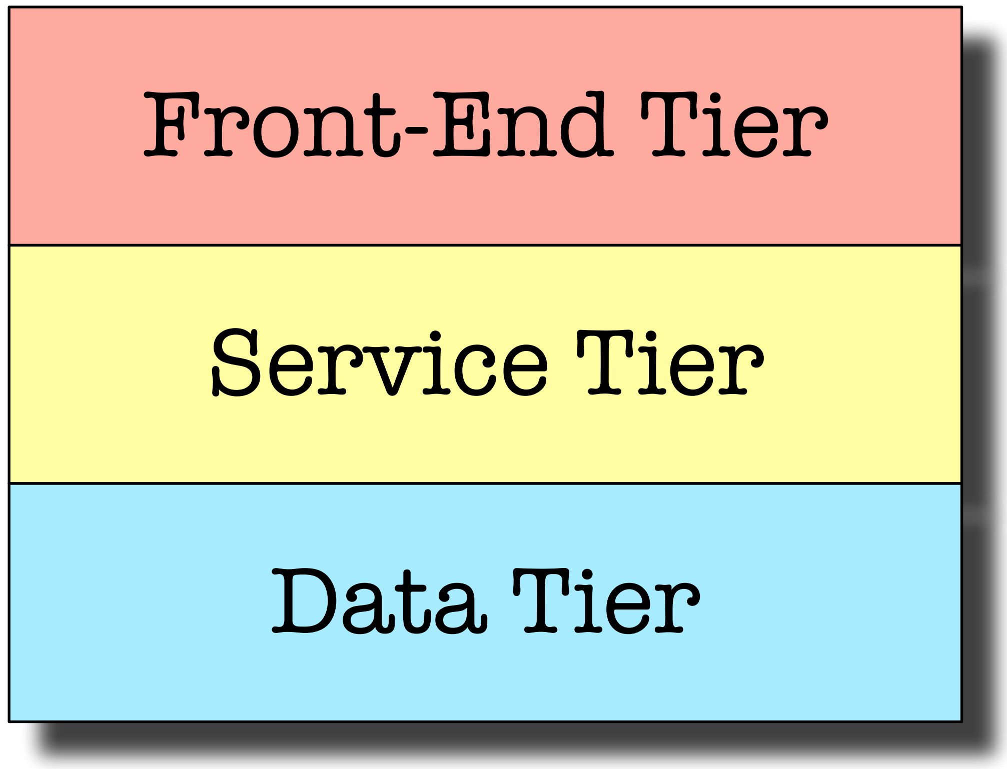

swift
user_service
order_service
payment_service
product_service
cart_service
pagegen
RAILS

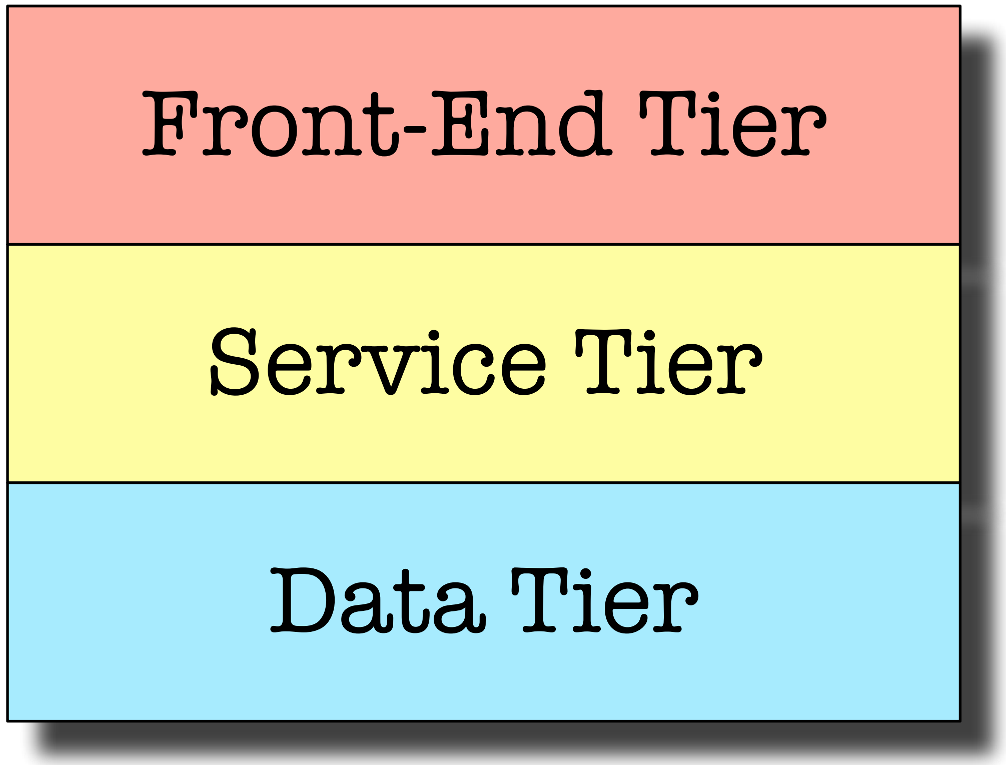
http://megmurph.com/wp-content/uploads/2013/03/success-1.jpg


Service X
Front End
Service Y
Service Z

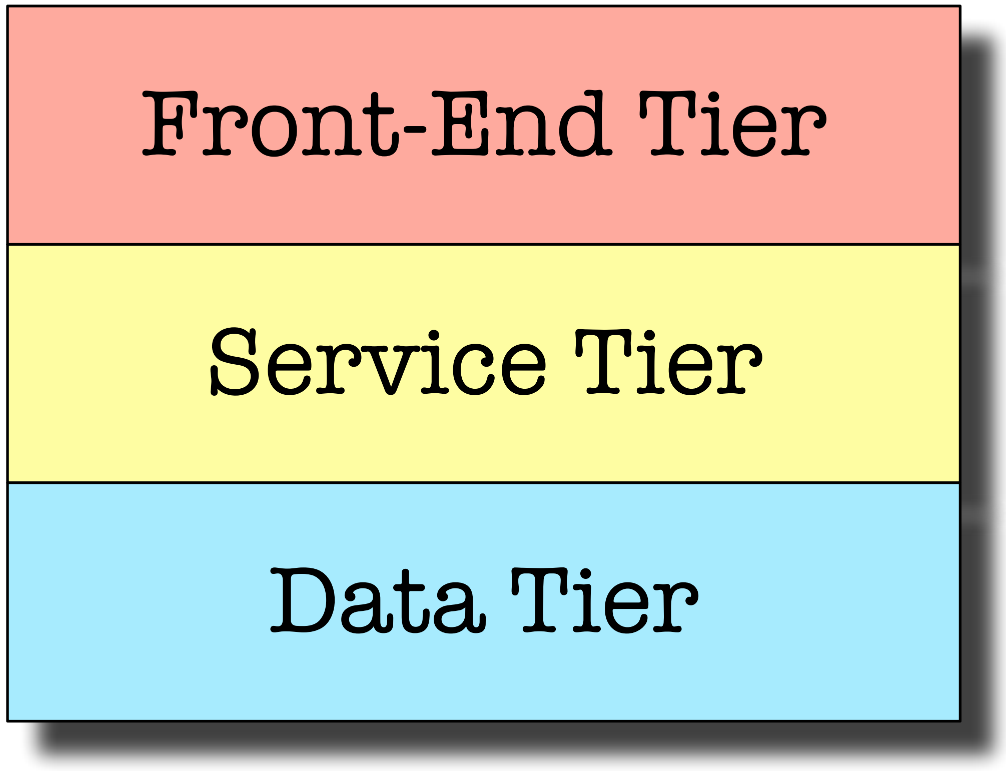

Front End
Service X
Service Y
Service Z


Front End
Service X
Service Y
Service Z
Content from Service X
Content from Service Y
Content from Z


Front-End Tier
Service Tier
Data Tier

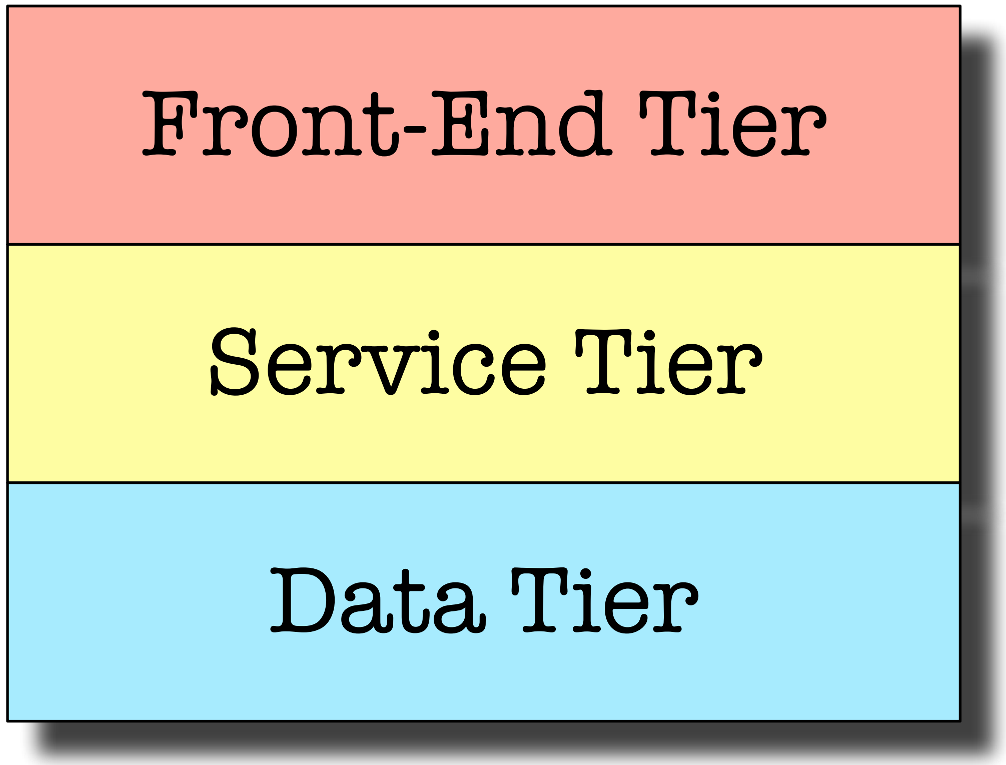


Caching
Light Computation
Orchestration



Caching
Light Computation
Orchestration

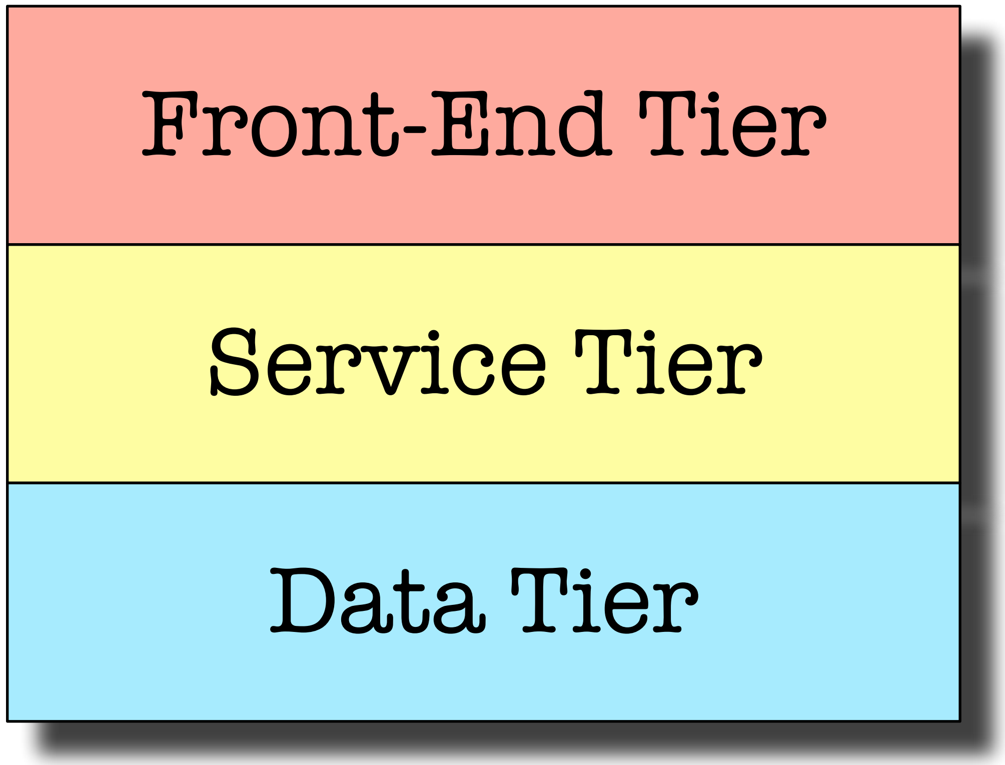
Caching
Heavier Computation
Separation of Concerns



Caching
Light Computation
Orchestration

Caching
Heavier Computation
Separation of Concerns

Data Access
(Disk/SSD/RAM)

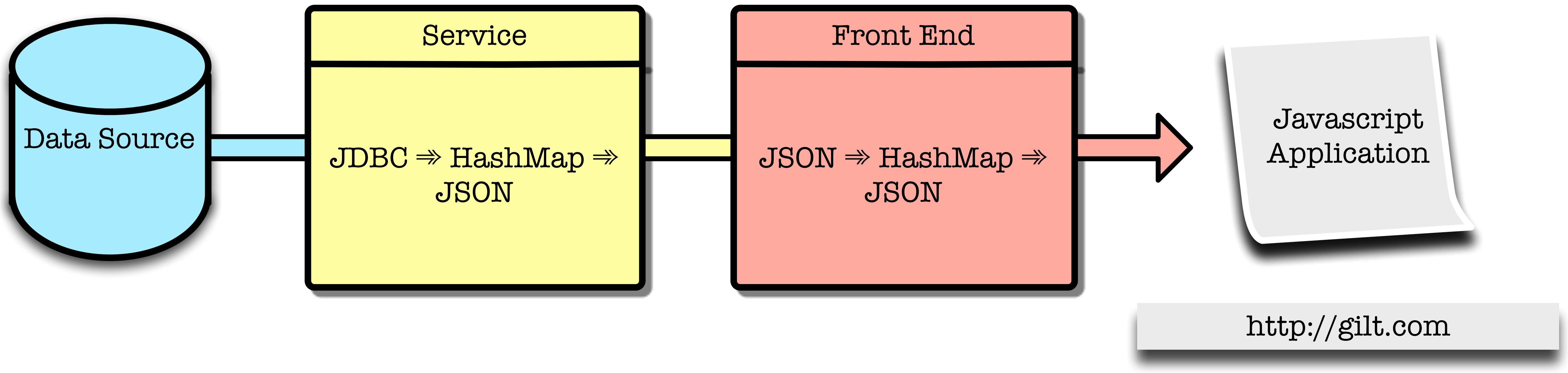


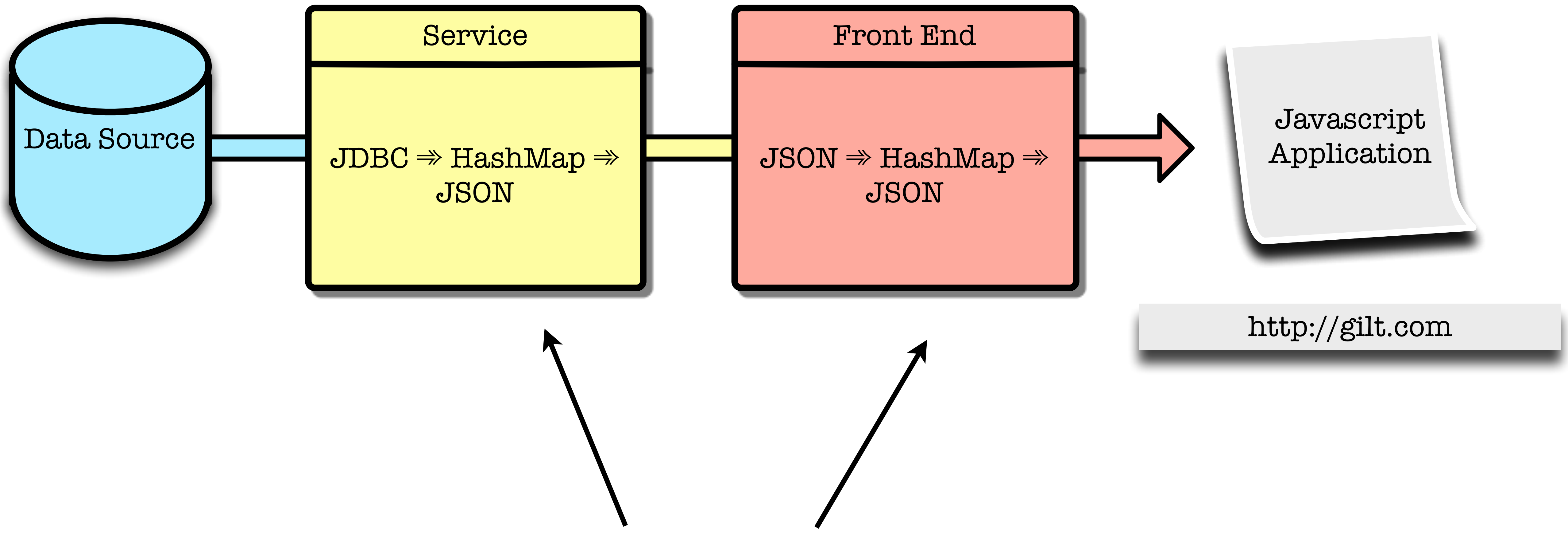
Caching
Light Computation
Orchestration

Caching
Heavier Computation
Separation of Concerns

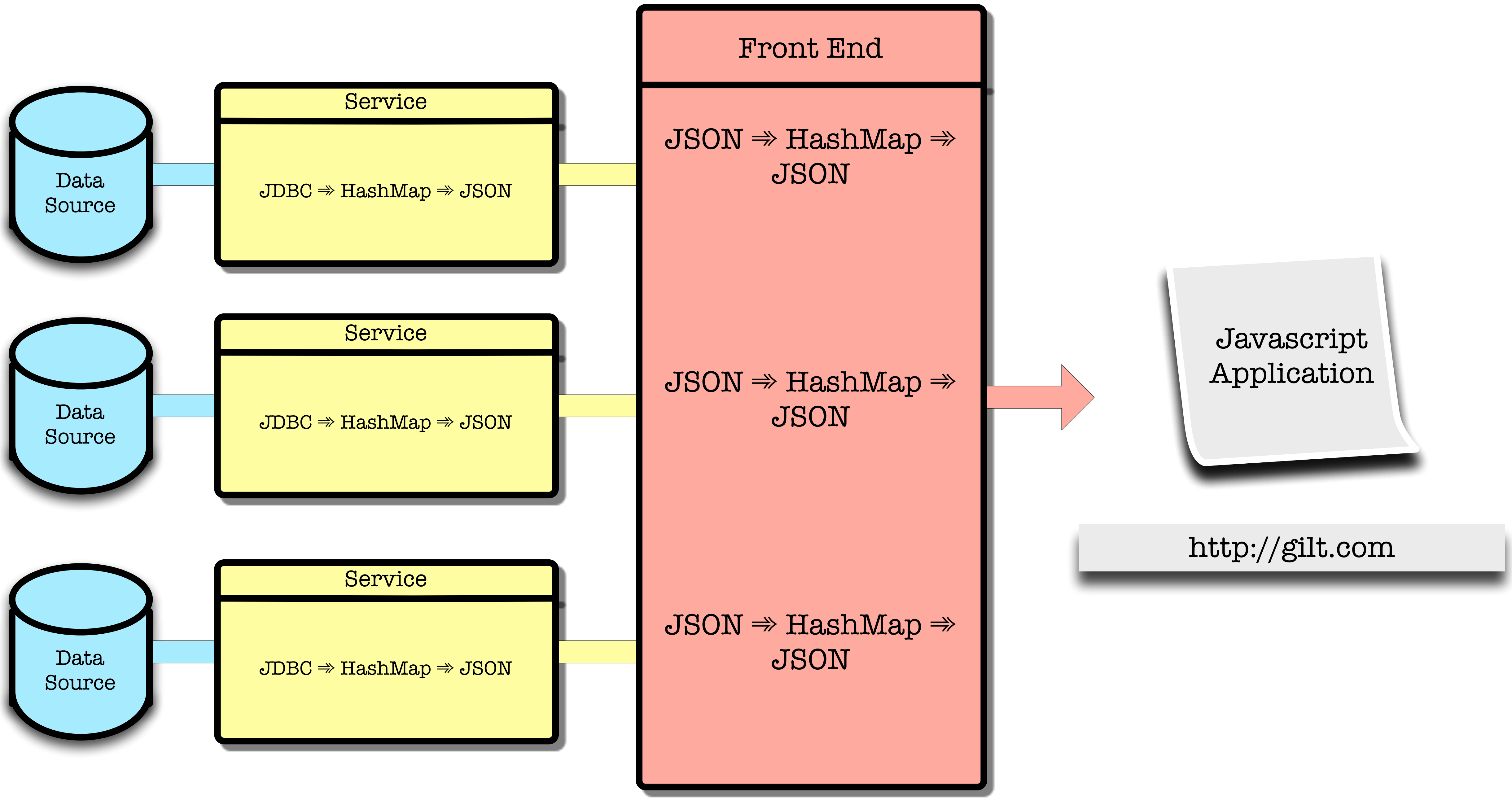
Data Access
(Disk/SSD/RAM)

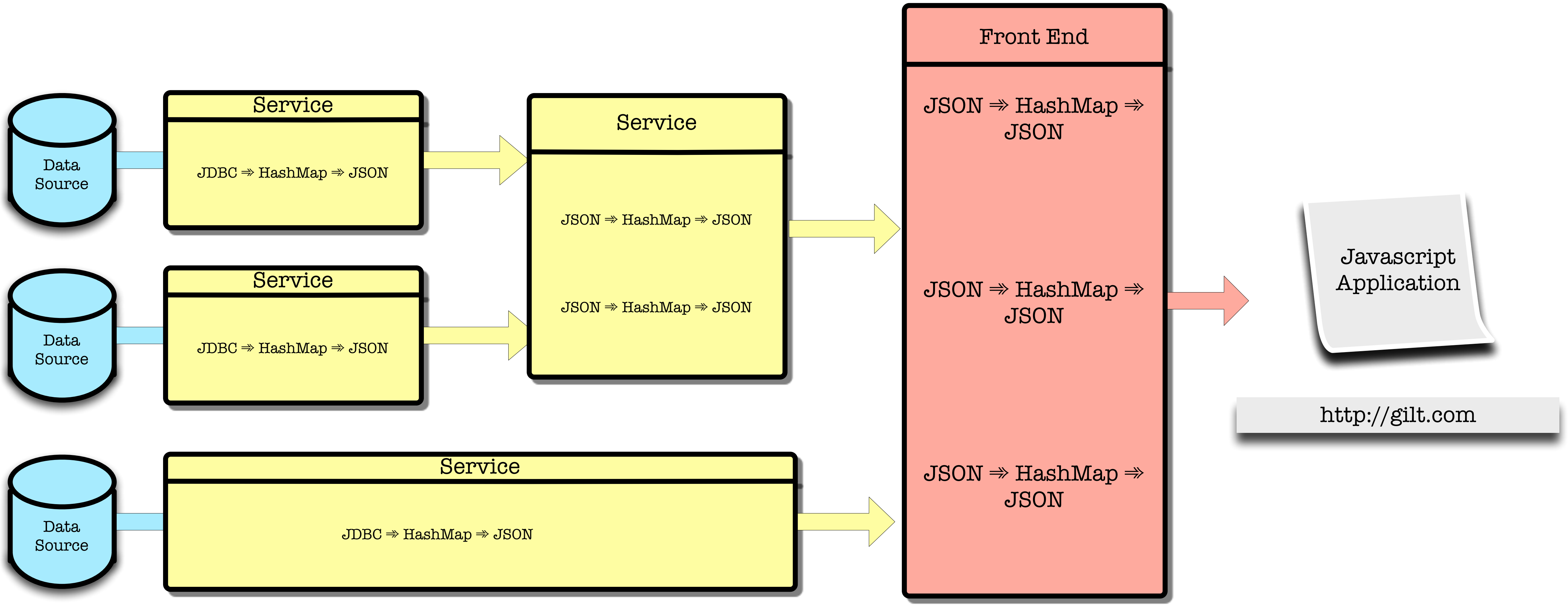
- Runtime Temperature
- Development Temperature


Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
Front End
JSON ⇛ HashMap ⇛ JSON
Javascript Application
http://gilt.com



Chaos Grows Quickly


Front End
Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
JSON ⇛ HashMap ⇛ JSON
JSON ⇛ HashMap ⇛ JSON
JSON ⇛ HashMap ⇛ JSON
Javascript Application
http://gilt.com


Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
Service
JSON ⇛ HashMap ⇛ JSON
JSON ⇛ HashMap ⇛ JSON
Data Source
Service
JDBC ⇛ HashMap ⇛ JSON
Front End
JSON ⇛ HashMap ⇛ JSON
JSON ⇛ HashMap ⇛ JSON
JSON ⇛ HashMap ⇛ JSON
Javascript Application
http://gilt.com

http://www.recruitmenttakeout.com/wp-content/uploads/2013/04/brian.png

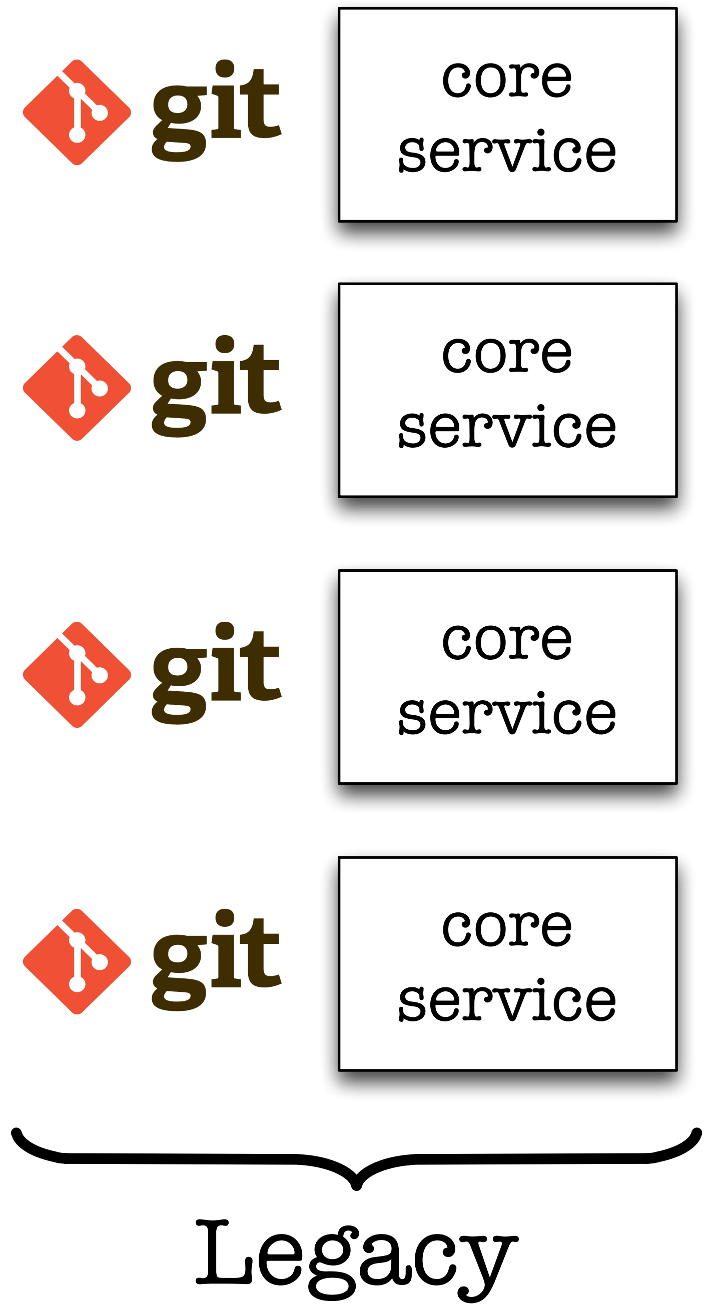
- Fast
- Service Decomposition
- Implicit core model was good

- Org Scaling
- APIs
- Implicit core model was ... implicit

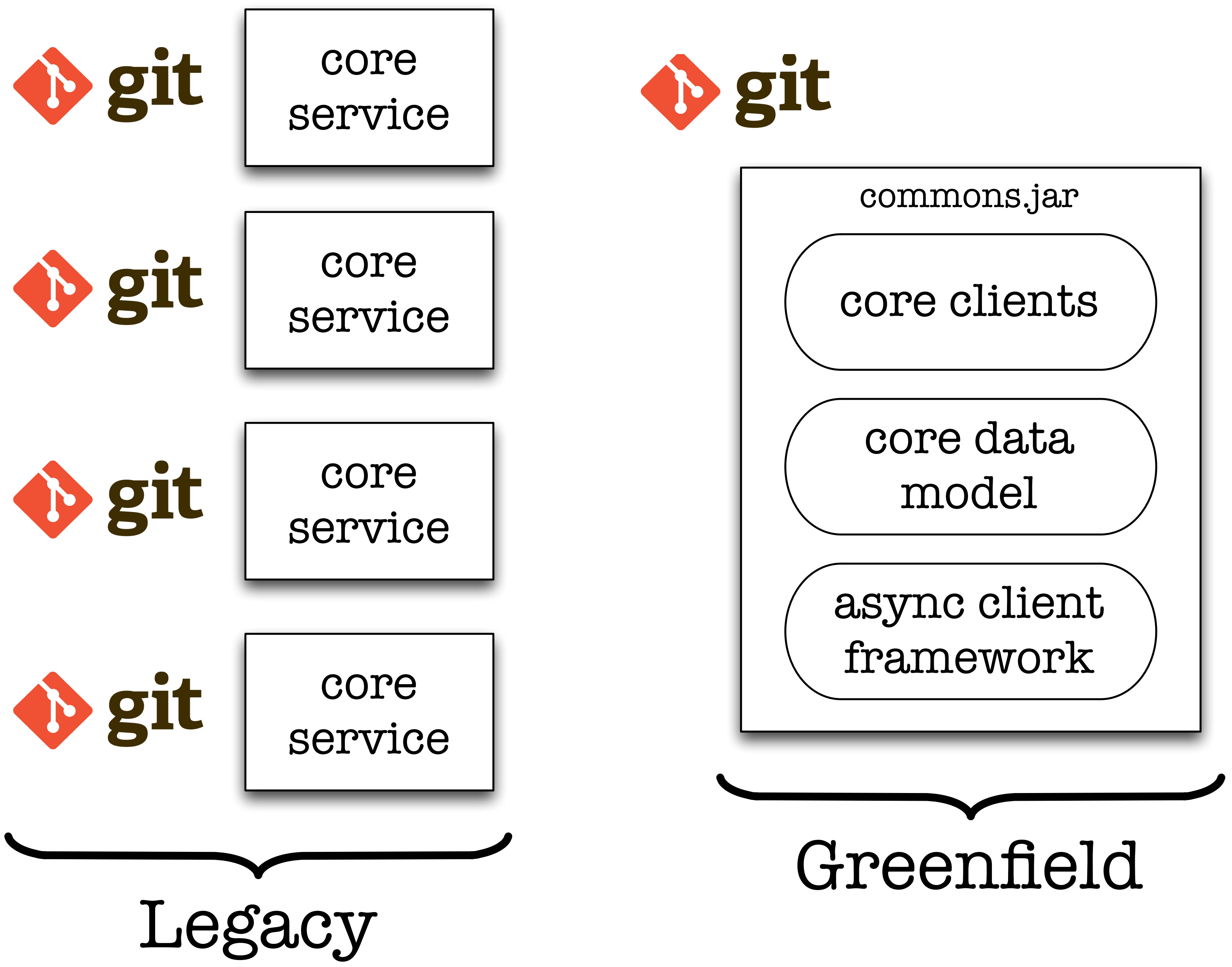
A data model and APIs for services

A data model and APIs for services

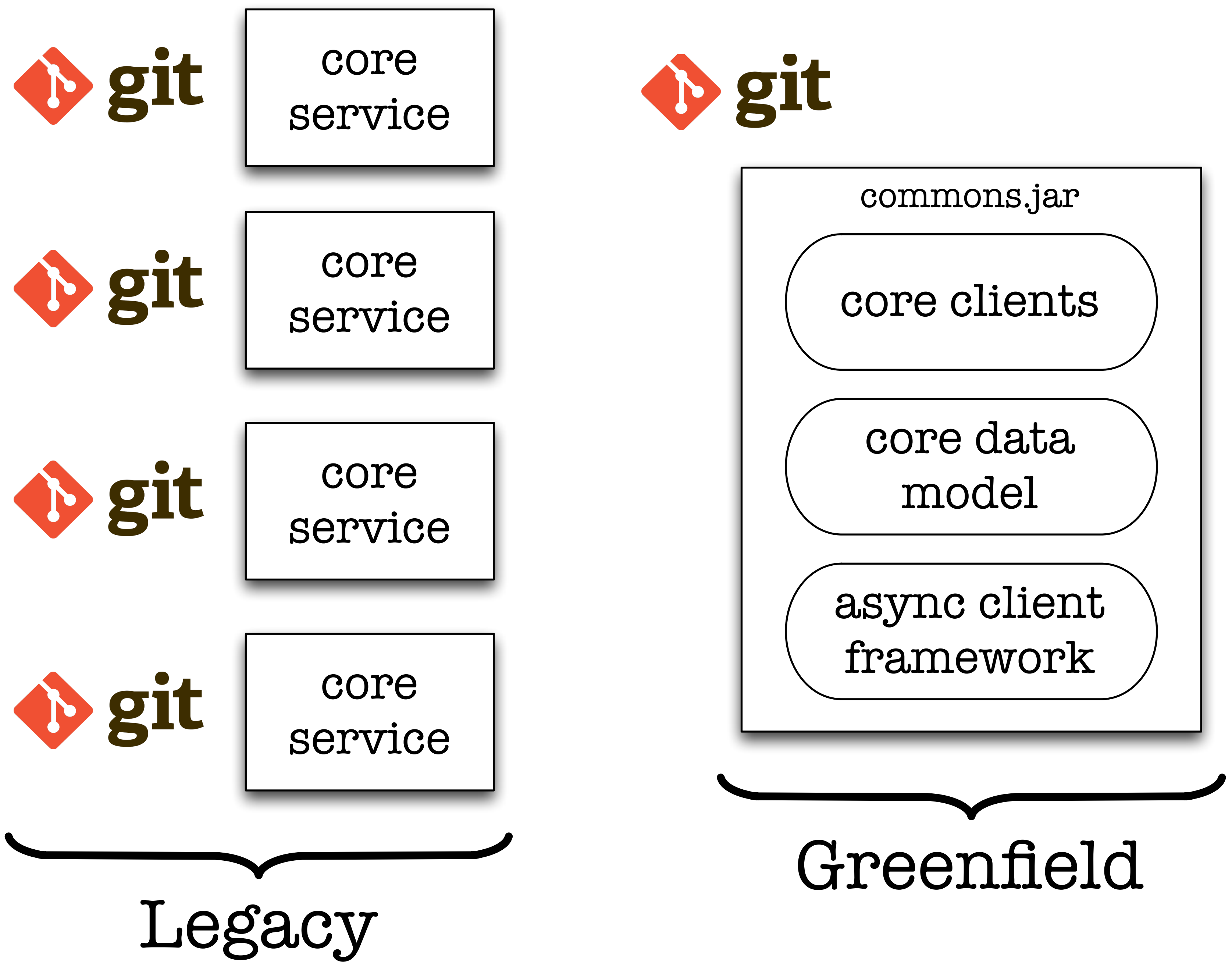
(aka, RPC)



http://git-scm.com/images/logos/downloads/Git-Logo-2Color.png

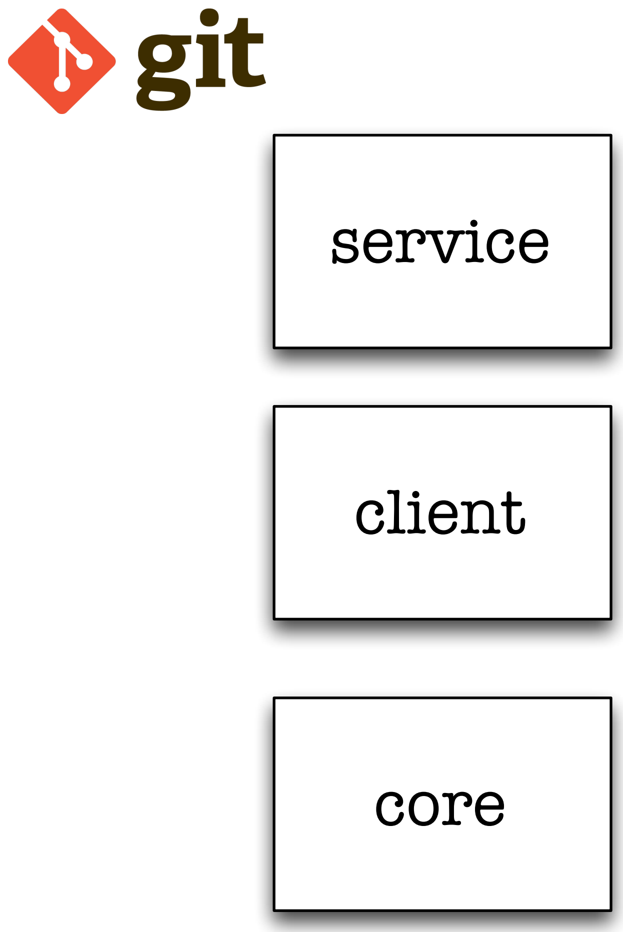


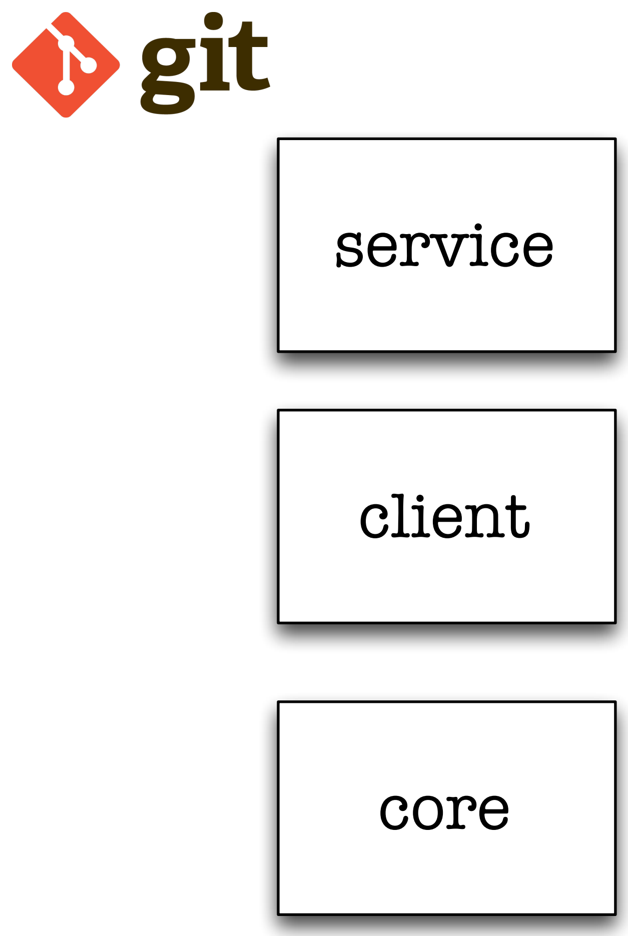
http://git-scm.com/images/logos/downloads/Git-Logo-2Color.png



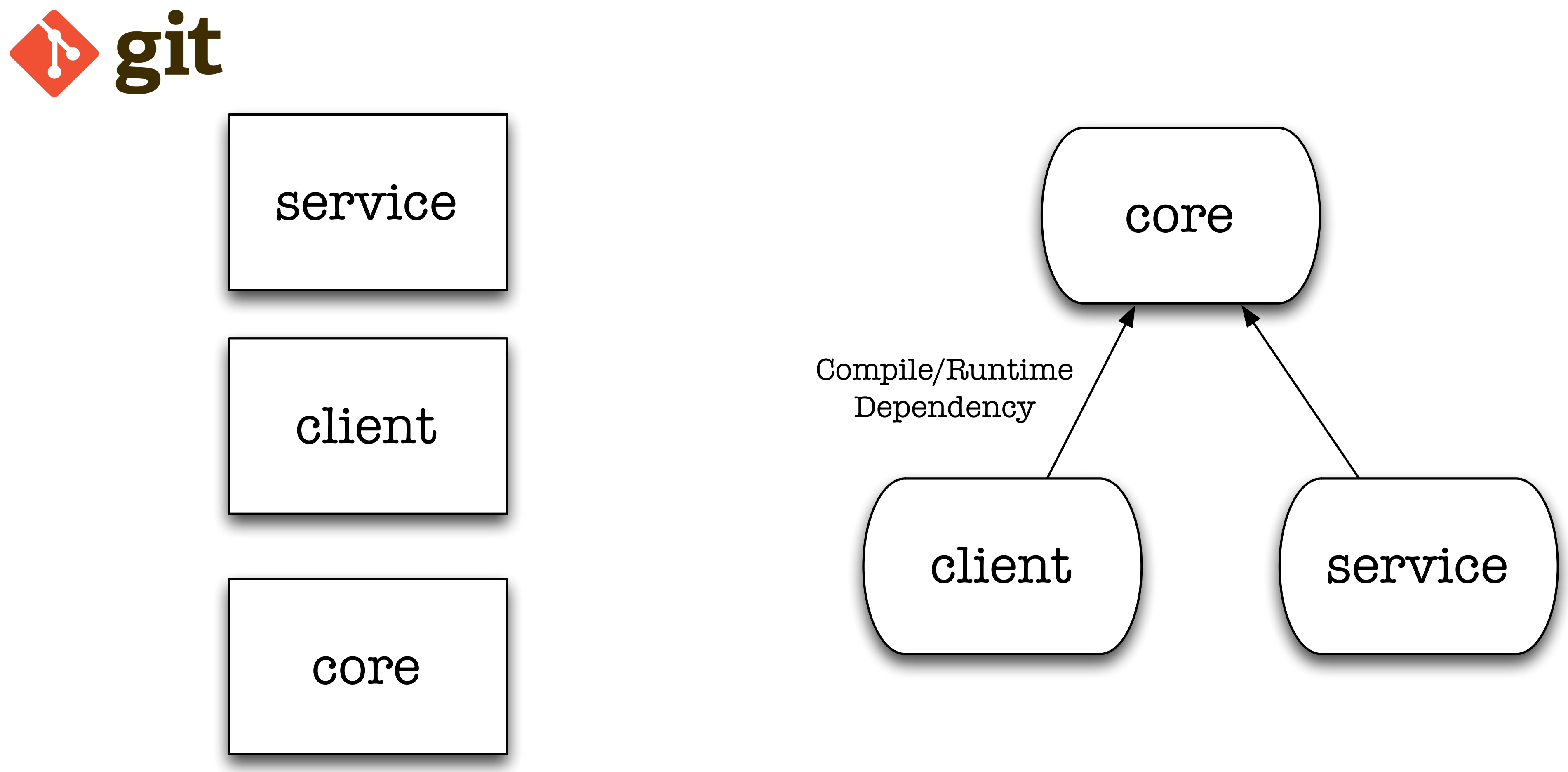
- Users
- Sales
- Products
- Skus
- Assets
- Targeting
- Auth

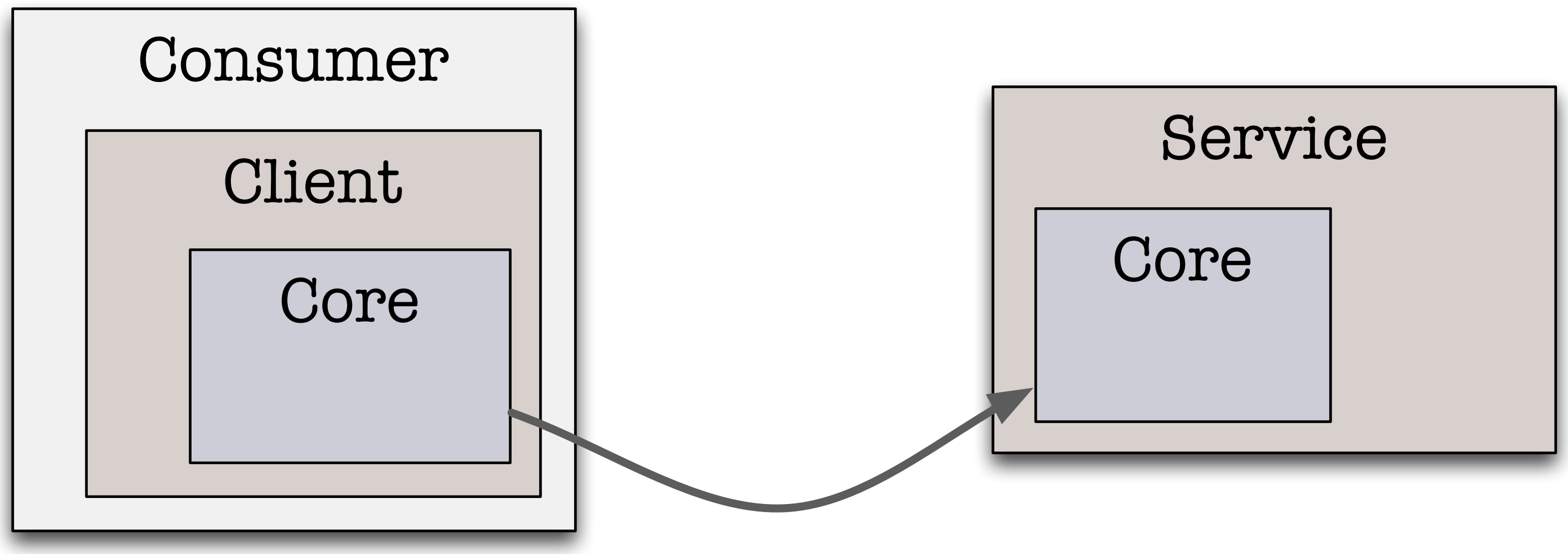
http://git-scm.com/images/logos/downloads/Git-Logo-2Color.png


git
service
client
core

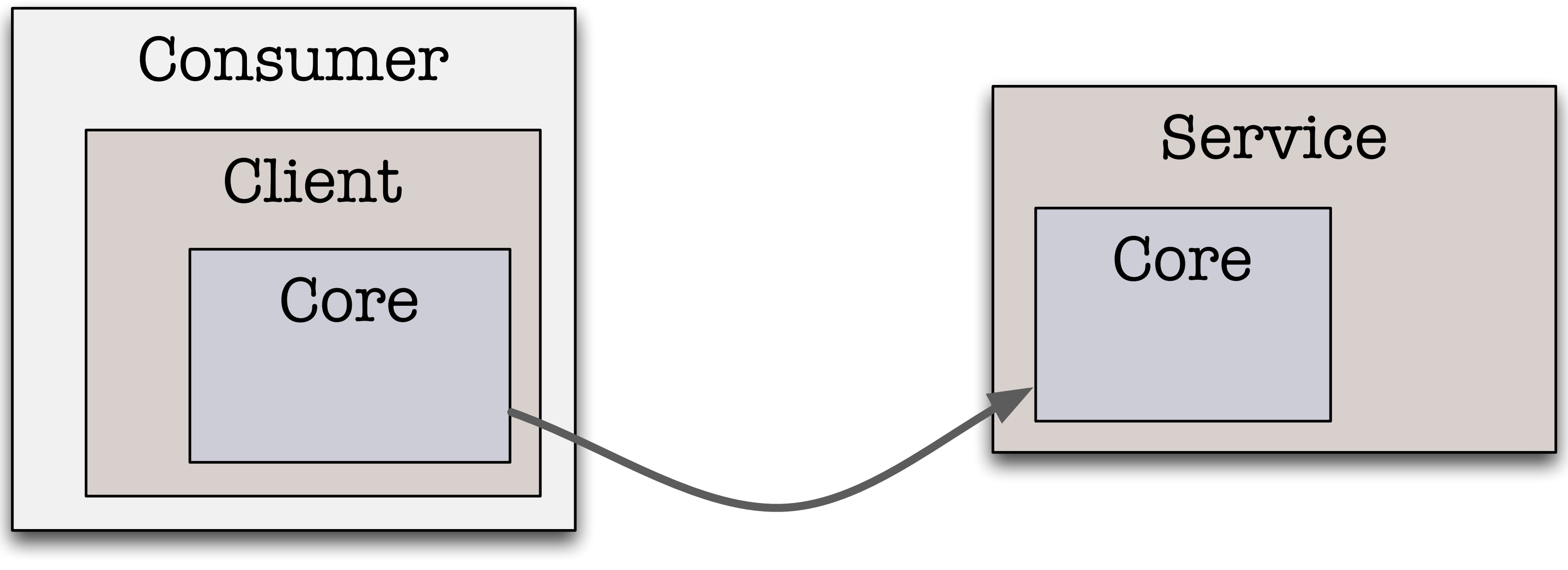


- RESTful
- Scala clients
- All APIs futures-based
- Case class schema

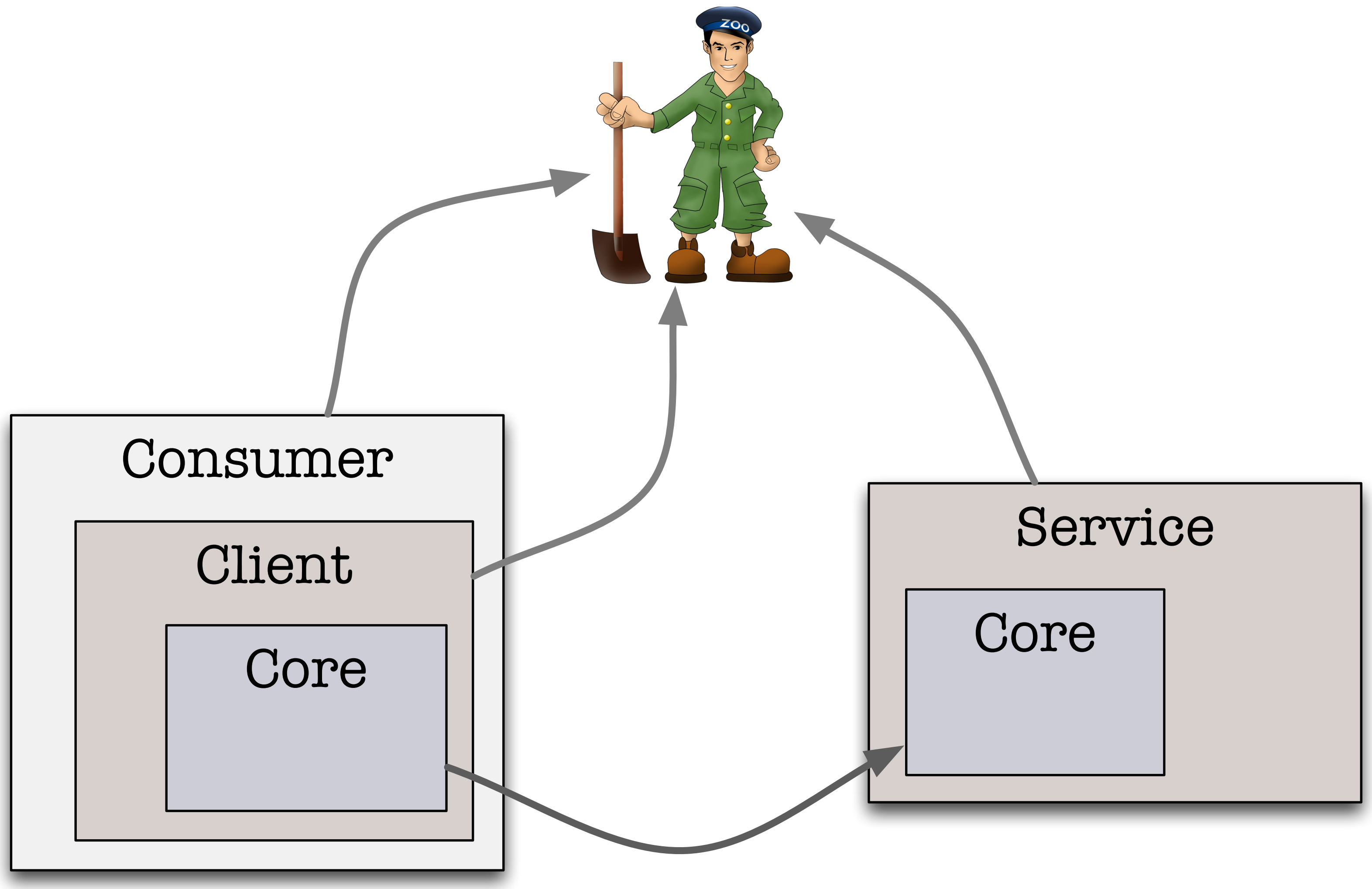

git
service
client
core
core
Compile/Runtime Dependency
client
service

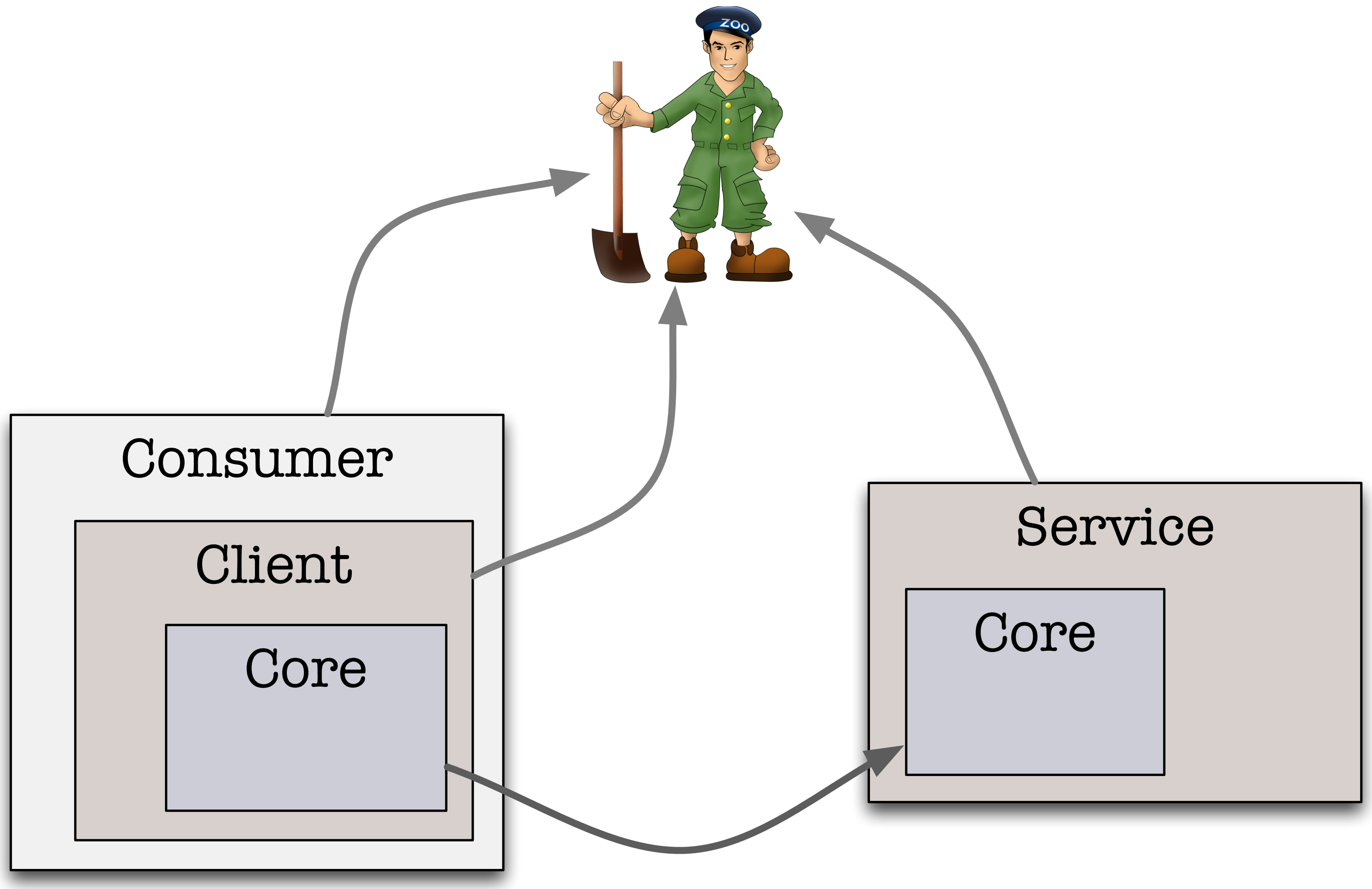


"Embassy Soil"



- Easy functional testing
- Service response capture
- Test linking
- Upgradable
- Emergent Regression
- Automated upgrades
- Compile farmers


Consumer
Client
Core
Service
Core



- Environment ⊕ Config
- Live updates
- Indirection
- Circuit Breaker

“All of this is completely wrong.” *

* Not an actual quote

1. The network is reliable
2. Latency is zero
3. Bandwidth is infinite
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. Transport cost is zero
8. The network is homogeneous

https://blogs.oracle.com/jag/resource/Fallacies.html

1. The network is reliable

https://blogs.oracle.com/jag/resource/Fallacies.html

1. The network is reliable

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

2.Latency is zero

https://blogs.oracle.com/jag/resource/Fallacies.html

2. Latency is zero

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

3.Bandwidth is infinite

https://blogs.oracle.com/jag/resource/Fallacies.html

3.Bandwidth is infinite

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

4. The network is secure

https://blogs.oracle.com/jag/resource/Fallacies.html

4. The network is secure

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

5.Topology doesn't change

https://blogs.oracle.com/jag/resource/Fallacies.html

5.Topology doesn't change

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

6.There is one administrator

https://blogs.oracle.com/jag/resource/Fallacies.html

6.There is one administrator

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

7.Transport cost is zero

https://blogs.oracle.com/jag/resource/Fallacies.html

7. Transport cost is zero

Resolved

https://blogs.oracle.com/jag/resource/Fallacies.html

8.The network is homogeneous

https://blogs.oracle.com/jag/resource/Fallacies.html

8.The network is homogeneous

Resolved

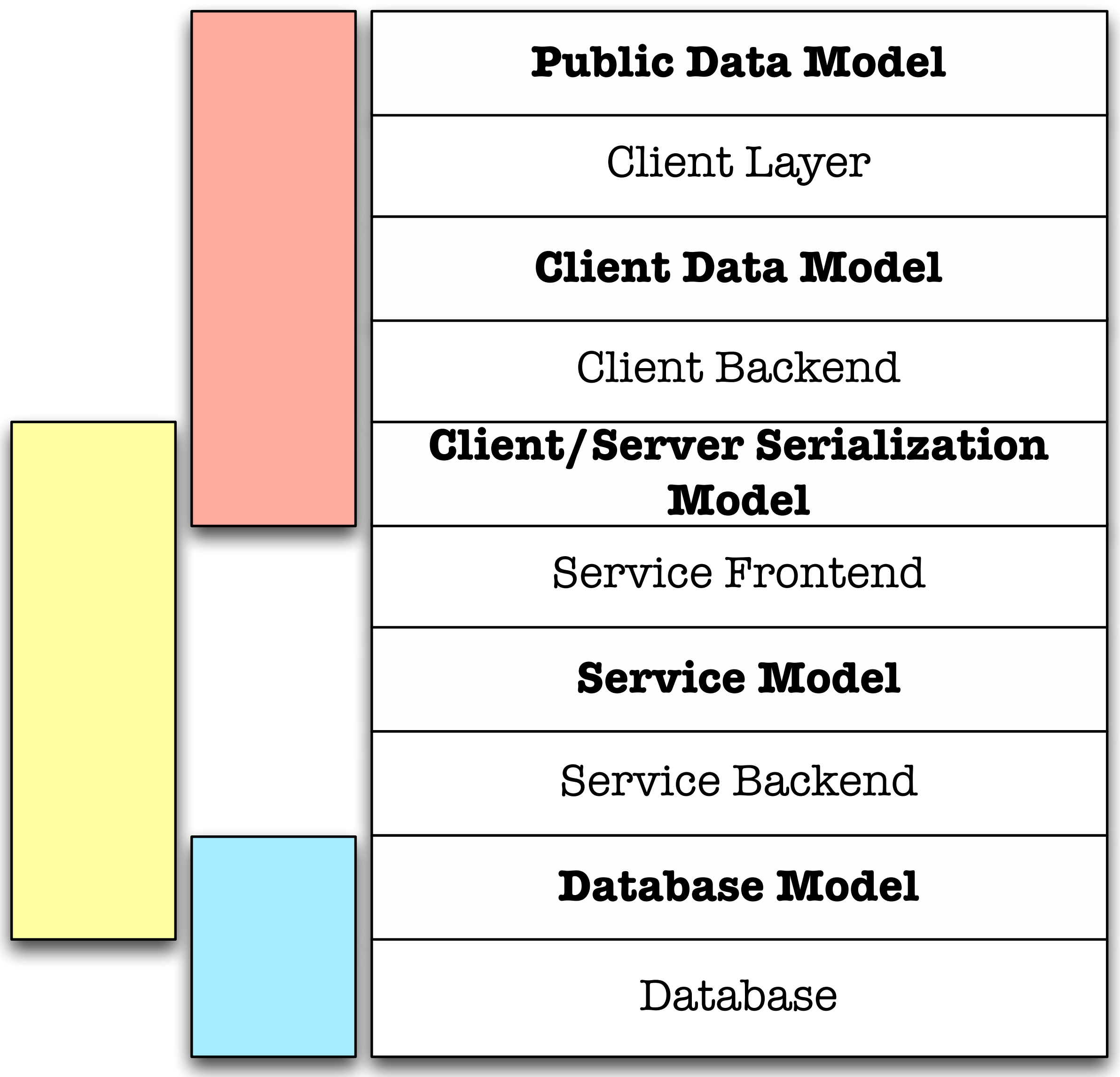
https://blogs.oracle.com/jag/resource/Fallacies.html

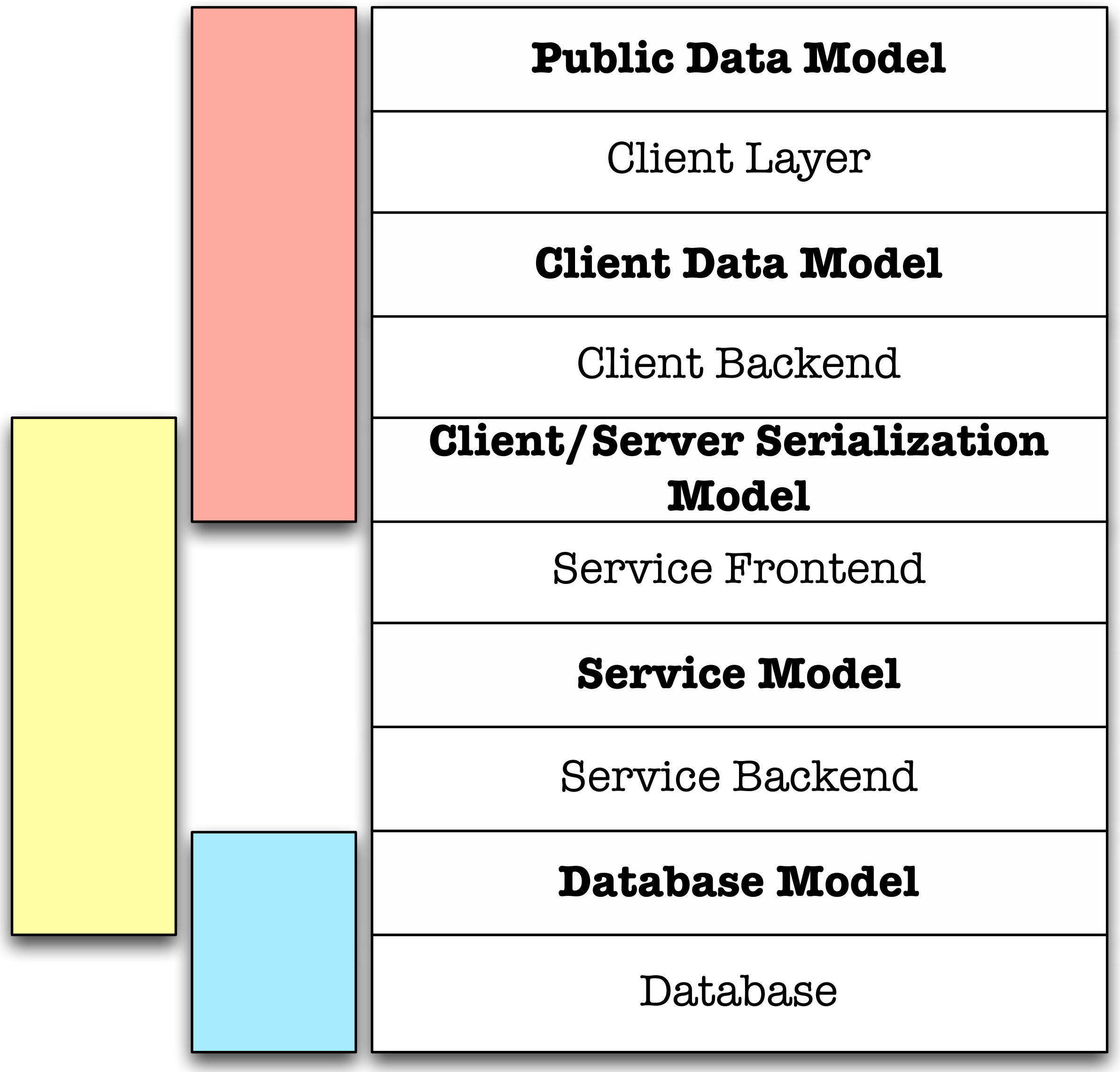
“Still Wrong.” *

* Possibly an actual quote

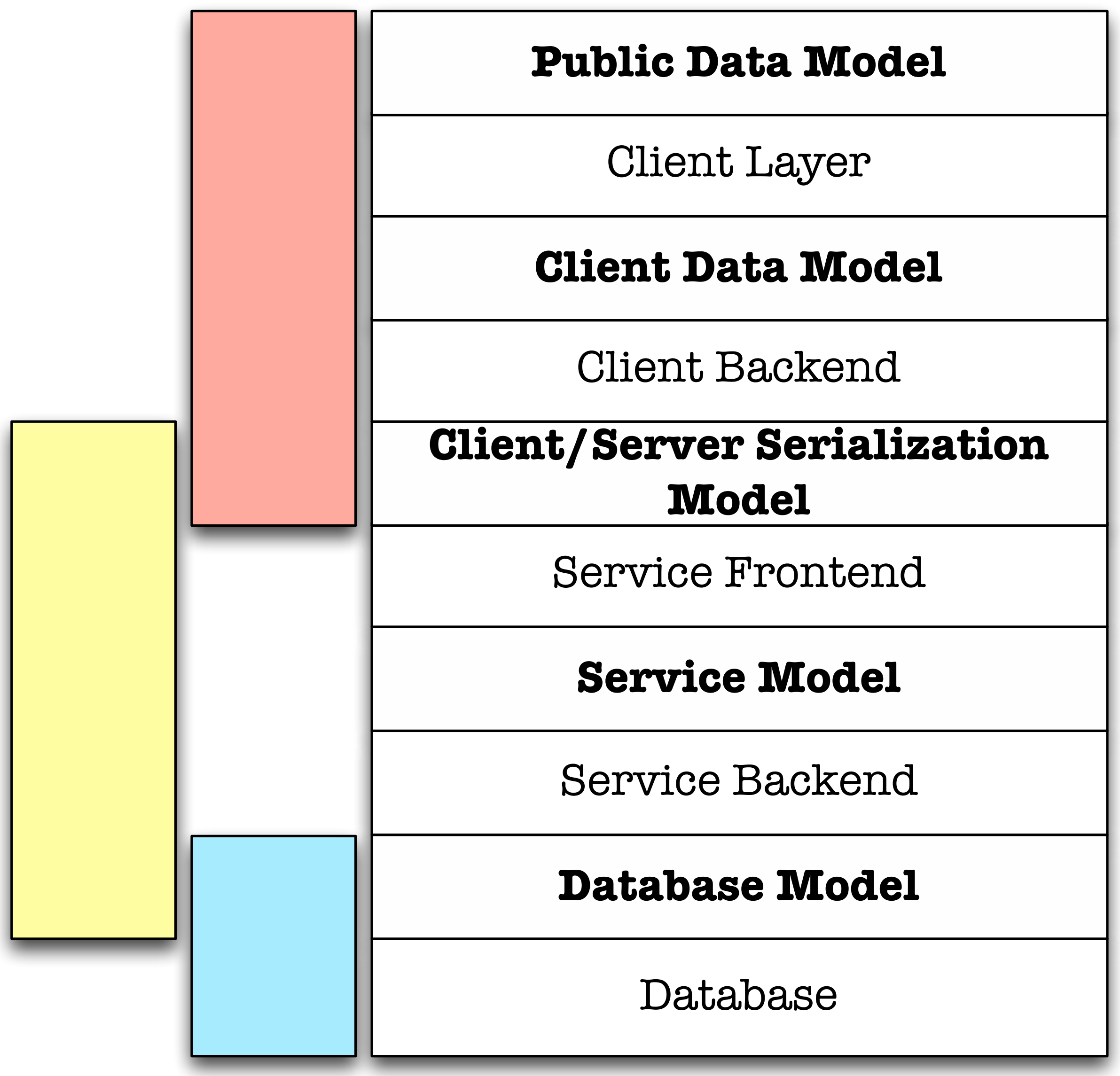
Convenience Over Correctness

http://steve.vinoski.net/blog/2008/07/01/convenience-over-correctness/


Public Data Model
Client Layer
Client Data Model
Client Backend
Client/Server Serialization Model
Service Frontend
Service Model
Service Backend
Database Model
Database

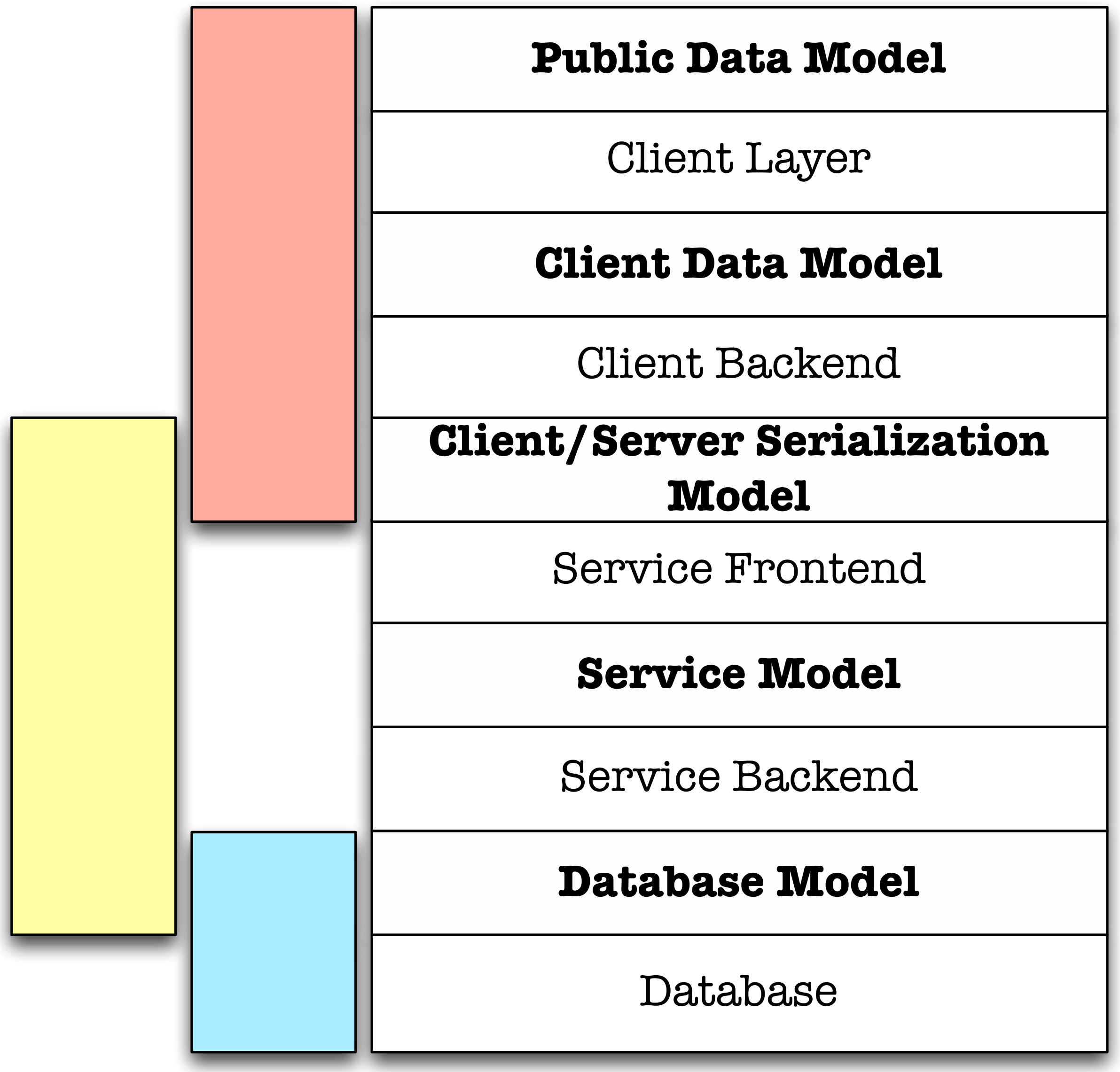


- So many models
- Corners are cut
- Typesafe helps
- Conflation?
- Just the data



- So many models
- Corners are cut
- Typesafe helps
- Conflation?
- Just the data

"Works in Practice for some use cases"



- So many models
- Corners are cut
- Typesafe helps
- Conflation?
- Just the data

"Works in Practice for some use cases"

"No free silver bullet lunches."

However...

However...

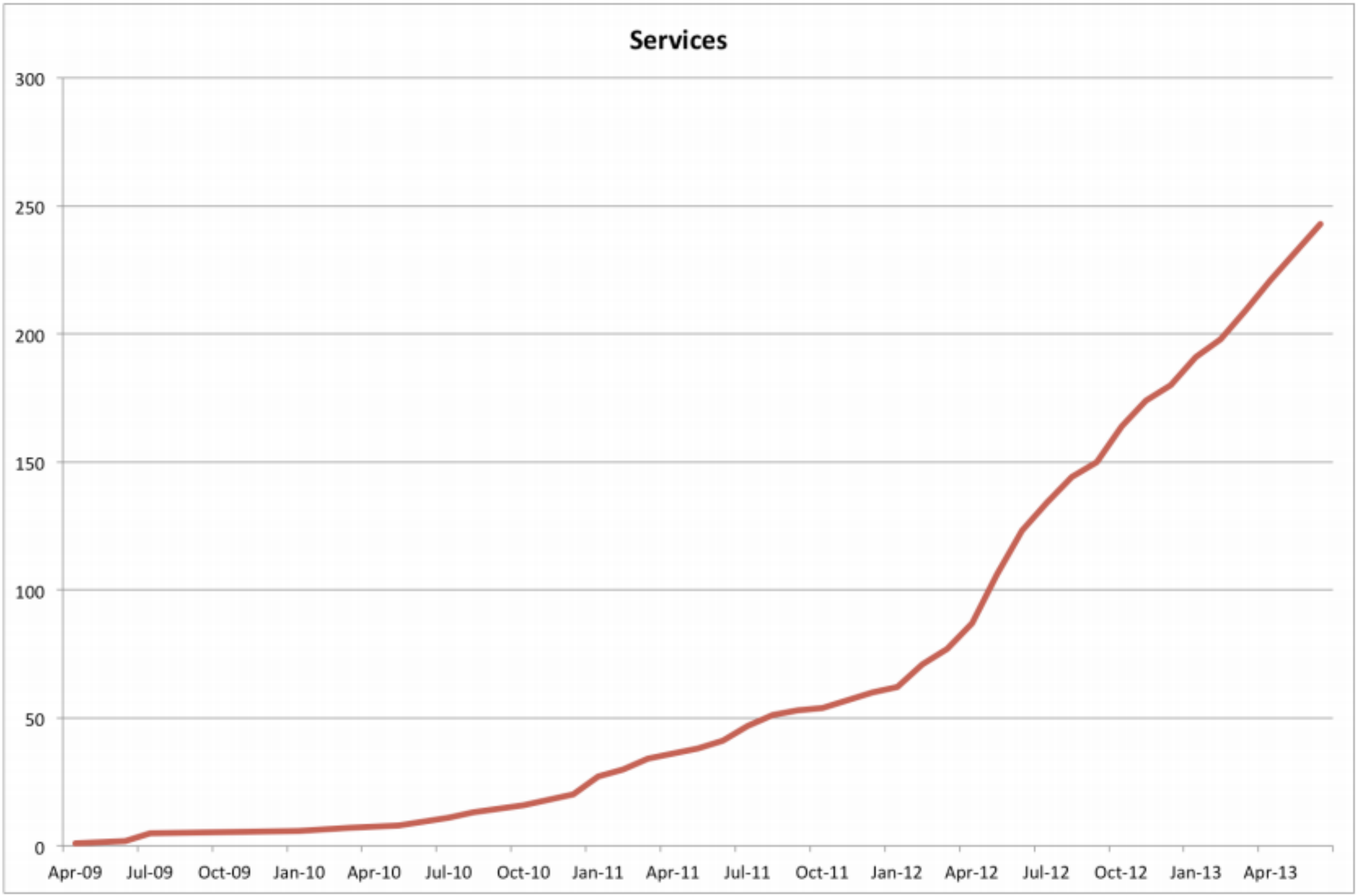
- No machine generated stubs
- Embassy-Oriented Programming
- Lots of indirection
- Type-system support for failures

What Actually Sucks about RPC:

What Actually Sucks about RPC:

- Remote objects
- Failures
- Idempotency

Ignoring all that was Too Easy.


Services
300
250
200
150
100
50
0
Apr-09
Jul-09
Oct-09
Jan-10
Apr-10
Jul-10
Oct-10
Jan-11
Apr-11
Jul-11
Oct-11
Jan-12
Apr-12
Jul-12
Oct-12
Jan-13
Apr-13

Batch Jobs

1. The network is reliable
2. Latency is zero
3. Bandwidth is infinite
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. Transport cost is zero
8. The network is homogeneous

Law of Instrument

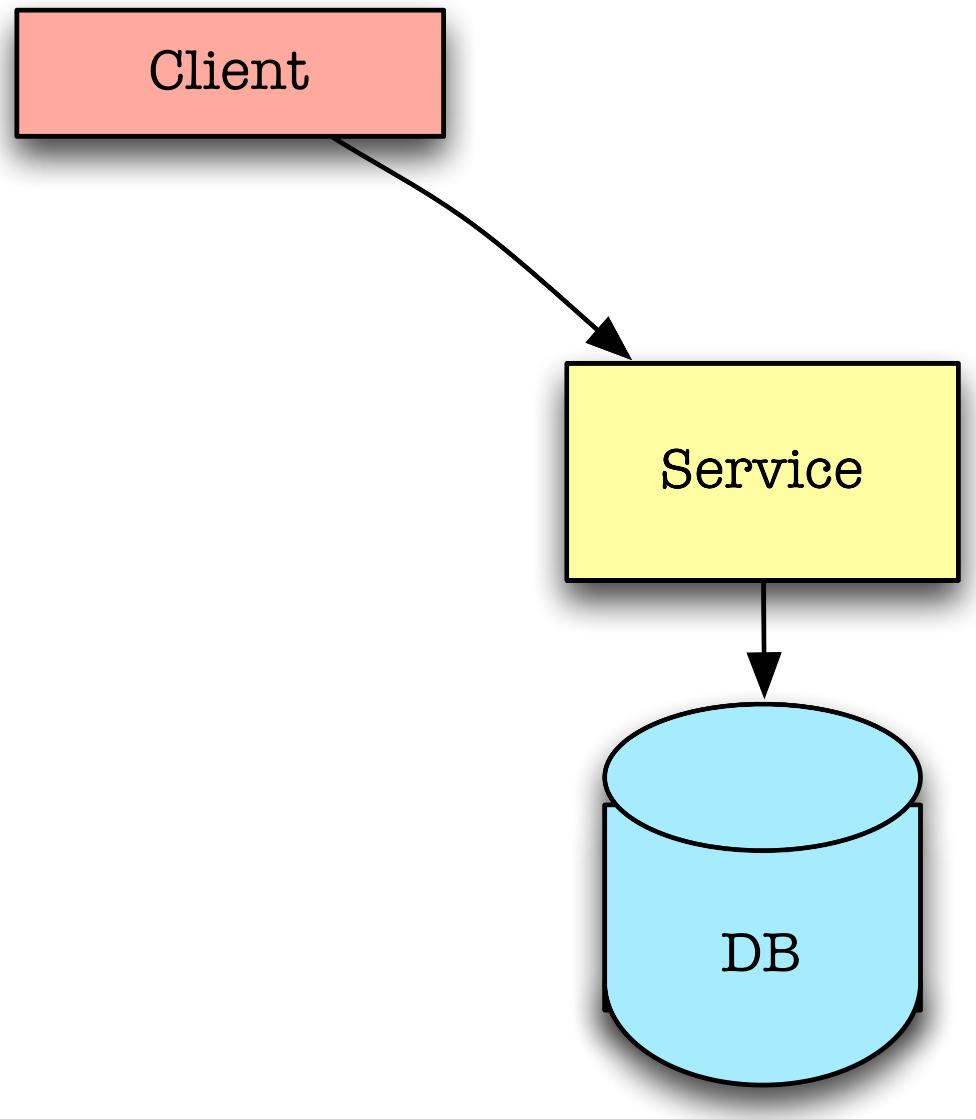
Pulling data.

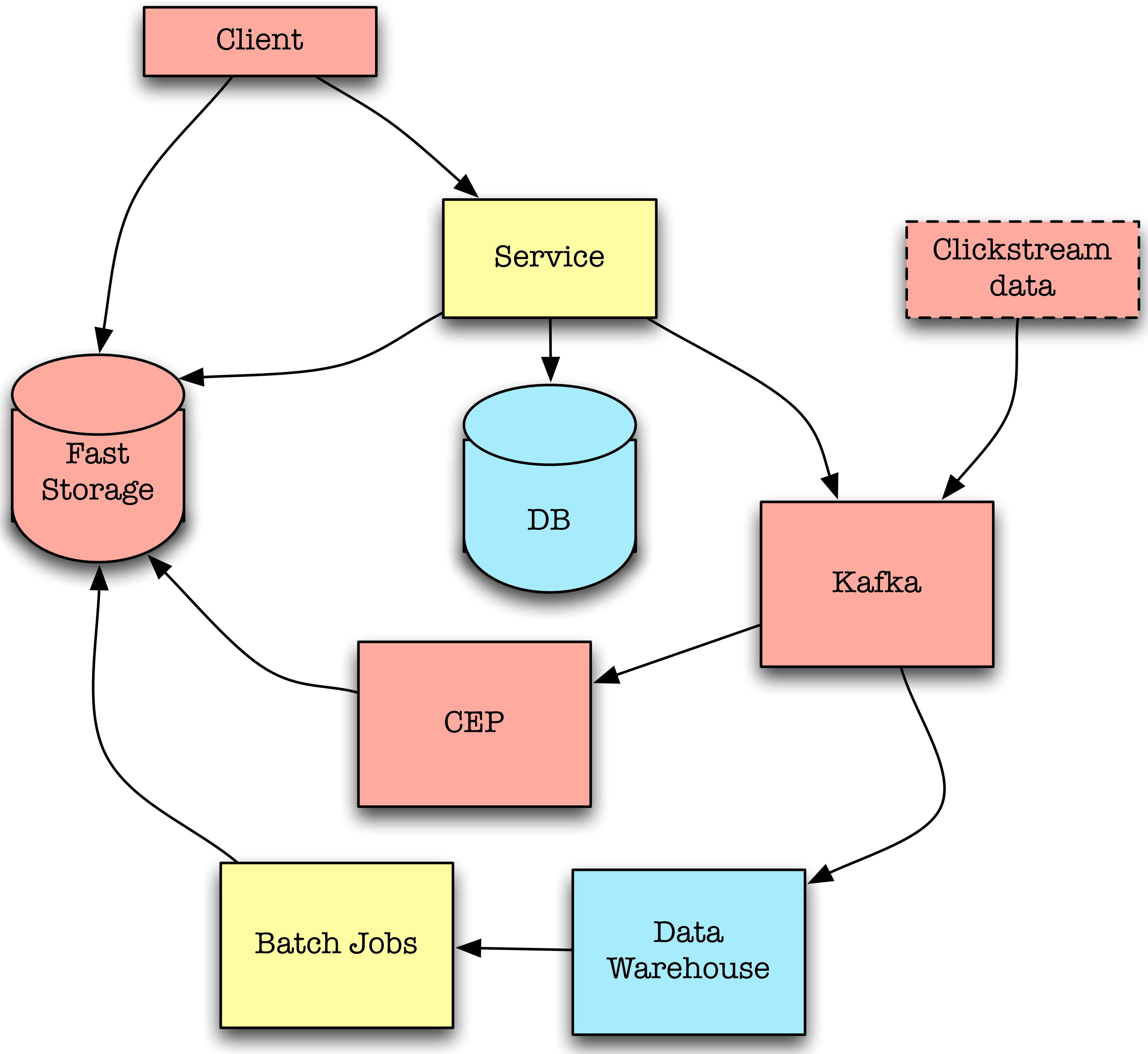
Pushing code.

SOA, Reloaded

SOA, Reloaded

- CR~~UD~~
- Event Streams
- Batch Processing
- Lambda Architecture
- CQRS


Client
Service
DB


Client
Service
Clickstream data
Fast Storage
DB
Kafka
CEP
Batch Jobs
Data Warehouse

http://tech.gilt.com

join us.
new york & dublin