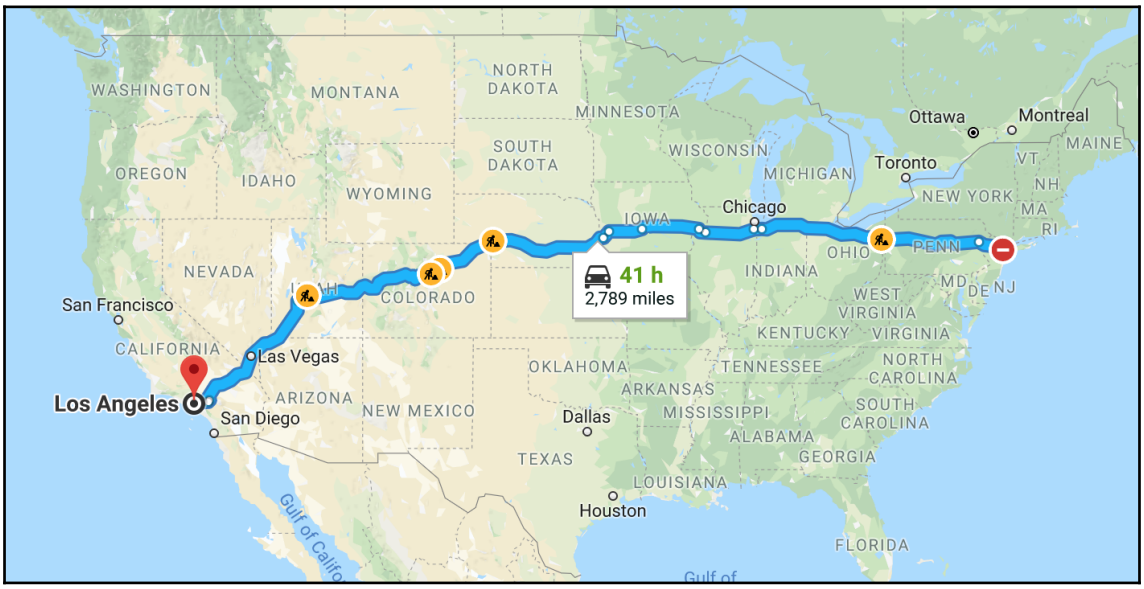
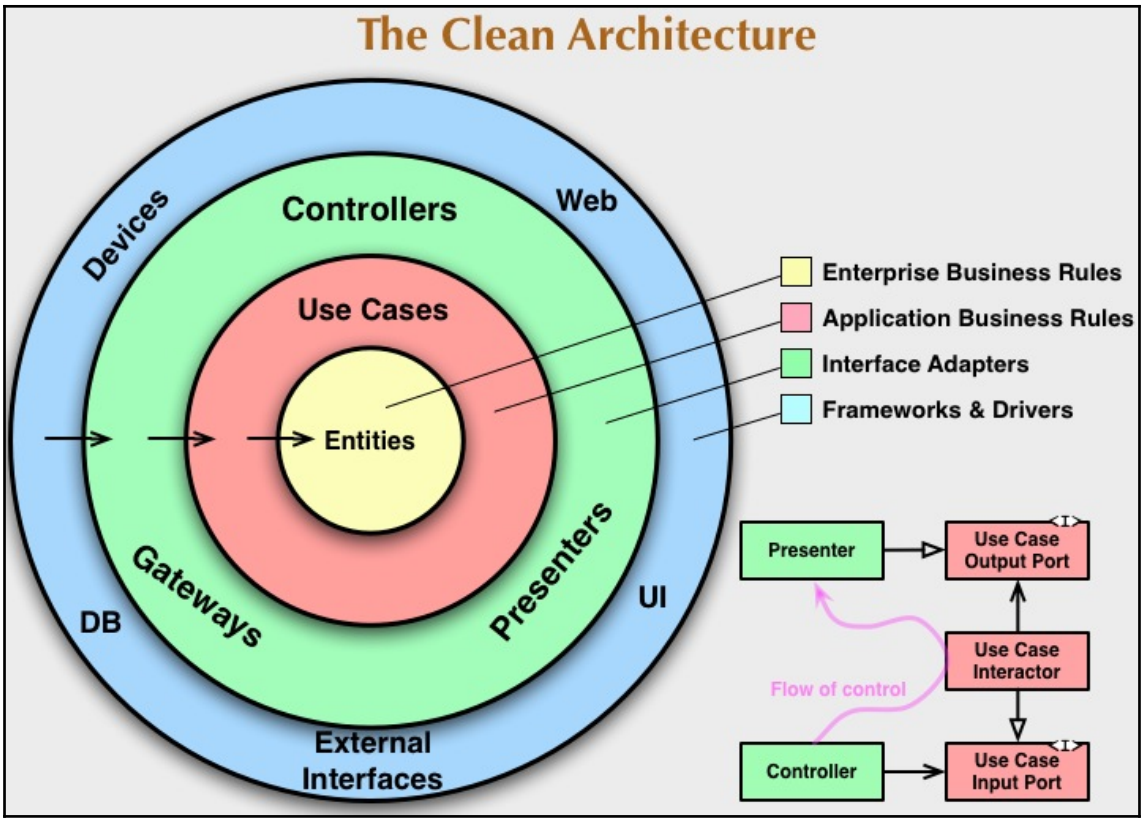
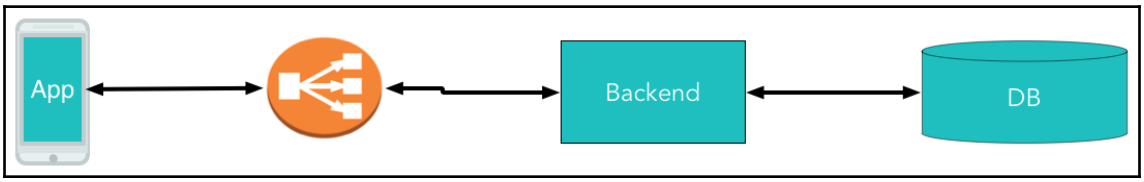
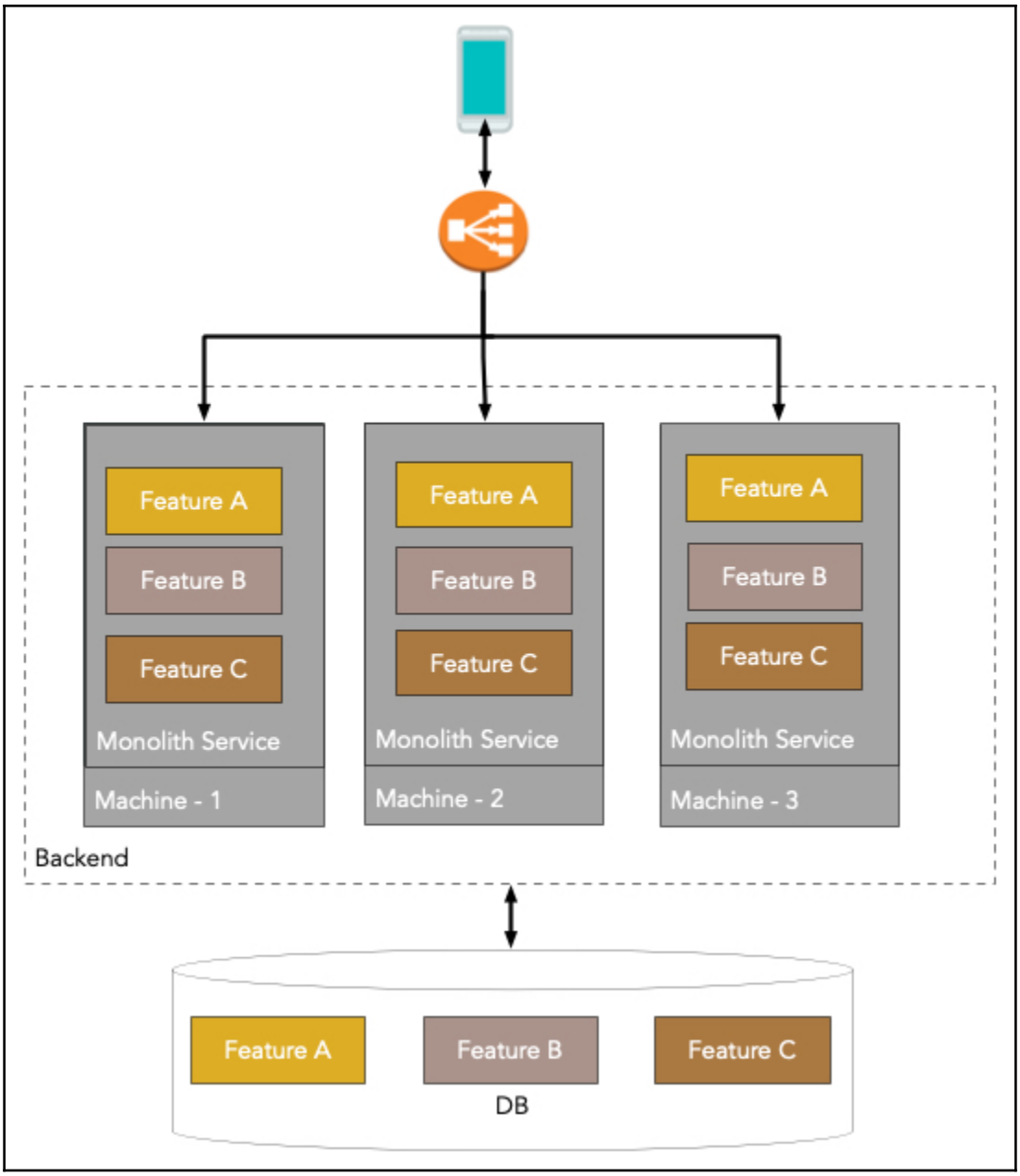
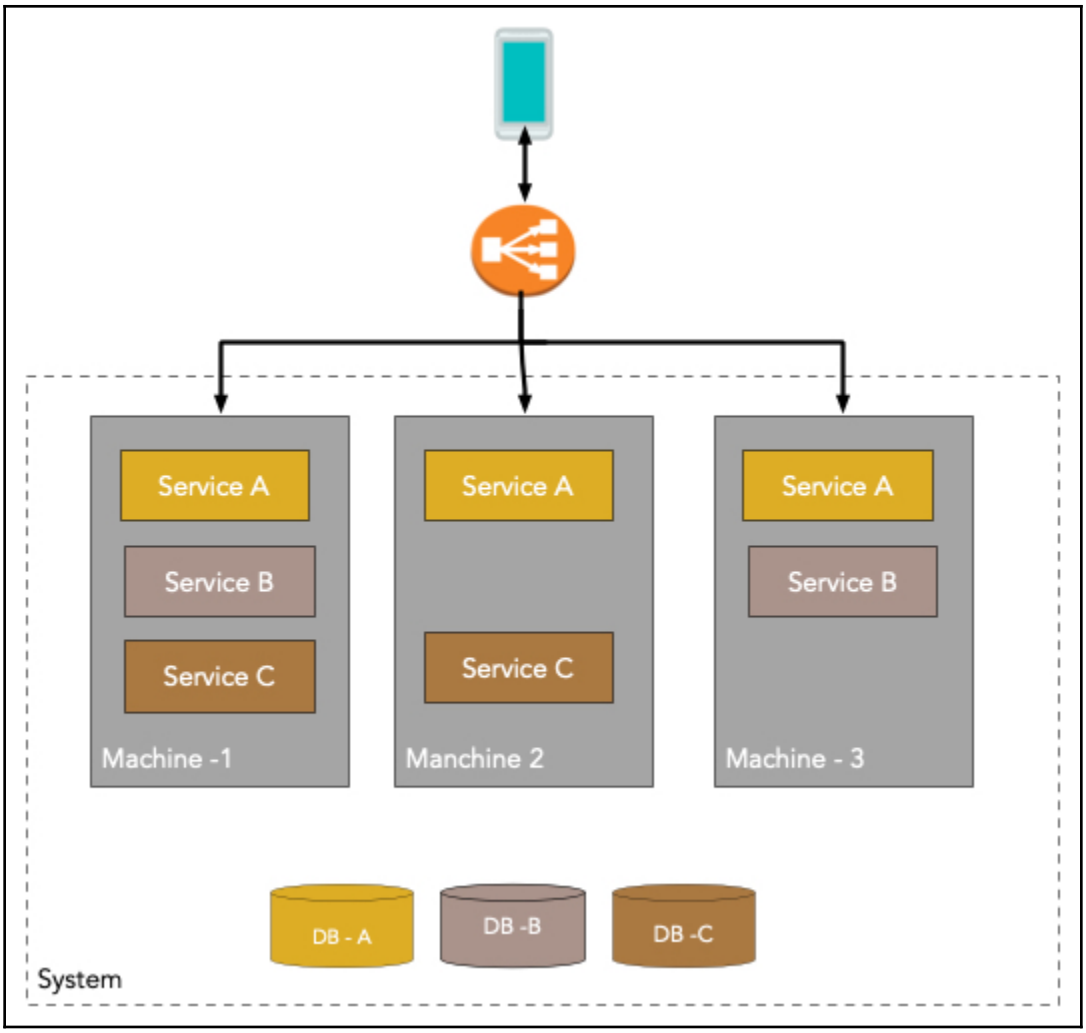
Chapter 1: Building Big with Go

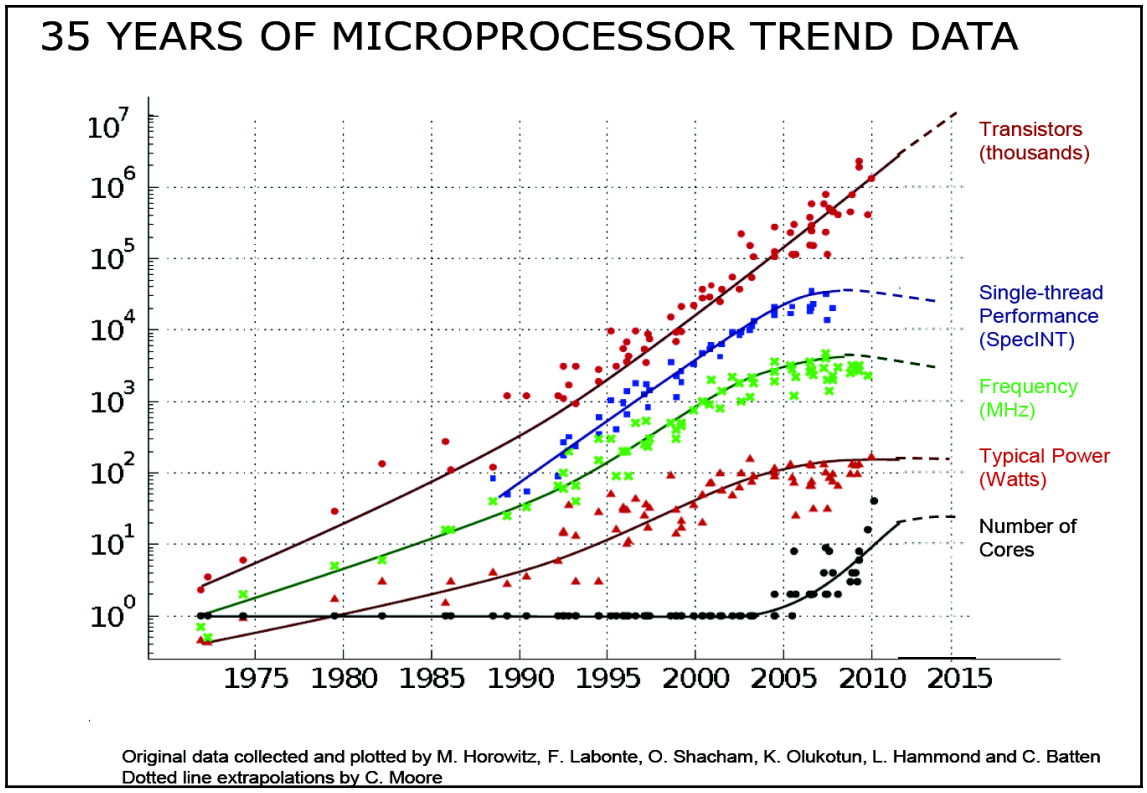


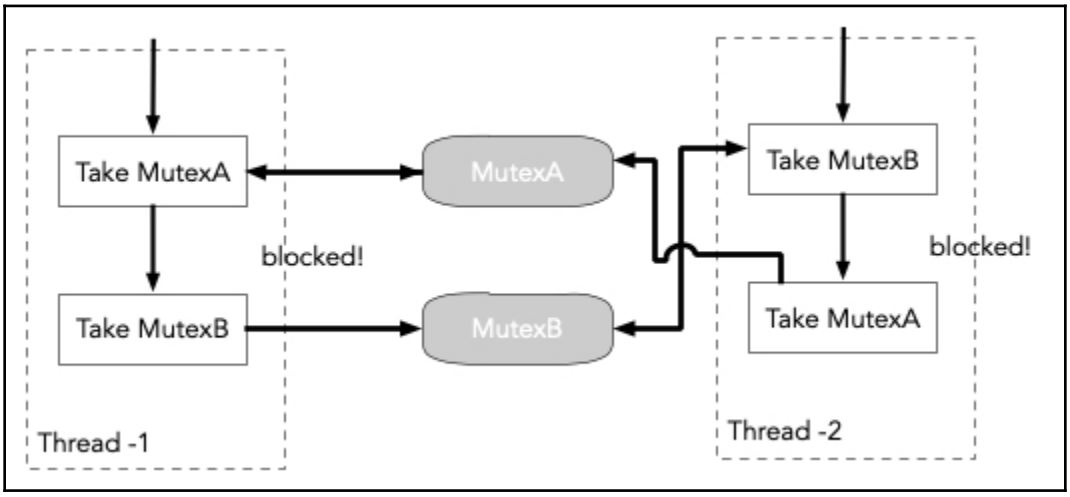

The Clean Architecture
Devices
Controllers
Web
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers
Presenter
Use Case
Output Port
Use Case
Interactor
Flow of control
Controller
Use Case
Input Port


App
Backend
DB

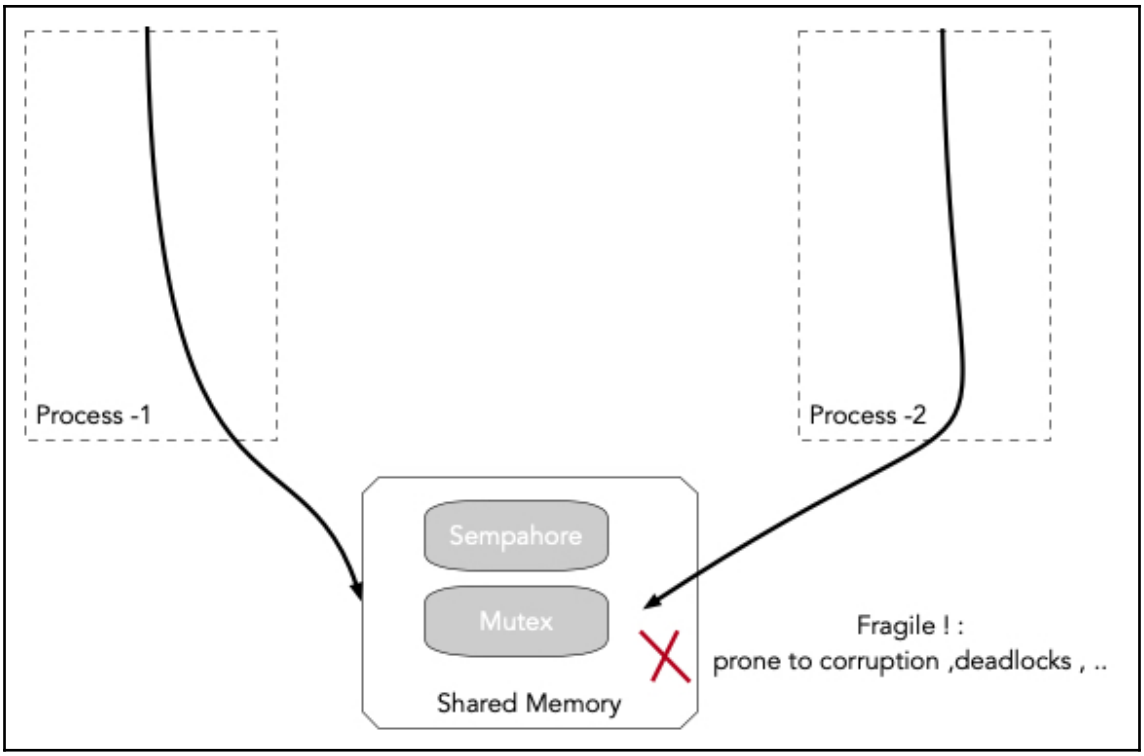

Feature A
Feature B
Feature C
Monolith Service
Machine - 1
Feature A
Feature B
Feature C
Monolith Service
Machine - 2
Feature A
Feature B
Feature C
Monolith Service
Machine - 3
Backend
Feature A
Feature B
Feature C
DB

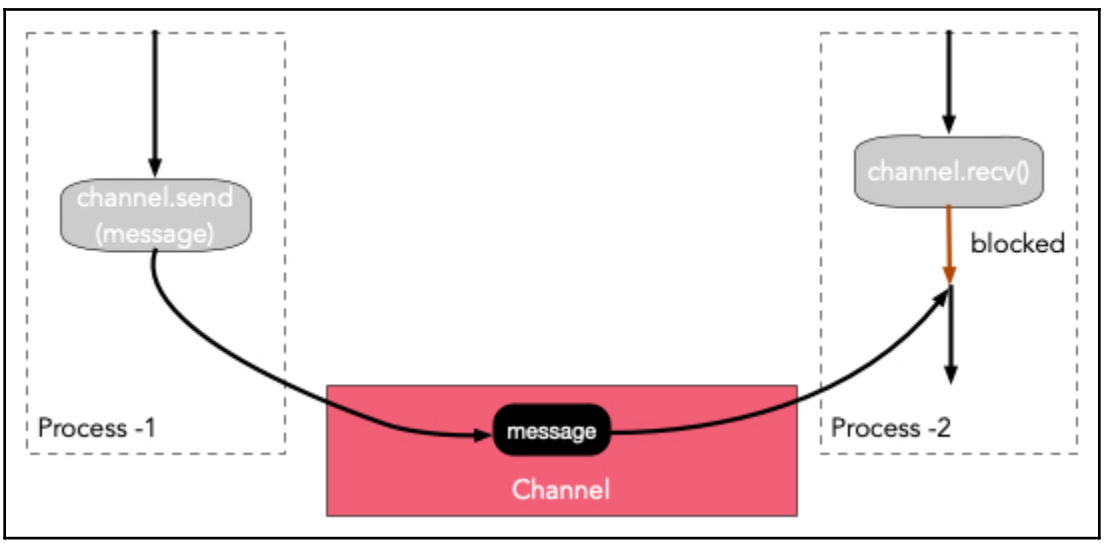

Service A
Service B
Service C
Machine -1
Service A
Service C
Manchine 2
Service A
Service B
Machine - 3
DB - A
DB -B
DB -C
System

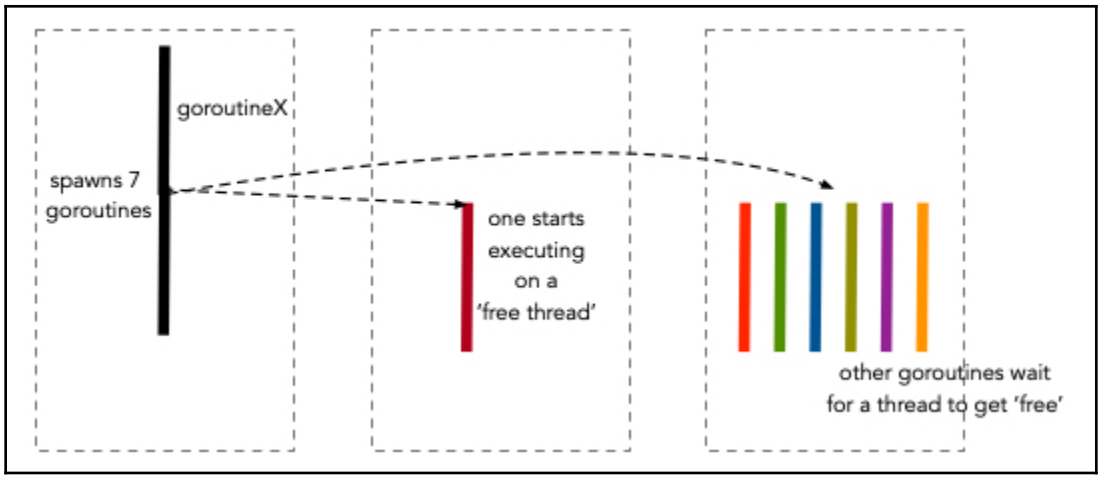

35 YEARS OF MICROPROCESSOR TREND DATA
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0
1975 1980 1985 1990 1995 2000 2005 2010 2015
Transistors (thousands)
Single-thread Performance (SpecINT)
Frequency (MHz)
Typical Power (Watts)
Number of Cores
Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore


Take MutexA
MutexA
Take MutexB
blocked!
Take MutexB
MutexB
Take MutexA
blocked!
Thread -1
Thread -2

# Ranking	Programming Language	Percentage (Change)	Trend
1	JavaScript	22.525% (-5.733%)	
2	Python	15.861% (+0.943%)	
3	Java	10.197% (+0.270%)	
4	Ruby	7.041% (-0.088%)	^
5	Go	6.941% (+1.967%)	^
6	PHP	6.717% (-0.730%)	v
7	C++	6.373% (+1.089%)	v
8	C	3.519% (+0.150%)	
9	TypeScript	3.381% (+1.241%)	^
10	C#	3.309% (-0.042%)	v
11	Shell	2.291% (+0.209%)	
12	Scala	1.558% (+0.187%)	^
13	Swift	1.094% (-0.324%)	v
14	Rust	1.074% (+0.208%)	^
15	DM	0.873% (+0.532%)	^^
16	Objective-C	0.831% (-0.176%)	v


Process -1
Process -2
Sempahore
Mutex
Shared Memory
Fragile ! :
prone to corruption ,deadlocks , ..


channel.send
(message)
channel.recv()
blocked
Process -1
message
Channel
Process -2



org.springframework.aop.framework

Class AbstractSingletonProxyFactoryBean

java.lang.Object
  └ org.springframework.aop.framework.ProxyConfig
    └ **org.springframework.aop.framework.AbstractSingletonProxyFactoryBean**

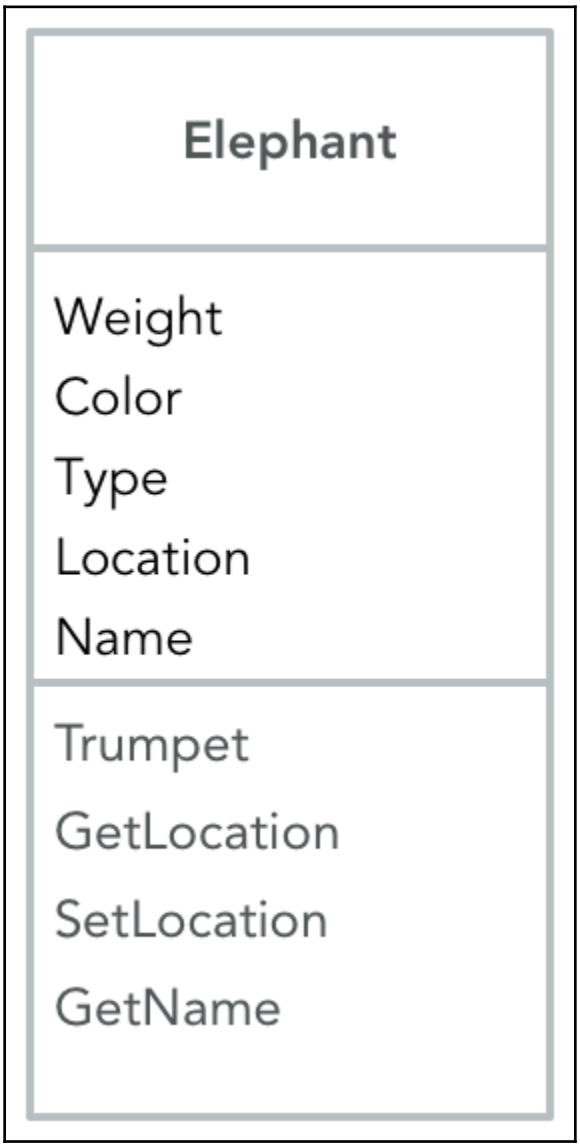
All Implemented Interfaces:
Serializable, BeanClassLoaderAware, FactoryBean, InitializingBean

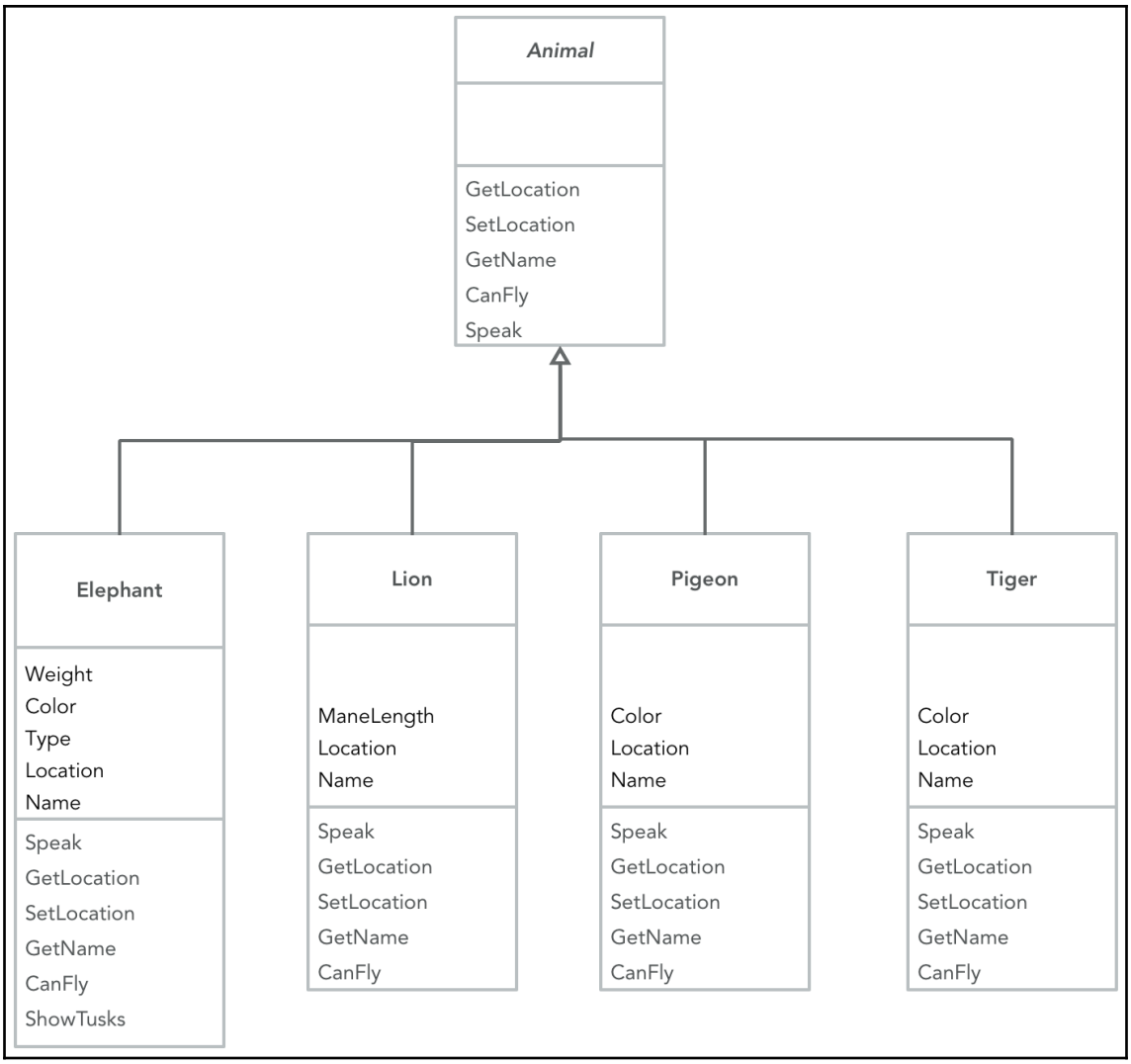
Direct Known Subclasses:
TransactionProxyFactoryBean

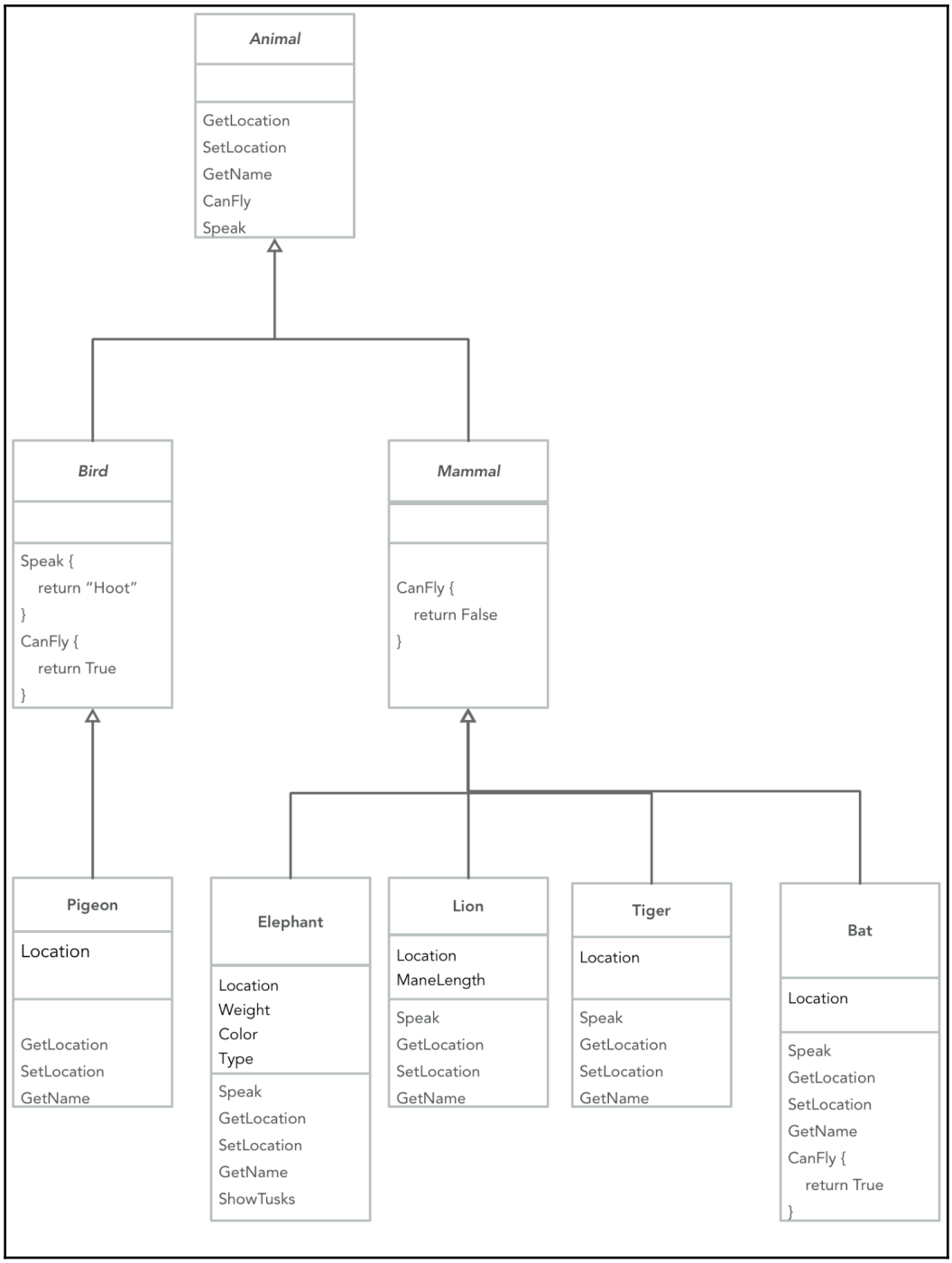
```
public abstract class AbstractSingletonProxyFactoryBean
extends ProxyConfig
implements FactoryBean, BeanClassLoaderAware, InitializingBean
```

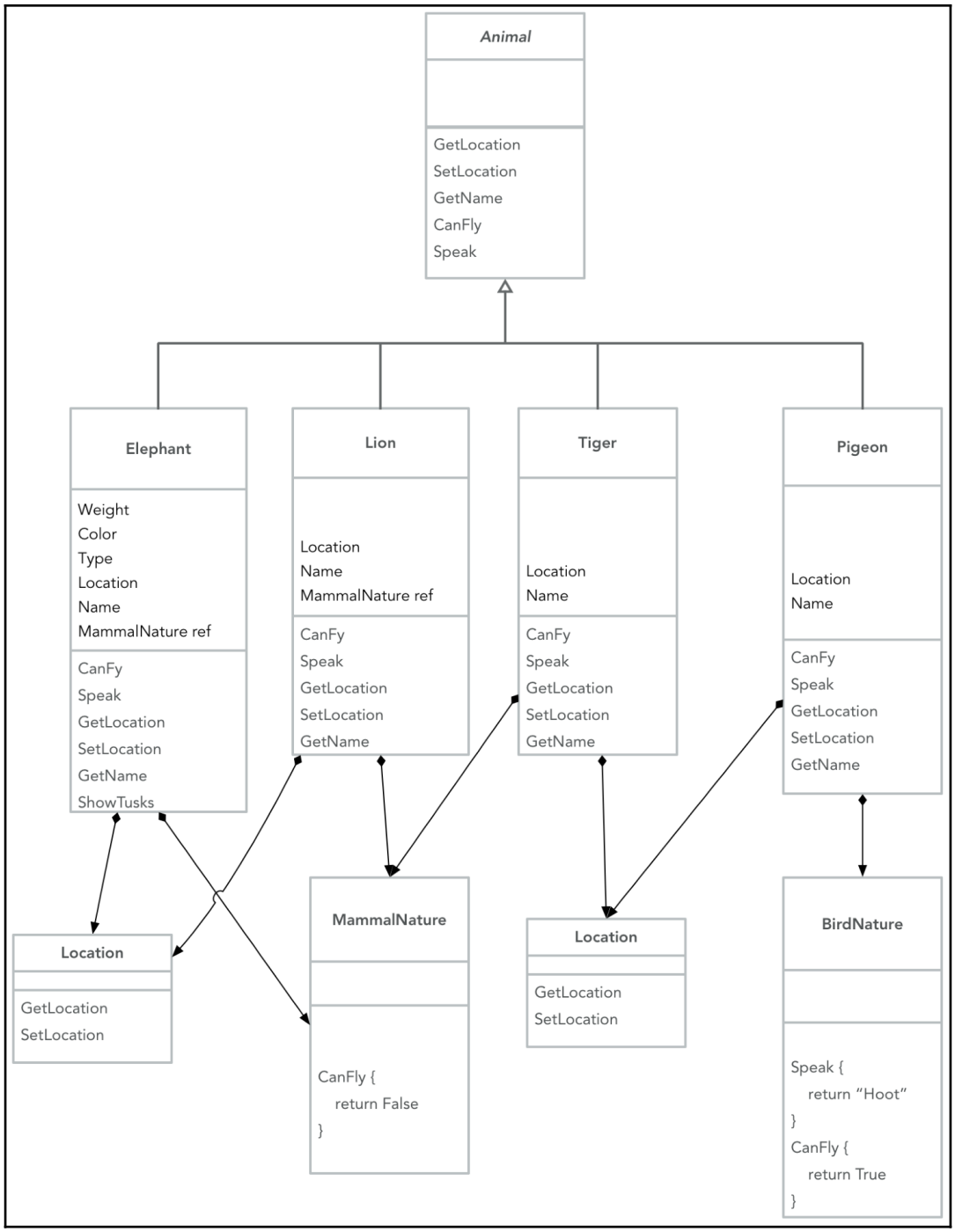
Convenient proxy factory bean superclass for proxy factory beans that create only singletons.

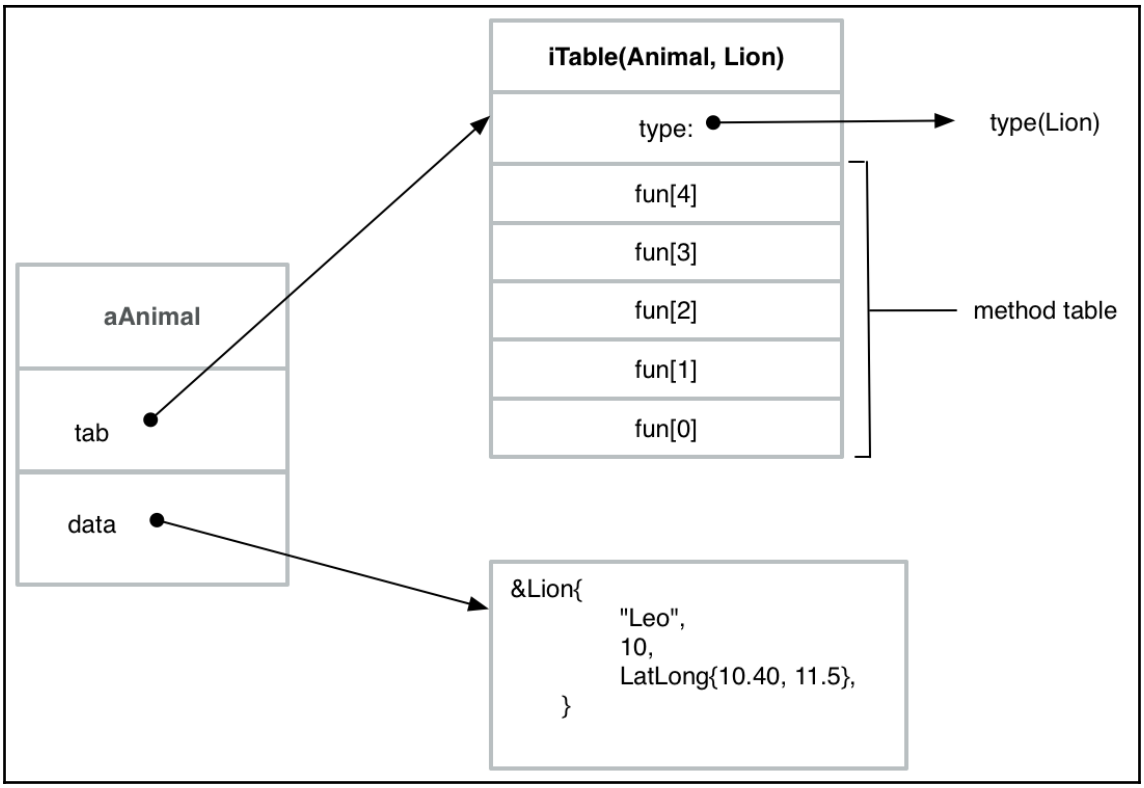
Chapter 2: Packaging Code

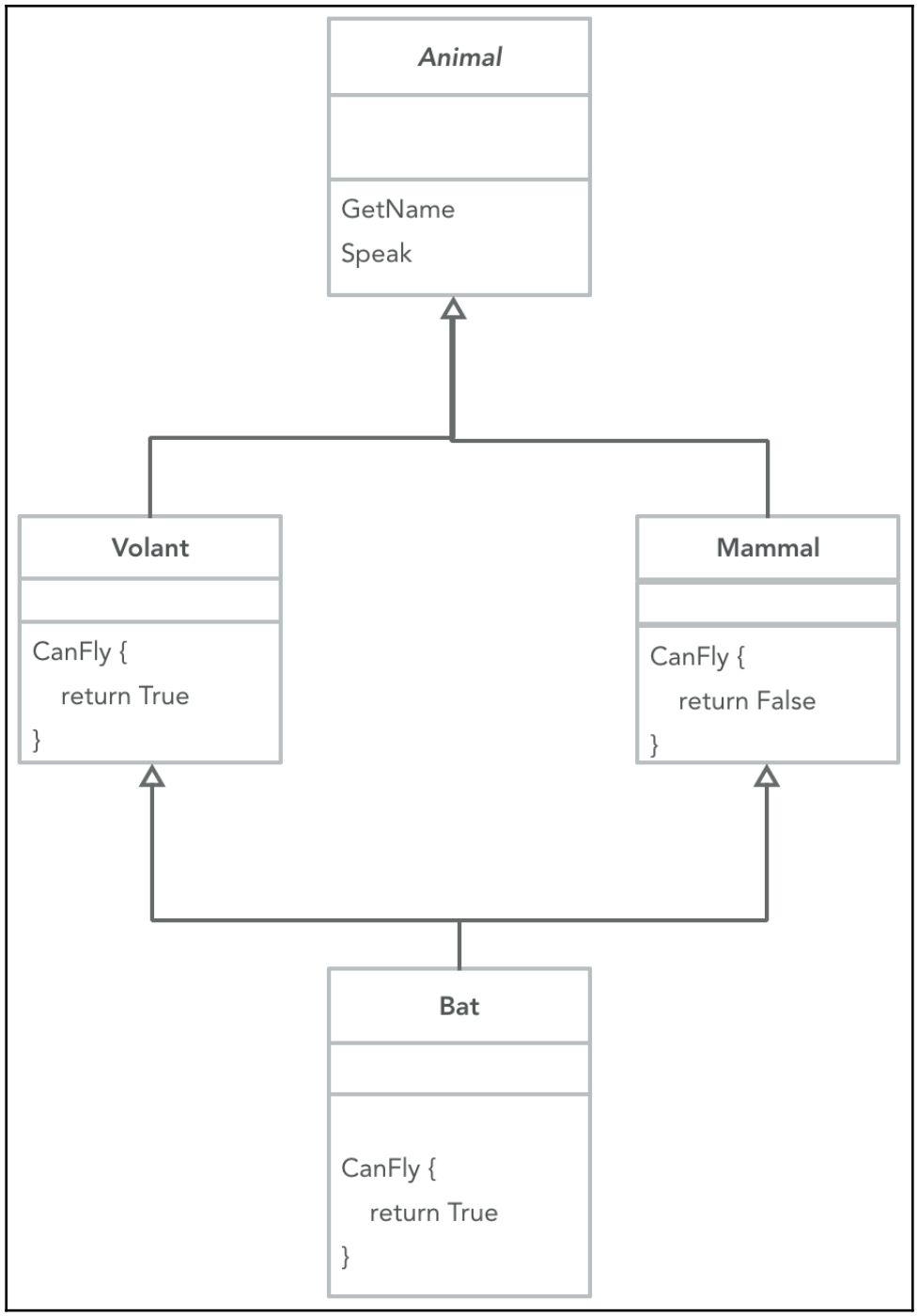



Animal
GetLocation
SetLocation
GetName
CanFly
Speak
Elephant
Weight
Color
Type
Location
Name
Speak
GetLocation
SetLocation
GetName
CanFly
ShowTusks
Lion
ManeLength
Location
Name
Speak
GetLocation
SetLocation
GetName
CanFly
Pigeon
Color
Location
Name
Speak
GetLocation
SetLocation
GetName
CanFly
Tiger
Color
Location
Name
Speak
GetLocation
SetLocation
GetName
CanFly

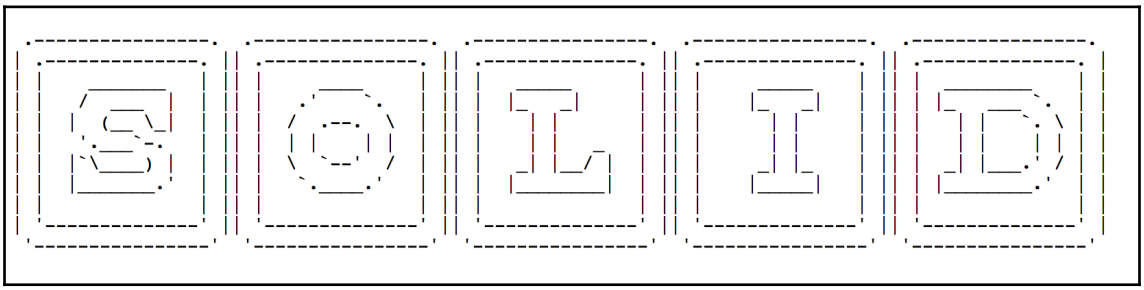

Animal
GetLocation
SetLocation
GetName
CanFly
Speak
Bird
Speak {
return "Hoot"
}
CanFly {
return True
}
Mammal
CanFly {
return False
}
Pigeon
Location
GetLocation
SetLocation
GetName
Elephant
Location
Weight
Color
Type
Speak
GetLocation
SetLocation
GetName
ShowTusks
Lion
Location
ManeLength
Speak
GetLocation
SetLocation
GetName
Tiger
Location
Speak
GetLocation
SetLocation
GetName
Bat
Location
Speak
GetLocation
SetLocation
GetName
CanFly {
return True
}

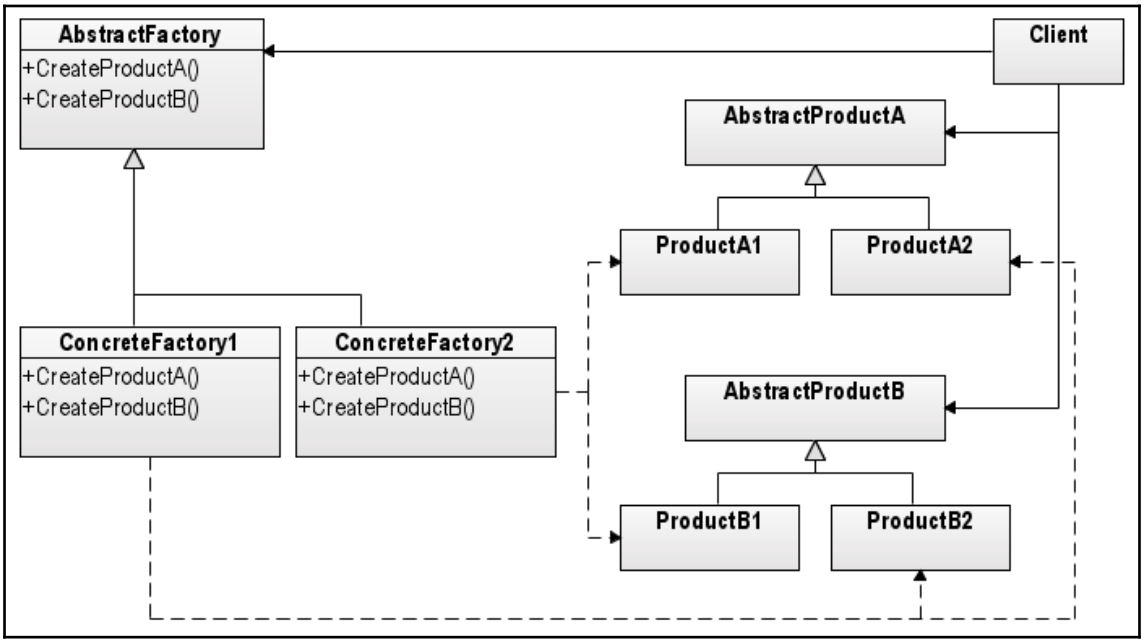

Animal
GetLocation
SetLocation
GetName
CanFly
Speak
Elephant
Weight
Color
Type
Location
Name
MammalNature ref
CanFy
Speak
GetLocation
SetLocation
GetName
ShowTusks
Lion
Location
Name
MammalNature ref
CanFy
Speak
GetLocation
SetLocation
GetName
Tiger
Location
Name
CanFy
Speak
GetLocation
SetLocation
GetName
Pigeon
Location
Name
CanFy
Speak
GetLocation
SetLocation
GetName
Location
GetLocation
SetLocation
MammalNature
CanFly {
return False
}
Location
GetLocation
SetLocation
BirdNature
Speak {
return “Hoot”
}
CanFly {
return True
}

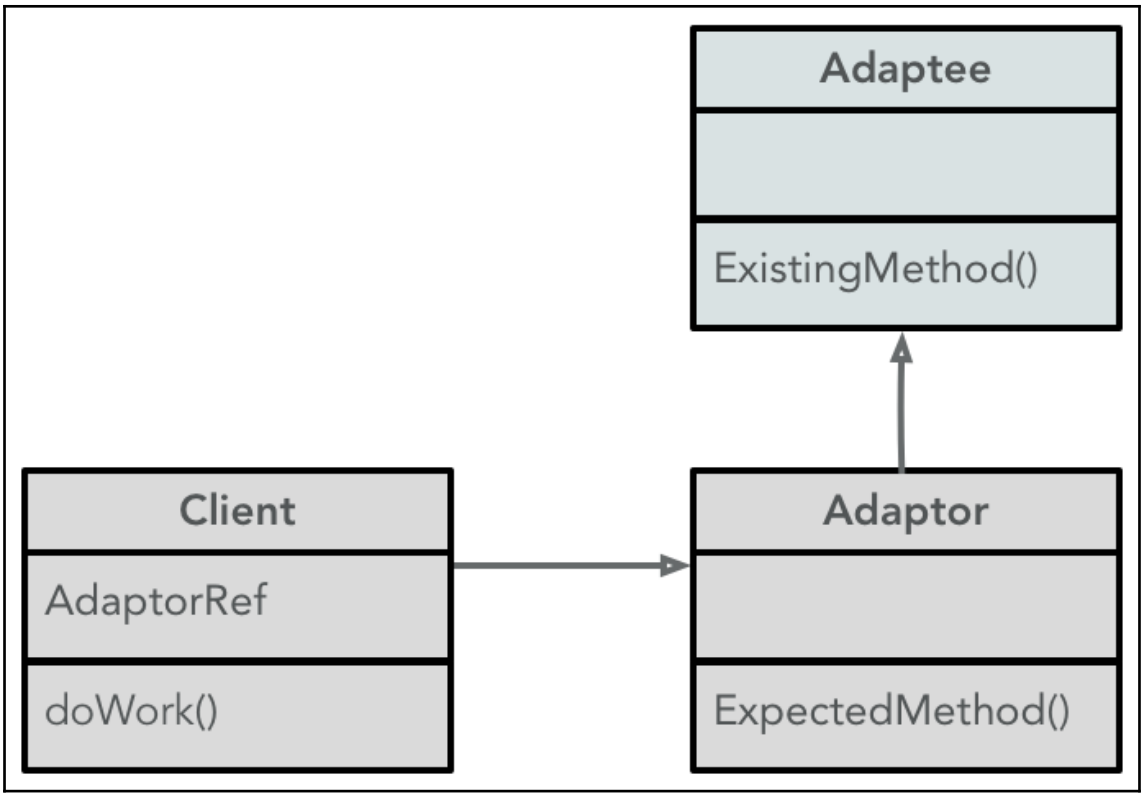

iTable(Animal, Lion)
type:
type(Lion)
fun[4]
fun[3]
fun[2]
fun[1]
fun[0]
method table
aAnimal
tab
data
&Lion{
"Leo",
10,
LatLong{10.40, 11.5},
}

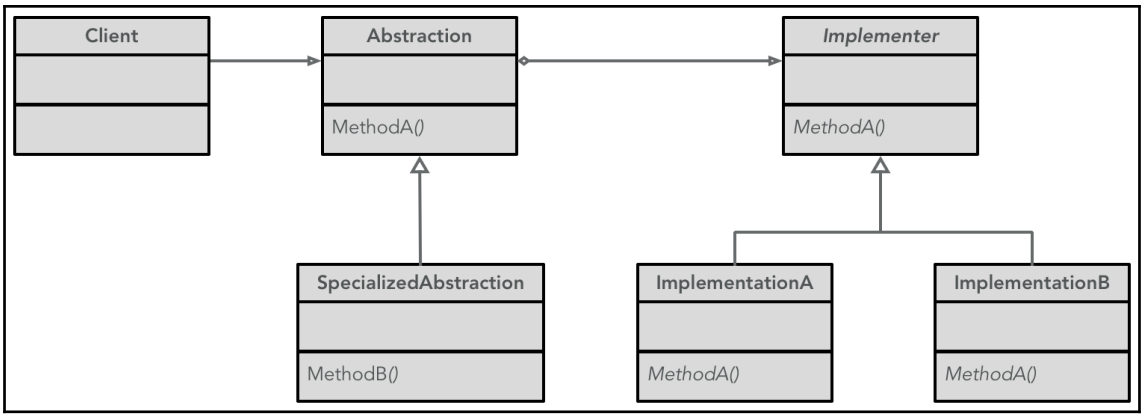

Animal
GetName
Speak
Volant
CanFly {
return True
}
Mammal
CanFly {
return False
}
Bat
CanFly {
return True
}

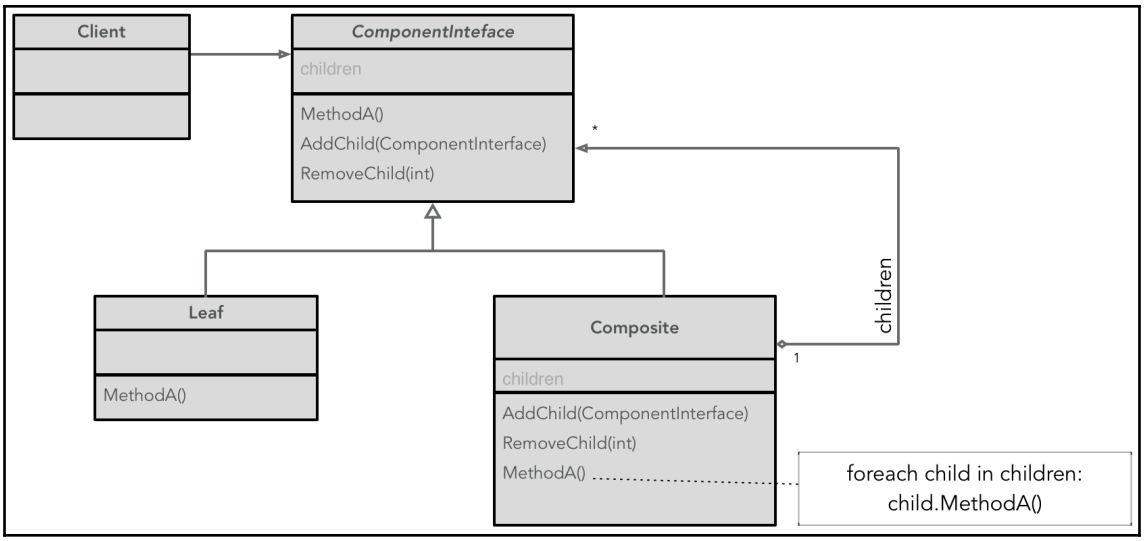
Chapter 3: Design Patterns

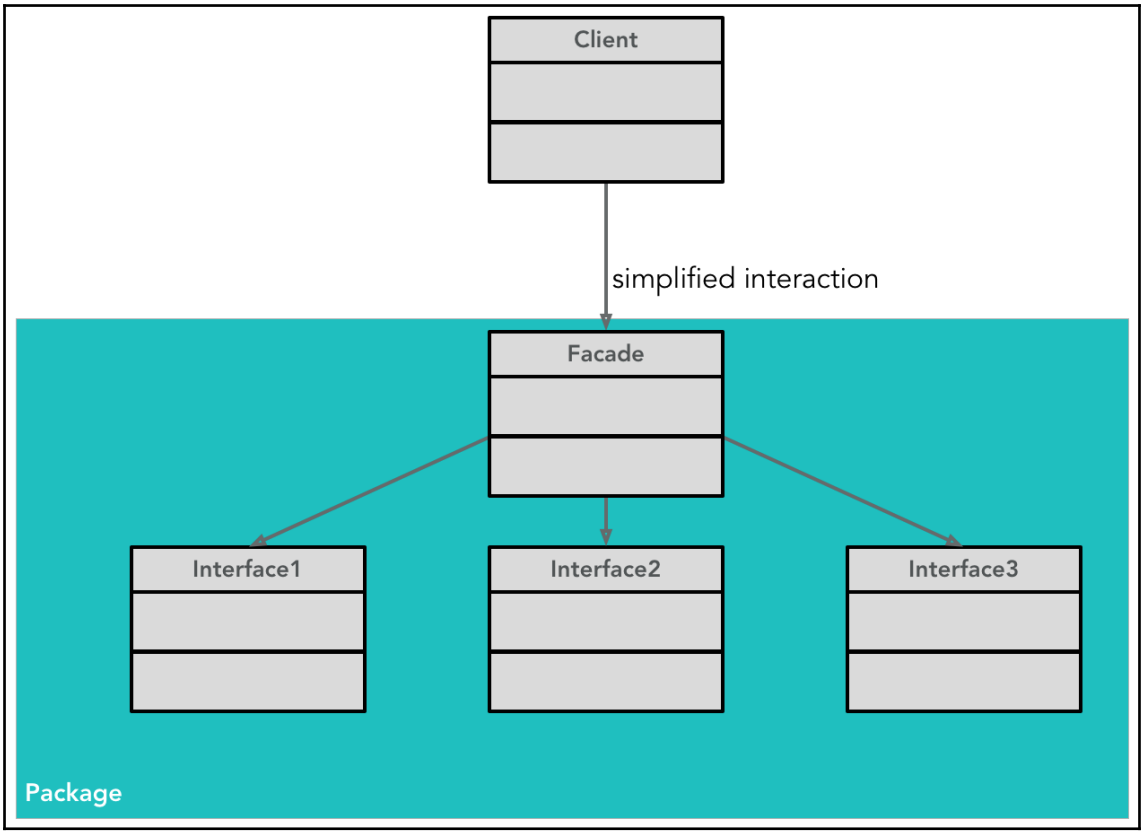


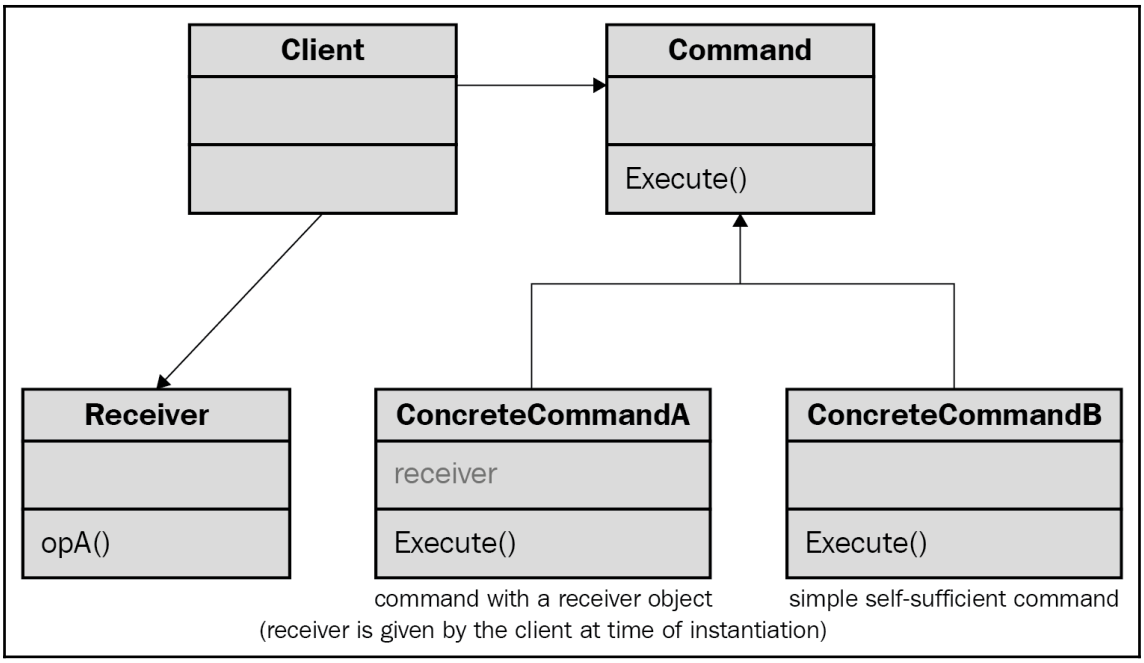


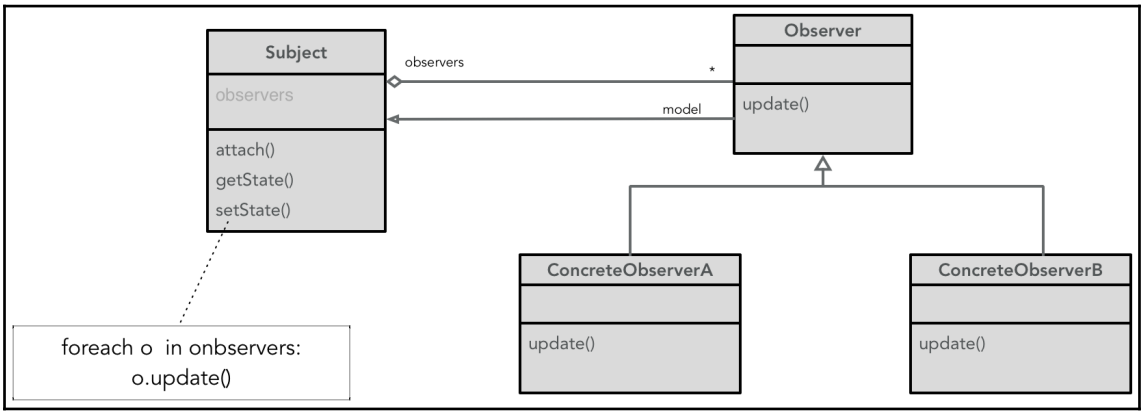

Adaptee
ExistingMethod()
Client
AdaptorRef
doWork()
Adaptor
ExpectedMethod()

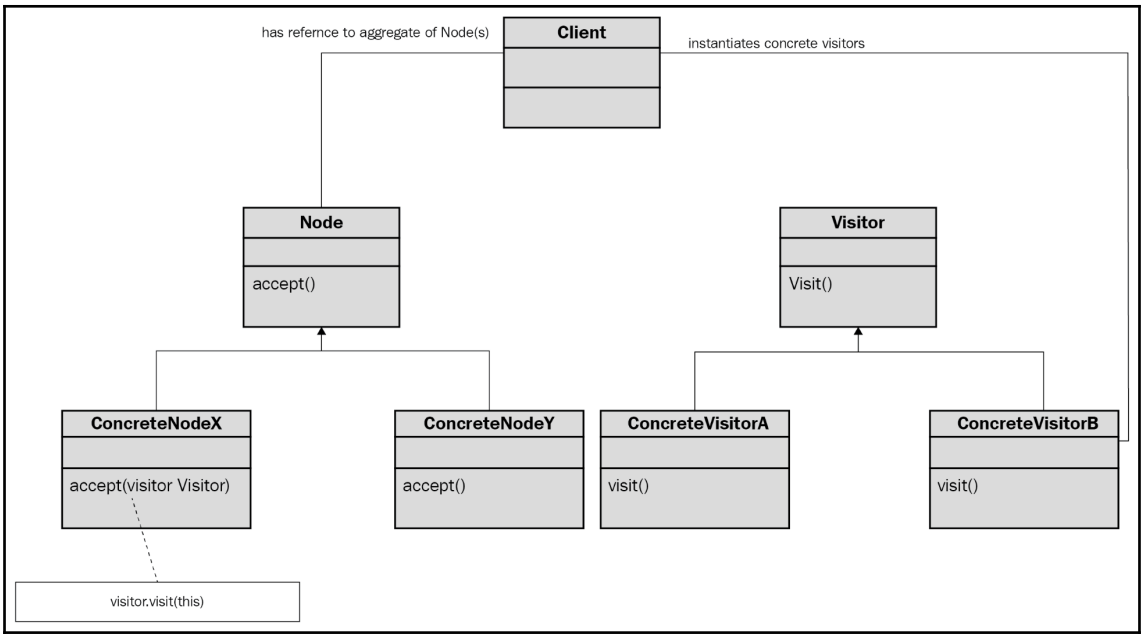

Client
Abstraction
MethodA()
Implementer
MethodA()
SpecializedAbstraction
MethodB()
ImplementationA
MethodA()
ImplementationB
MethodA()

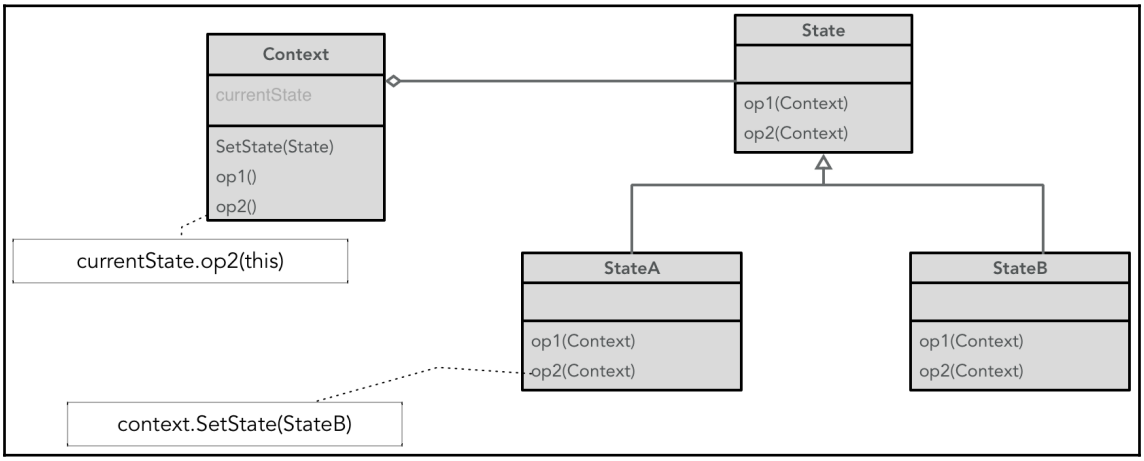

Client
ComponentInteface
children
MethodA()
AddChild(ComponentInterface)
RemoveChild(int)
*
children
1
Leaf
MethodA()
Composite
children
AddChild(ComponentInterface)
RemoveChild(int)
MethodA()
foreach child in children:
child.MethodA()


Client
simplified interaction
Facade
Interface1
Interface2
Interface3
Package

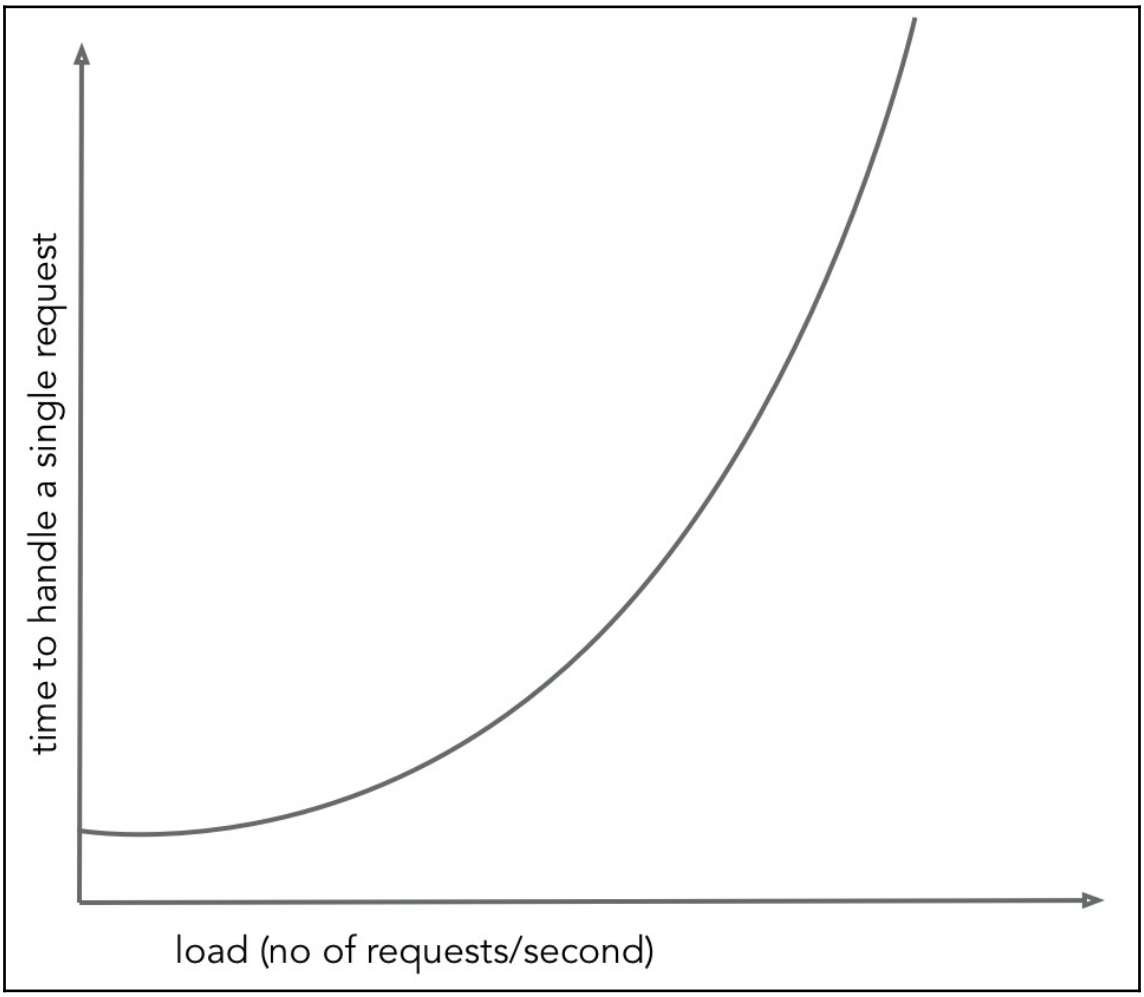

Client
Command
Execute()
Receiver
opA()
ConcreteCommandA
receiver
Execute()
ConcreteCommandB
Execute()
command with a receiver object
(receiver is given by the client at time of instantiation)
simple self-sufficient command

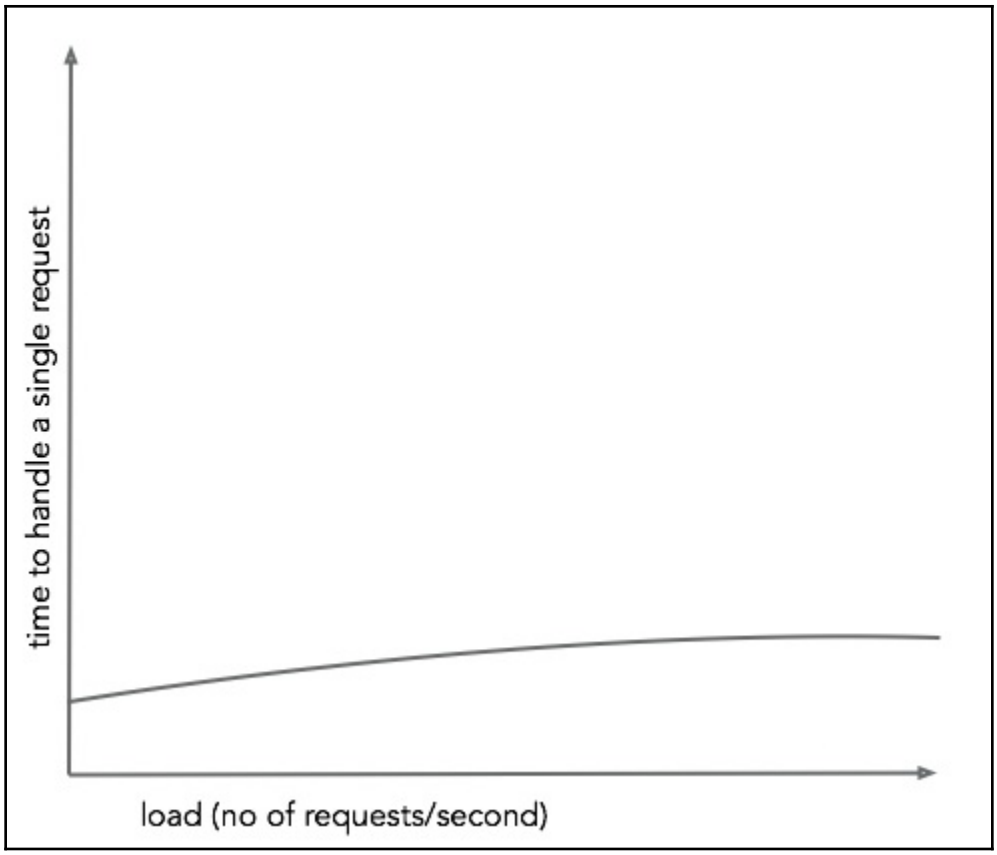

Subject
observers
attach()
getState()
setState()
observers
*
model
Observer
update()
foreach o in onbservers:
o.update()
ConcreteObserverA
update()
ConcreteObserverB
update()

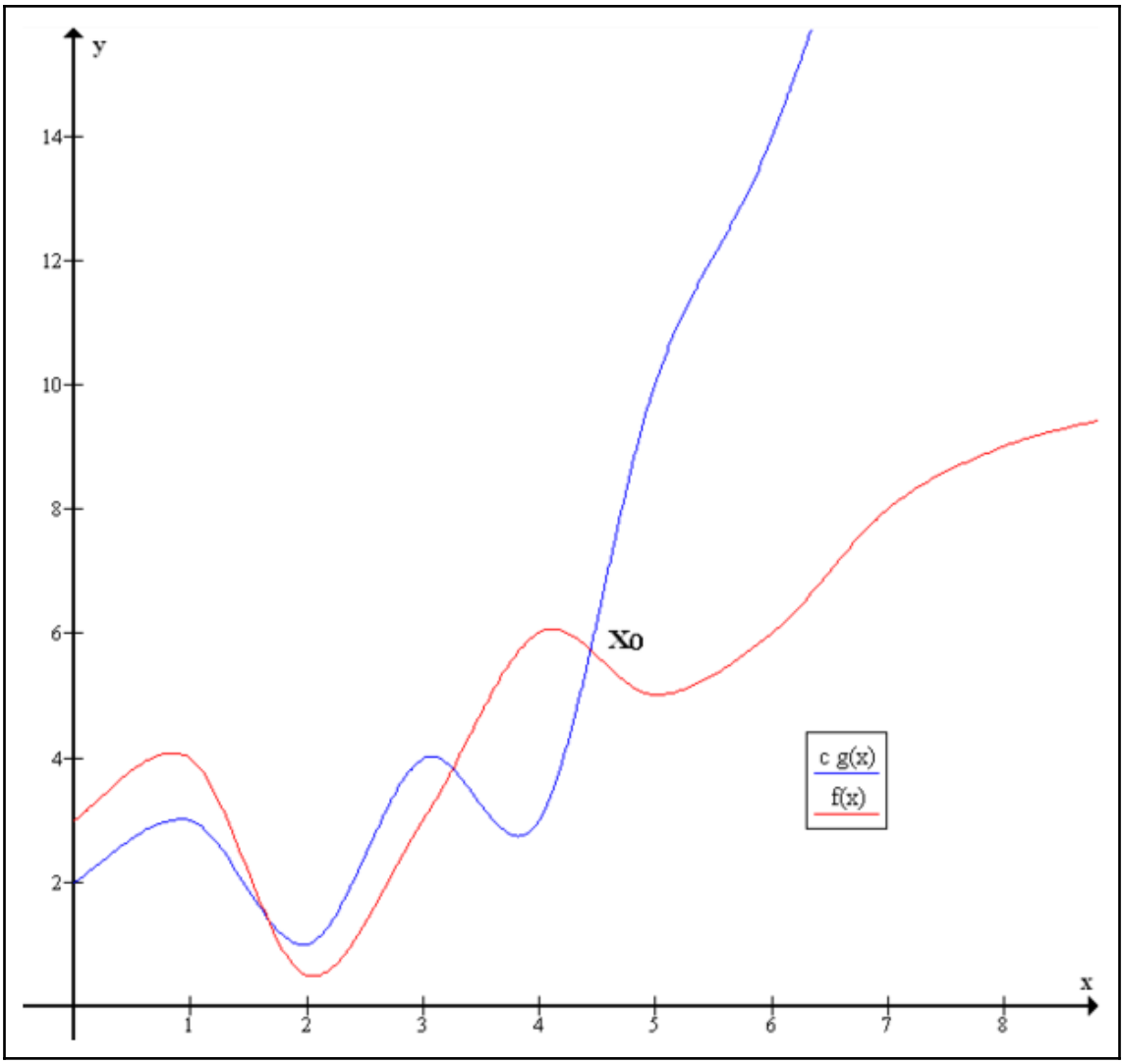

has refernce to aggregate of Node(s)
Client
instantiates concrete visitors
Node
accept()
Visitor
Visit()
ConcreteNodeX
accept(visitor Visitor)
ConcreteNodeY
accept()
ConcreteVisitorA
visit()
ConcreteVisitorB
visit()
visitor.visit(this)

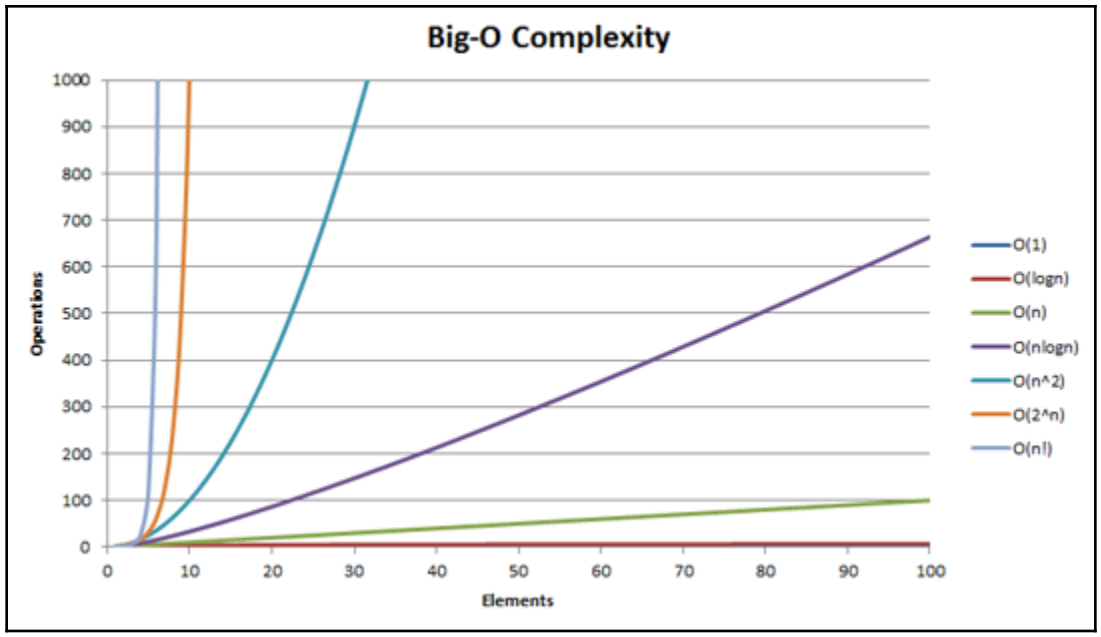

Context
currentState
SetState(State)
op1()
op2()
State
op1(Context)
op2(Context)
currentState.op2(this)
StateA
op1(Context)
op2(Context)
StateB
op1(Context)
op2(Context)
context.SetState(StateB)

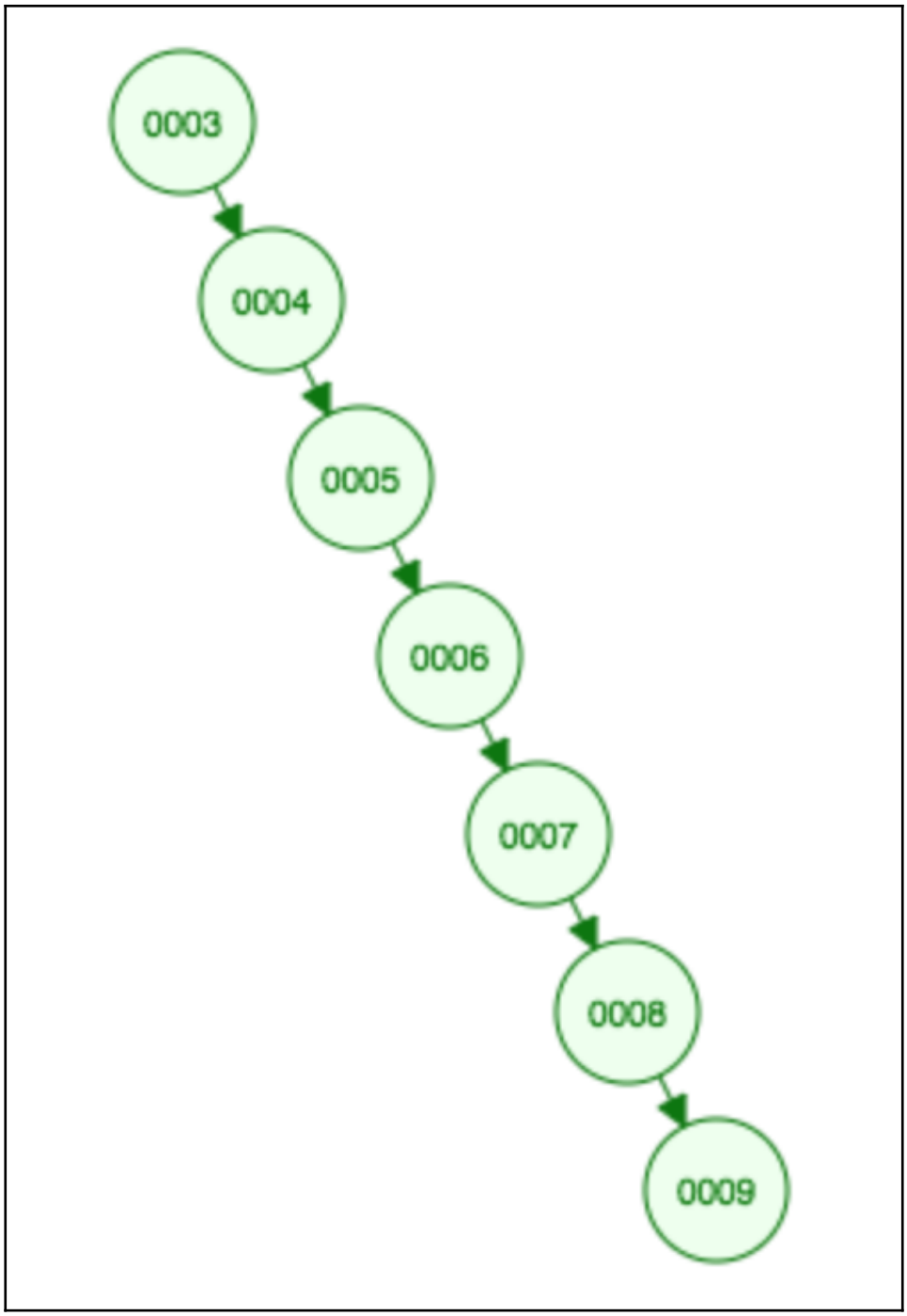
Chapter 4: Scaling Applications

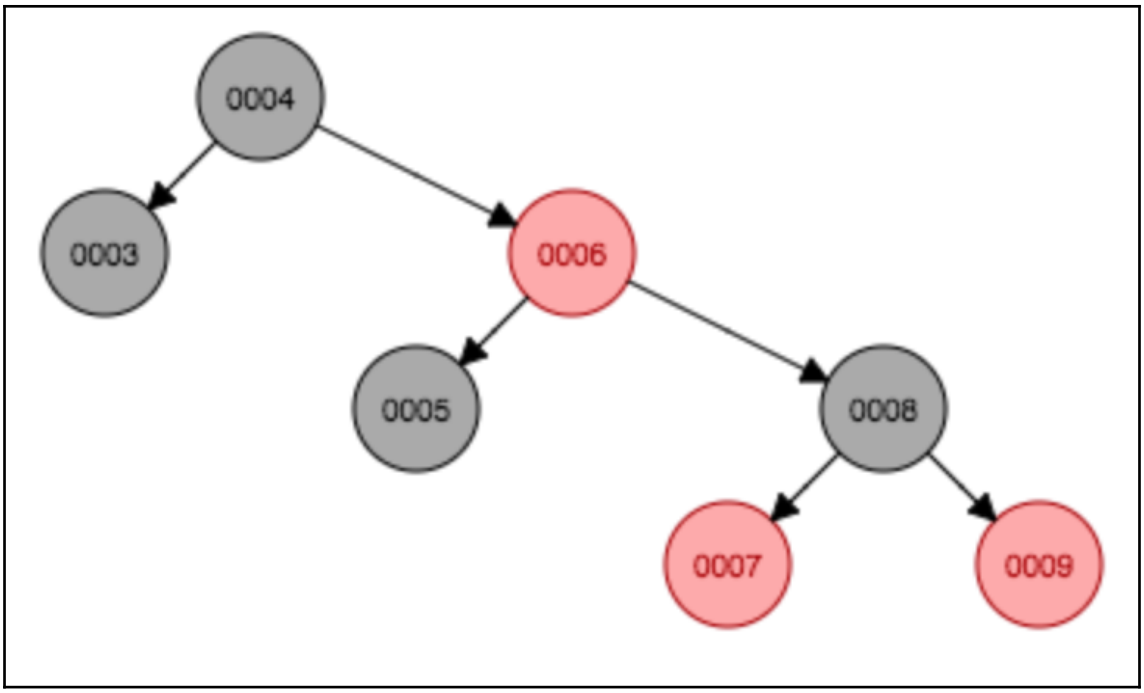


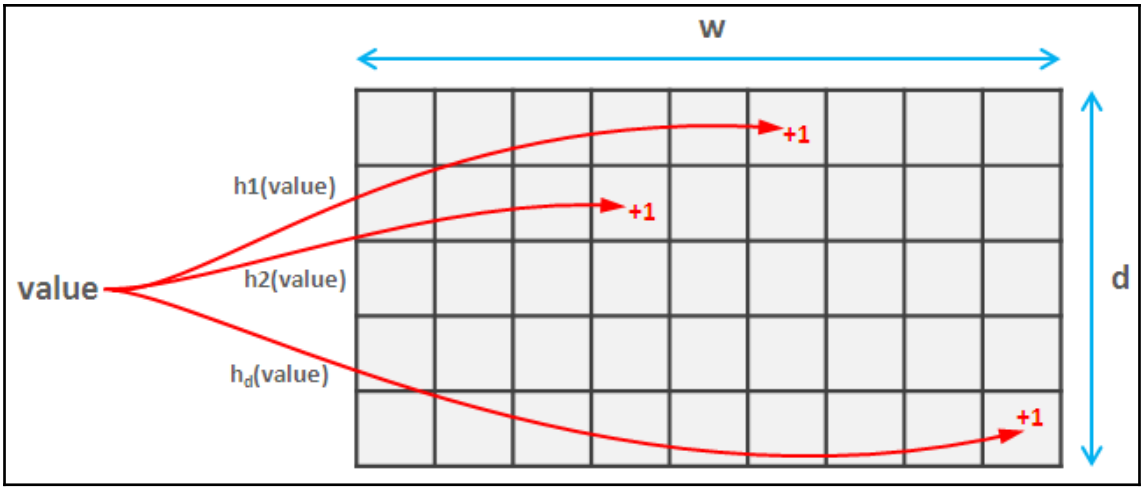

time to handle a single request
load (no of requests/second)

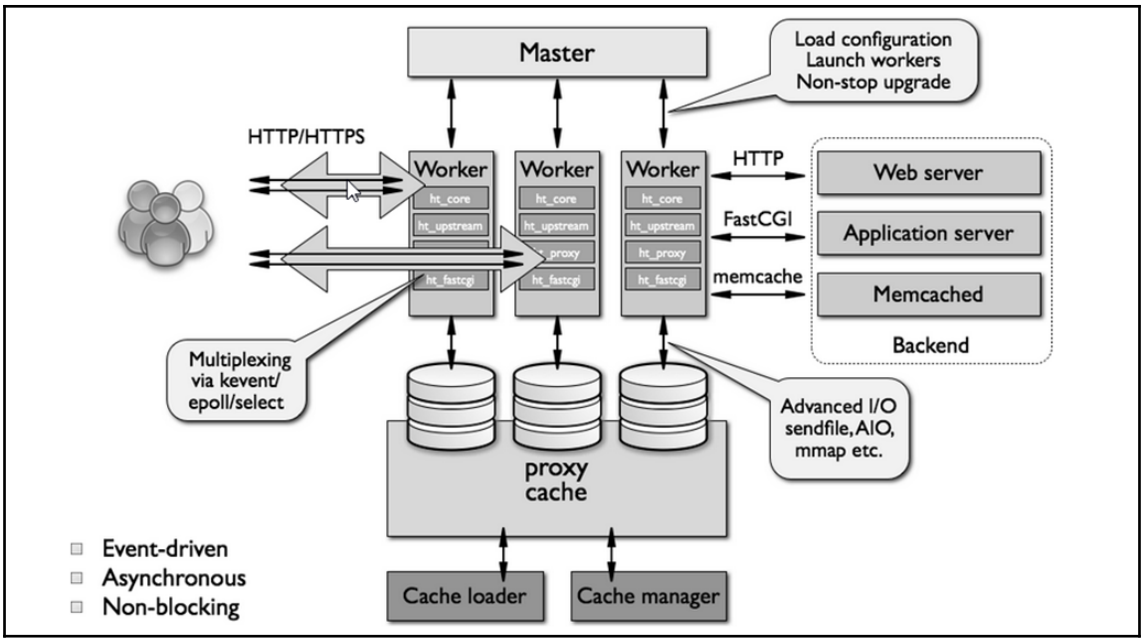

y
14
12
10
8
6
4
2
Xo
c g(x)
f(x)
x
1
2
3
4
5
6
7
8

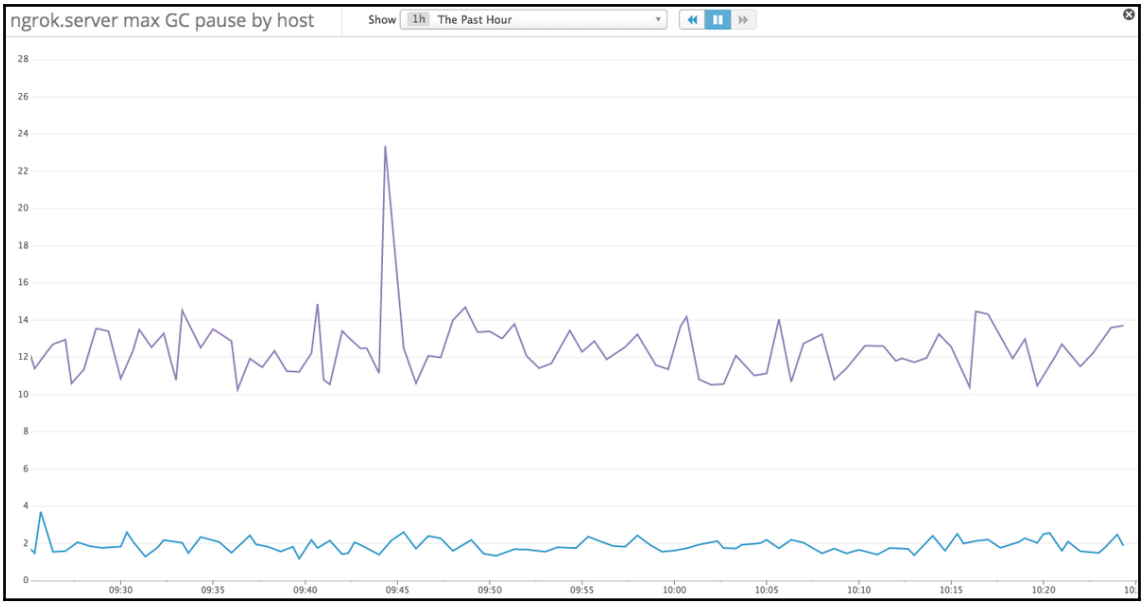

Big-O Complexity
Operations
Elements
1000
900
800
700
600
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
90
100
O(1)
O(logn)
O(n)
O(nlogn)
O(n^2)
O(2^n)
O(n!)

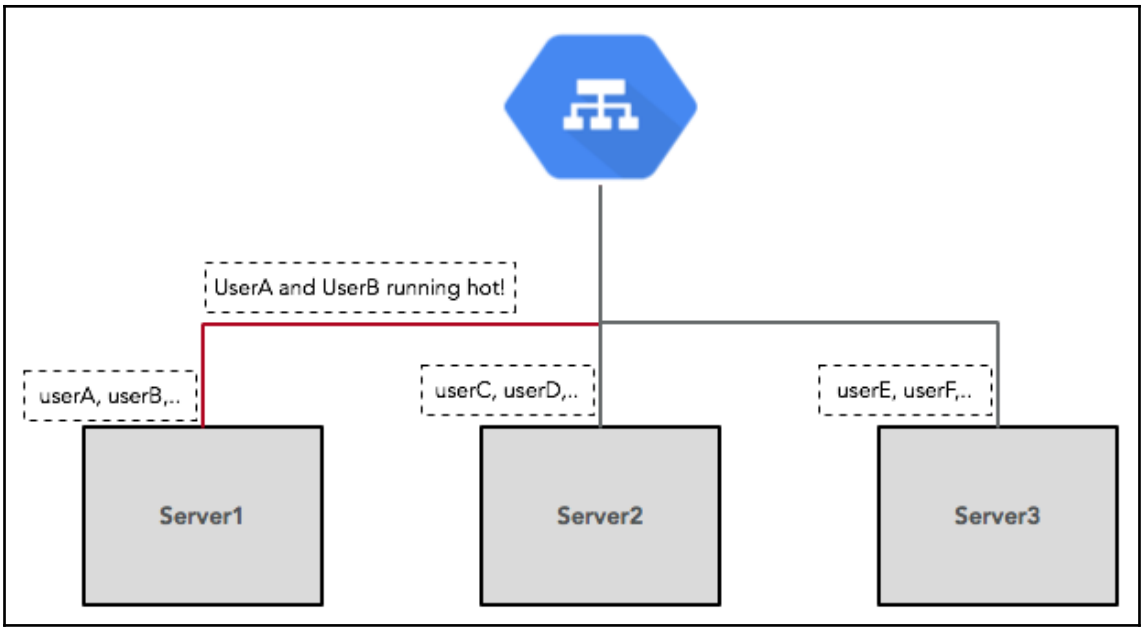

0003
0004
0005
0006
0007
0008
0009

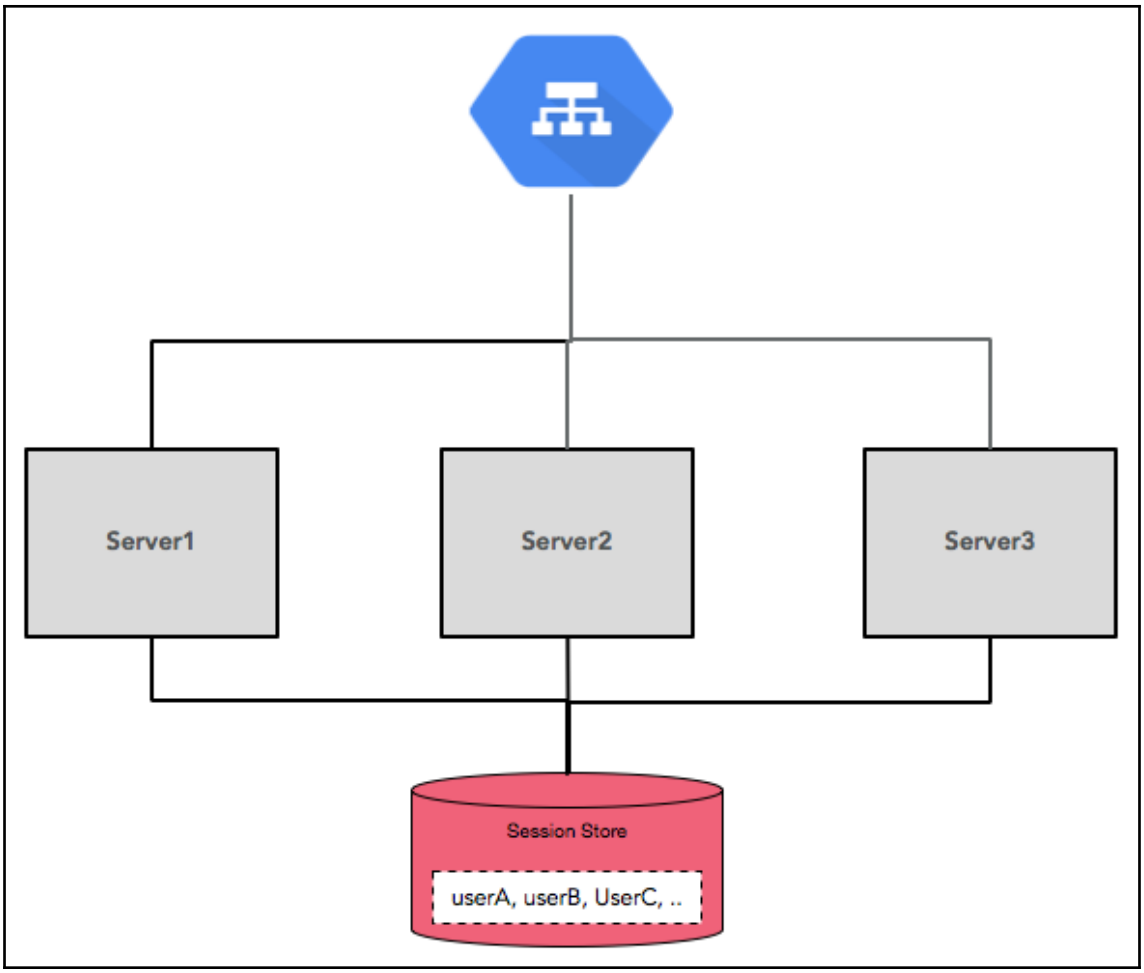

0004
0003
0006
0005
0008
0007
0009

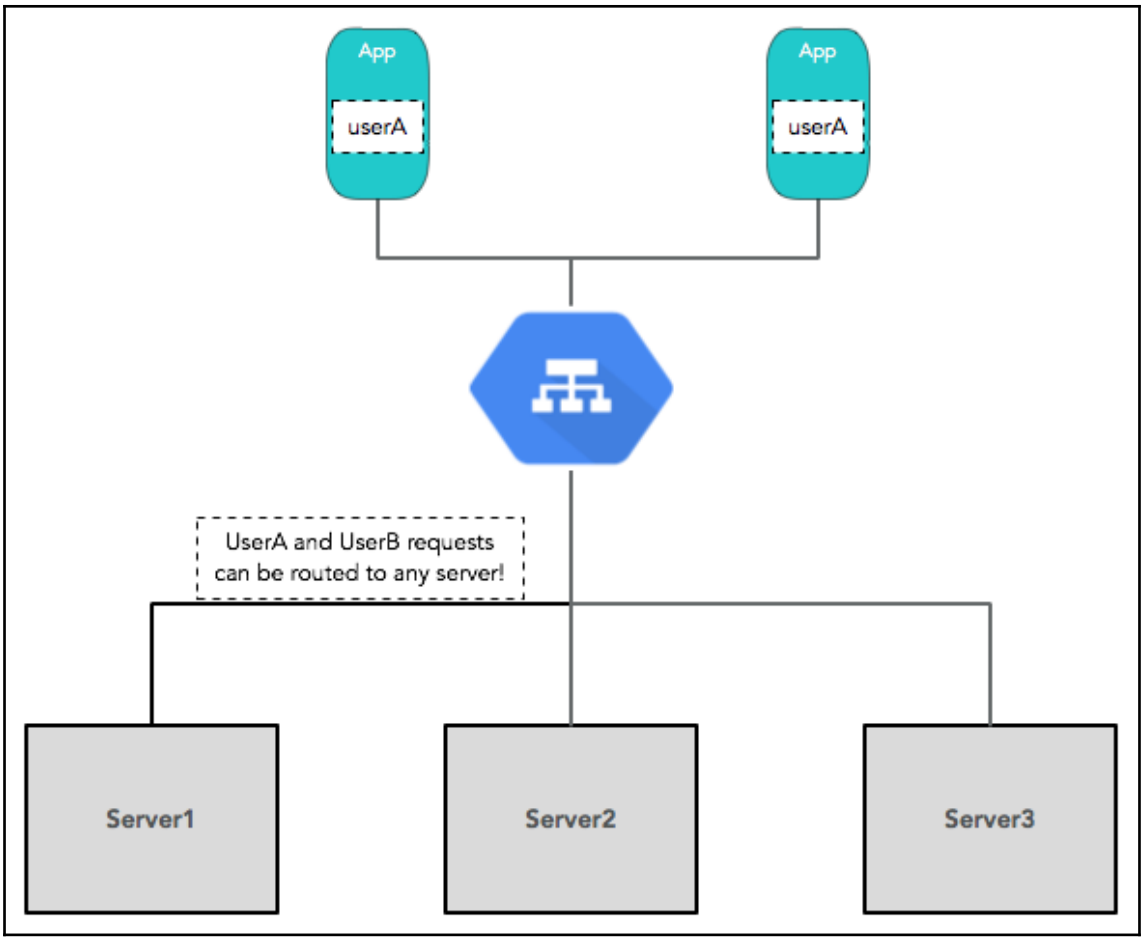

w
d
value
h1(value)
h2(value)
hd(value)
+1
+1
+1

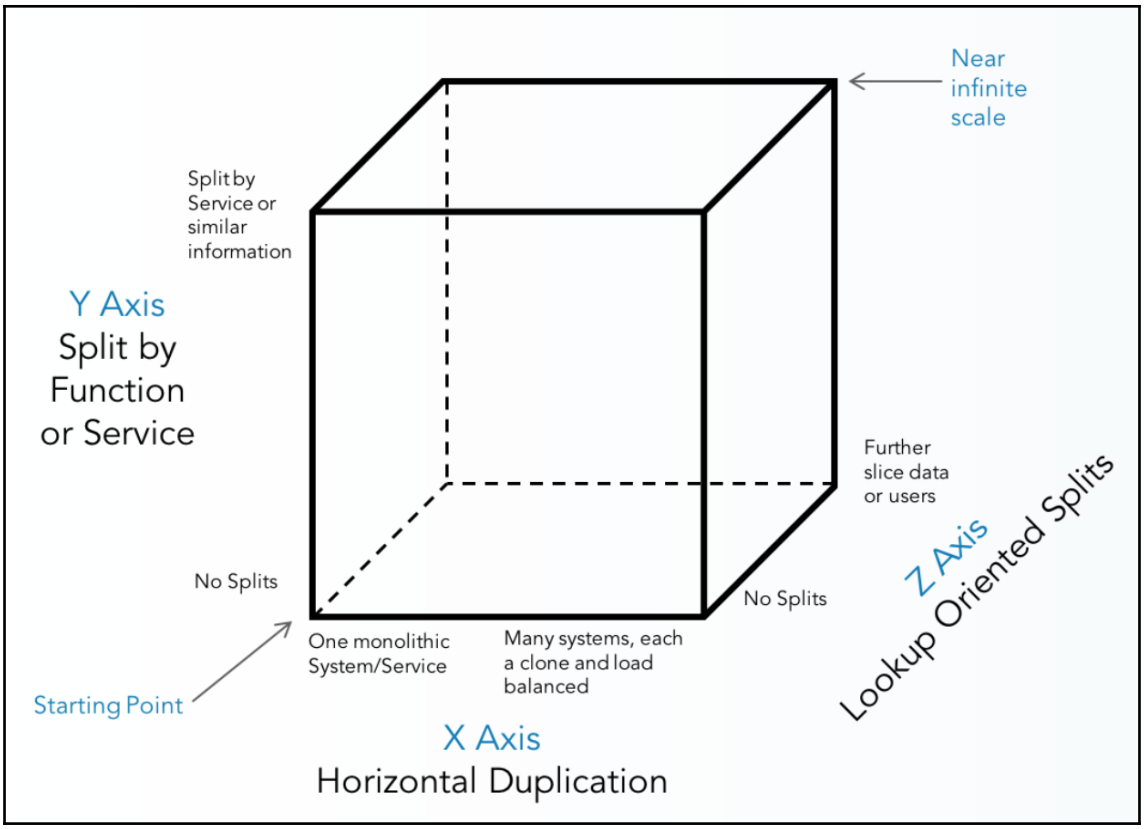

Master
Load configuration
Launch workers
Non-stop upgrade
HTTP/HTTPS
Worker
Worker
Worker
ht_core
ht_upstream
ht_proxy
ht_fastcgi
HTTP
FastCGI
memcache
Web server
Application server
Memcached
Backend
Multiplexing via kevent/ epoll/select
Advanced I/O sendfile, AIO, mmap etc.
proxy cache
Cache loader
Cache manager
Event-driven
Asynchronous
Non-blocking

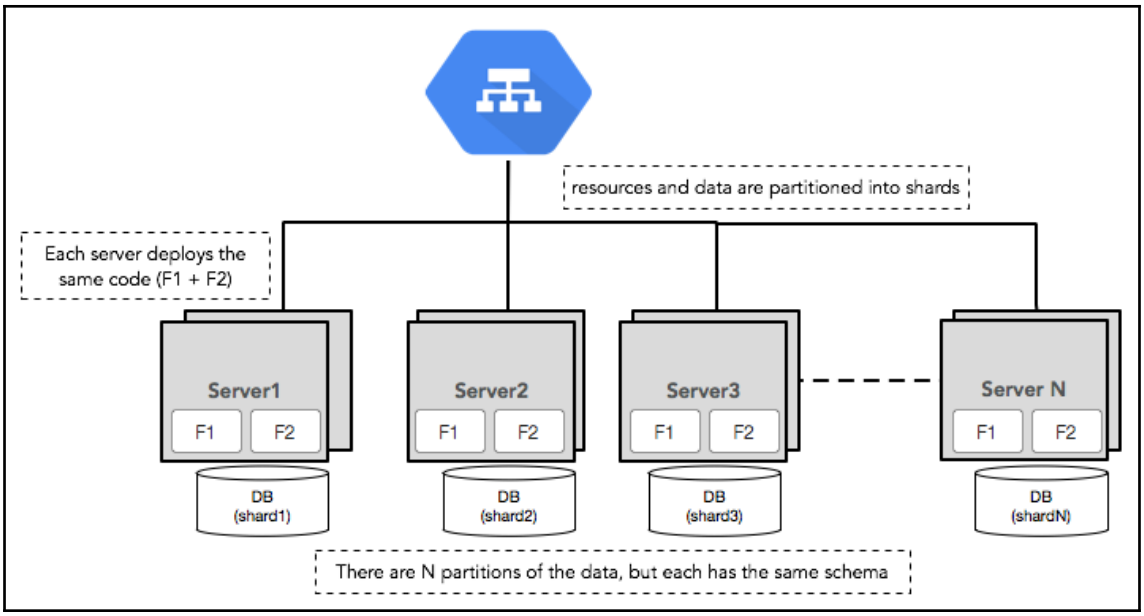

ngrok.server max GC pause by host
Show
1h
The Past Hour
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
09:30
09:35
09:40
09:45
09:50
09:55
10:00
10:05
10:10
10:15
10:20

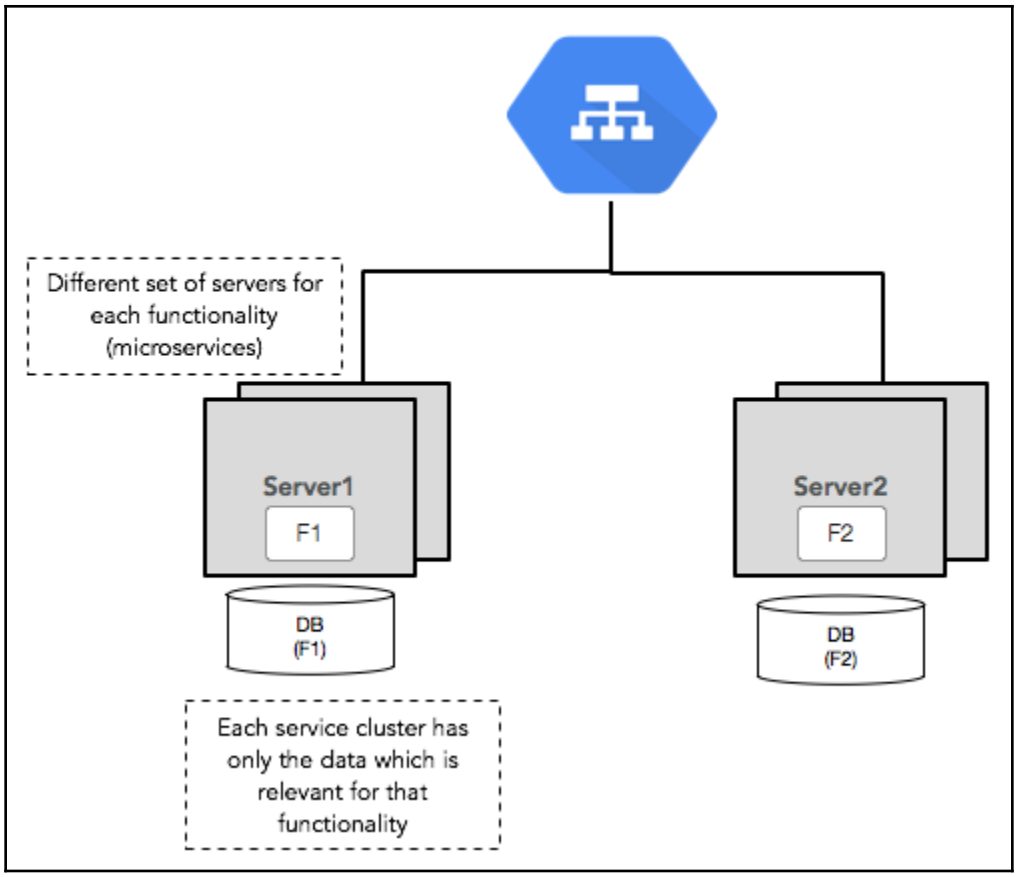

UserA and UserB running hot!
userA, userB,..
userC, userD,..
userE, userF,..
Server1
Server2
Server3

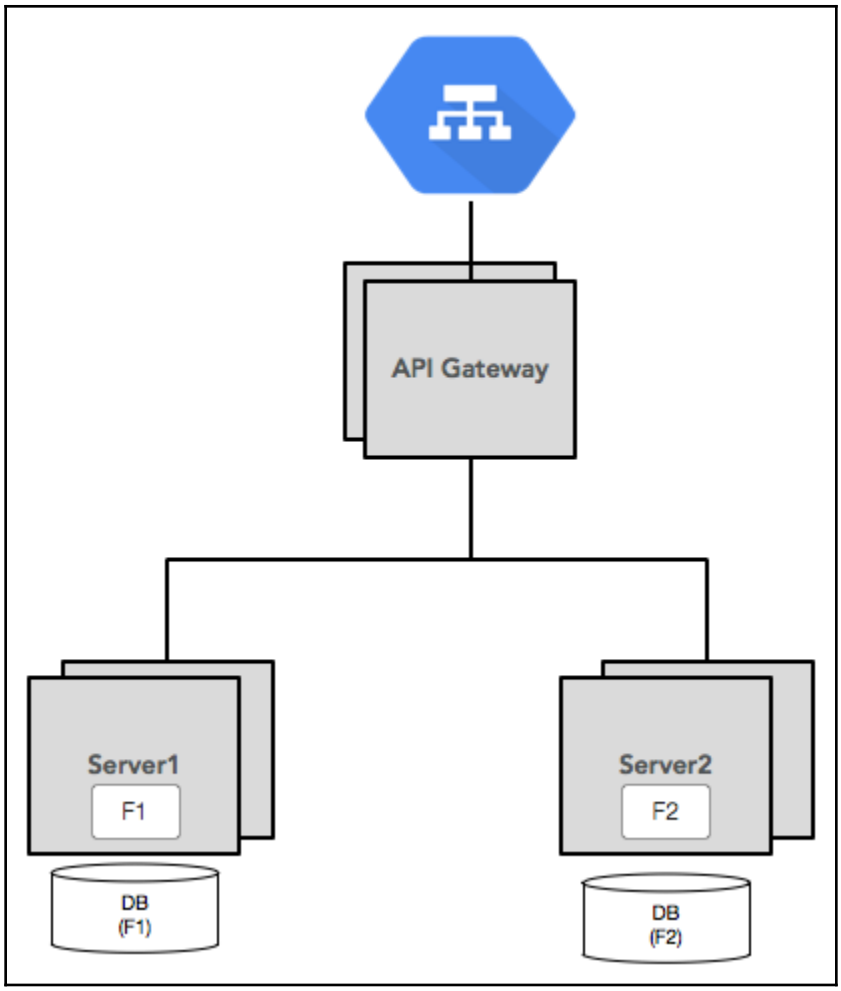

Server1
Server2
Server3
Session Store
userA, userB, UserC, ..

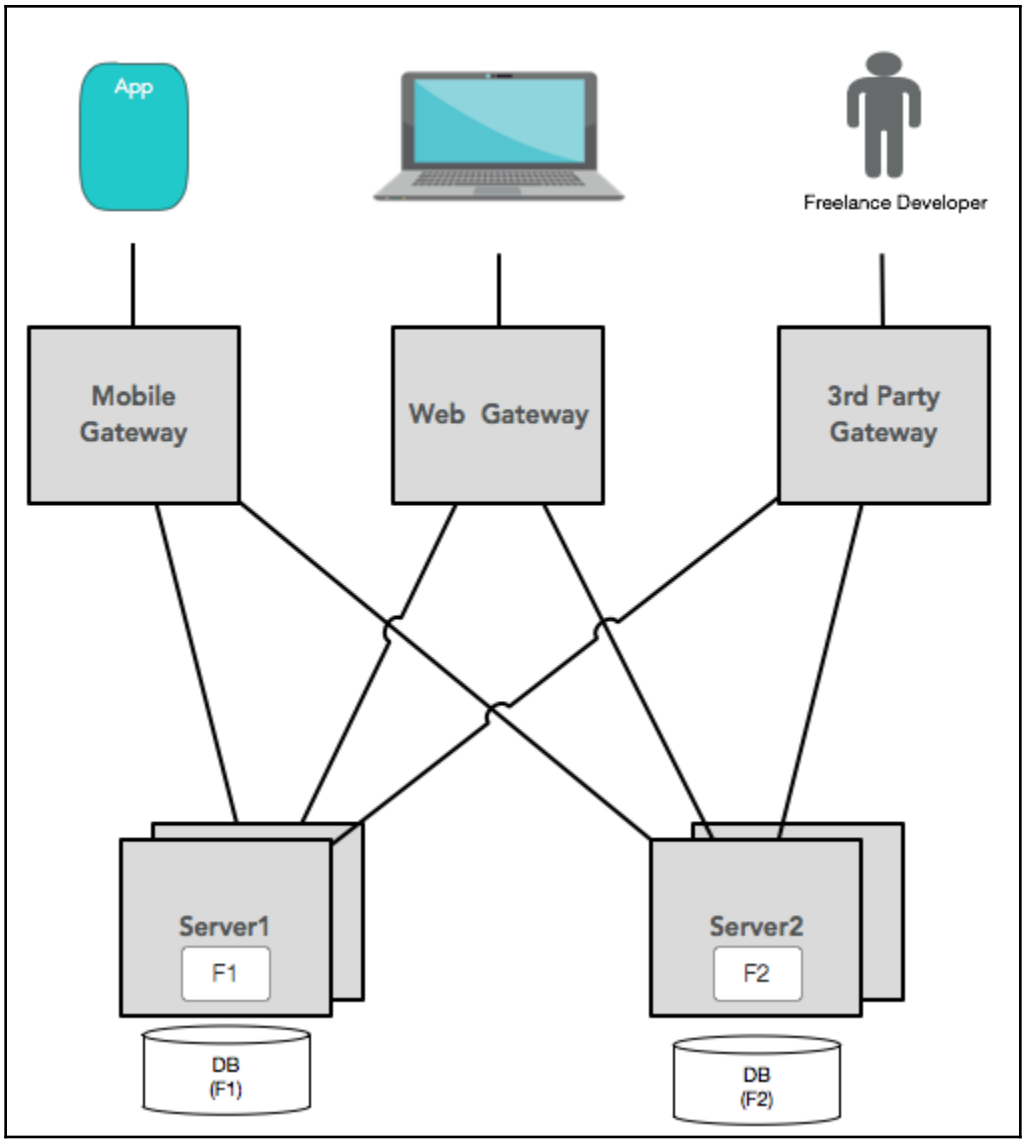

App
userA
App
userA
UserA and UserB requests
can be routed to any server!
Server1
Server2
Server3

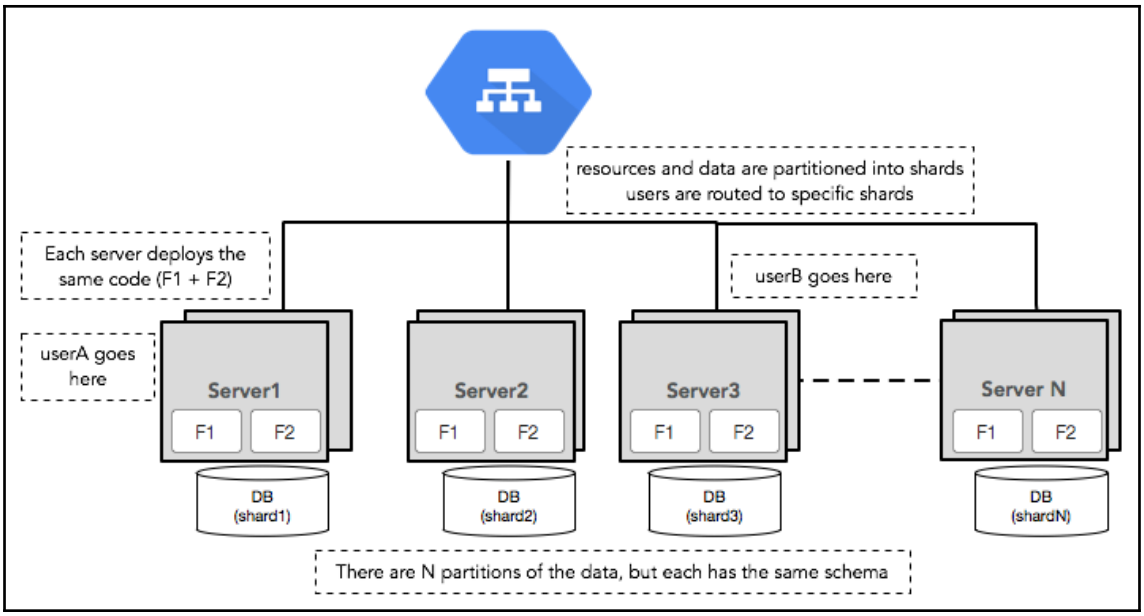

Near infinite scale
Split by Service or similar information
Y Axis
Split by Function or Service
Further slice data or users
Z Axis
Lookup Oriented Splits
No Splits
No Splits
One monolithic System/Service
Many systems, each a clone and load balanced
Starting Point
X Axis
Horizontal Duplication


resources and data are partitioned into shards
Each server deploys the same code (F1 + F2)
Server1
F1
F2
DB (shard1)
Server2
F1
F2
DB (shard2)
Server3
F1
F2
DB (shard3)
Server N
F1
F2
DB (shardN)
There are N partitions of the data, but each has the same schema

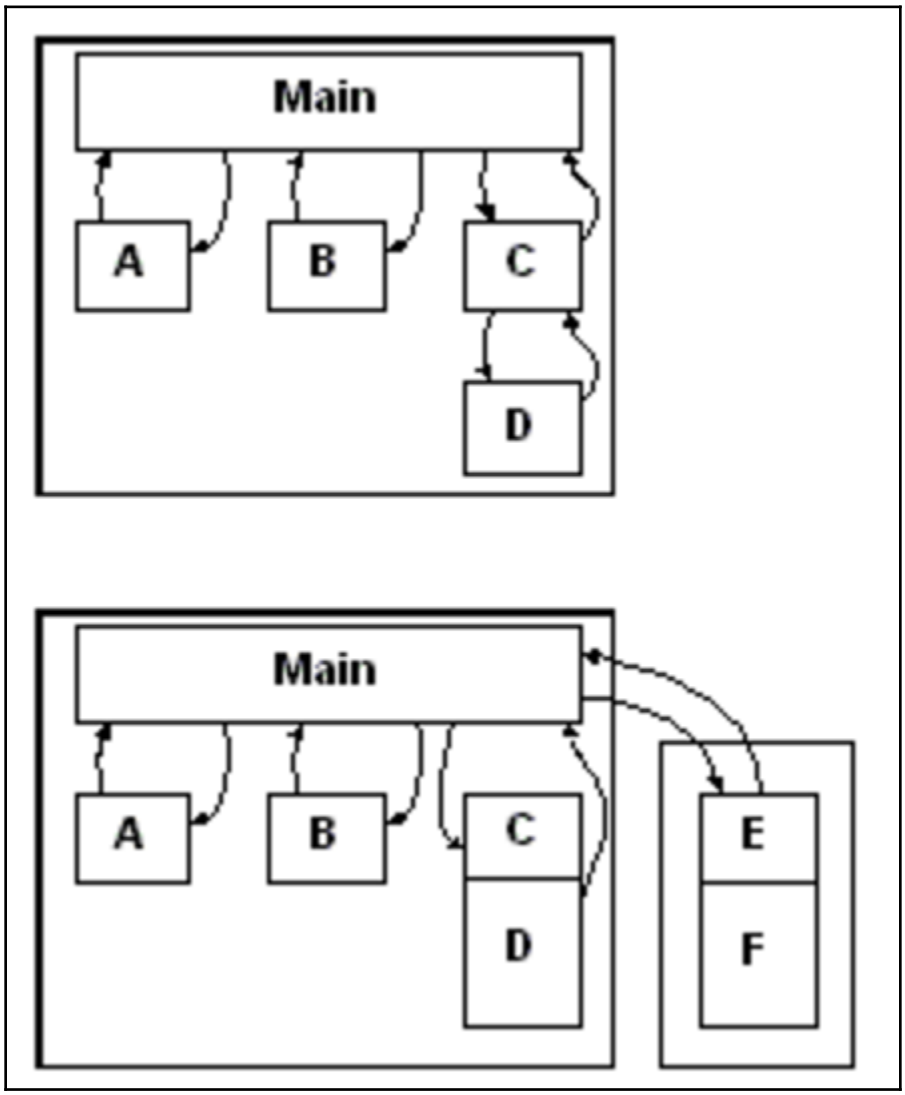

Different set of servers for each functionality (microservices)
Server1
F1
DB (F1)
Server2
F2
DB (F2)
Each service cluster has only the data which is relevant for that functionality

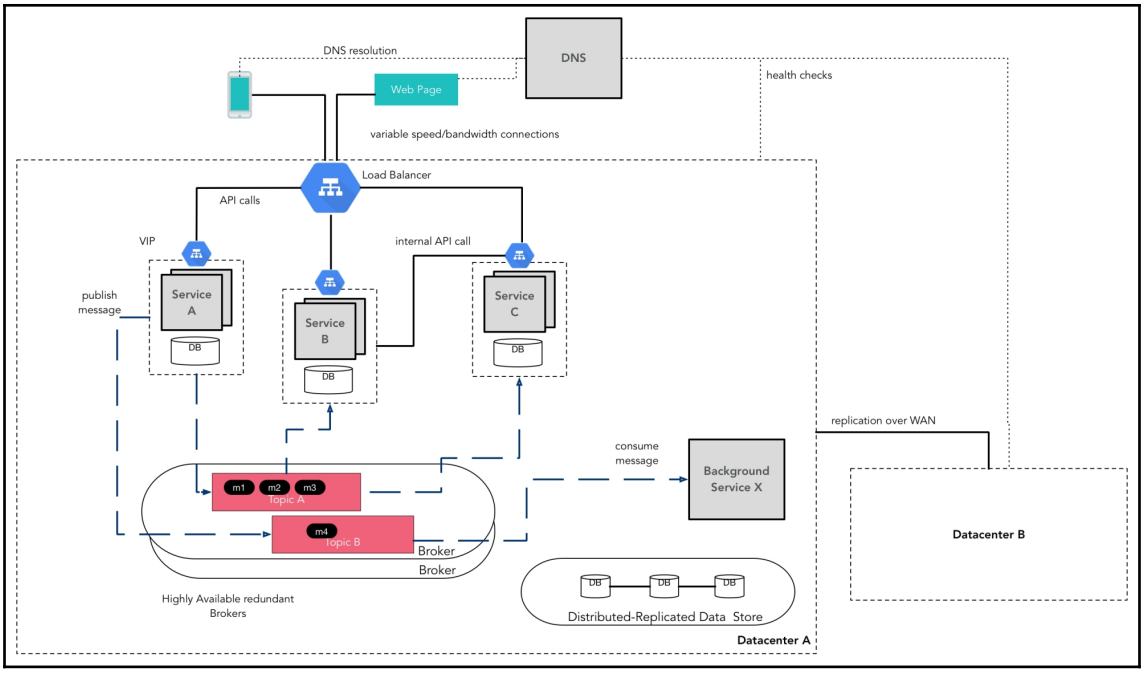

API Gateway
Server1
F1
DB
(F1)
Server2
F2
DB
(F2)

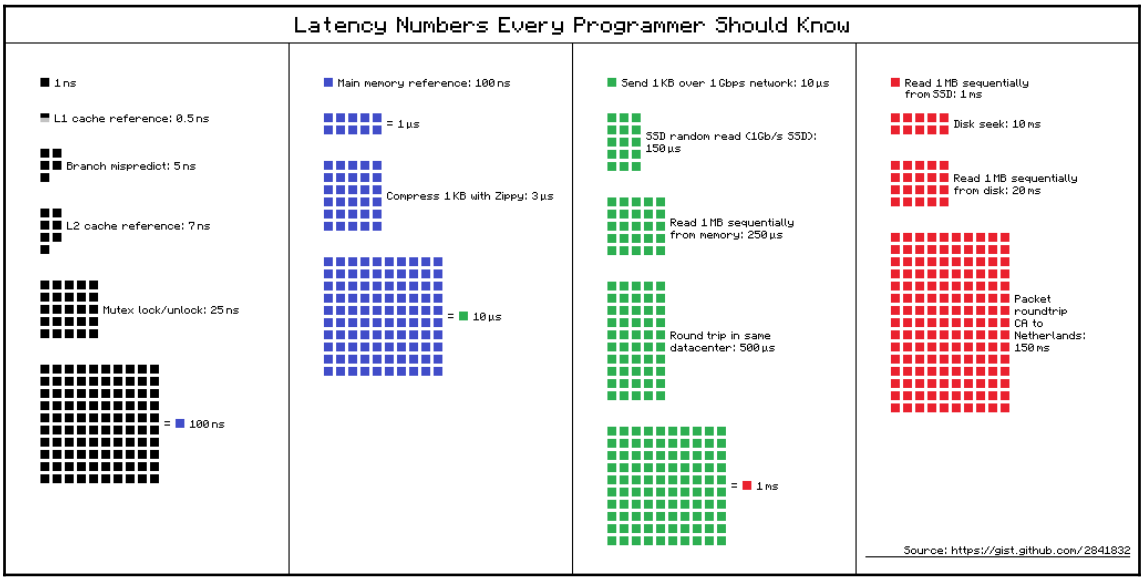

App
Freelance Developer
Mobile Gateway
Web Gateway
3rd Party Gateway
Server1
F1
Server2
F2
DB (F1)
DB (F2)

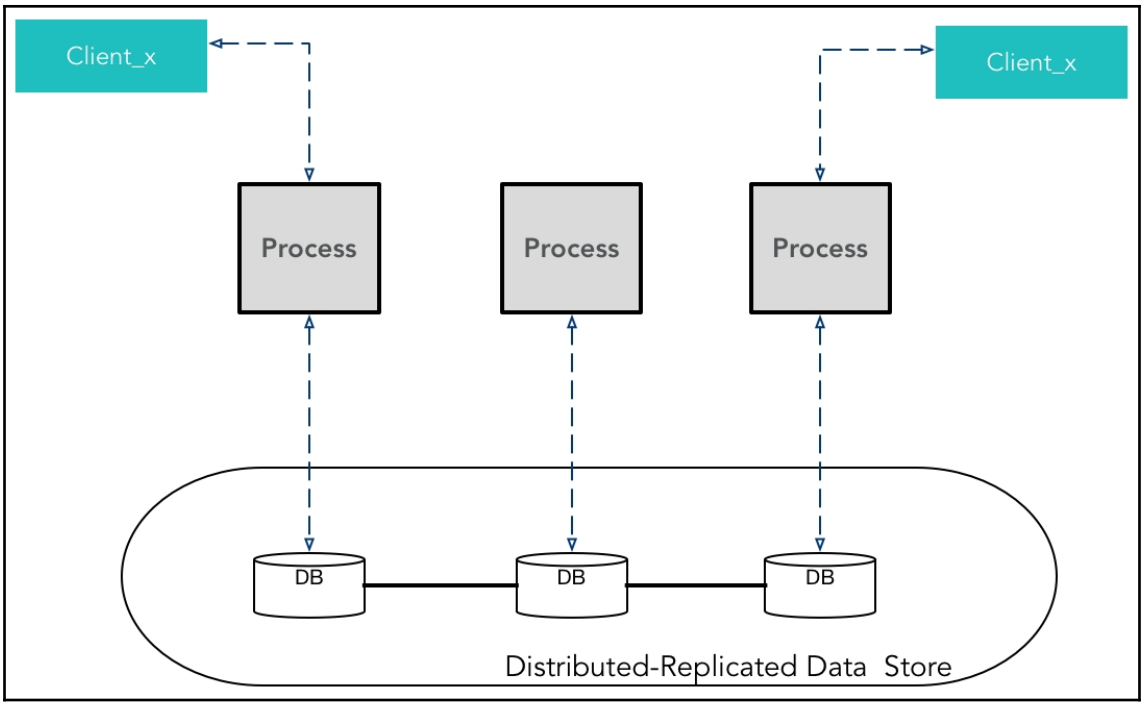

resources and data are partitioned into shards
users are routed to specific shards
Each server deploys the same code (F1 + F2)
userB goes here
userA goes here
Server1
F1
F2
DB (shard1)
Server2
F1
F2
DB (shard2)
Server3
F1
F2
DB (shard3)
Server N
F1
F2
DB (shardN)
There are N partitions of the data, but each has the same schema

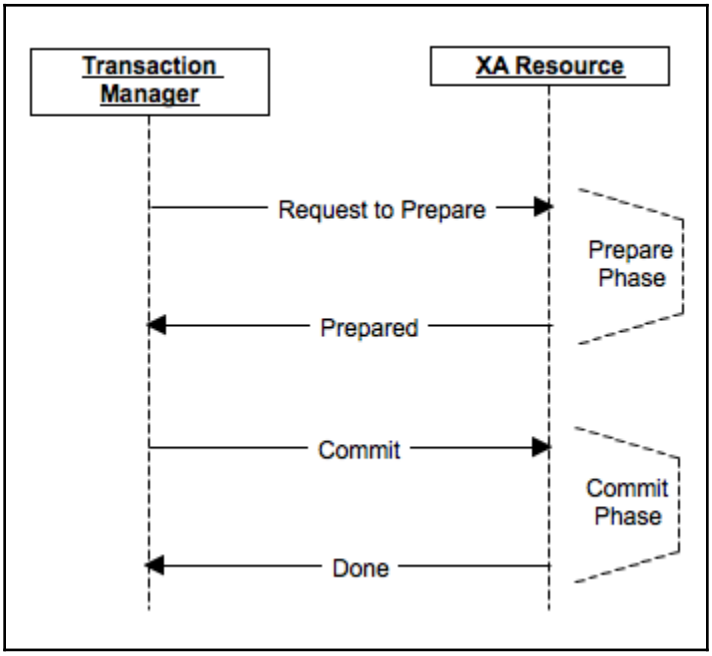
Chapter 5: Going Distributed

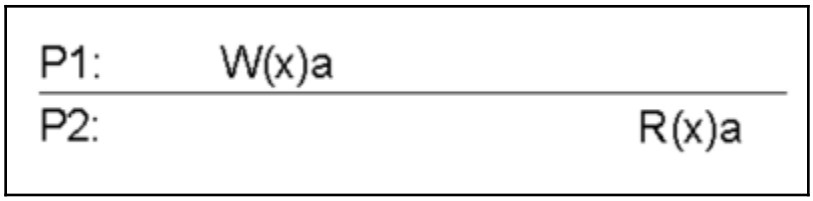


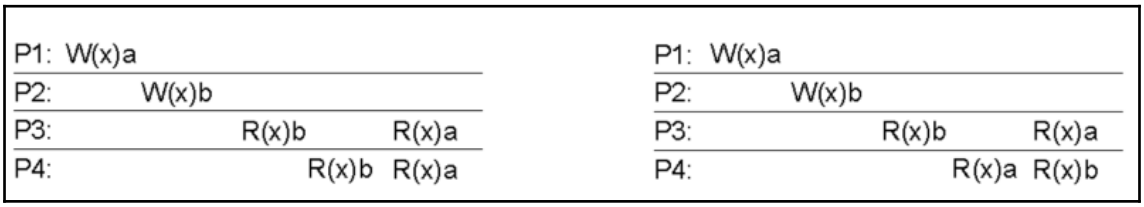

DNS resolution
DNS
Web Page
health checks
variable speed/bandwidth connections
Load Balancer
API calls
VIP
internal API call
publish message
Service A
Service B
Service C
DB
replication over WAN
consume message
Background Service X
m1
m2
m3
Topic A
m4
Topic B
Broker
Broker
Highly Available redundant Brokers
Distributed-Replicated Data Store
Datacenter B
Datacenter A

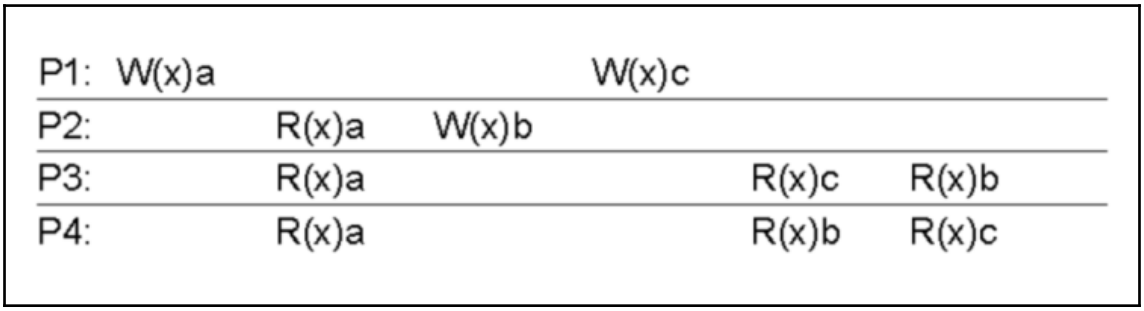

Latency Numbers Every Programmer Should Know
1 ns
L1 cache reference: 0.5 ns
Branch mispredict: 5 ns
L2 cache reference: 7 ns
Mutex lock/unlock: 25 ns
= 100 ns
Main memory reference: 100 ns
= 1 µs
Compress 1K bytes with Zippy: 3 µs
= 10 µs
Send 1K bytes over 1 Gbps network: 10 µs
SSD random read (1Gb/s SSD): 150 µs
Read 1 MB sequentially from memory: 250 µs
Round trip in same datacenter: 500 µs
= 1 ms
Read 1 MB sequentially from SSD: 1 ms
Disk seek: 10 ms
Read 1 MB sequentially from disk: 20 ms
Packet roundtrip CA to Netherlands: 150 ms
Source: https://gist.github.com/2841832

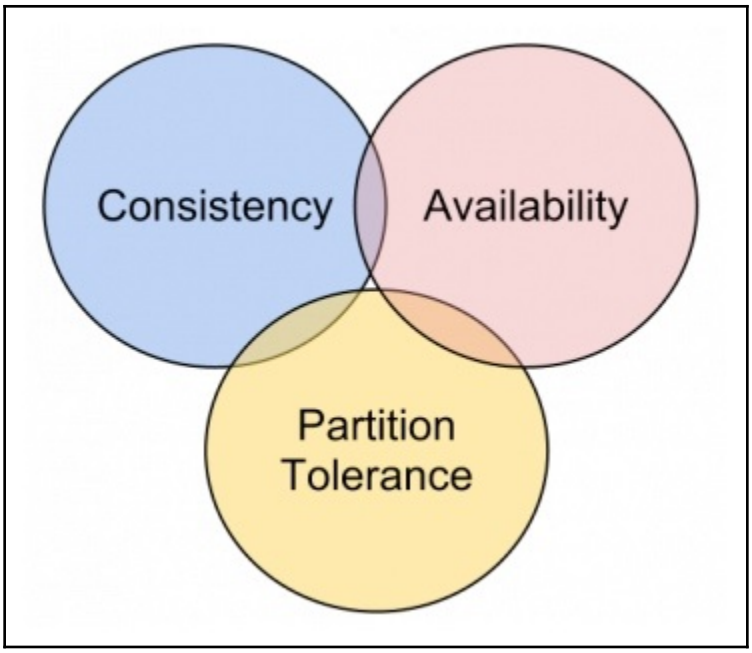

Client_x
Client_x
Process
Process
Process
DB
DB
DB
Distributed-Replicated Data Store

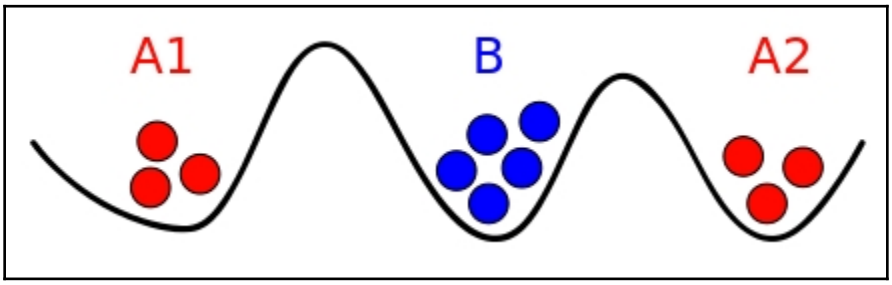

Transaction Manager
XA Resource
Request to Prepare
Prepare Phase
Prepared
Commit
Commit Phase
Done

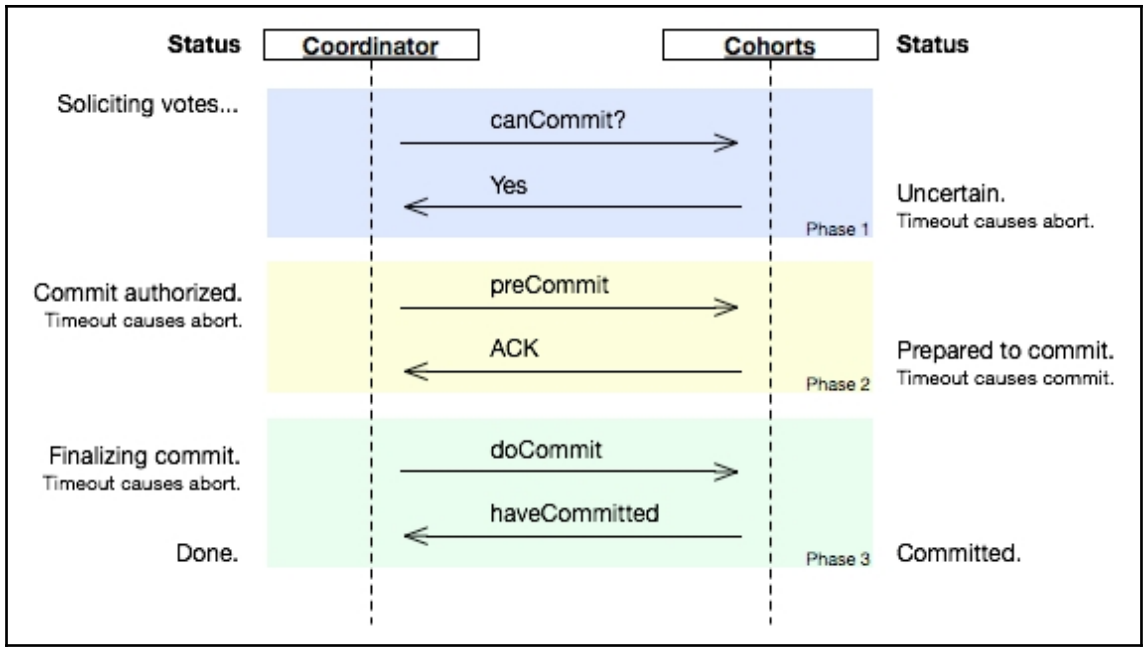

P1: W(x)a
P2: R(x)a

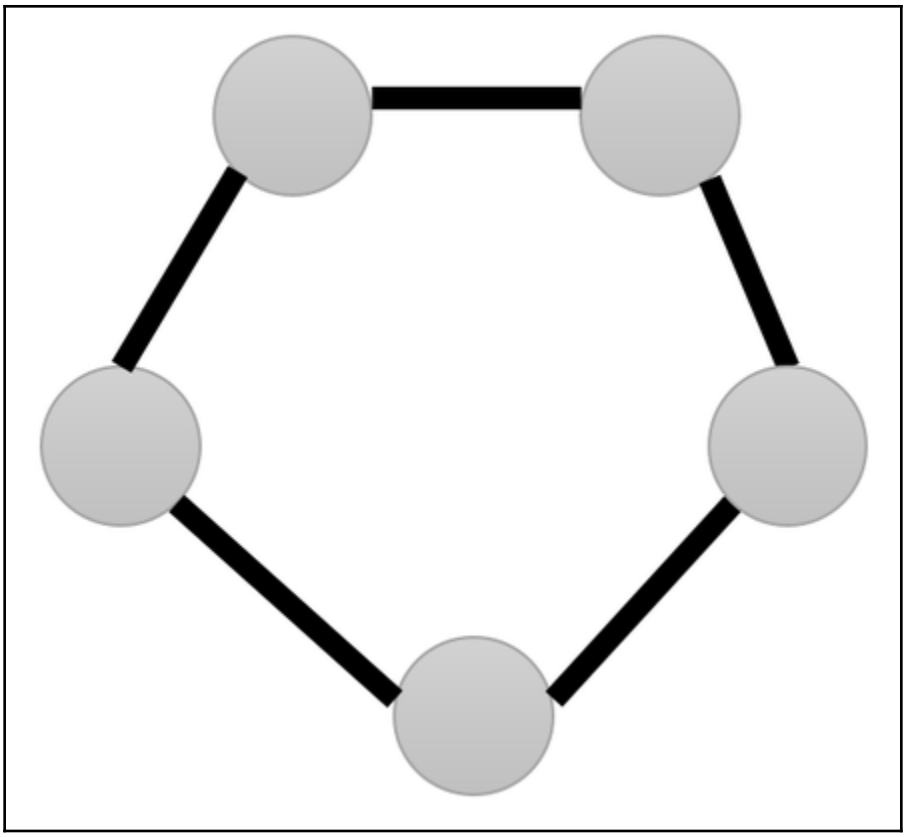

P1: W(x)a
P2: W(x)b
P3: R(x)b R(x)a
P4: R(x)b R(x)a
P1: W(x)a
P2: W(x)b
P3: R(x)b R(x)a
P4: R(x)a R(x)b

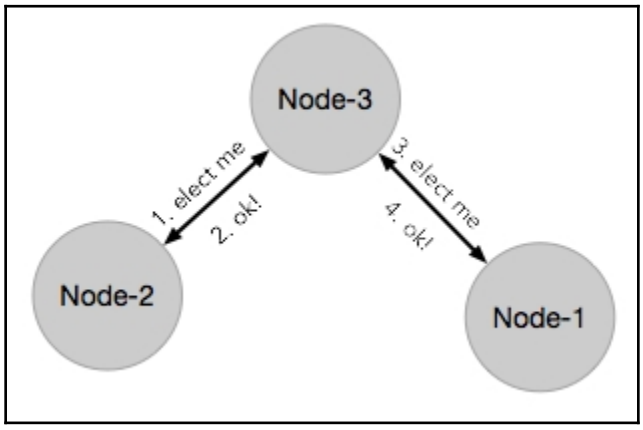

P1: W(x)a W(x)c
P2: R(x)a W(x)b
P3: R(x)a R(x)c R(x)b
P4: R(x)a R(x)b R(x)c

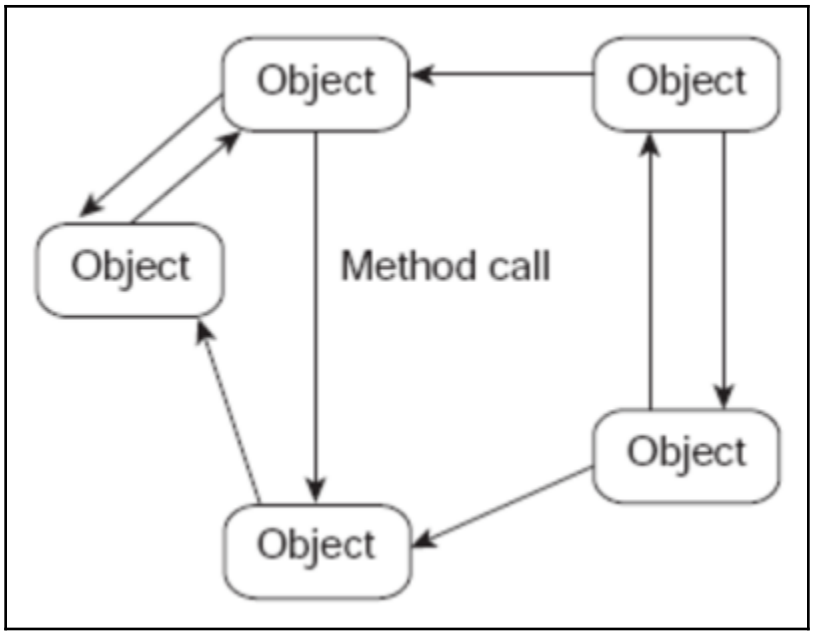

Consistency
Availability
Partition
Tolerance

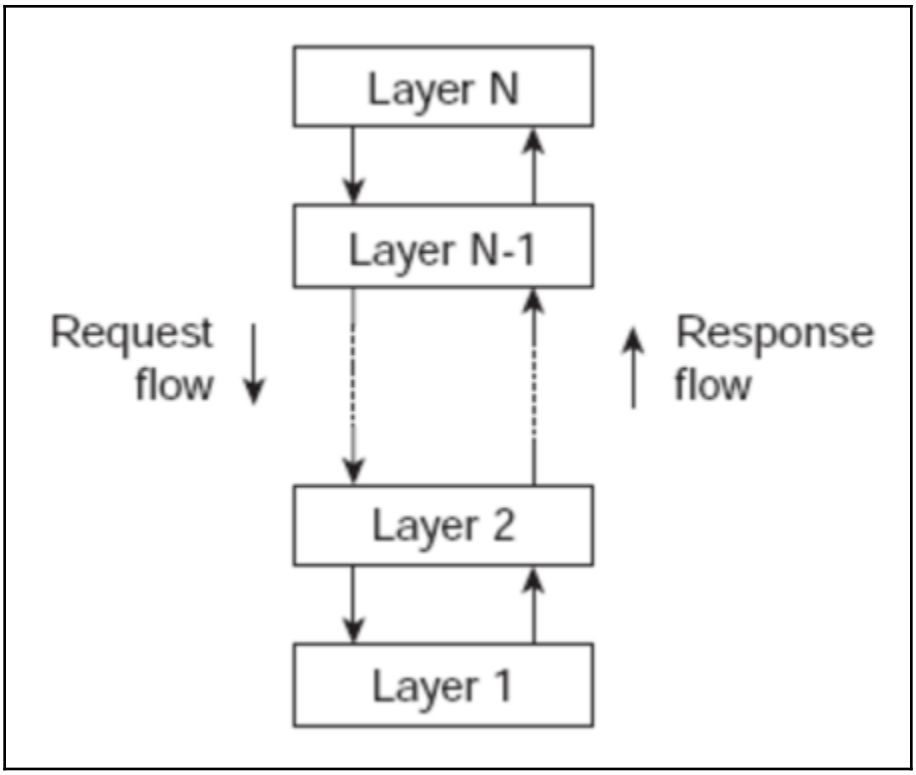

A1
B
A2

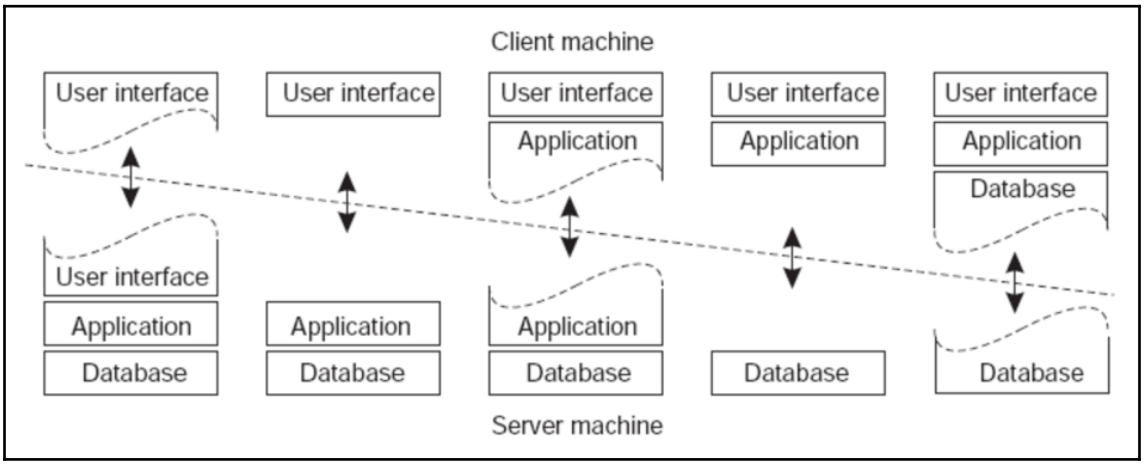

Status
Coordinator
Cohorts
Status
Soliciting votes...
canCommit?
Yes
Uncertain.
Timeout causes abort.
Phase 1
Commit authorized.
Timeout causes abort.
preCommit
ACK
Prepared to commit.
Timeout causes commit.
Phase 2
Finalizing commit.
Timeout causes abort.
doCommit
haveCommitted
Done.
Committed.
Phase 3

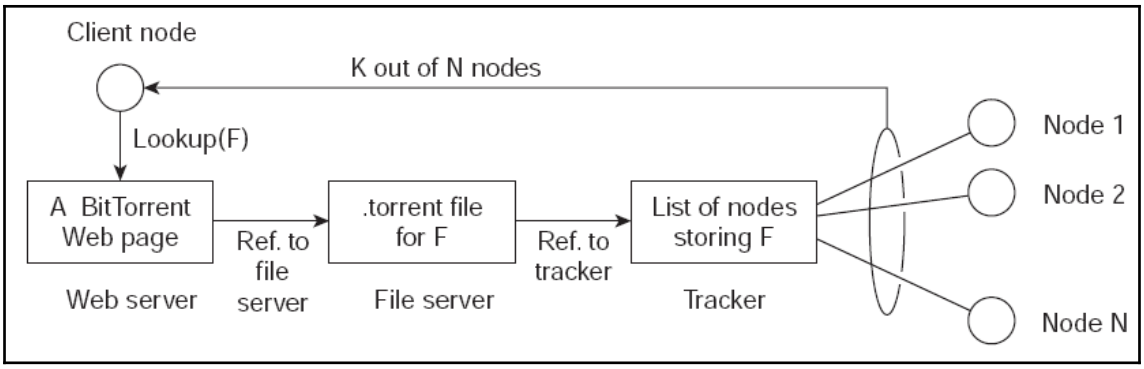


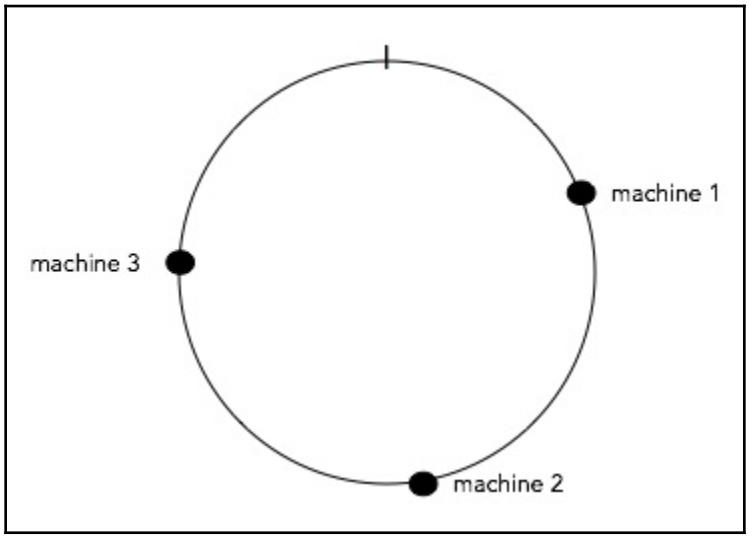

Node-3
1. elect me
2. ok!
3. elect me
4. ok!
Node-2
Node-1

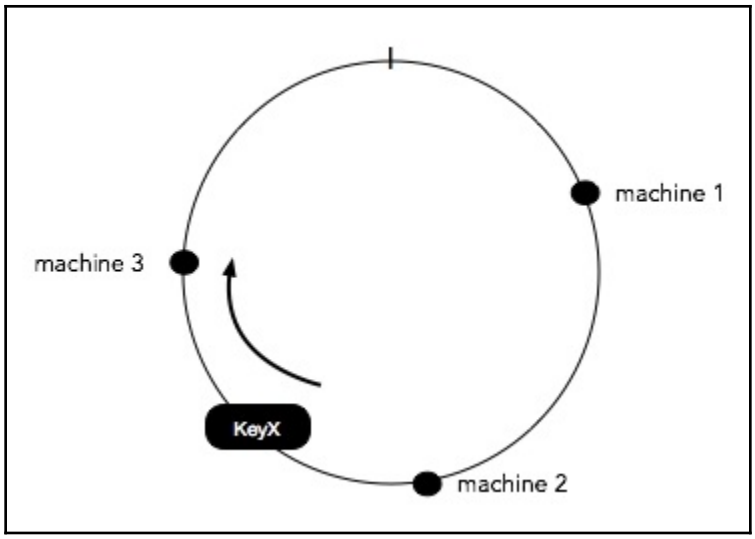

Object
Object
Object
Method call
Object
Object

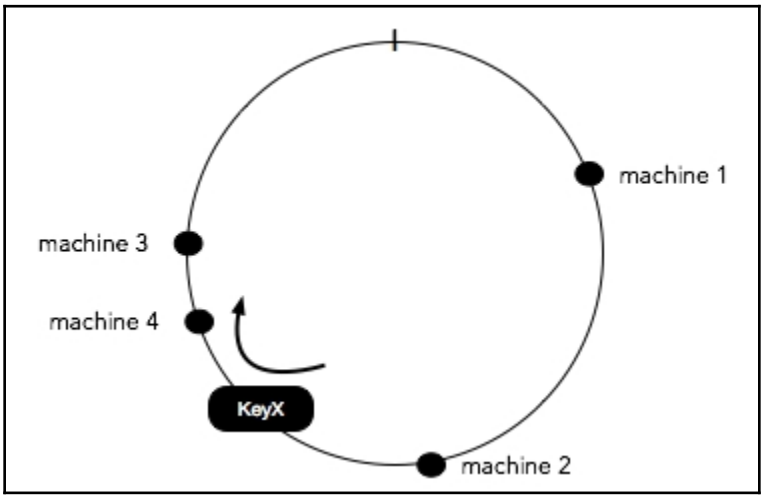

Layer N
Layer N-1
Request flow
Response flow
Layer 2
Layer 1

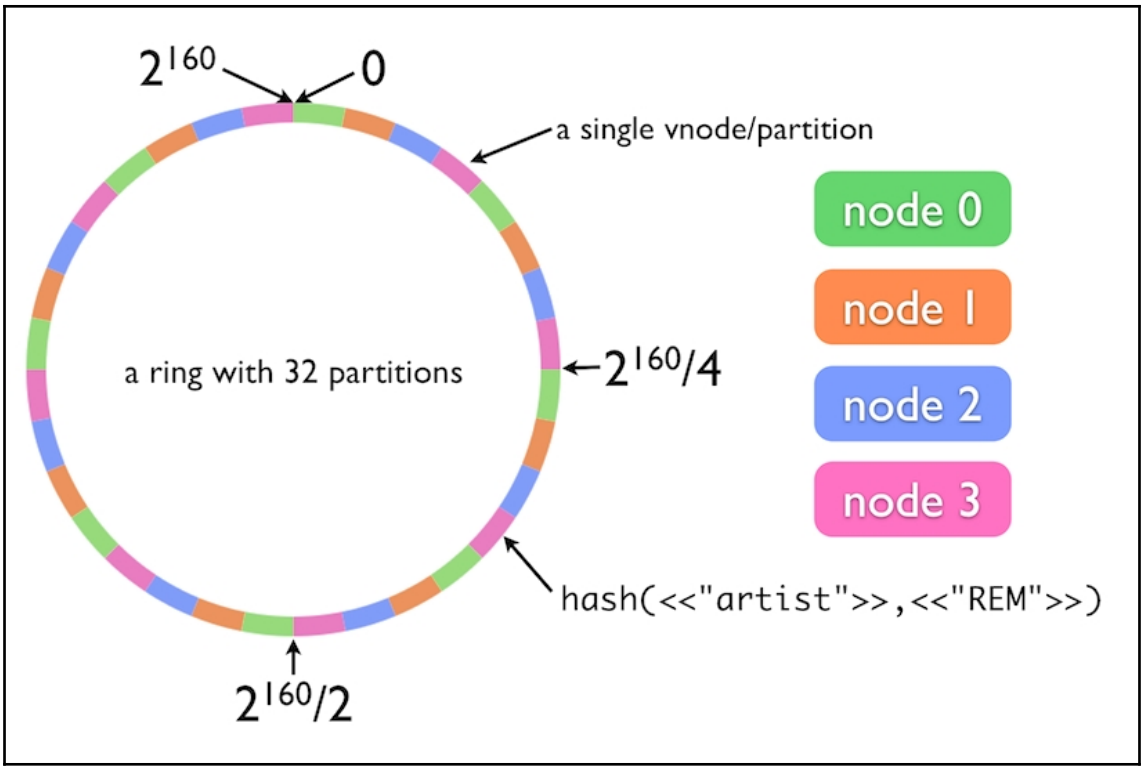

Client machine
User interface
User interface
User interface
Application
User interface
Application
User interface
Application
Database
User interface
Application
Database
Application
Database
Application
Database
Database
Database
Server machine

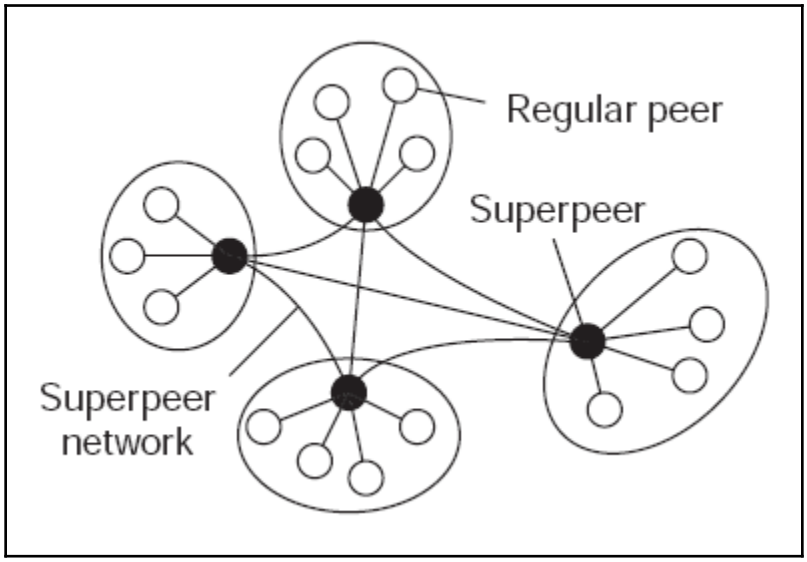

Client node
K out of N nodes
Lookup(F)
A BitTorrent Web page
Ref. to file server
.torrent file for F
Ref. to tracker
List of nodes storing F
Web server
File server
Tracker
Node 1
Node 2
Node N

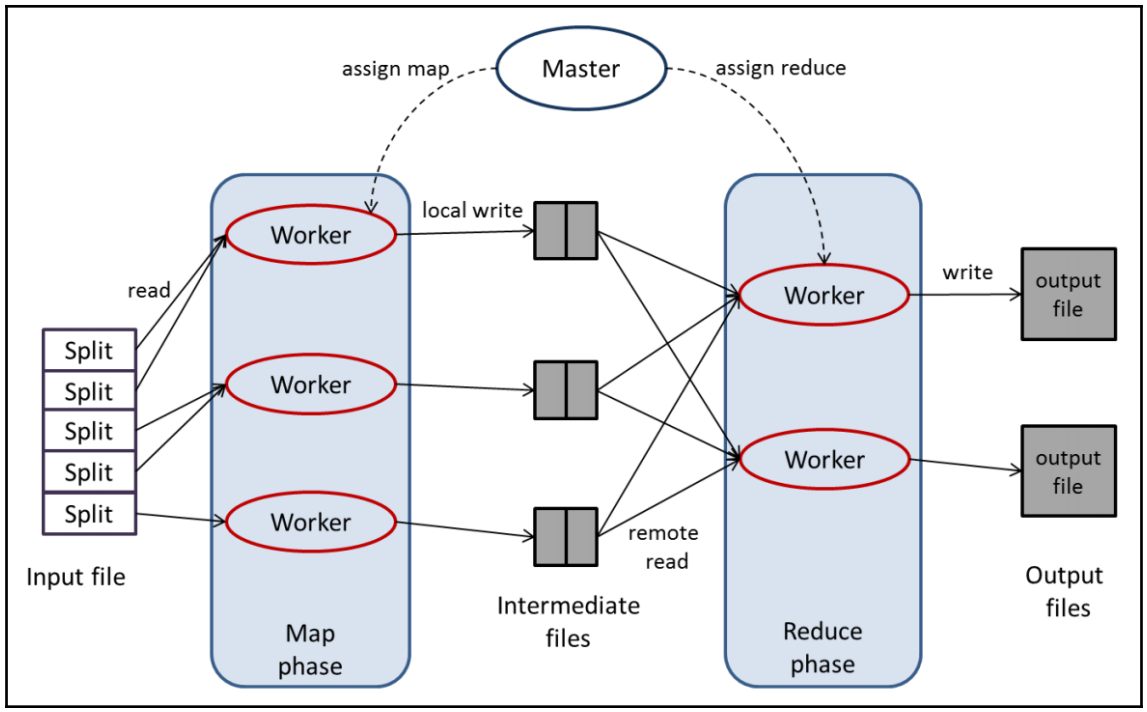

machine 1
machine 3
machine 2

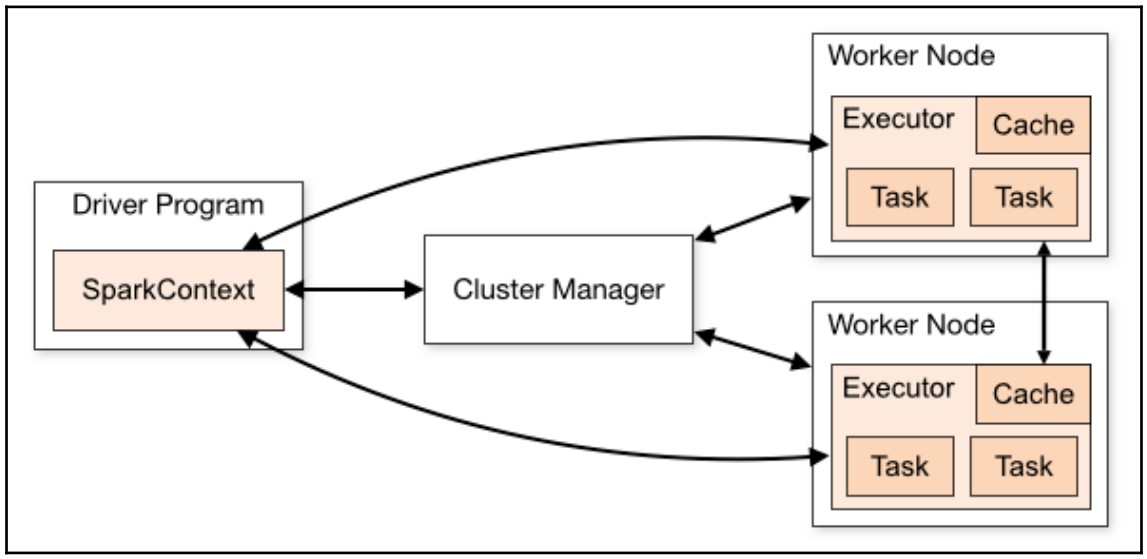

machine 1
machine 3
KeyX
machine 2

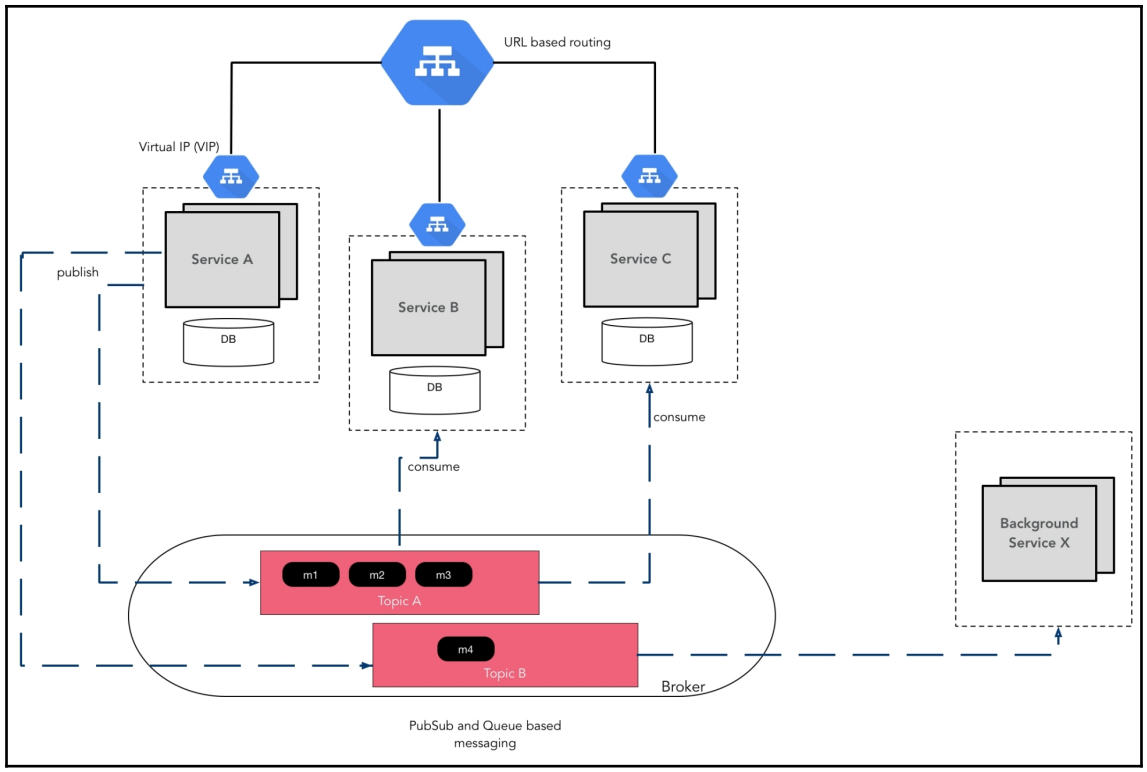

machine 1
machine 3
machine 4
KeyX
machine 2

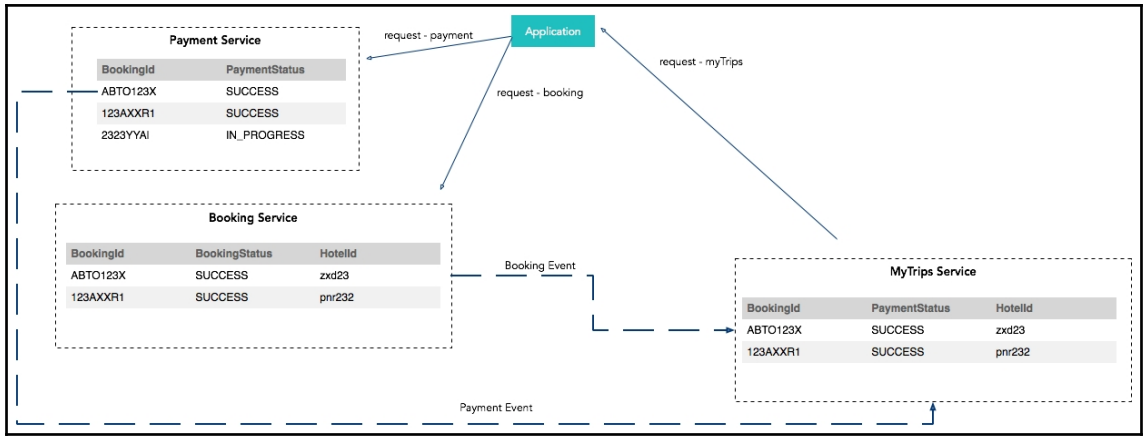

2^160
0
a single vnode/partition
node 0
node 1
node 2
node 3
a ring with 32 partitions
2^160/4
hash(<<"artist">>,<<"REM">>)
2^160/2

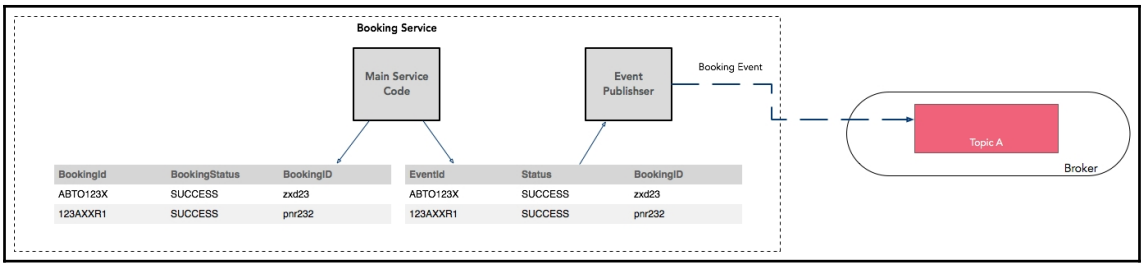

Regular peer
Superpeer
Superpeer
network

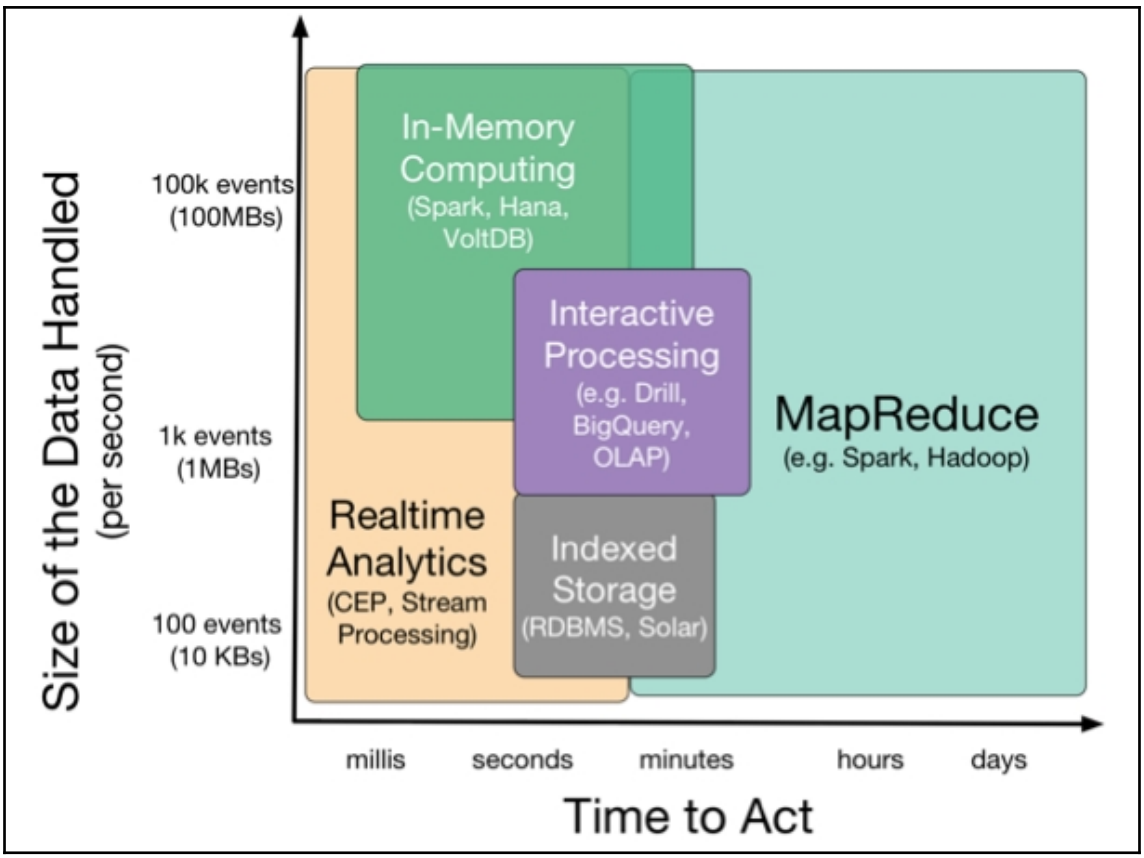

Master
assign map
assign reduce
Worker
Worker
Worker
local write
read
Split
Split
Split
Split
Split
Input file
Map
phase
Intermediate
files
remote
read
Worker
Worker
write
output
file
output
file
Output
files
Reduce
phase


Worker Node
Executor
Cache
Task
Task
Driver Program
SparkContext
Cluster Manager
Worker Node
Executor
Cache
Task
Task

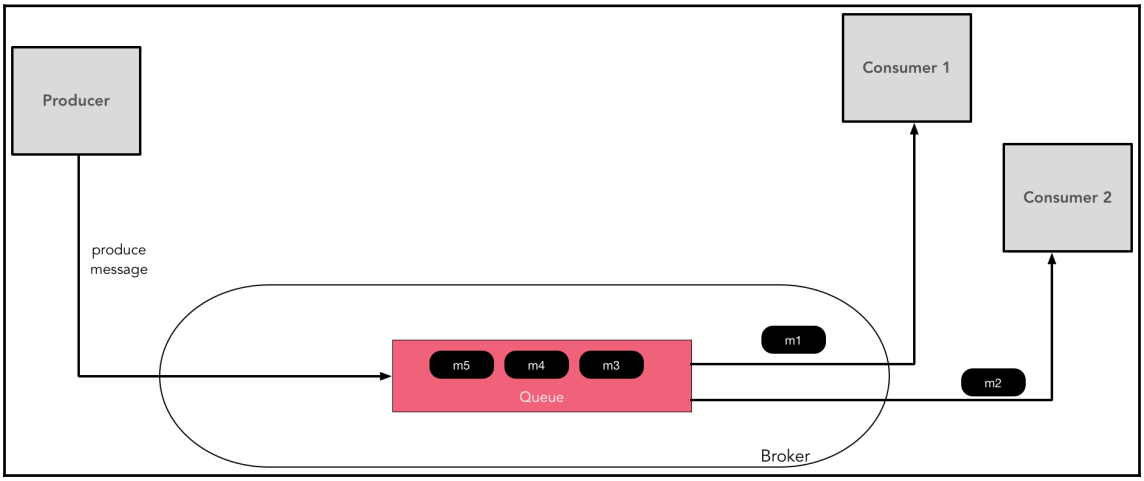

URL based routing
Virtual IP (VIP)
Service A
Service B
Service C
DB
DB
DB
publish
consume
consume
Background Service X
m1
m2
m3
Topic A
m4
Topic B
Broker
PubSub and Queue based messaging

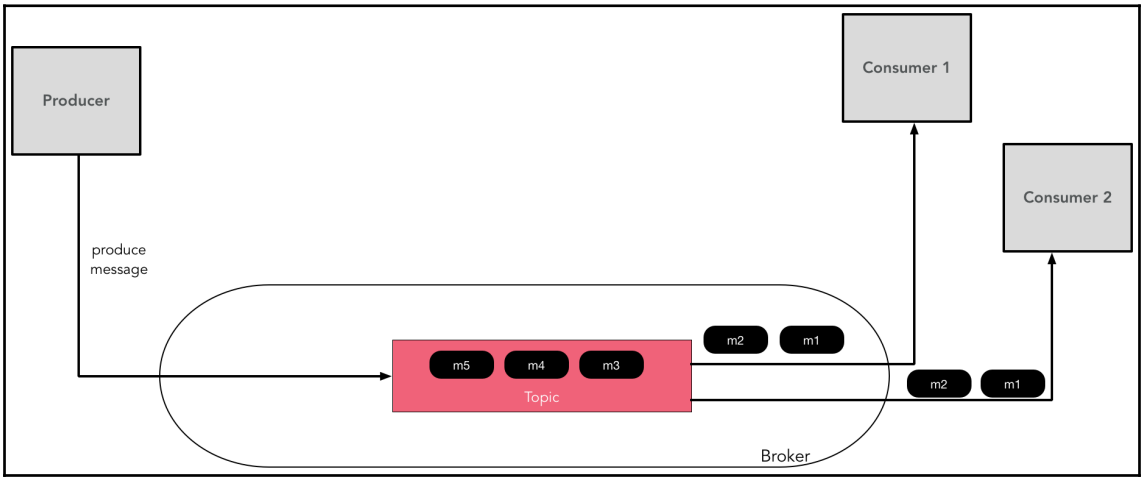

Payment Service
BookingId PaymentStatus
ABTO123X SUCCESS
123AXXR1 SUCCESS
2323YYAI IN_PROGRESS
request - payment
Application
request - myTrips
request - booking
Booking Service
BookingId BookingStatus HotelId
ABTO123X SUCCESS zxd23
123AXXR1 SUCCESS pnr232
Booking Event
MyTrips Service
BookingId PaymentStatus HotelId
ABTO123X SUCCESS zxd23
123AXXR1 SUCCESS pnr232
Payment Event

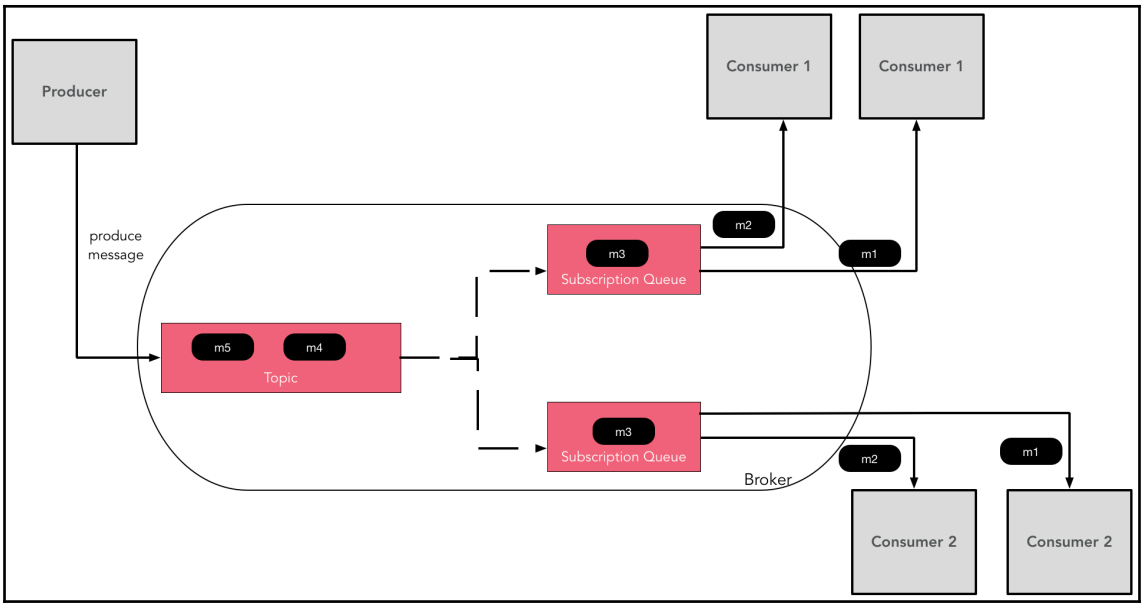

Booking Service
Main Service Code
Event Publishser
Booking Event
Topic A
Broker
BookingId
BookingStatus
BookingID
ABTO123X
SUCCESS
zxd23
123AXXR1
SUCCESS
pnr232
EventId
Status
BookingID
ABTO123X
SUCCESS
zxd23
123AXXR1
SUCCESS
pnr232

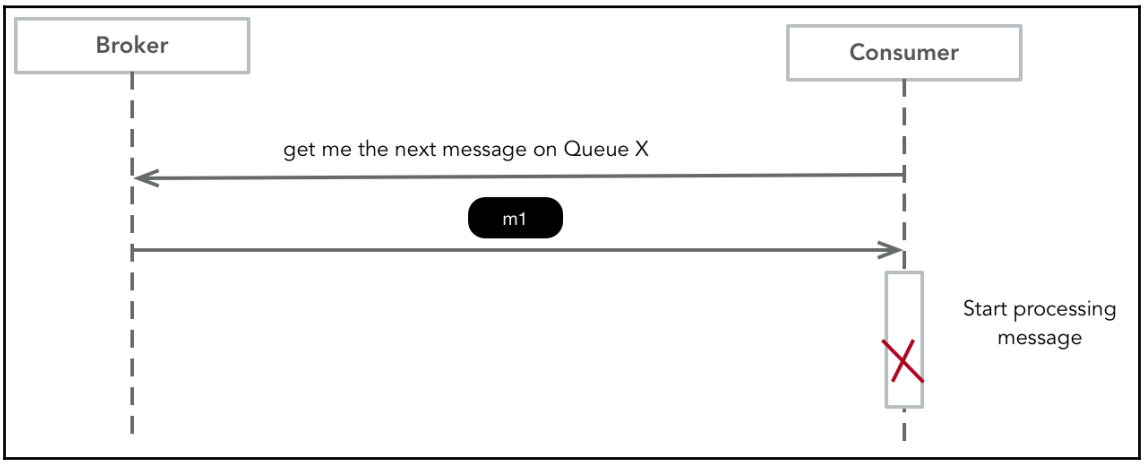

Size of the Data Handled
(per second)
100k events
(100MBs)
1k events
(1MBs)
100 events
(10 KBs)
In-Memory Computing
(Spark, Hana, VoltDB)
Interactive Processing
(e.g. Drill, BigQuery, OLAP)
MapReduce
(e.g. Spark, Hadoop)
Realtime Analytics
(CEP, Stream Processing)
Indexed Storage
(RDBMS, Solar)
millis
seconds
minutes
hours
days
Time to Act

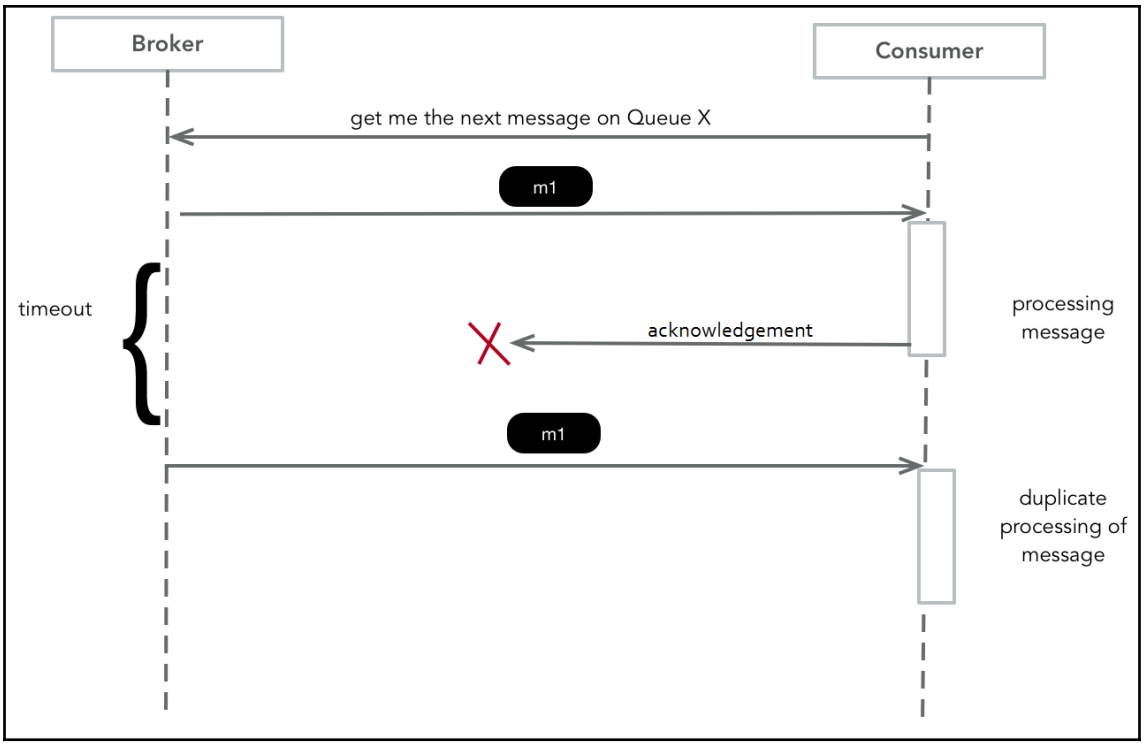
Chapter 6: Messaging

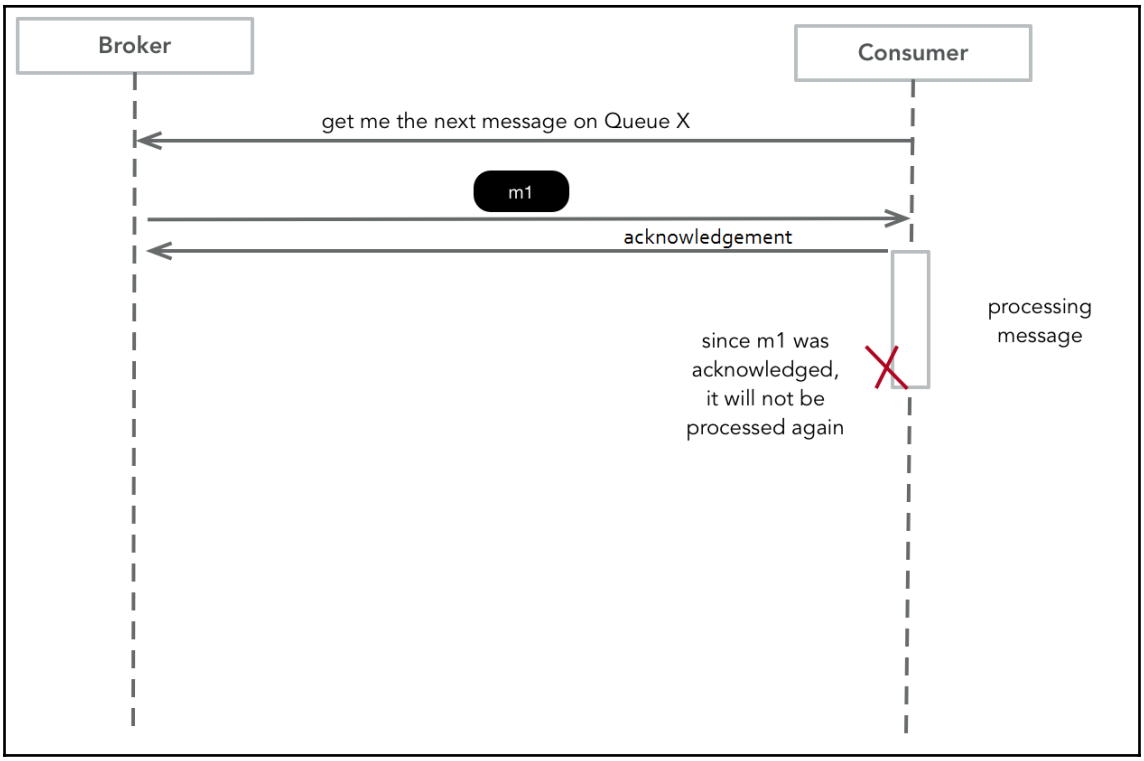


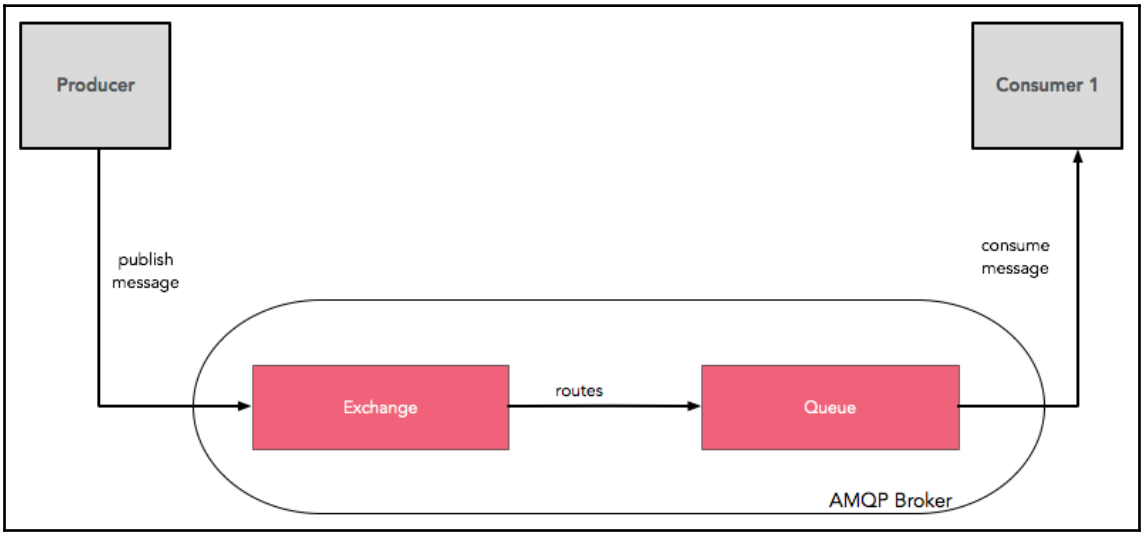


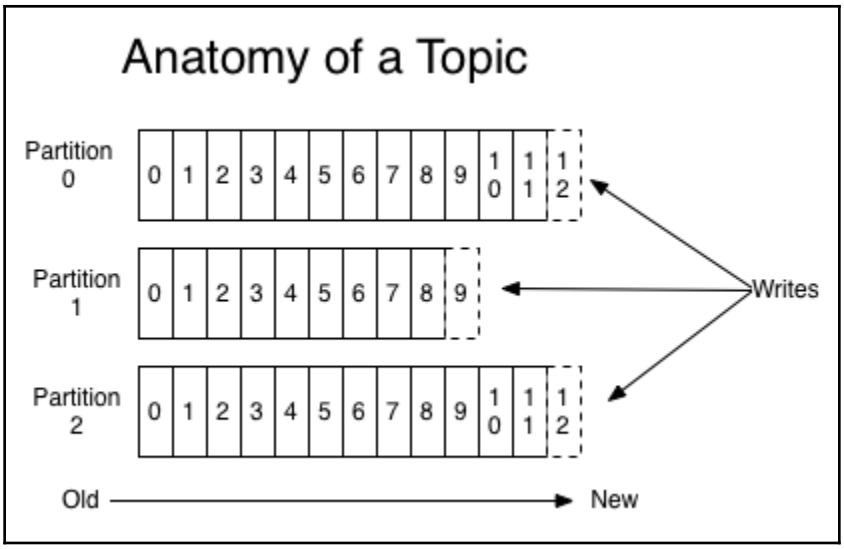

Producer
Consumer 1
Consumer 1
produce
message
m2
m3
Subscription Queue
m1
m5
m4
Topic
m3
Subscription Queue
m2
m1
Broker
Consumer 2
Consumer 2

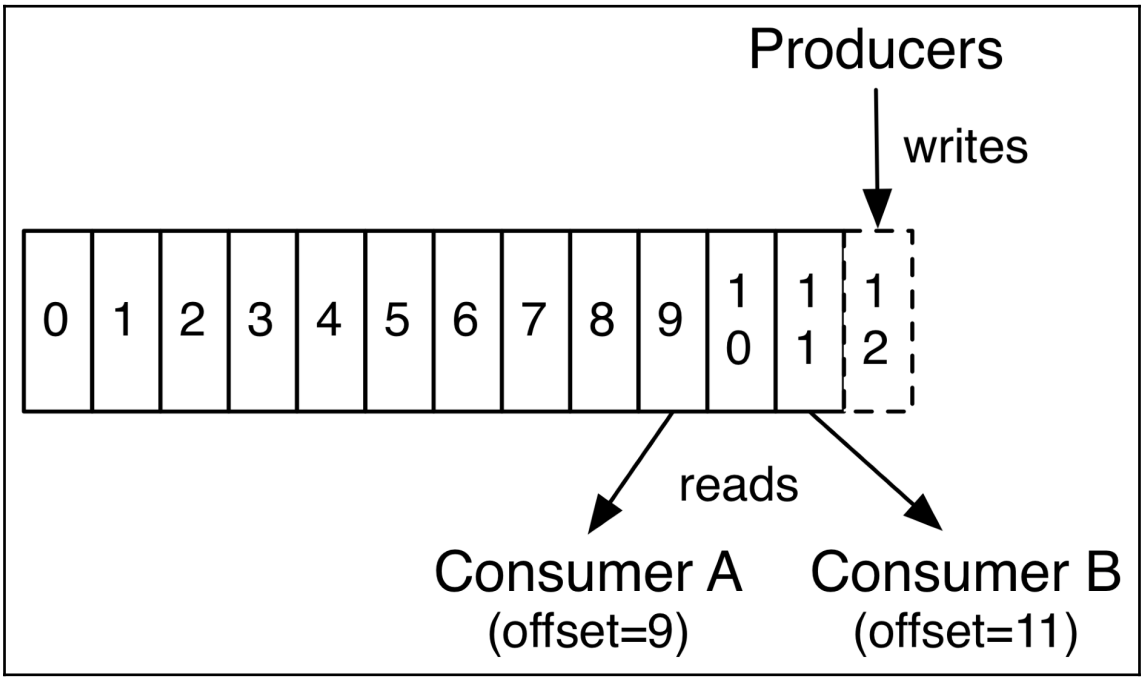

Broker
Consumer
get me the next message on Queue X
m1
Start processing
message

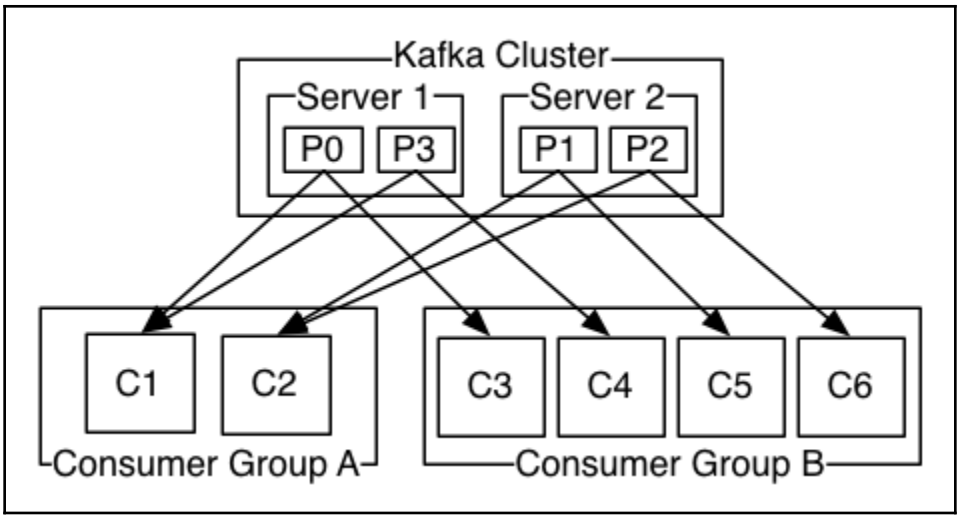

Broker
Consumer
get me the next message on Queue X
m1
timeout
acknowledgement
processing message
m1
duplicate processing of message

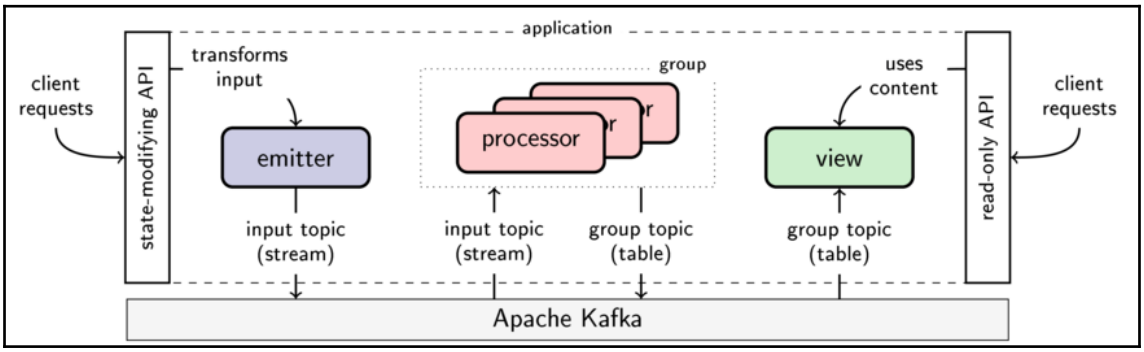

Broker
Consumer
get me the next message on Queue X
m1
acknowledgement
processing message
since m1 was acknowledged, it will not be processed again

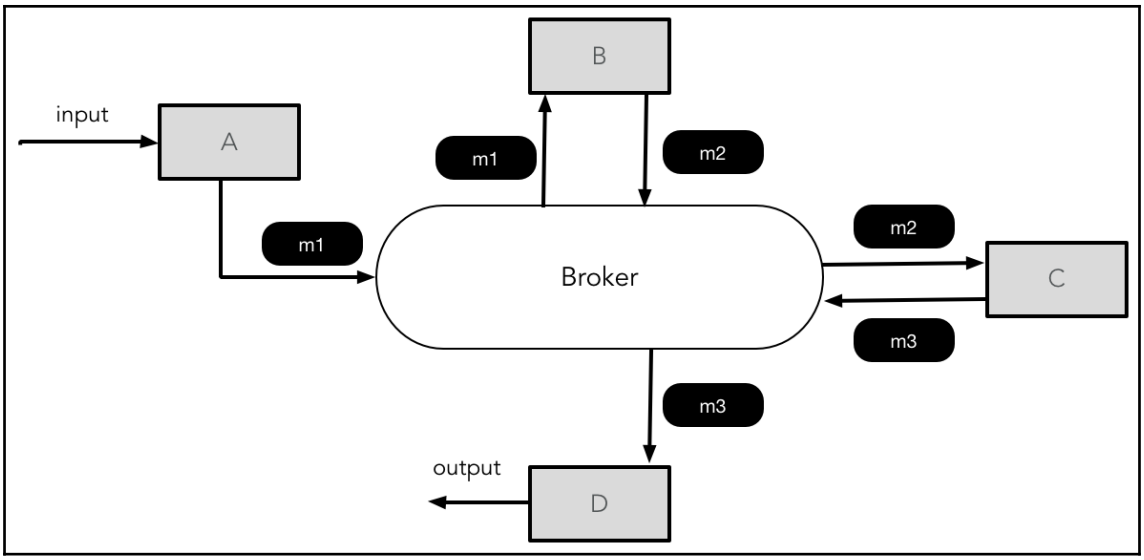

Producer
Consumer 1
publish message
consume message
Exchange
routes
Queue
AMQP Broker

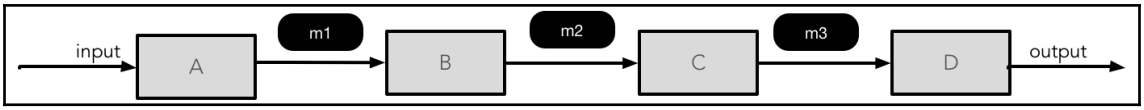

Anatomy of a Topic
Partition 0
0 1 2 3 4 5 6 7 8 9 10 11 12
Partition 1
0 1 2 3 4 5 6 7 8 9
Partition 2
0 1 2 3 4 5 6 7 8 9 10 11 12
Writes
Old
New

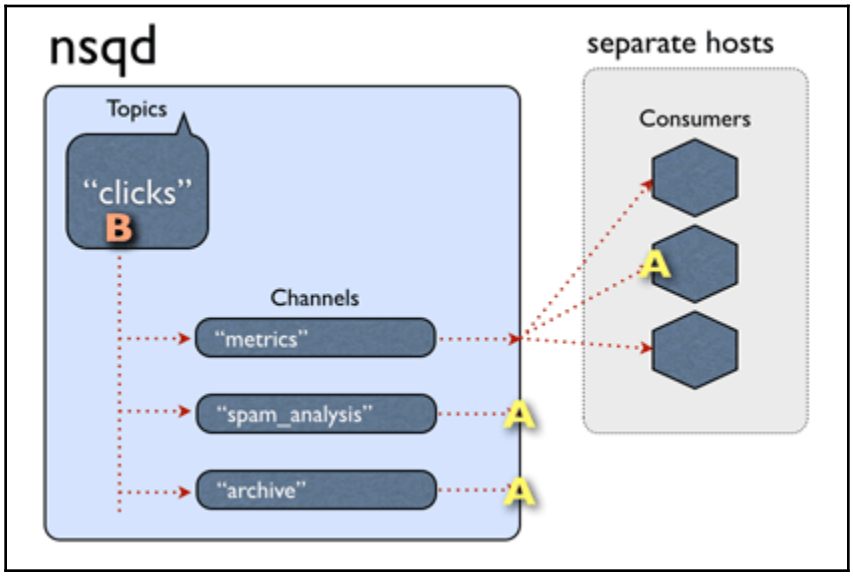

Producers
writes
0 1 2 3 4 5 6 7 8 9 10 11 12
reads
Consumer A
(offset=9)
Consumer B
(offset=11)

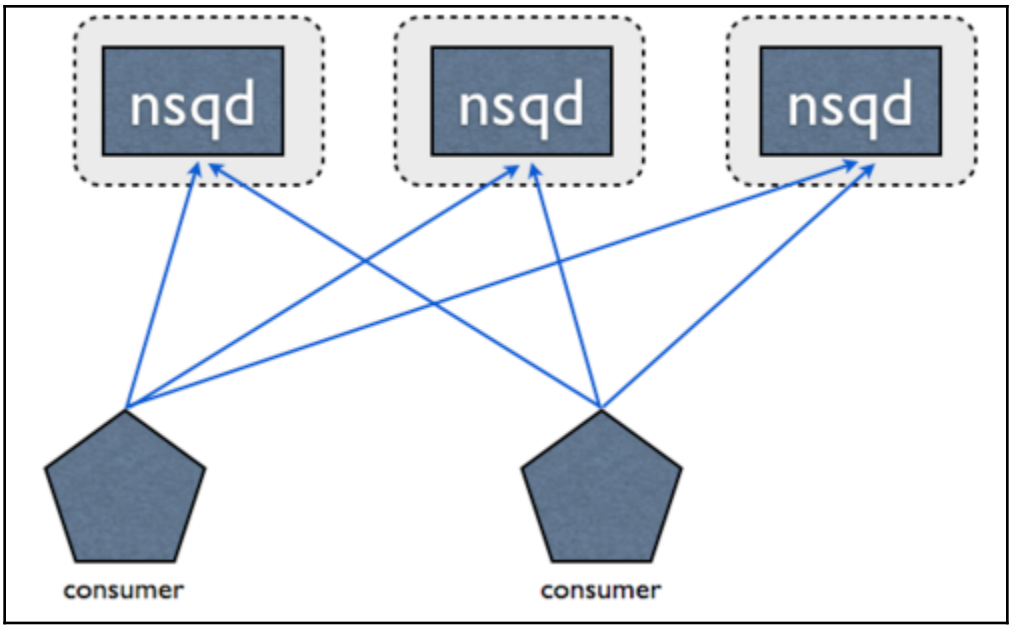

Kafka Cluster
Server 1
Server 2
P0
P3
P1
P2
C1
C2
C3
C4
C5
C6
Consumer Group A
Consumer Group B

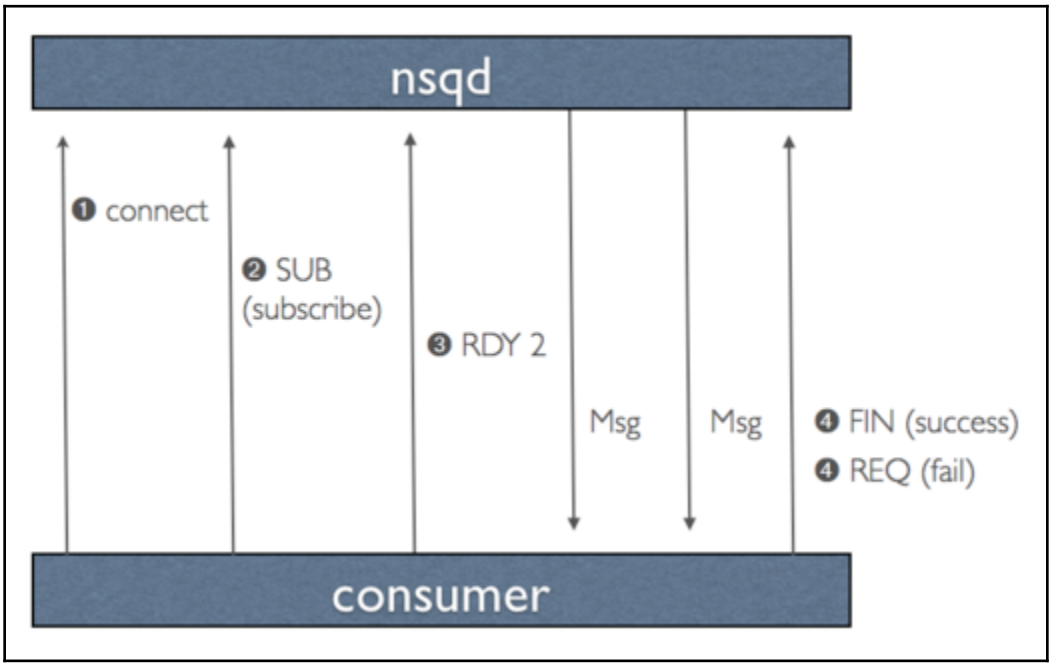

application
client requests
state-modifying API
transforms input
emitter
input topic (stream)
group
processor
input topic (stream)
group topic (table)
uses content
view
group topic (table)
read-only API
client requests
Apache Kafka

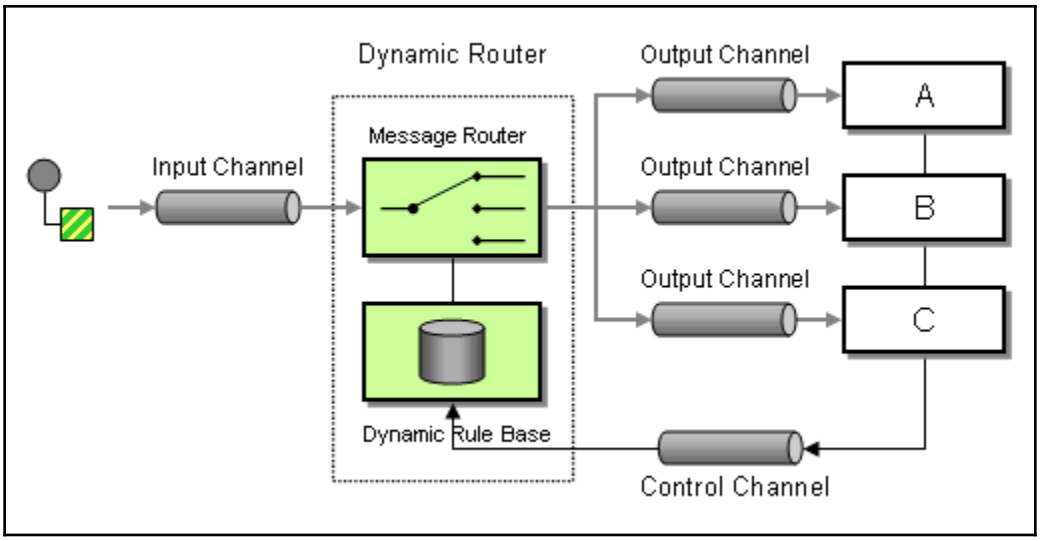

B
input
A
m1
m2
m1
Broker
m2
C
m3
m3
output
D


input
A
m1
B
m2
C
m3
D
output

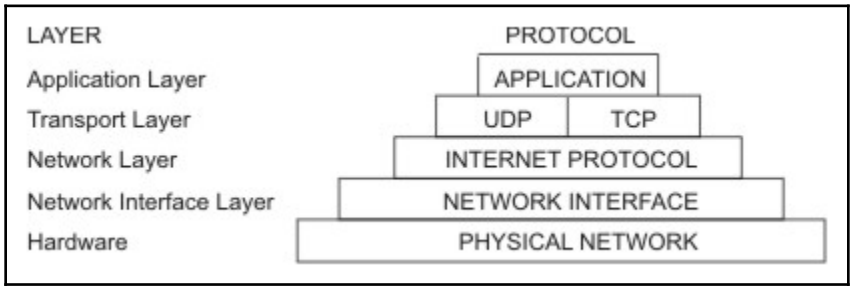

nsqd
separate hosts
Topics
"clicks"
B
Consumers
A
Channels
"metrics"
"spam_analysis"
A
"archive"
A

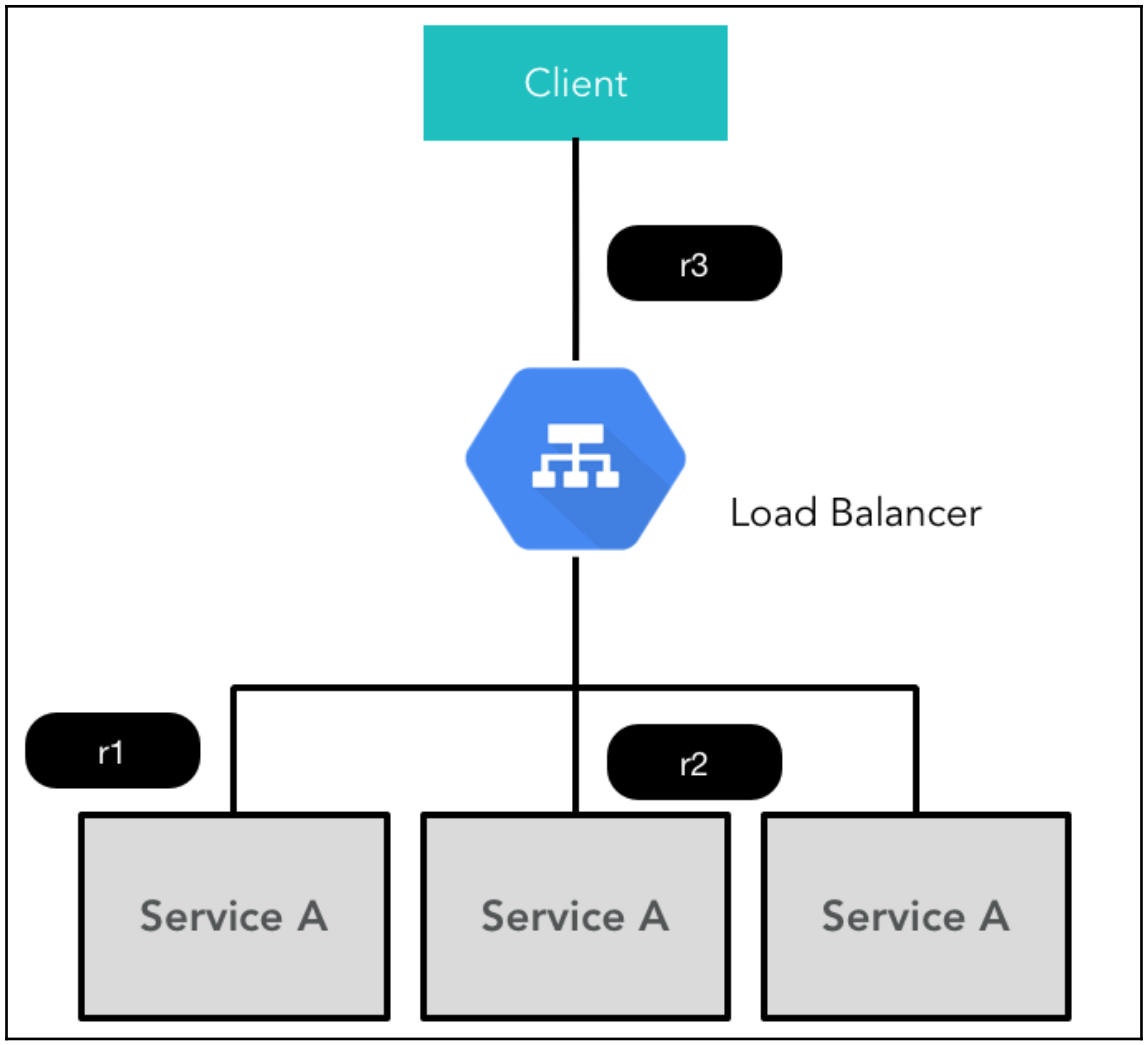

nsqd
nsqd
nsqd
consumer
consumer

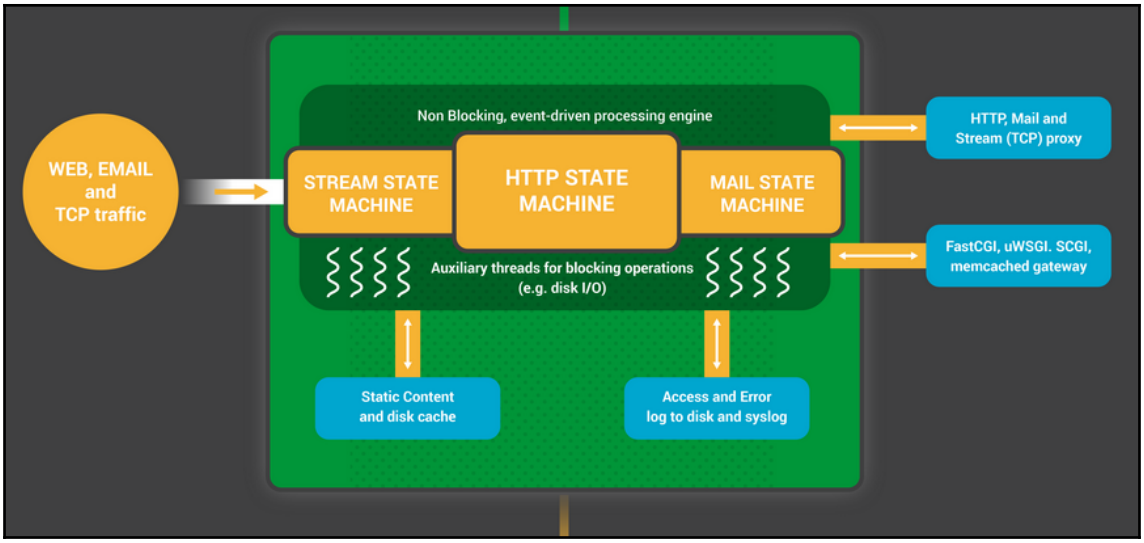

nsqd
❶ connect
❷ SUB
(subscribe)
❸ RDY 2
Msg
Msg
❹ FIN (success)
❹ REQ (fail)
consumer

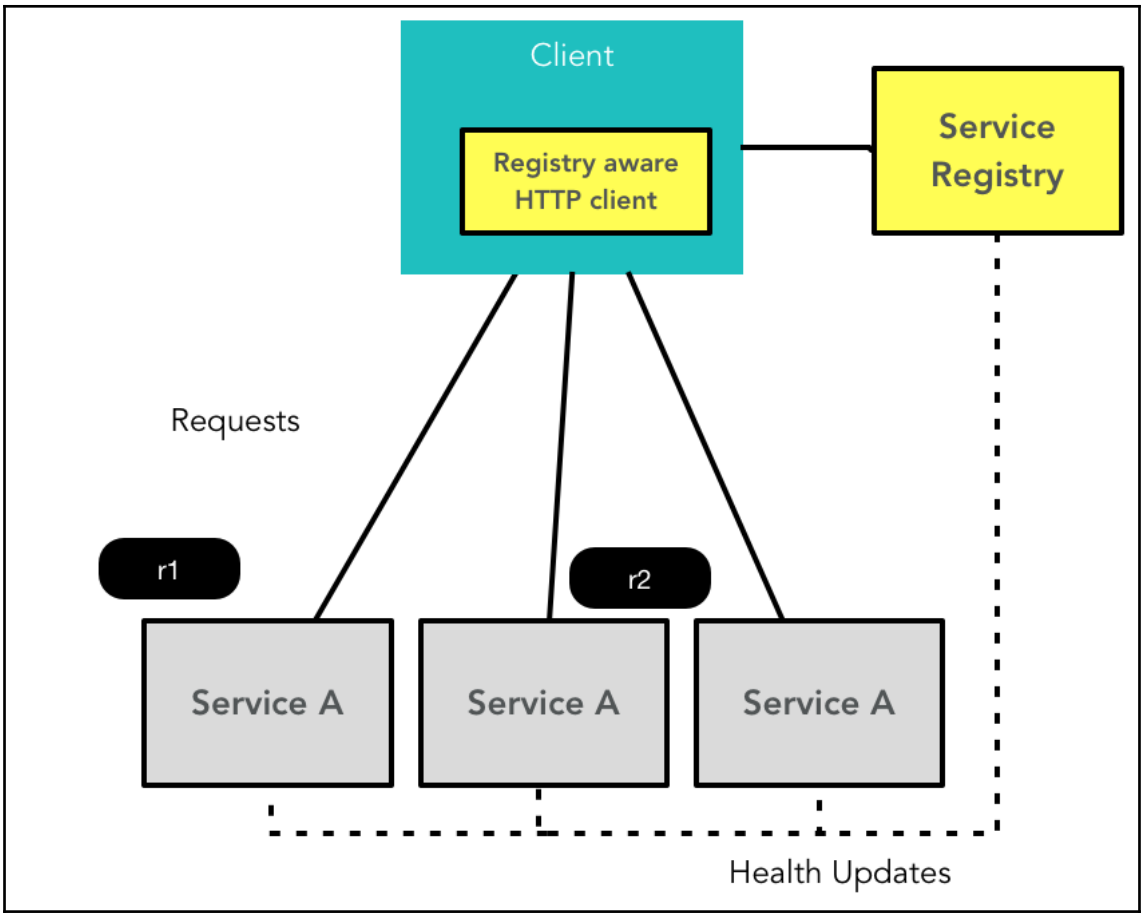

Dynamic Router
Output Channel
A
Message Router
Input Channel
Output Channel
B
Output Channel
C
Dynamic Rule Base
Control Channel

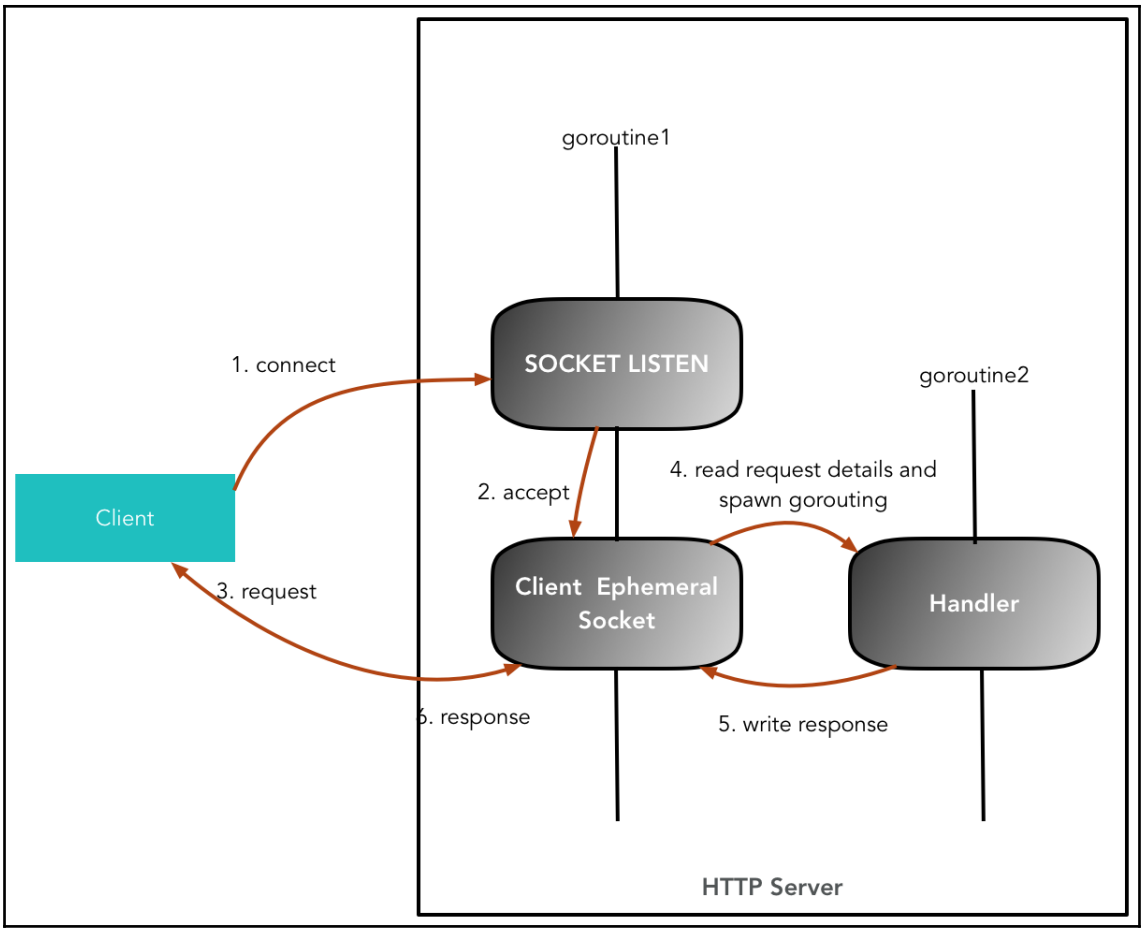
Chapter 7: Building APIs

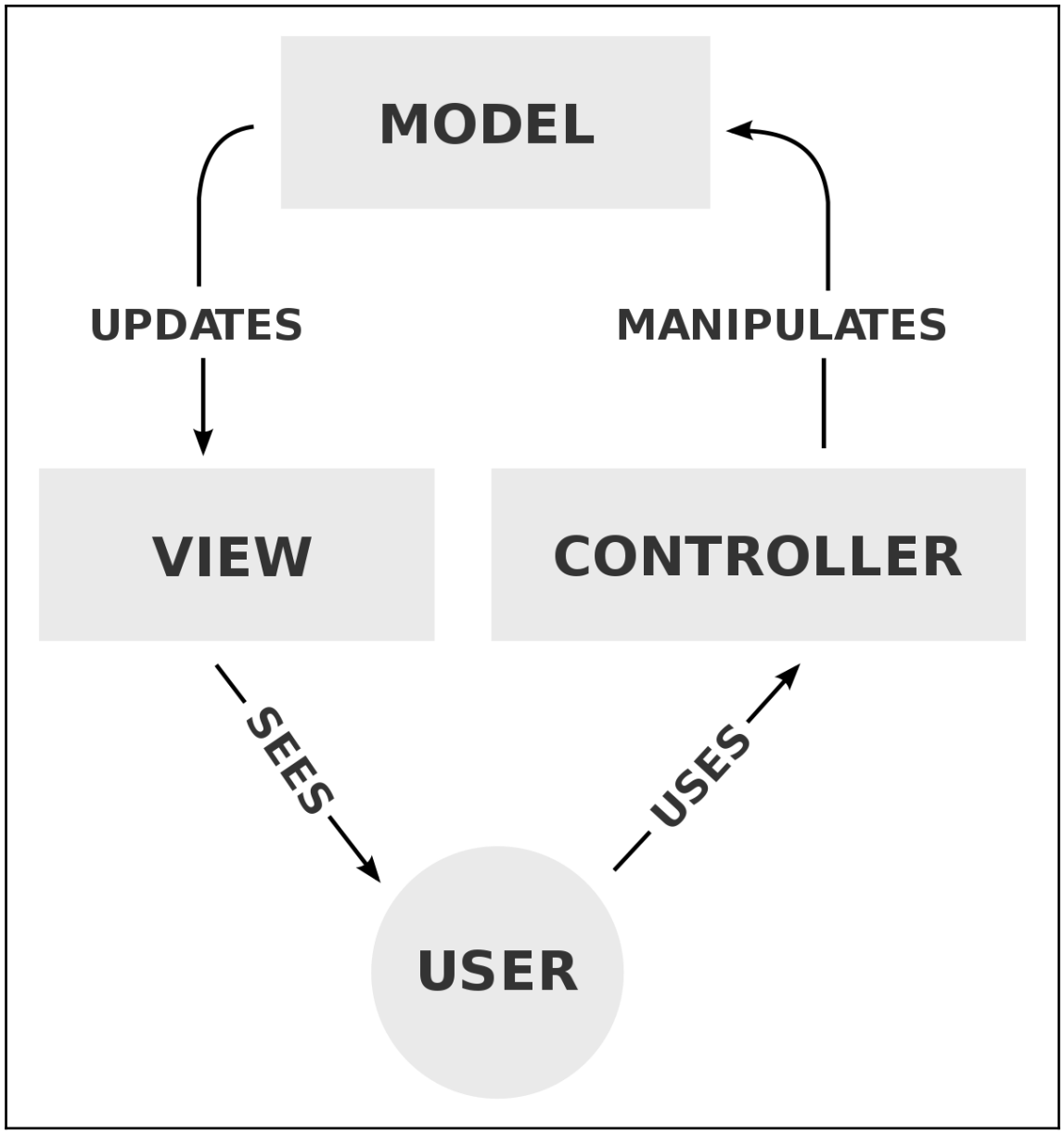


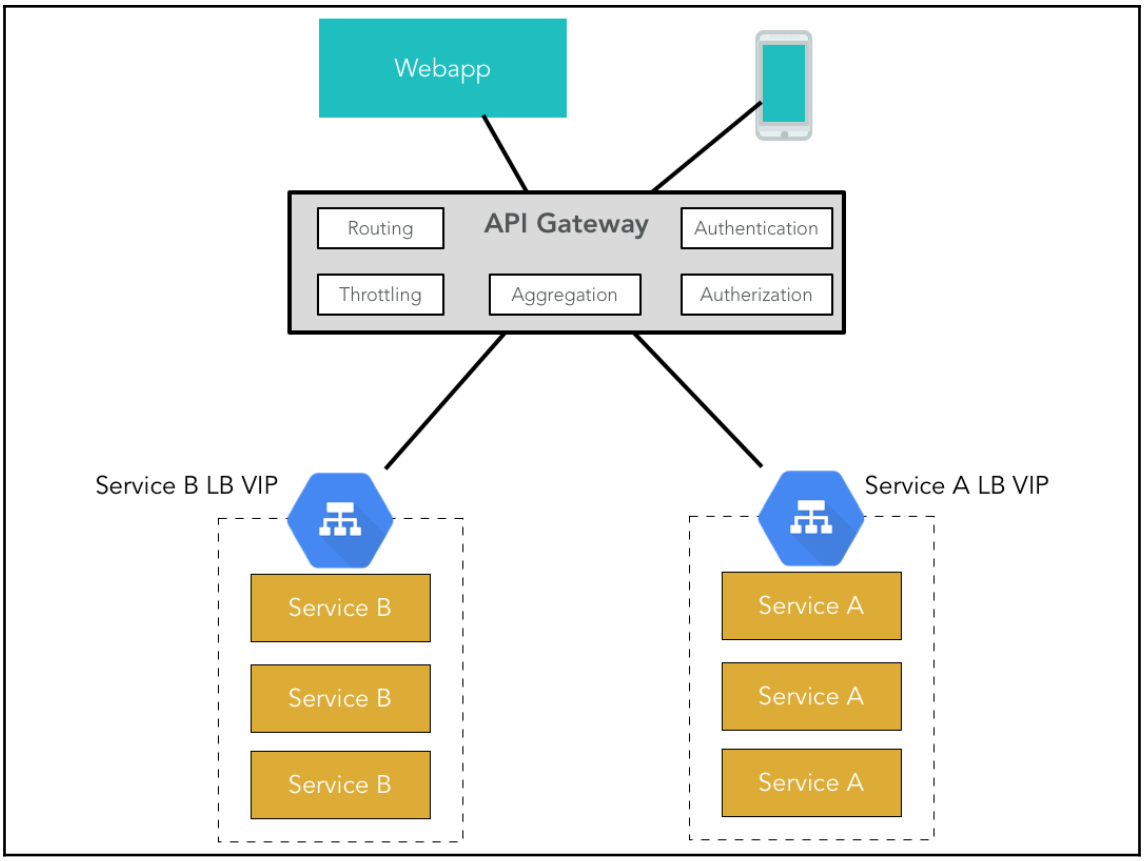

Client
r3
Load Balancer
r1
r2
Service A
Service A
Service A

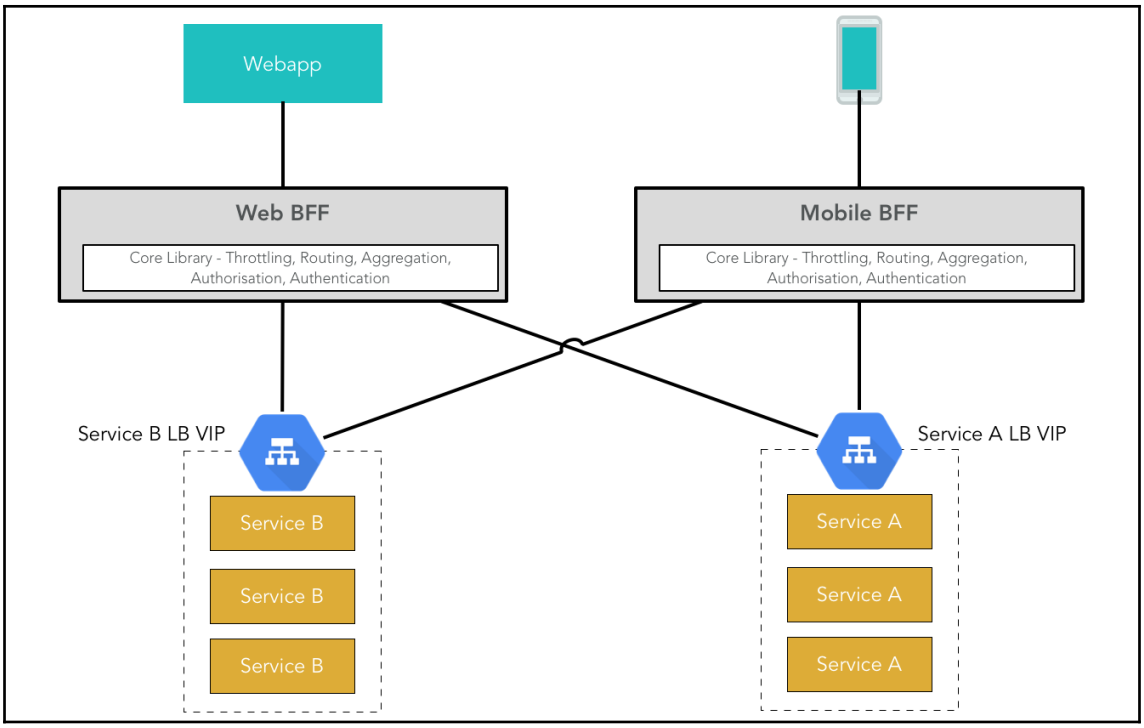

WEB, EMAIL
and
TCP traffic
Non Blocking, event-driven processing engine
STREAM STATE MACHINE
HTTP STATE MACHINE
MAIL STATE MACHINE
Auxiliary threads for blocking operations
(e.g. disk I/O)
HTTP, Mail and
Stream (TCP) proxy
FastCGI, uWSGI. SCGI,
memcached gateway
Static Content
and disk cache
Access and Error
log to disk and syslog

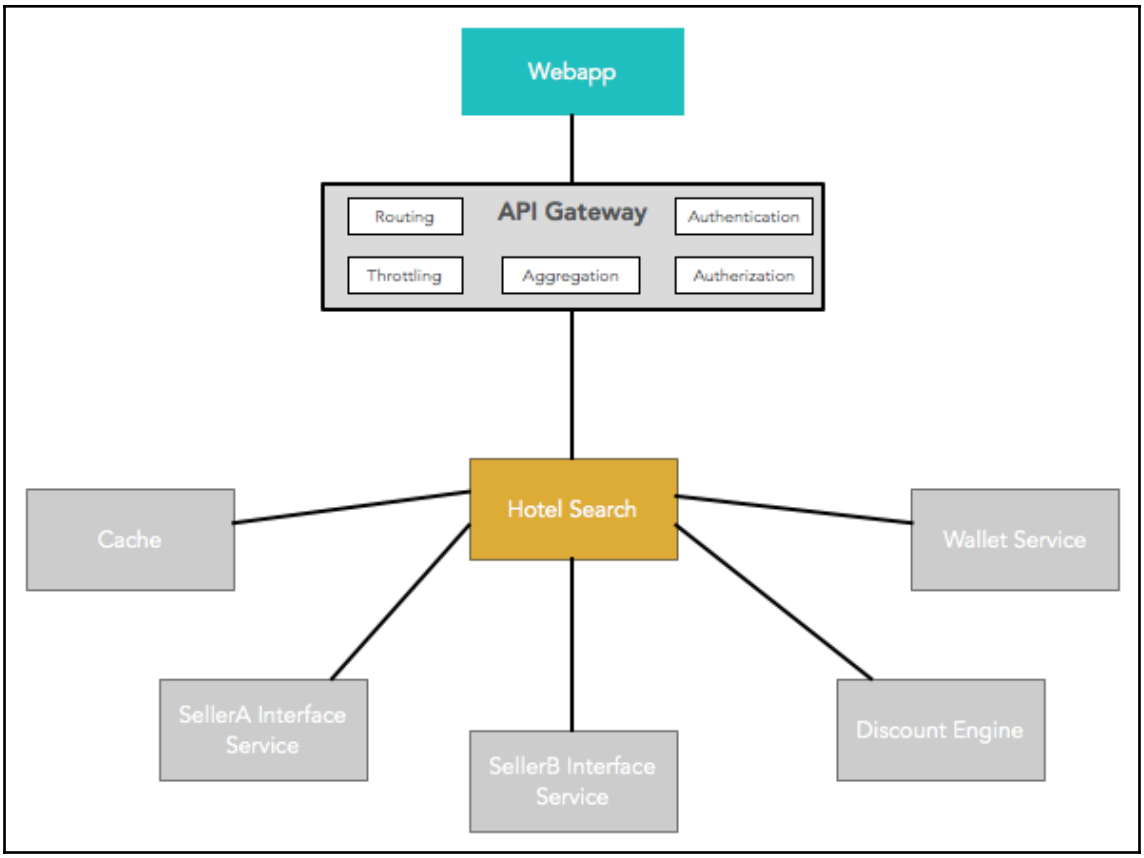

Client
Registry aware HTTP client
Service Registry
Requests
r1
r2
Service A
Service A
Service A
Health Updates

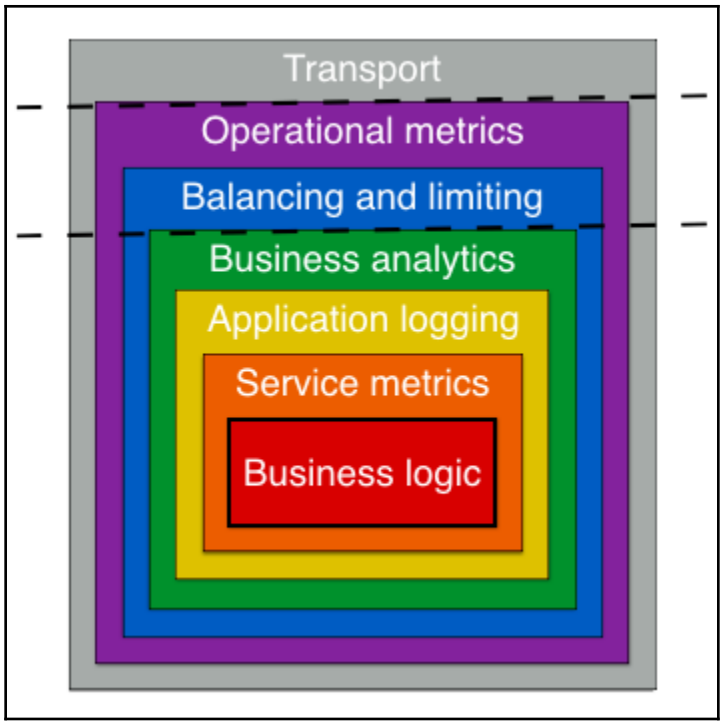

goroutine1
SOCKET LISTEN
1. connect
goroutine2
Client
2. accept
4. read request details and
spawn gorouting
Client Ephemeral
Socket
Handler
3. request
6. response
5. write response
HTTP Server


MODEL
UPDATES
MANIPULATES
VIEW
CONTROLLER
SEES
USES
USER

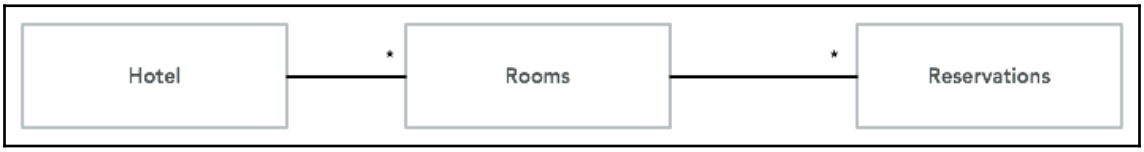

Webapp
API Gateway
Routing
Authentication
Throttling
Aggregation
Autherization
Service B LB VIP
Service A LB VIP
Service B
Service B
Service B
Service A
Service A
Service A

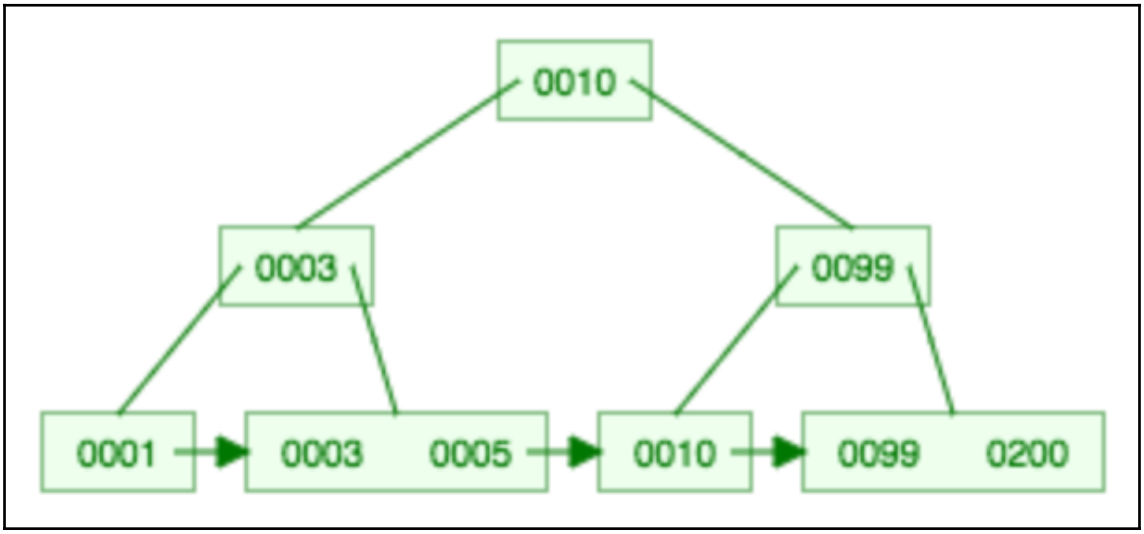

Webapp
Web BFF
Core Library - Throttling, Routing, Aggregation, Authorisation, Authentication
Mobile BFF
Core Library - Throttling, Routing, Aggregation, Authorisation, Authentication
Service B LB VIP
Service A LB VIP
Service B
Service B
Service B
Service A
Service A
Service A


Webapp
API Gateway
Routing
Authentication
Throttling
Aggregation
Autherization
Hotel Search
Cache
Wallet Service
SellerA Interface Service
SellerB Interface Service
Discount Engine


Transport
Operational metrics
Balancing and limiting
Business analytics
Application logging
Service metrics
Business logic

Chapter 8: Modeling Data





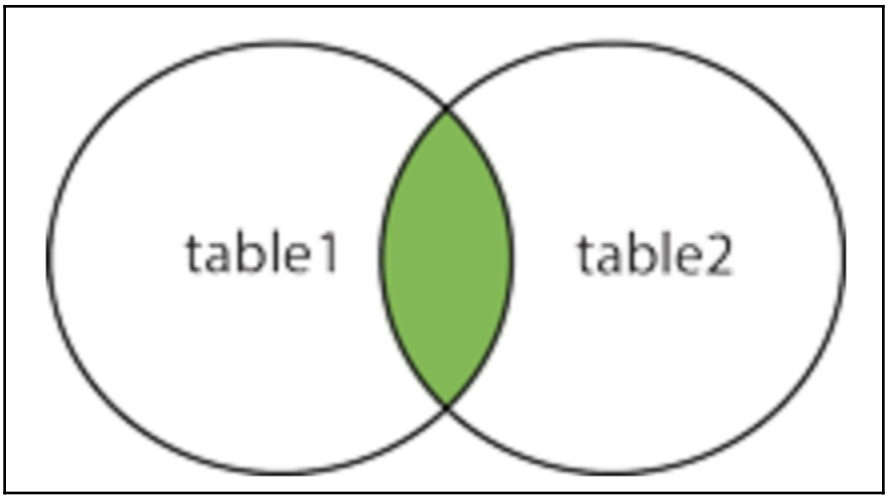


table1
table2

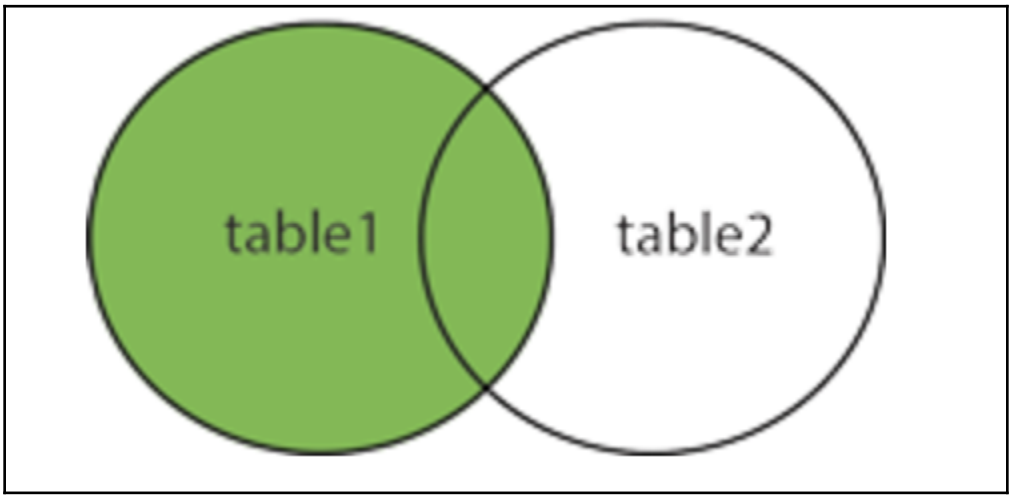


table1
table2

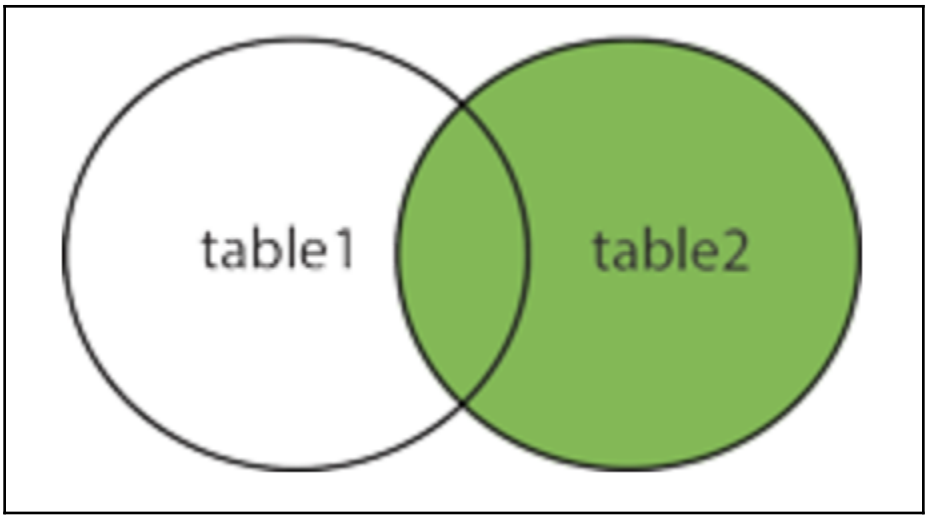


table1
table2

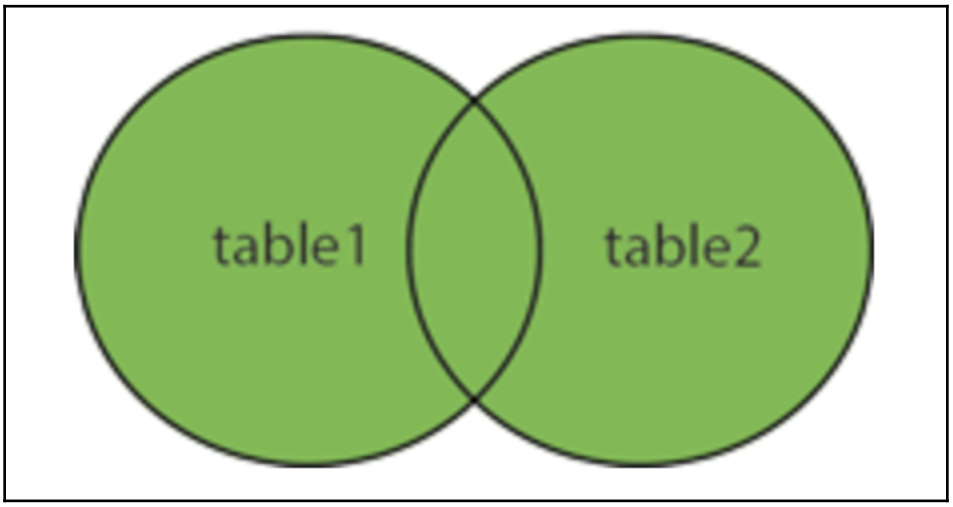
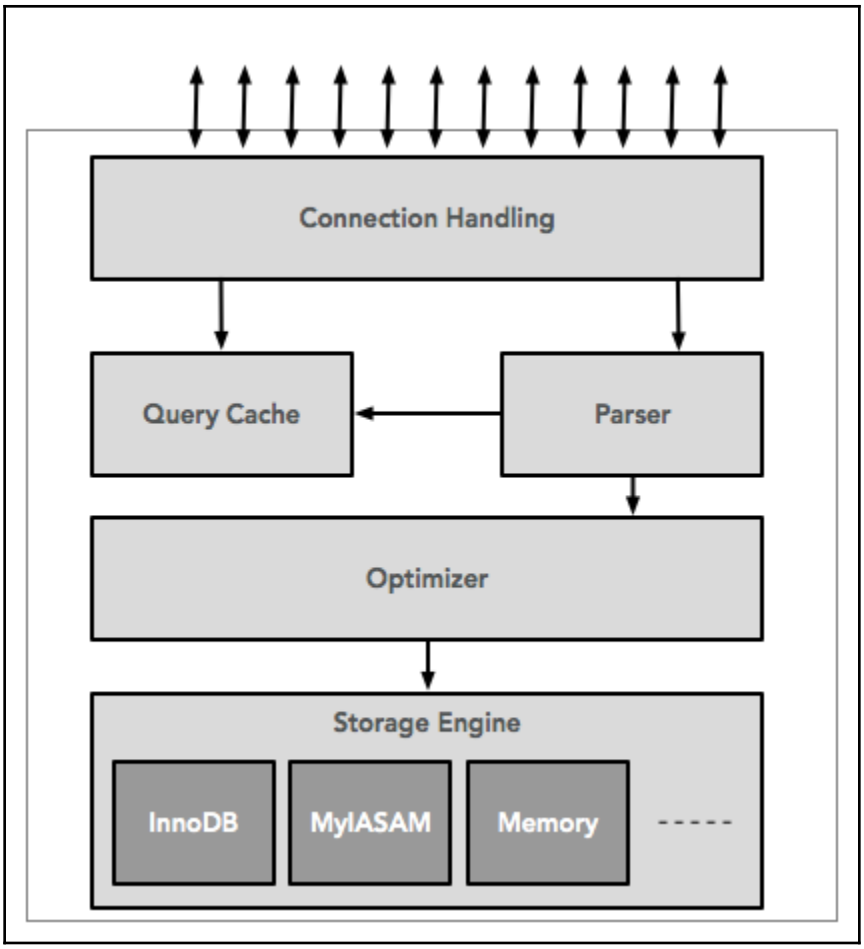
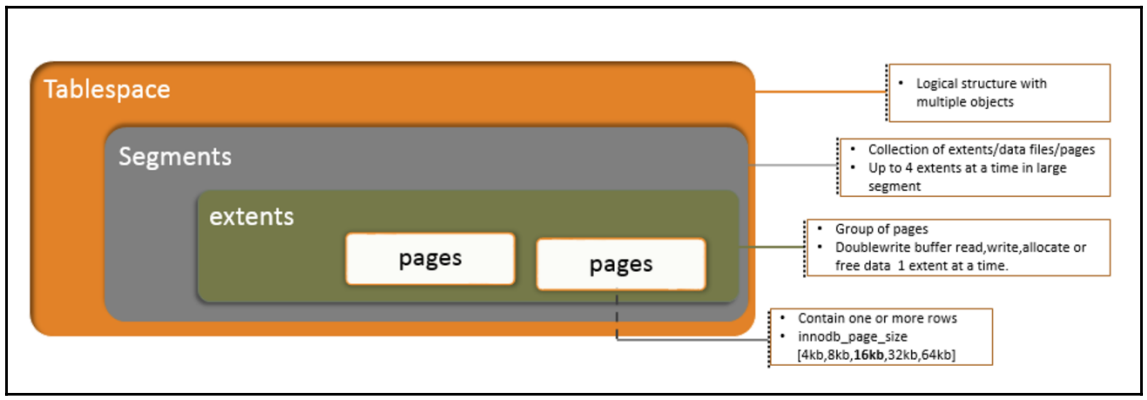
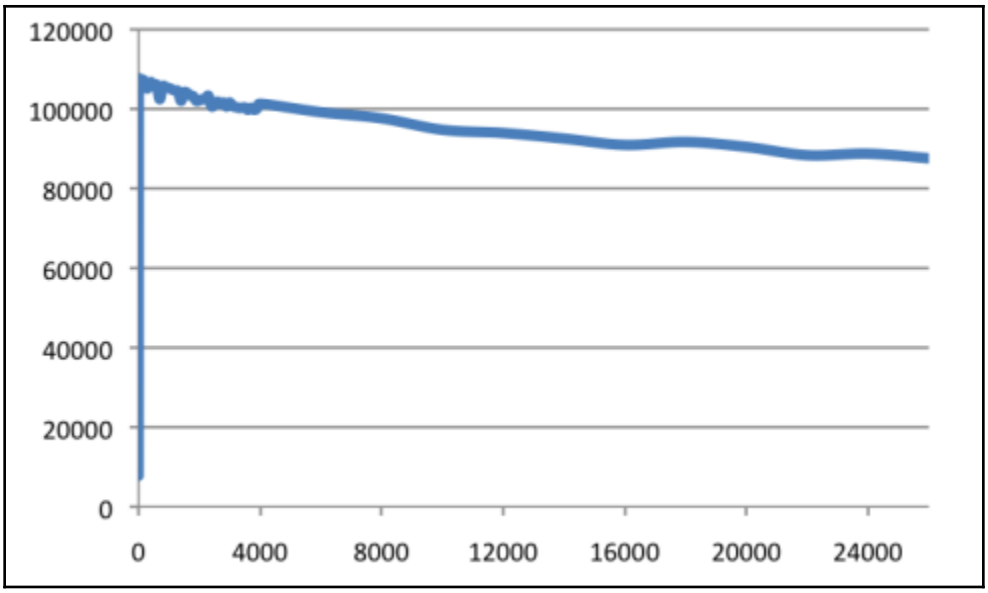
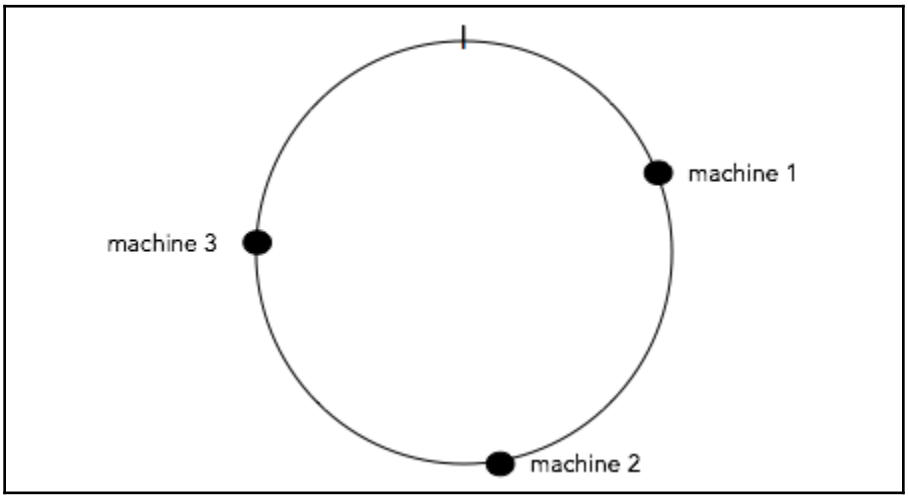


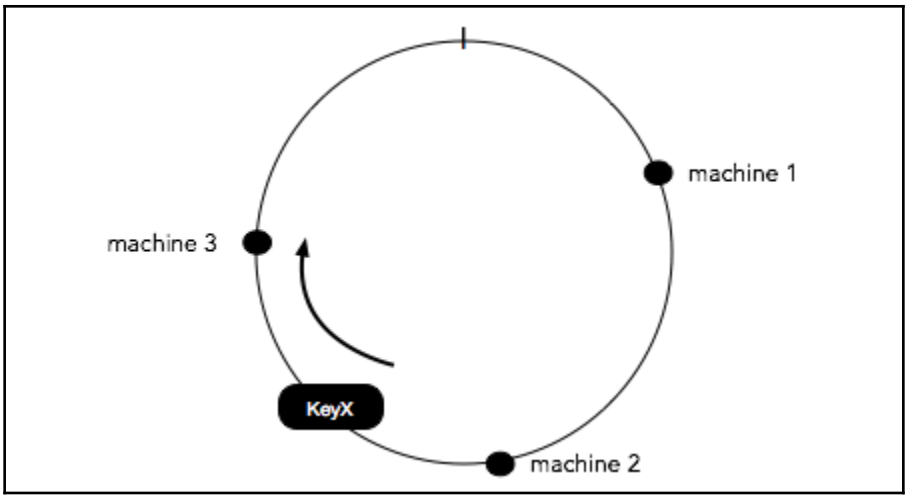
table1
table2

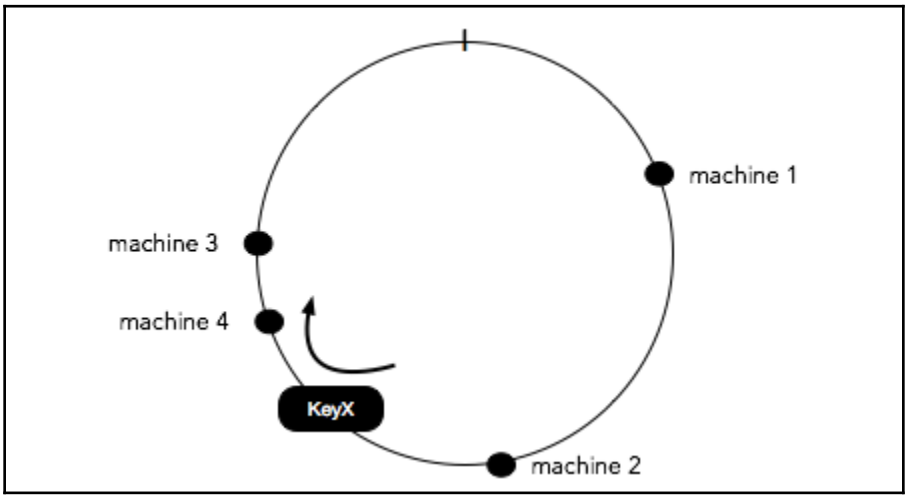

Connection Handling
Query Cache
Parser
Optimizer
Storage Engine
InnoDB
MyIASAM
Memory
- - - - -

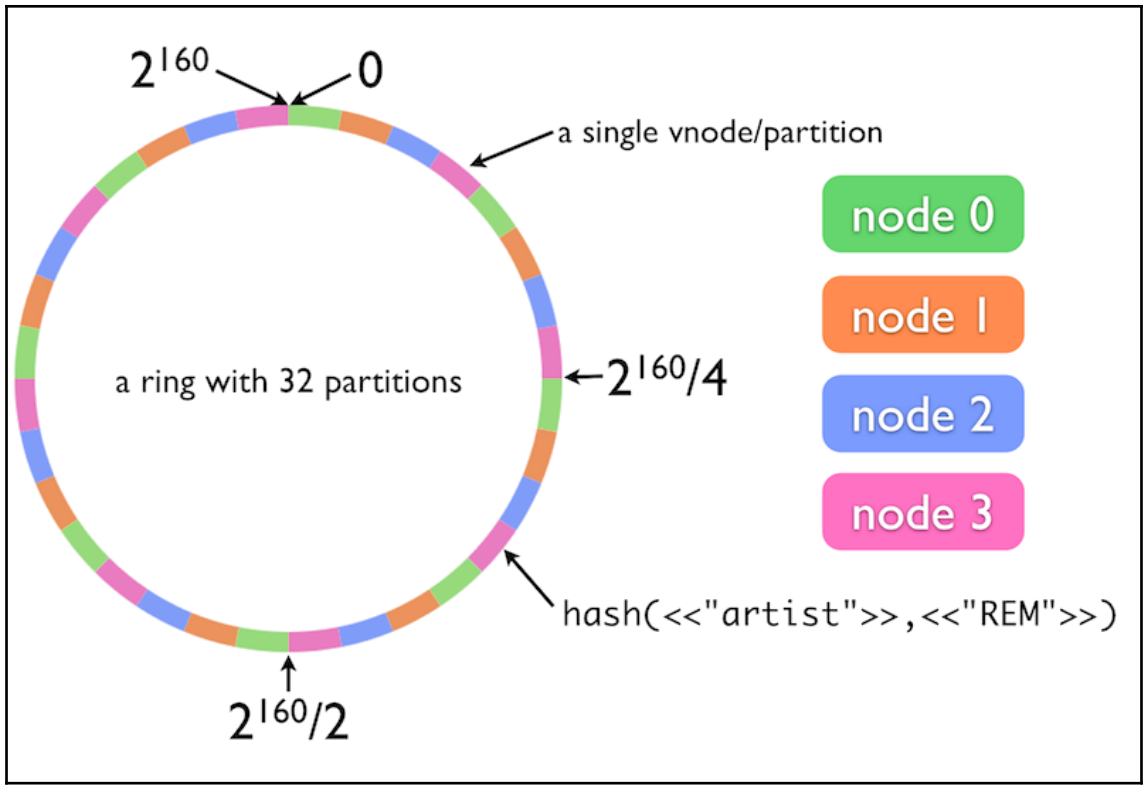

Tablespace
Segments
extents
pages
pages
• Logical structure with multiple objects
• Collection of extents/data files/pages
• Up to 4 extents at a time in large segment
• Group of pages
• Doublewrite buffer read,write,allocate or free data 1 extent at a time.
• Contain one or more rows
• innodb_page_size [4kb,8kb,16kb,32kb,64kb]

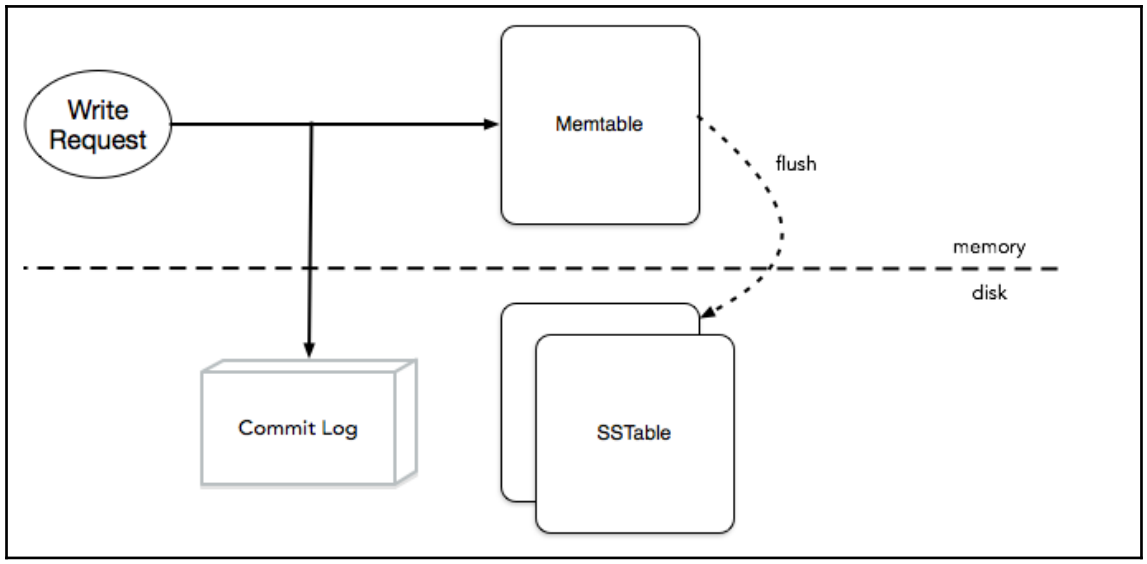

120000
100000
80000
60000
40000
20000
0
0
4000
8000
12000
16000
20000
24000

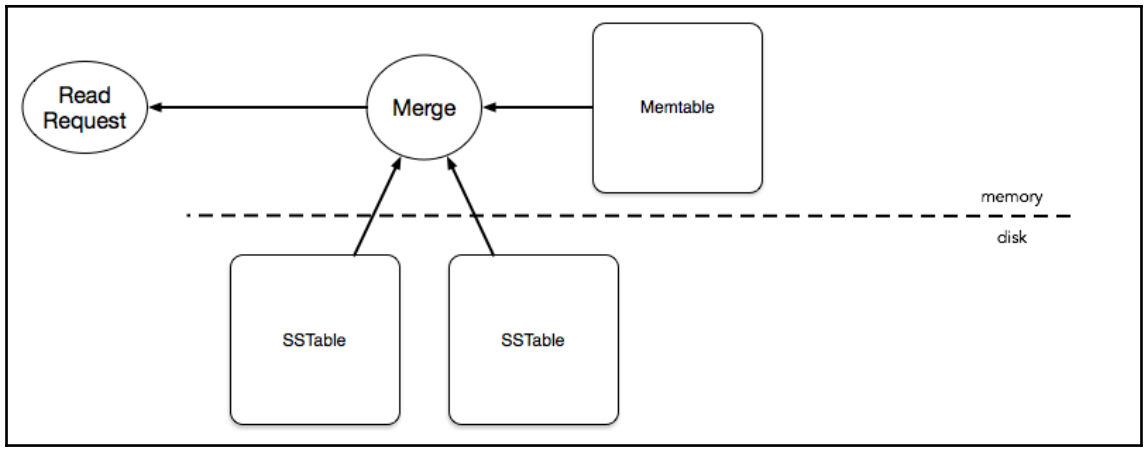

machine 1
machine 3
machine 2


machine 1
machine 3
KeyX
machine 2

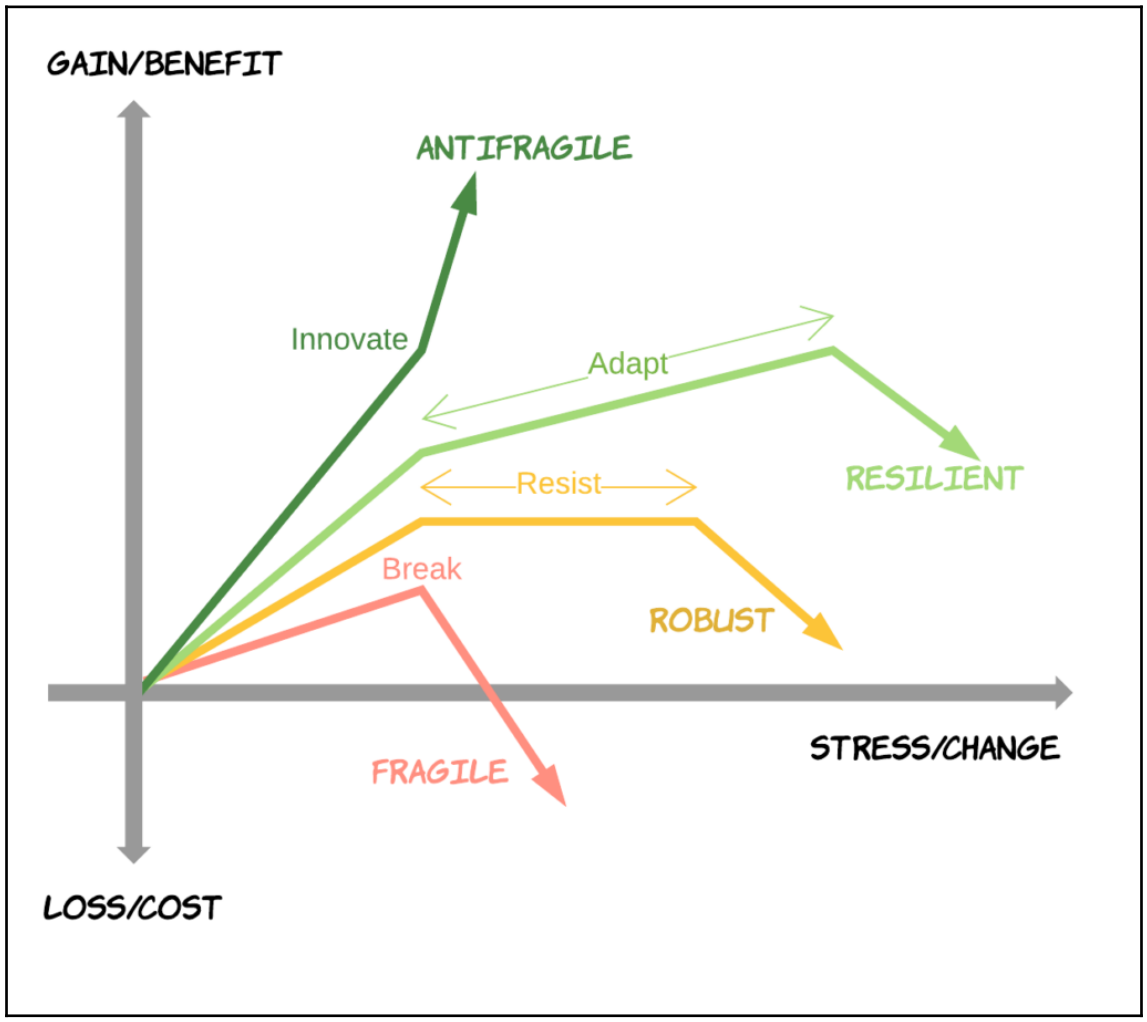

machine 1
machine 3
machine 4
KeyX
machine 2

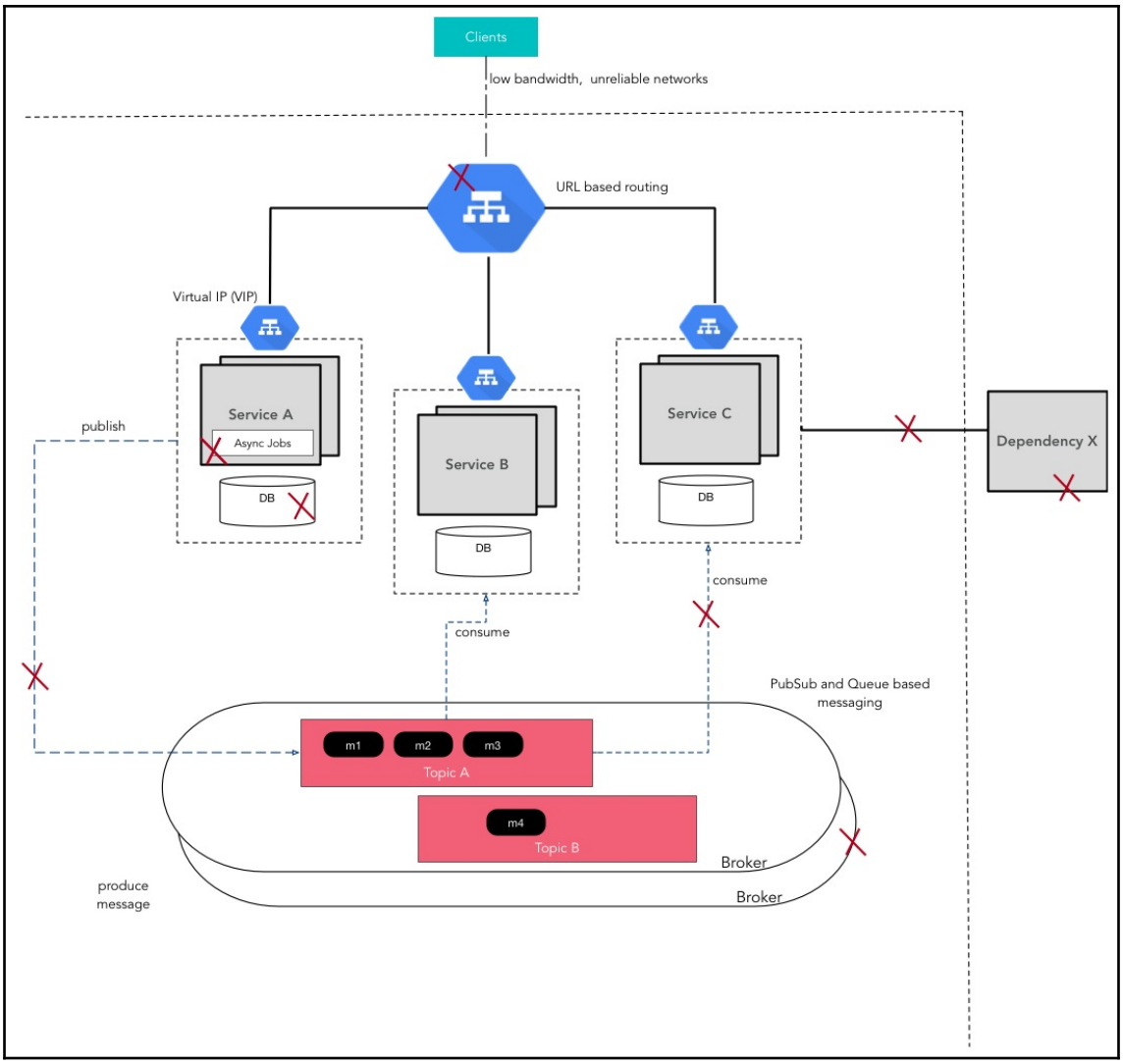

2^160
0
a single vnode/partition
node 0
node 1
node 2
node 3
a ring with 32 partitions
2^160/4
hash(<<"artist">>,<<"REM">>)
2^160/2

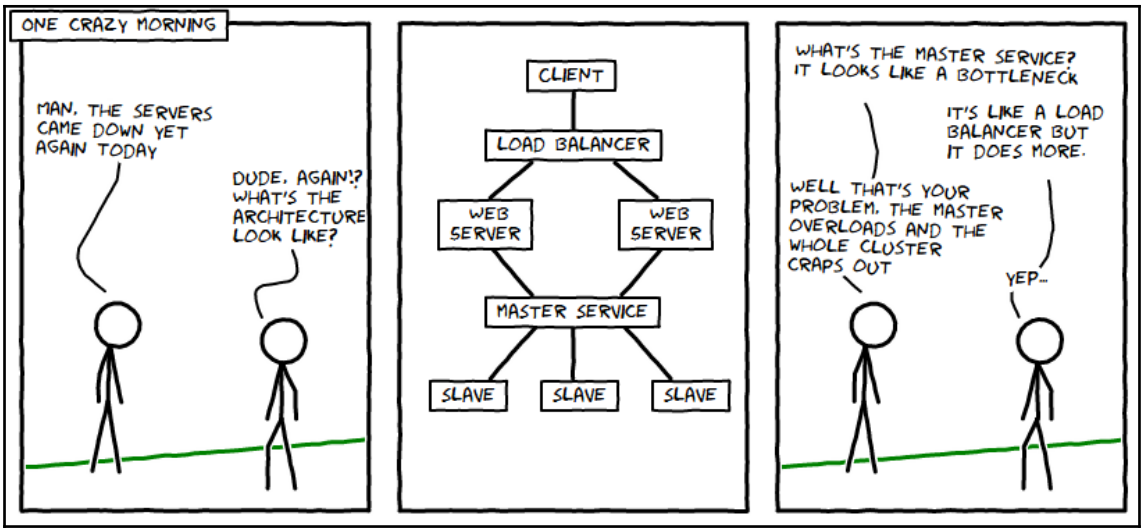

Write Request
Memtable
flush
memory
disk
Commit Log
SSTable

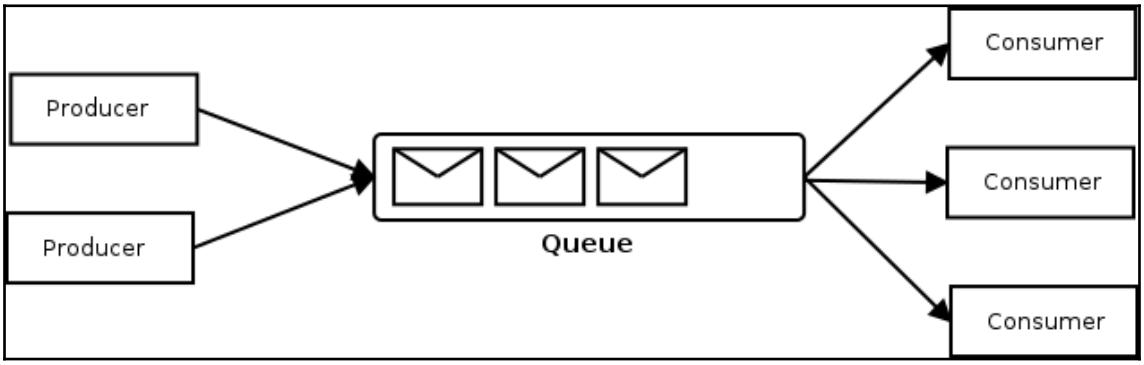

Read
Request
Merge
Memtable
memory
disk
SSTable
SSTable

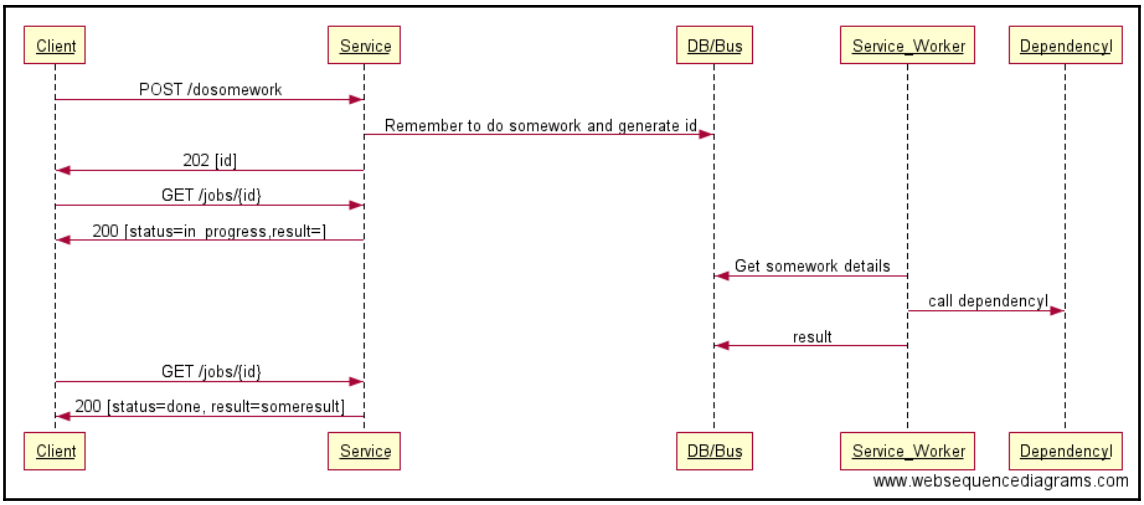
Chapter 9: Anti-Fragile Systems

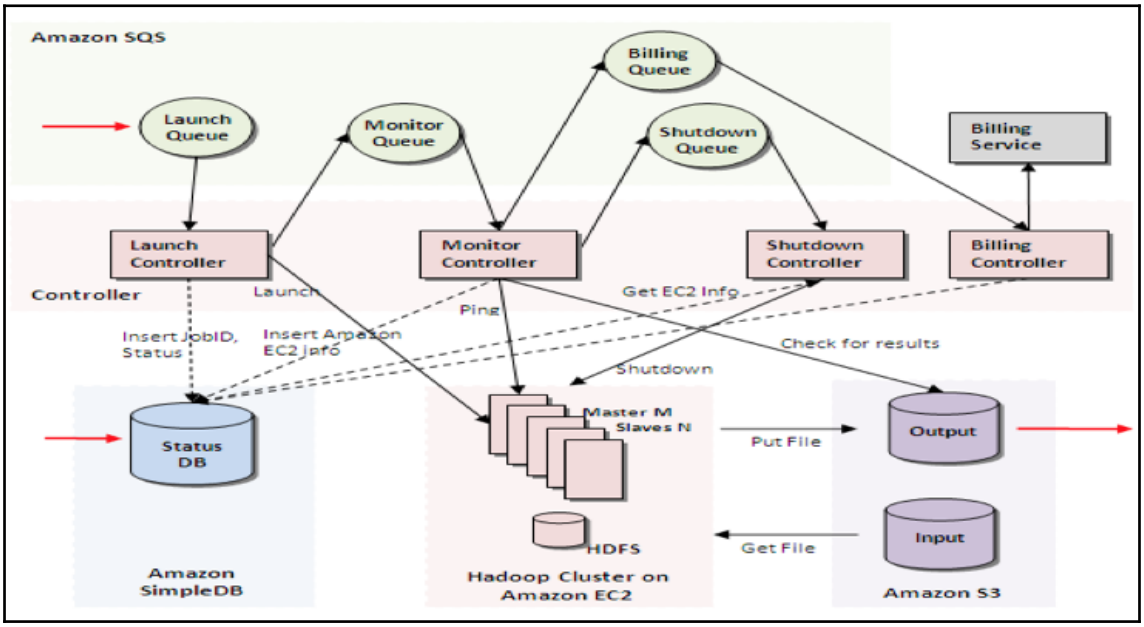


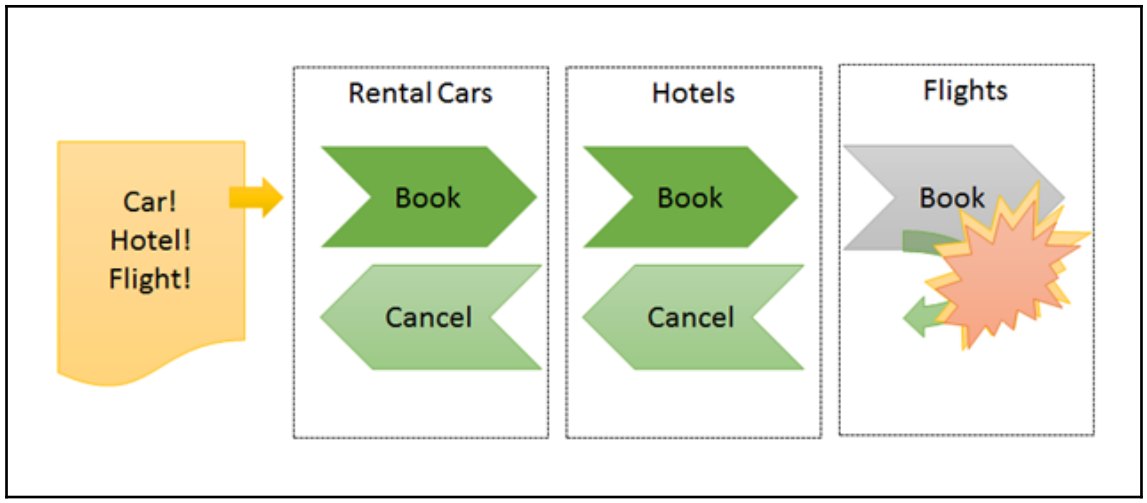

Clients
low bandwidth, unreliable networks
URL based routing
Virtual IP (VIP)
Service A
Async Jobs
DB
Service B
DB
Service C
DB
Dependency X
publish
consume
consume
PubSub and Queue based messaging
m1
m2
m3
Topic A
m4
Topic B
Broker
Broker
produce message

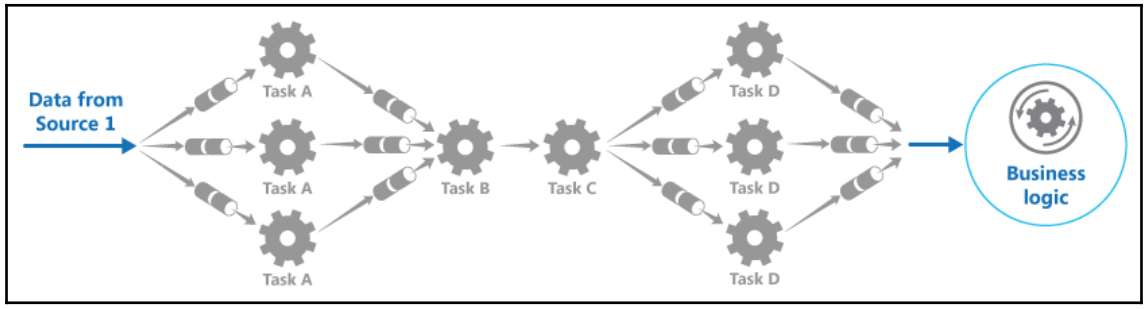

ONE CRAZY MORNING
MAN. THE SERVERS CAME DOWN YET AGAIN TODAY
DUDE. AGAIN!? WHAT'S THE ARCHITECTURE LOOK LIKE?
CLIENT
LOAD BALANCER
WEB SERVER
WEB SERVER
MASTER SERVICE
SLAVE
SLAVE
SLAVE
WHAT'S THE MASTER SERVICE? IT LOOKS LIKE A BOTTLENECK
IT'S LIKE A LOAD BALANCER BUT IT DOES MORE.
WELL THAT'S YOUR PROBLEM. THE MASTER OVERLOADS AND THE WHOLE CLUSTER CRAPS OUT
YEP...

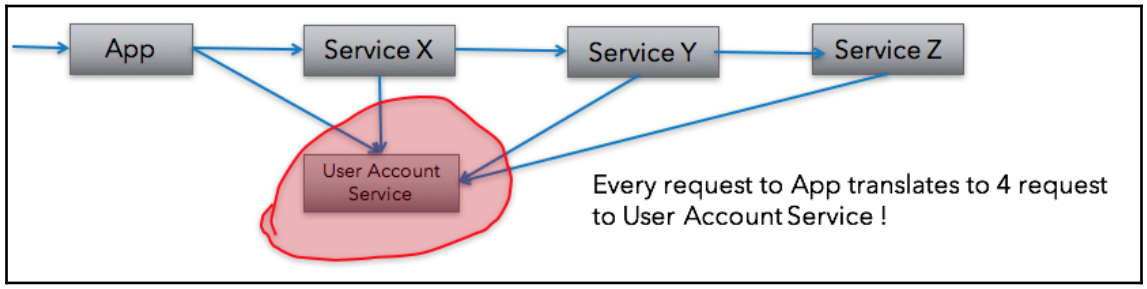

Producer
Producer
Queue
Consumer
Consumer
Consumer

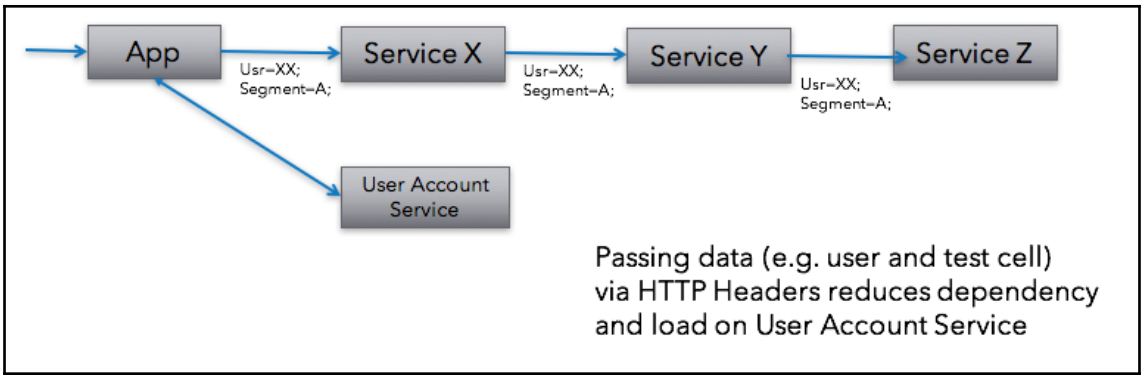

Client
Service
DB/Bus
Service_Worker
Dependency1
POST /dosomework
Remember to do somework and generate id
202 [id]
GET /jobs/{id}
200 [status=in_progress,result=]
Get somework details
call dependency1
result
GET /jobs/{id}
200 [status=done, result=someresult]
Client
Service
DB/Bus
Service_Worker
Dependency1
www.websequencediagrams.com

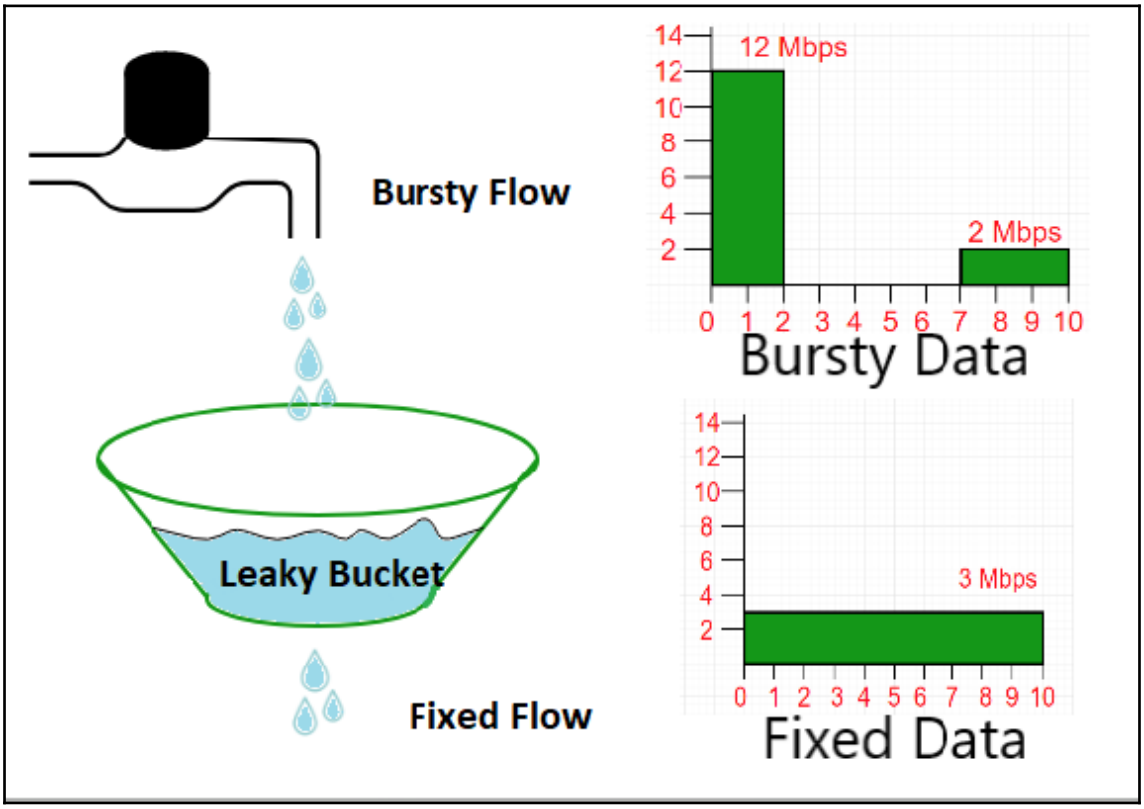

Amazon SQS
Launch Queue
Monitor Queue
Billing Queue
Shutdown Queue
Billing Service
Launch Controller
Monitor Controller
Shutdown Controller
Billing Controller
Controller
Launch
Ping
Get EC2 Info
Insert JobID, Status
Insert Amazon EC2 Info
Check for results
Shutdown
Status DB
Master M
Slaves N
HDFS
Put File
Get File
Output
Input
Amazon SimpleDB
Hadoop Cluster on Amazon EC2
Amazon S3

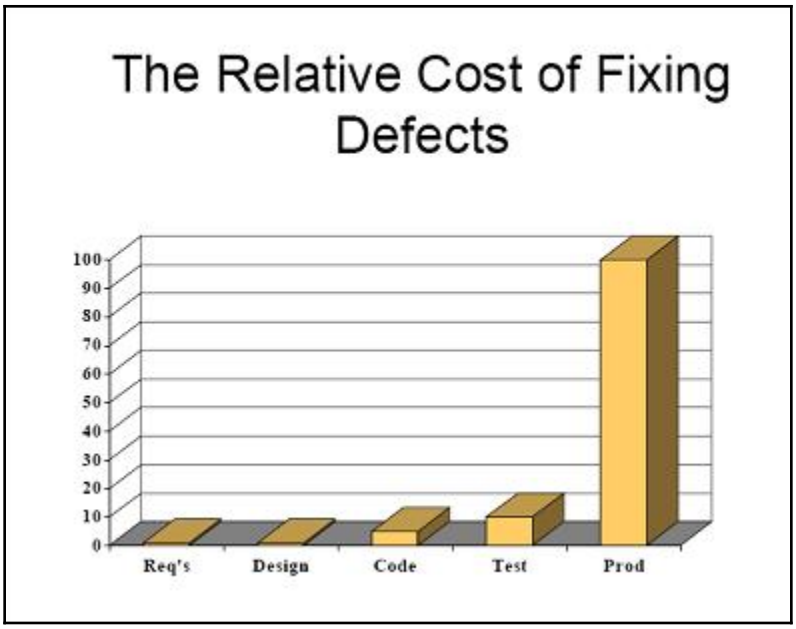

Car!
Hotel!
Flight!
Rental Cars
Book
Cancel
Hotels
Book
Cancel
Flights
Book

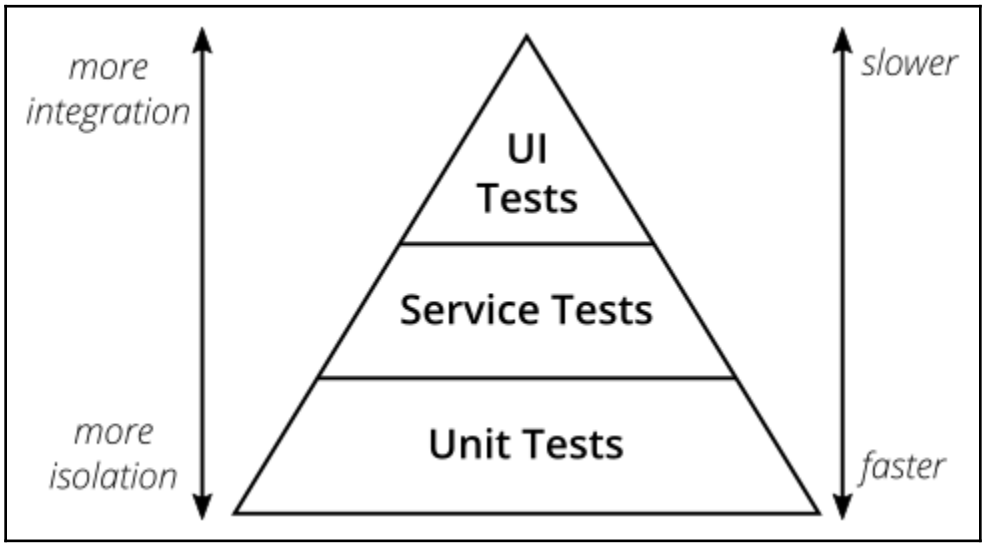

Data from
Source 1
Task A
Task A
Task A
Task B
Task C
Task D
Task D
Task D
Business
logic

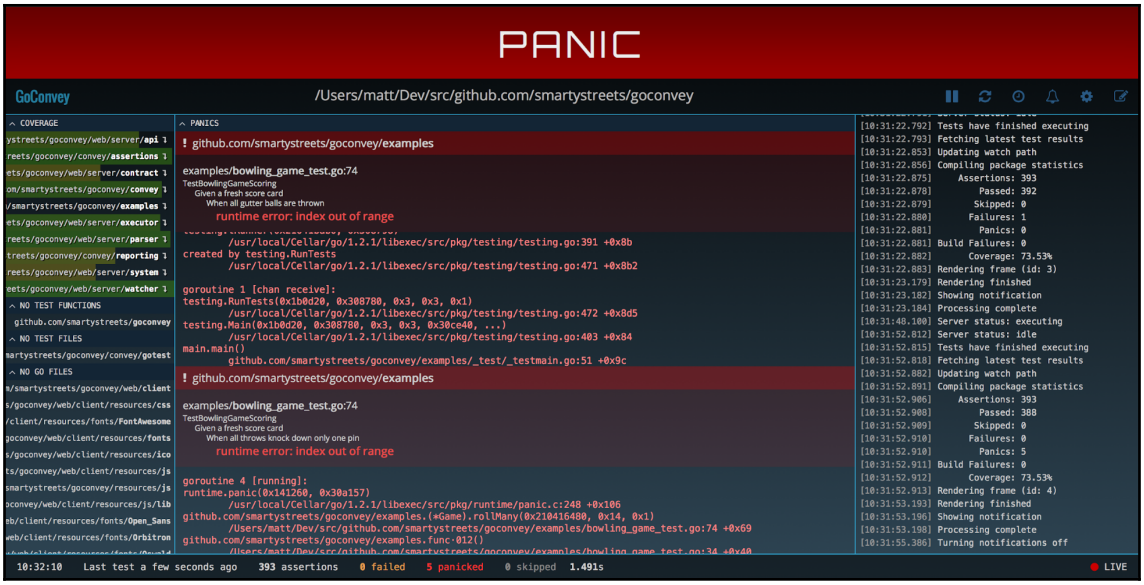

App
Service X
Service Y
Service Z
User Account
Service
Every request to App translates to 4 request
to User Account Service !

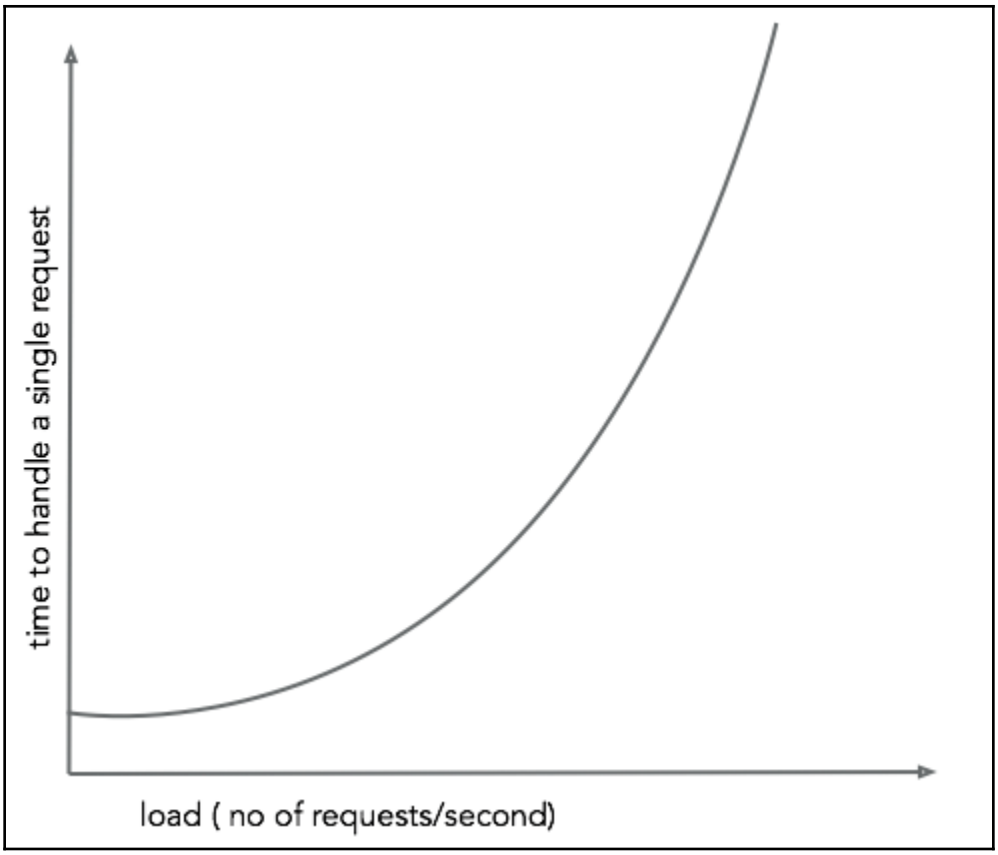

App
Usr=XX;
Segment=A;
Service X
Usr=XX;
Segment=A;
Service Y
Usr=XX;
Segment=A;
Service Z
User Account
Service
Passing data (e.g. user and test cell)
via HTTP Headers reduces dependency
and load on User Account Service

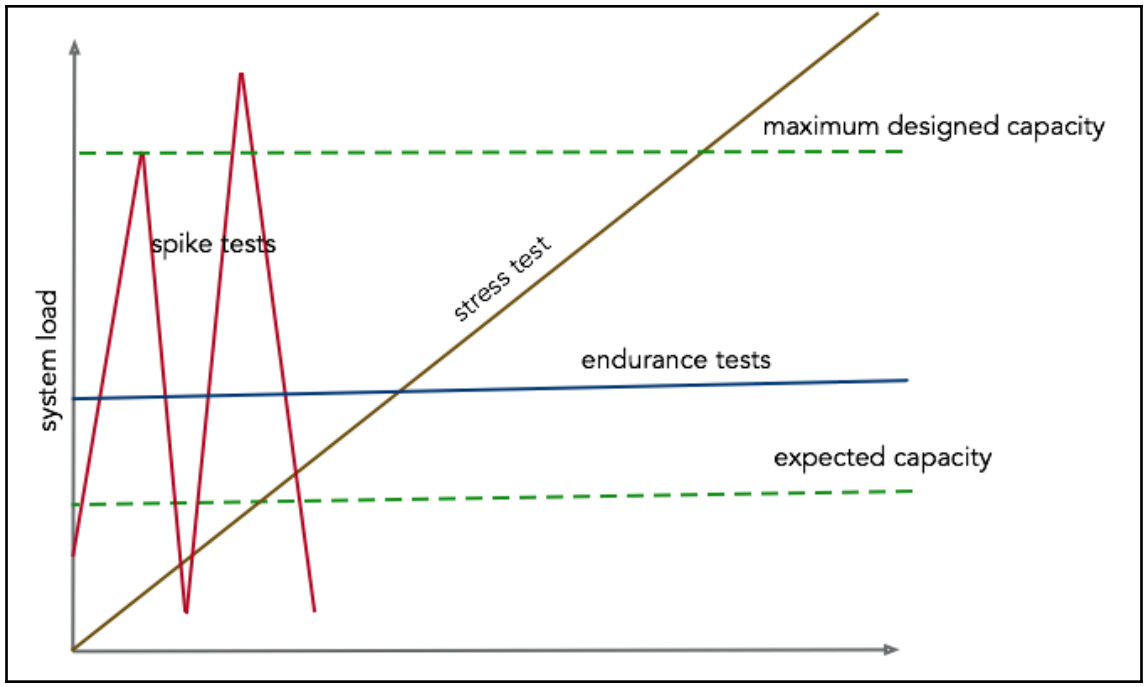

Bursty Flow
Leaky Bucket
Fixed Flow
12 Mbps
2 Mbps
Bursty Data
3 Mbps
Fixed Data

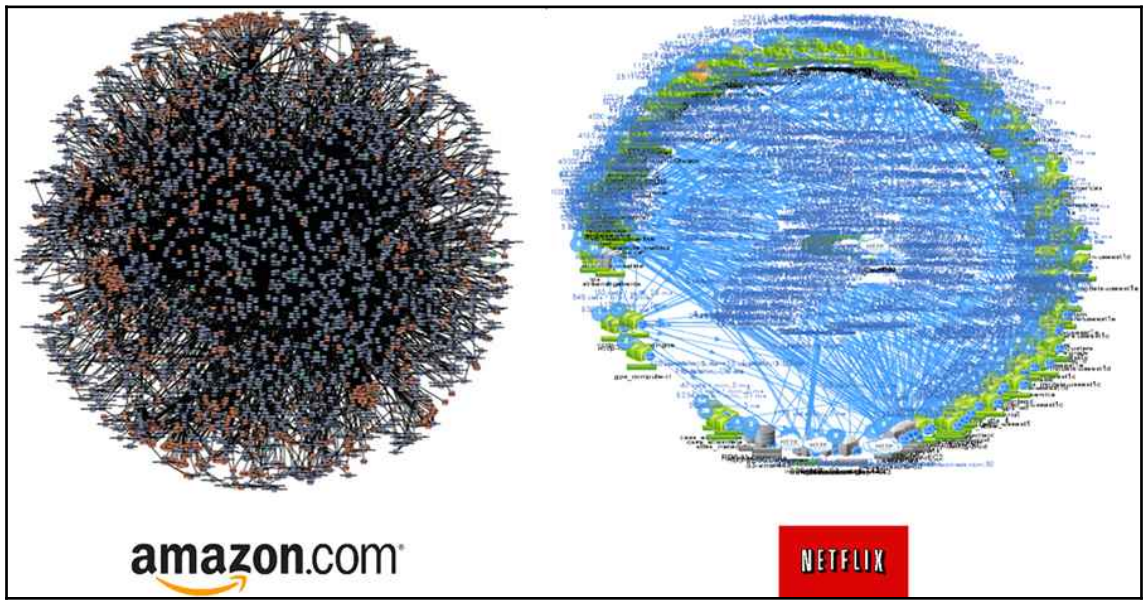

The Relative Cost of Fixing Defects
100
90
80
70
60
50
40
30
20
10
0
Req's
Design
Code
Test
Prod

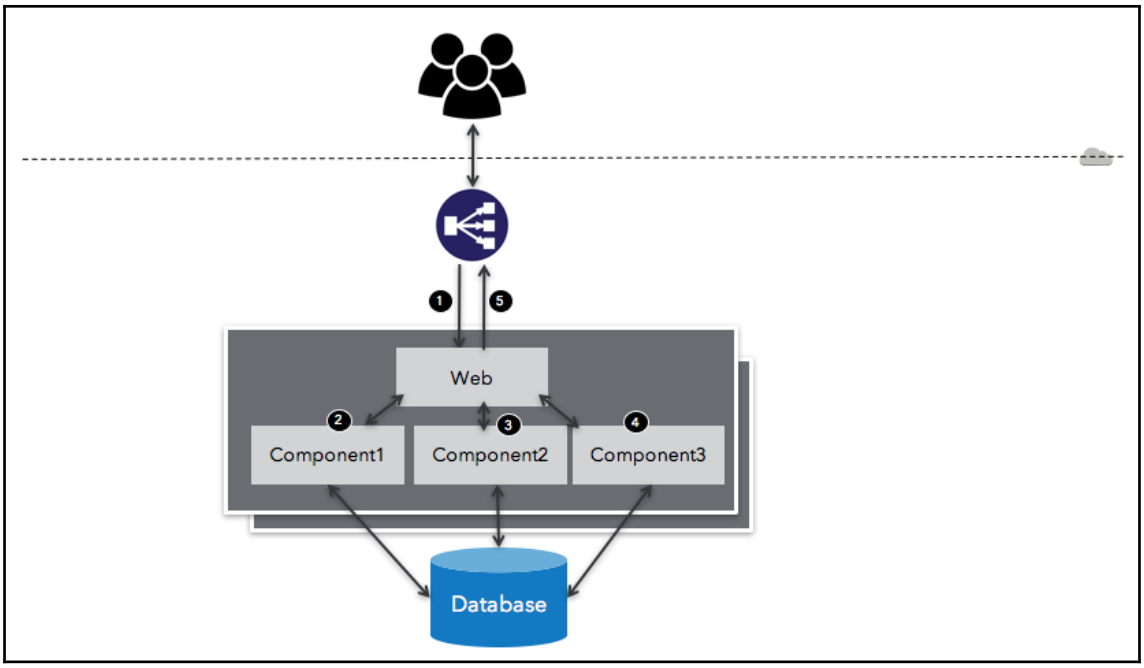

more integration
more isolation
UI Tests
Service Tests
Unit Tests
slower
faster

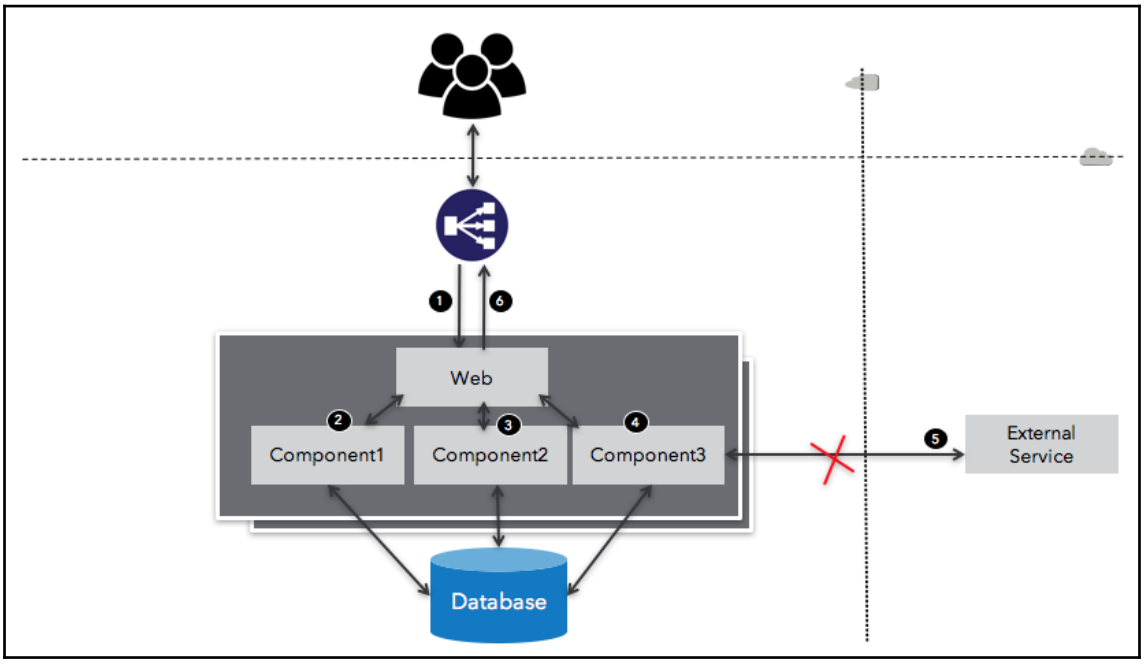

PANIC
GoConvey
/Users/matt/Dev/src/github.com/smartystreets/goconvey
COVERAGE
NO TEST FUNCTIONS
NO TEST FILES
NO GO FILES
PANICS
! github.com/smartystreets/goconvey/examples
examples/bowling_game_test.go:74
TestBowlingGameScoring
Given a fresh score card
When all gutter balls are thrown
runtime error: index out of range
/usr/local/Cellar/go/1.2.1/libexec/src/pkg/testing/testing.go:391 +0x8b
created by testing.RunTests
/usr/local/Cellar/go/1.2.1/libexec/src/pkg/testing/testing.go:471 +0x8b2
goroutine 1 [chan receive]:
testing.RunTests(0x1b0d20, 0x308780, 0x3, 0x3, 0x1)
/usr/local/Cellar/go/1.2.1/libexec/src/pkg/testing/testing.go:472 +0x8d5
testing.Main(0x1b0d20, 0x308780, 0x3, 0x3, 0x30ce40, ...)
/usr/local/Cellar/go/1.2.1/libexec/src/pkg/testing/testing.go:403 +0x84
main.main()
github.com/smartystreets/goconvey/examples/_test/_testmain.go:51 +0x9c
! github.com/smartystreets/goconvey/examples
examples/bowling_game_test.go:74
TestBowlingGameScoring
Given a fresh score card
When all throws knock down only one pin
runtime error: index out of range
goroutine 4 [running]:
runtime.panic(0x141260, 0x30a157)
/usr/local/Cellar/go/1.2.1/libexec/src/pkg/runtime/panic.c:248 +0x106
github.com/smartystreets/goconvey/examples.(*Game).rollMany(0x210416480, 0x14, 0x1)
/Users/matt/Dev/src/github.com/smartystreets/goconvey/examples/bowling_game_test.go:74 +0x69
github.com/smartystreets/goconvey/examples.func·012()
[10:31:22.792] Tests have finished executing
[10:31:22.793] Fetching latest test results
[10:31:22.853] Updating watch path
[10:31:22.856] Compiling package statistics
[10:31:22.875] Assertions: 393
[10:31:22.878] Passed: 392
[10:31:22.879] Skipped: 0
[10:31:22.880] Failures: 1
[10:31:22.881] Panics: 0
[10:31:22.881] Build Failures: 0
[10:31:22.882] Coverage: 73.53%
[10:31:22.883] Rendering frame (id: 3)
[10:31:23.179] Rendering finished
[10:31:23.182] Showing notification
[10:31:23.184] Processing complete
[10:31:48.100] Server status: executing
[10:31:52.812] Server status: idle
[10:31:52.815] Tests have finished executing
[10:31:52.818] Fetching latest test results
[10:31:52.882] Updating watch path
[10:31:52.891] Compiling package statistics
[10:31:52.906] Assertions: 393
[10:31:52.908] Passed: 388
[10:31:52.909] Skipped: 0
[10:31:52.910] Failures: 0
[10:31:52.910] Panics: 5
[10:31:52.911] Build Failures: 0
[10:31:52.912] Coverage: 73.53%
[10:31:52.913] Rendering frame (id: 4)
[10:31:53.193] Rendering finished
[10:31:53.196] Showing notification
[10:31:53.198] Processing complete
[10:31:55.386] Turning notifications off
10:32:10 Last test a few seconds ago 393 assertions 0 failed 5 panicked 0 skipped 1.491s
LIVE

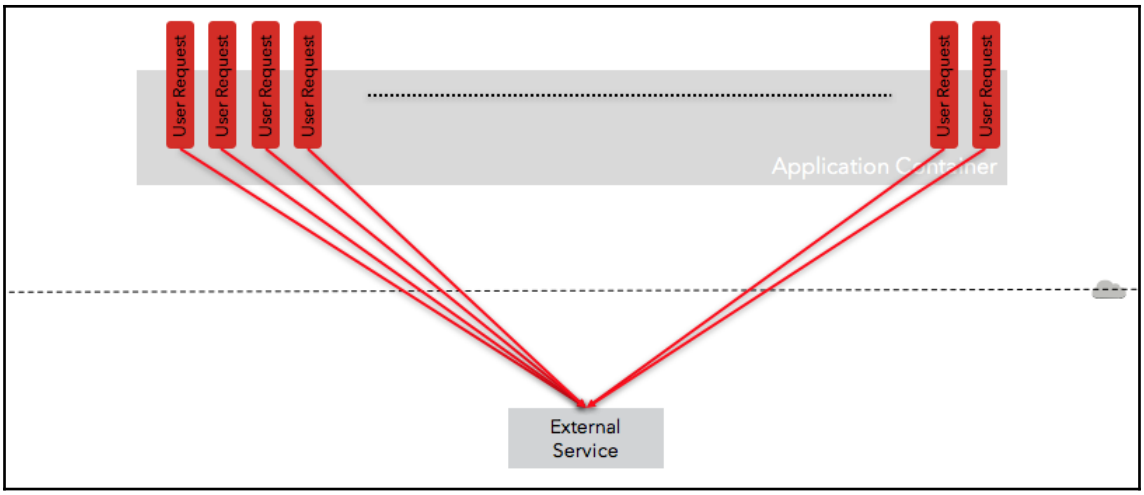

time to handle a single request
load (no of requests/second)

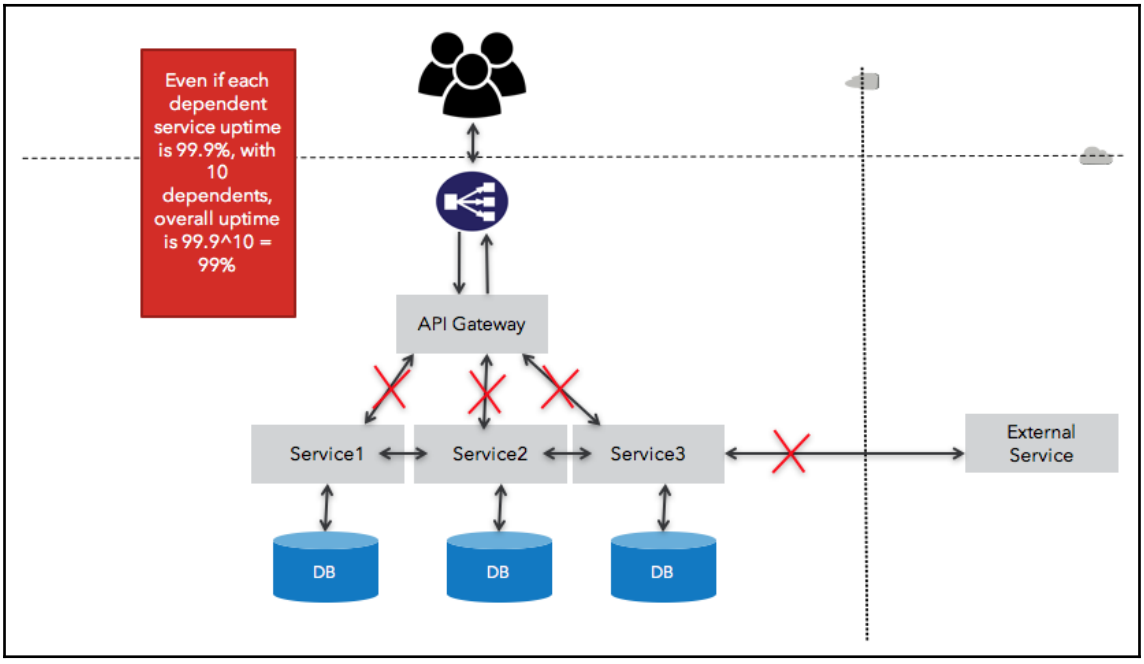

maximum designed capacity
spike tests
stress test
system load
endurance tests
expected capacity

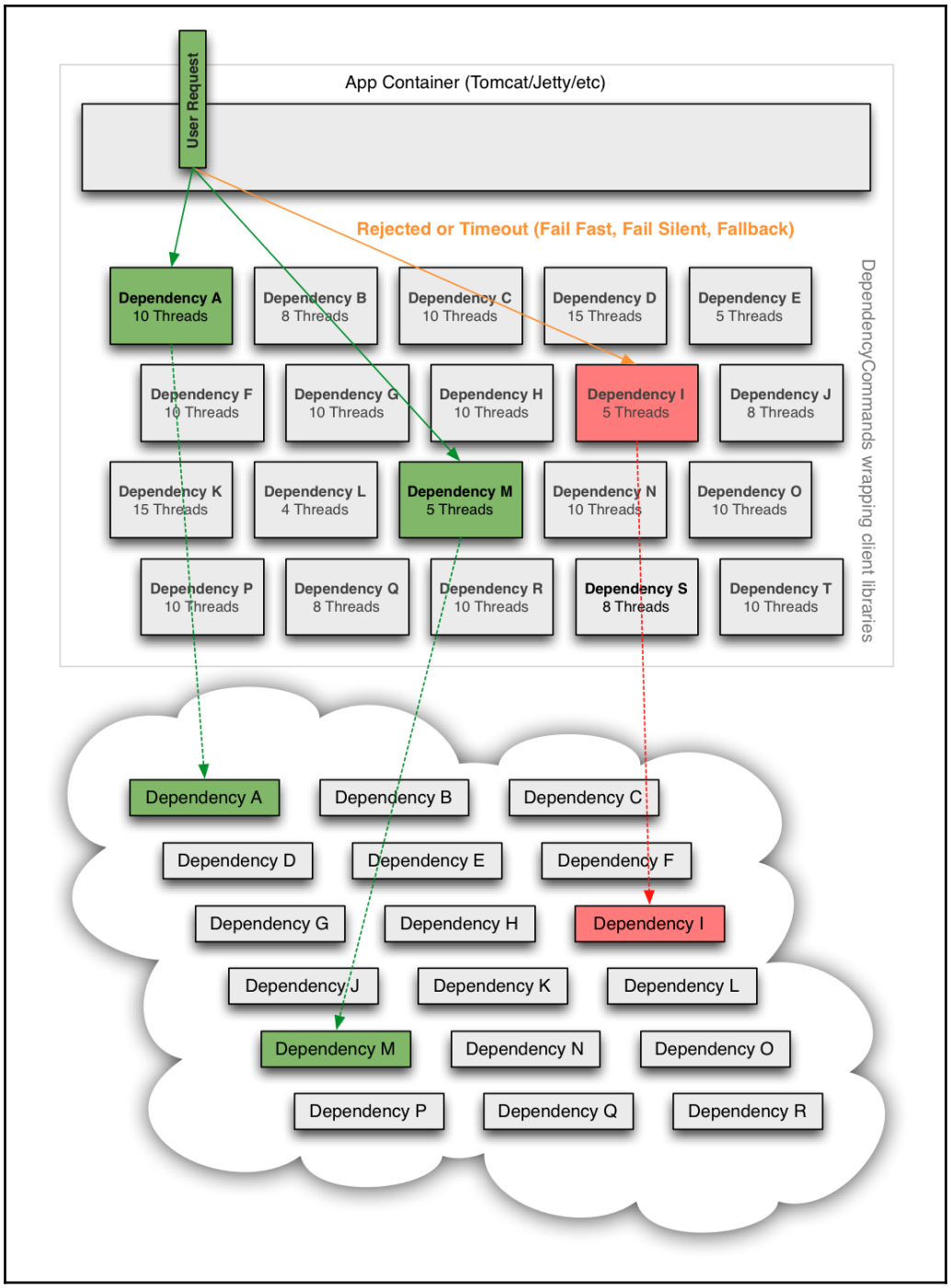

amazon.com
NETFLIX

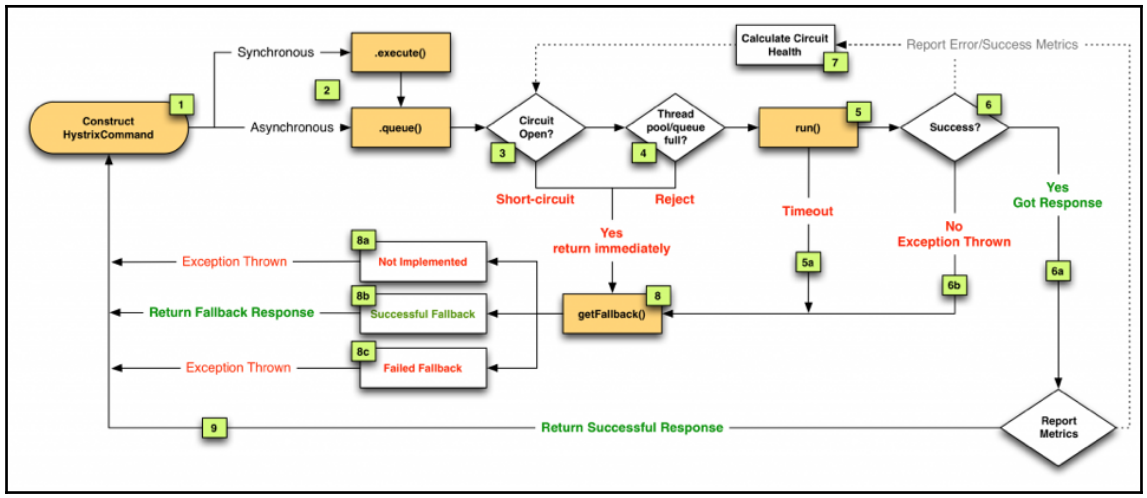

Web
Component1
Component2
Component3
Database

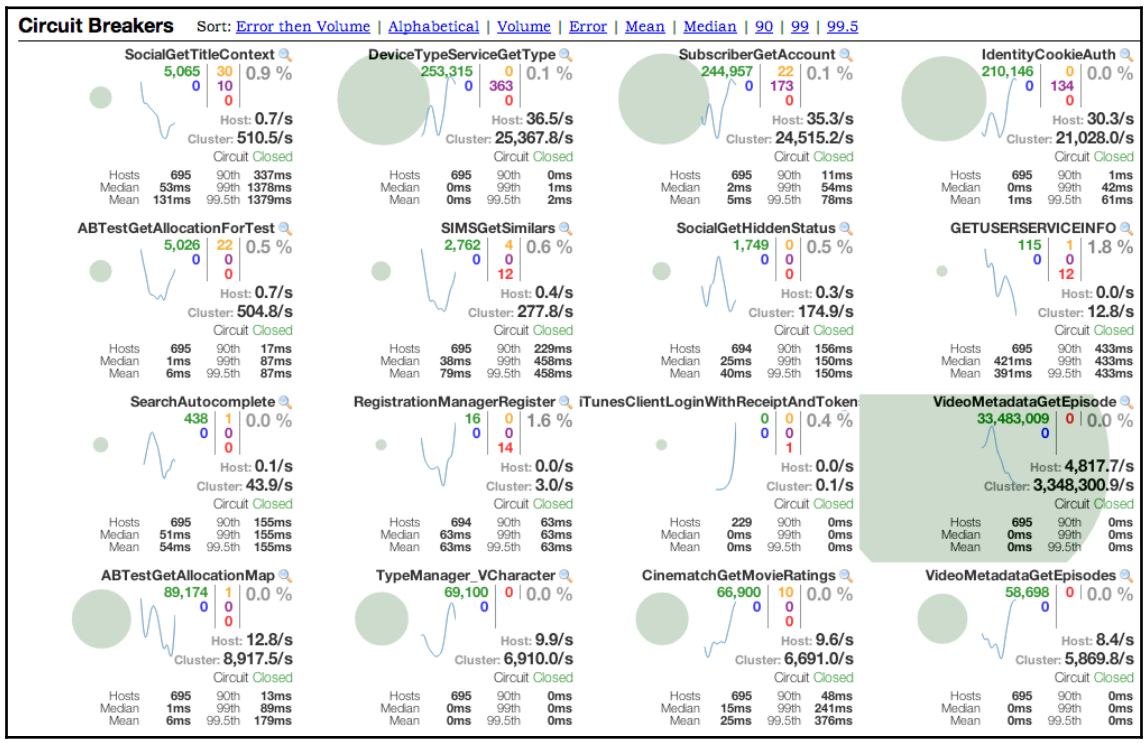

Web
Component1
Component2
Component3
External Service
Database

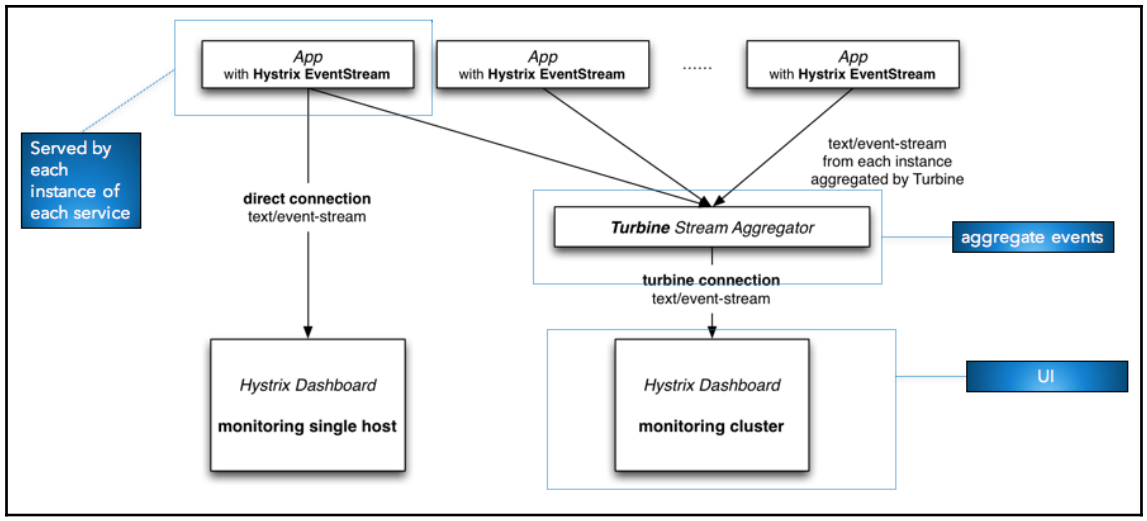

User Request
User Request
User Request
User Request
User Request
User Request
Application Container
External Service

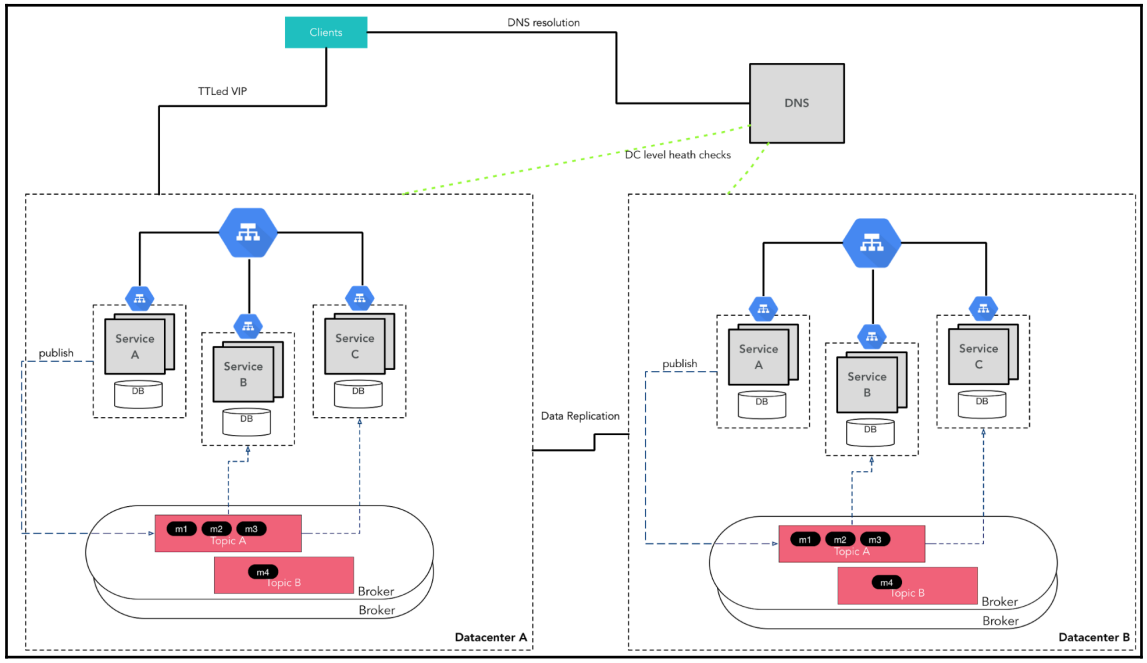

Even if each dependent service uptime is 99.9%, with 10 dependents, overall uptime is 99.9^10 = 99%
API Gateway
Service1
Service2
Service3
External Service
DB
DB
DB


User Request
App Container (Tomcat/Jetty/etc)
Rejected or Timeout (Fail Fast, Fail Silent, Fallback)
Dependency A
10 Threads
Dependency B
8 Threads
Dependency C
10 Threads
Dependency D
15 Threads
Dependency E
5 Threads
Dependency F
10 Threads
Dependency G
10 Threads
Dependency H
10 Threads
Dependency I
5 Threads
Dependency J
8 Threads
Dependency K
15 Threads
Dependency L
4 Threads
Dependency M
5 Threads
Dependency N
10 Threads
Dependency O
10 Threads
Dependency P
10 Threads
Dependency Q
8 Threads
Dependency R
10 Threads
Dependency S
8 Threads
Dependency T
10 Threads
DependencyCommands wrapping client libraries
Dependency A
Dependency B
Dependency C
Dependency D
Dependency E
Dependency F
Dependency G
Dependency H
Dependency I
Dependency J
Dependency K
Dependency L
Dependency M
Dependency N
Dependency O
Dependency P
Dependency Q
Dependency R

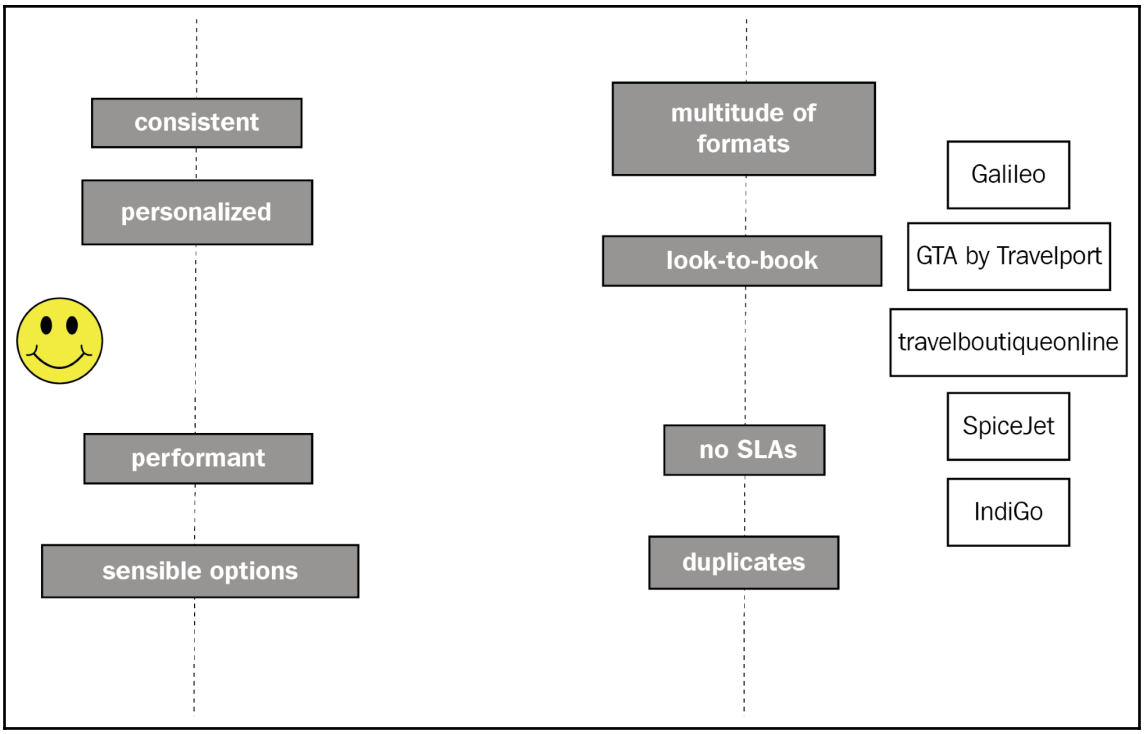

Synchronous
.execute()
Calculate Circuit Health
Report Error/Success Metrics
Construct HystrixCommand
Asynchronous
.queue()
Circuit Open?
Thread pool/queue full?
run()
Success?
Short-circuit
Reject
Timeout
Yes
Got Response
Yes
return immediately
No
Exception Thrown
Exception Thrown
Not Implemented
Return Fallback Response
Successful Fallback
getFallback()
Exception Thrown
Failed Fallback
Report Metrics
Return Successful Response

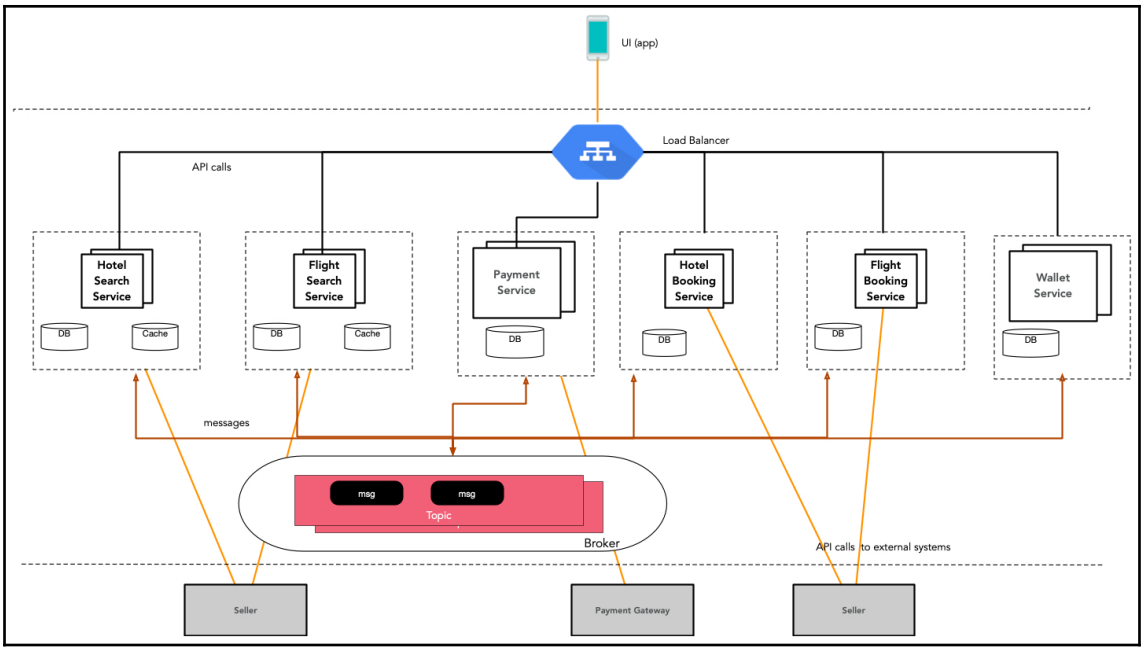

Circuit Breakers
Sort: Error then Volume | Alphabetical | Volume | Error | Mean | Median | 90 | 99 | 99.5
SocialGetTitleContext
5,065 30 0.9 %
0 10
0
Host: 0.7/s
Cluster: 510.5/s
Circuit Closed
Hosts 695 90th 337ms
Median 53ms 99th 1378ms
Mean 131ms 99.5th 1379ms
DeviceTypeServiceGetType
253,315 0 0.1 %
0 363
0
Host: 36.5/s
Cluster: 25,367.8/s
Circuit Closed
Hosts 695 90th 0ms
Median 0ms 99th 1ms
Mean 0ms 99.5th 2ms
SubscriberGetAccount
244,957 22 0.1 %
0 173
0
Host: 35.3/s
Cluster: 24,515.2/s
Circuit Closed
Hosts 695 90th 11ms
Median 2ms 99th 54ms
Mean 5ms 99.5th 78ms
IdentityCookieAuth
210,146 0 0.0 %
0 134
0
Host: 30.3/s
Cluster: 21,028.0/s
Circuit Closed
Hosts 695 90th 1ms
Median 0ms 99th 42ms
Mean 1ms 99.5th 61ms
ABTestGetAllocationForTest
5,026 22 0.5 %
0 0
0
Host: 0.7/s
Cluster: 504.8/s
Circuit Closed
Hosts 695 90th 17ms
Median 1ms 99th 87ms
Mean 6ms 99.5th 87ms
SIMSGetSimilars
2,762 4 0.6 %
0 0
12
Host: 0.4/s
Cluster: 277.8/s
Circuit Closed
Hosts 695 90th 229ms
Median 38ms 99th 458ms
Mean 79ms 99.5th 458ms
SocialGetHiddenStatus
1,749 0 0.5 %
0 0
0
Host: 0.3/s
Cluster: 174.9/s
Circuit Closed
Hosts 694 90th 156ms
Median 25ms 99th 150ms
Mean 40ms 99.5th 150ms
GETUSERSERVICEINFO
115 1 1.8 %
0 0
12
Host: 0.0/s
Cluster: 12.8/s
Circuit Closed
Hosts 695 90th 433ms
Median 421ms 99th 433ms
Mean 391ms 99.5th 433ms
SearchAutocomplete
438 1 0.0 %
0 0
0
Host: 0.1/s
Cluster: 43.9/s
Circuit Closed
Hosts 695 90th 155ms
Median 51ms 99th 155ms
Mean 54ms 99.5th 155ms
RegistrationManagerRegister
16 0 1.6 %
0 0
14
Host: 0.0/s
Cluster: 3.0/s
Circuit Closed
Hosts 694 90th 63ms
Median 63ms 99th 63ms
Mean 63ms 99.5th 63ms
iTunesClientLoginWithReceiptAndToken
0 0 0.4 %
0 0
1
Host: 0.0/s
Cluster: 0.1/s
Circuit Closed
Hosts 229 90th 0ms
Median 0ms 99th 0ms
Mean 0ms 99.5th 0ms
VideoMetadataGetEpisode
33,483,009 0 0.0 %
0
Host: 4,817.7/s
Cluster: 3,348,300.9/s
Circuit Closed
Hosts 695 90th 0ms
Median 0ms 99th 0ms
Mean 0ms 99.5th 0ms
ABTestGetAllocationMap
89,174 1 0.0 %
0 0
0
Host: 12.8/s
Cluster: 8,917.5/s
Circuit Closed
Hosts 695 90th 13ms
Median 1ms 99th 89ms
Mean 6ms 99.5th 179ms
TypeManager_VCharacter
69,100 0 0.0 %
0
Host: 9.9/s
Cluster: 6,910.0/s
Circuit Closed
Hosts 695 90th 0ms
Median 0ms 99th 0ms
Mean 0ms 99.5th 0ms
CinematchGetMovieRatings
66,900 10 0.0 %
0 0
0
Host: 9.6/s
Cluster: 6,691.0/s
Circuit Closed
Hosts 695 90th 48ms
Median 15ms 99th 241ms
Mean 25ms 99.5th 376ms
VideoMetadataGetEpisodes
58,698 0 0.0 %
0
Host: 8.4/s
Cluster: 5,869.8/s
Circuit Closed
Hosts 695 90th 0ms
Median 0ms 99th 0ms
Mean 0ms 99.5th 0ms

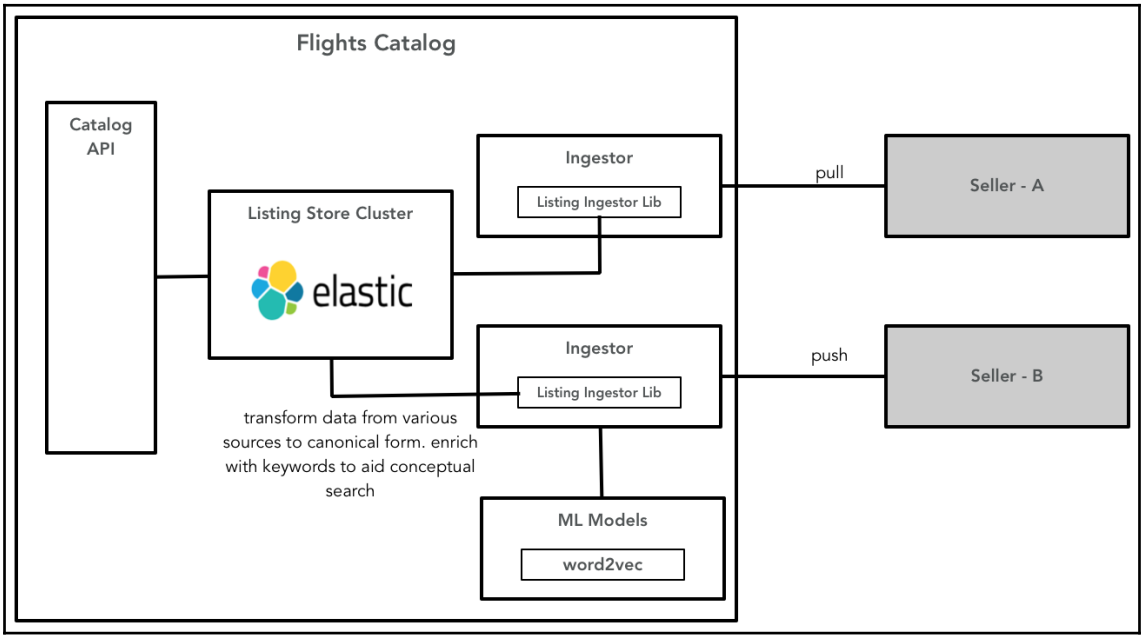

App
with Hystrix EventStream
App
with Hystrix EventStream
.....
App
with Hystrix EventStream
Served by each instance of each service
direct connection
text/event-stream
text/event-stream
from each instance
aggregated by Turbine
Turbine Stream Aggregator
aggregate events
turbine connection
text/event-stream
Hystrix Dashboard
monitoring single host
Hystrix Dashboard
monitoring cluster
UI

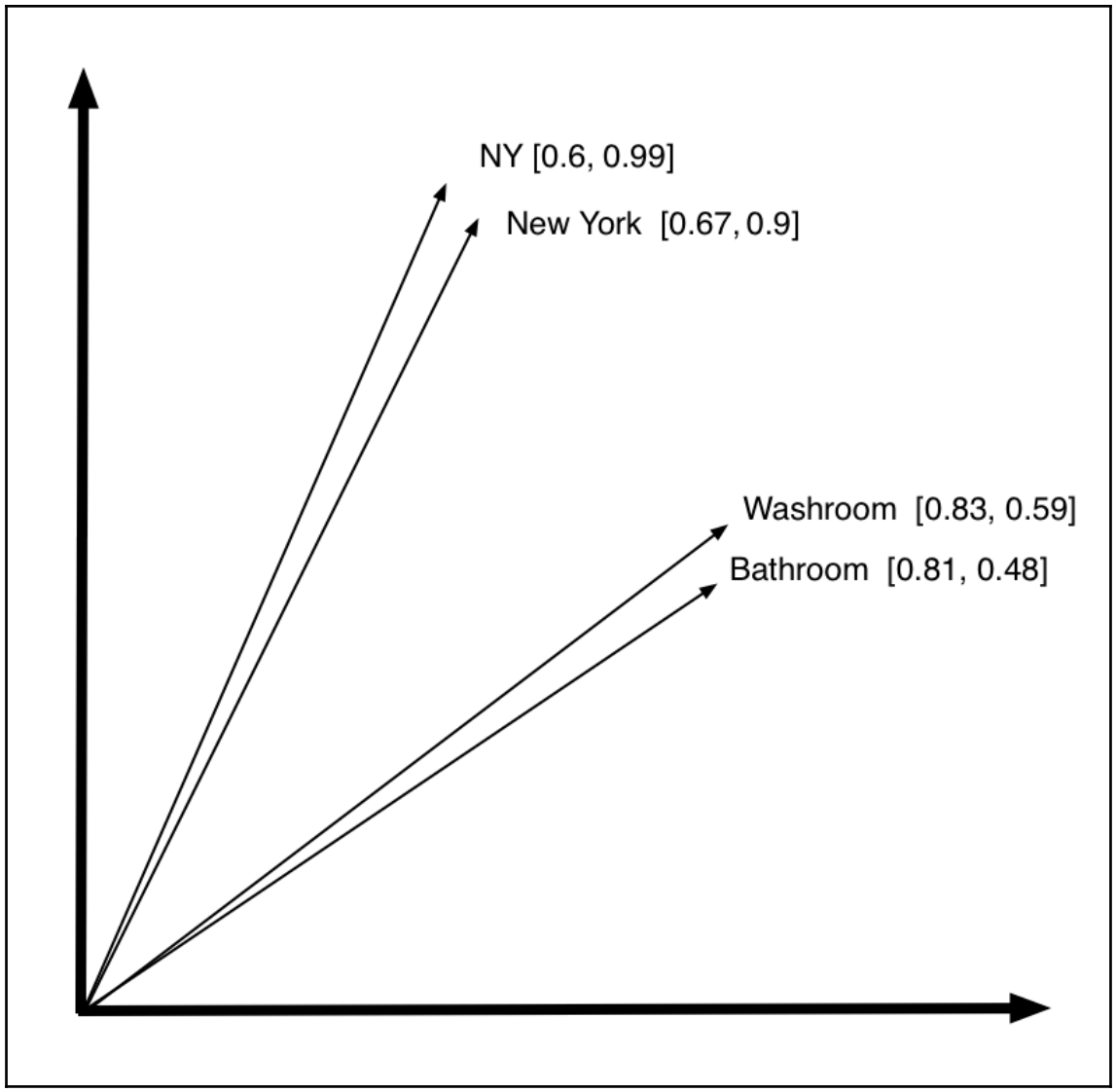

Clients
DNS resolution
TTLed VIP
DNS
DC level heath checks
publish
Service A
Service B
Service C
DB
Data Replication
m1
m2
m3
Topic A
m4
Topic B
Broker
Broker
Datacenter A
Datacenter B

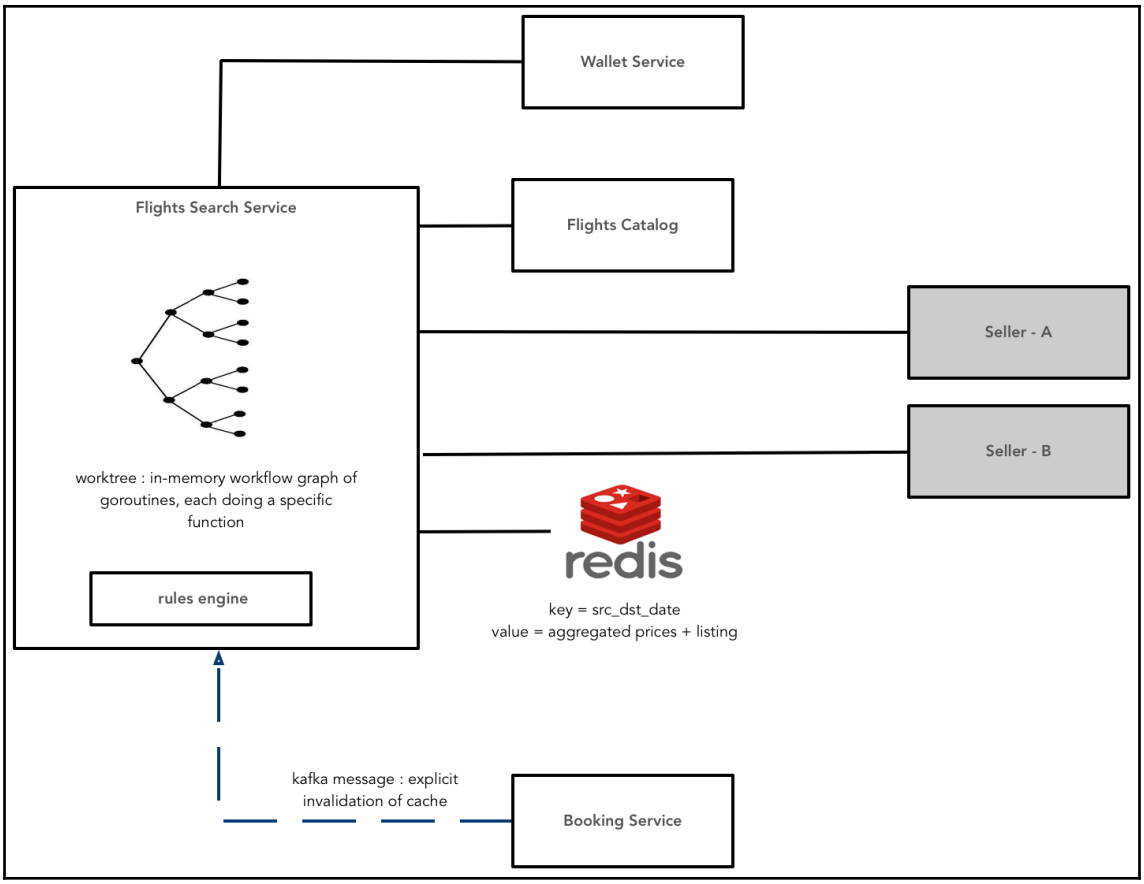
Chapter 10: Case Study – Travel Website

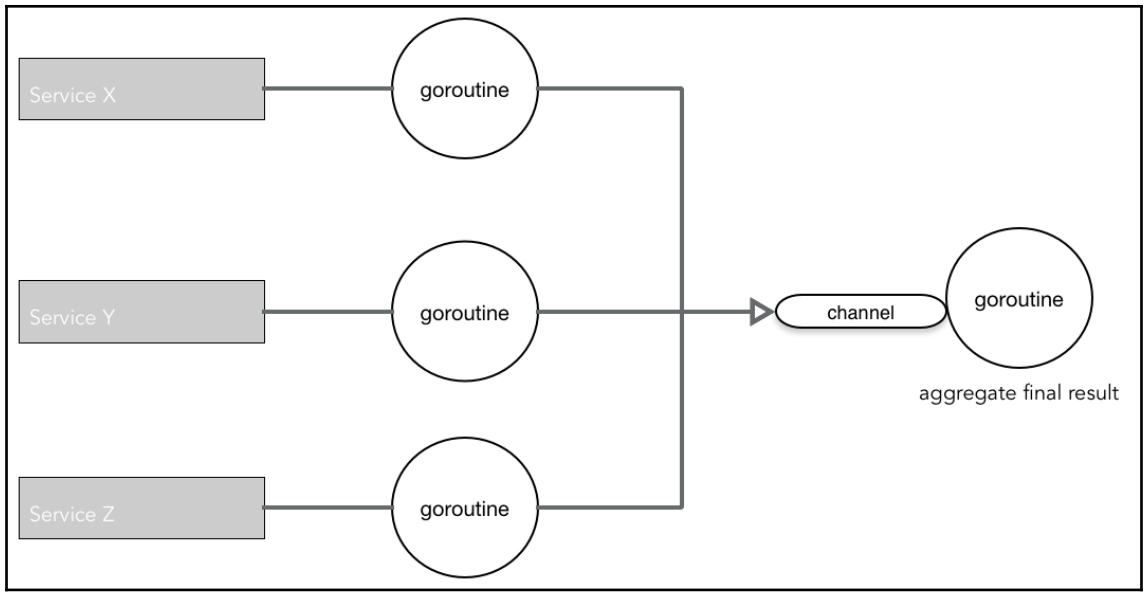


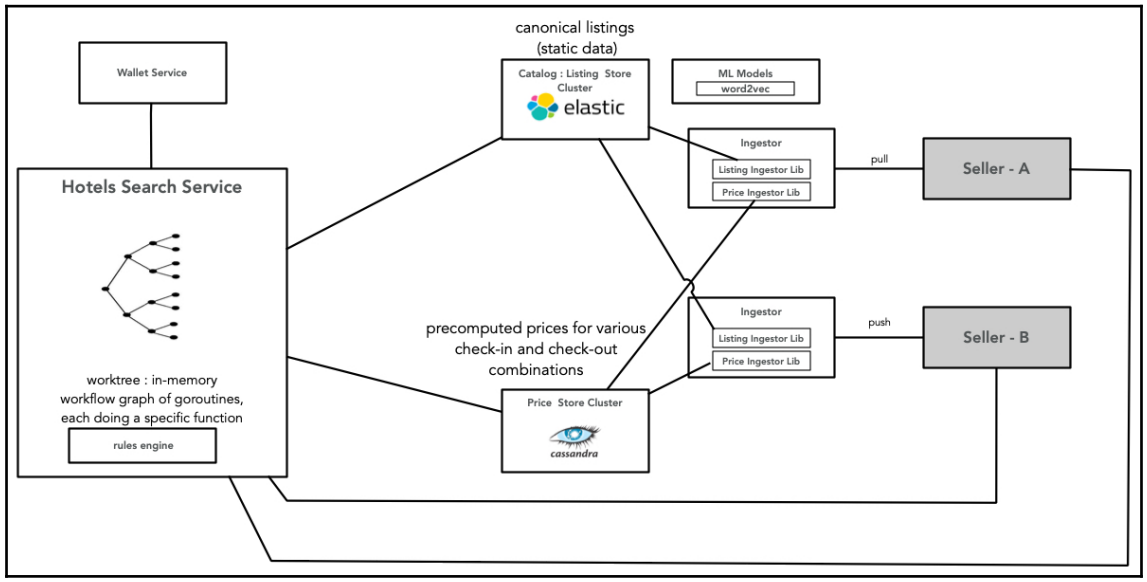

UI (app)
Load Balancer
API calls
Hotel Search Service
Flight Search Service
Payment Service
Hotel Booking Service
Flight Booking Service
Wallet Service
DB
Cache
messages
msg
msg
Topic
Broker
API calls to external systems
Seller
Payment Gateway
Seller

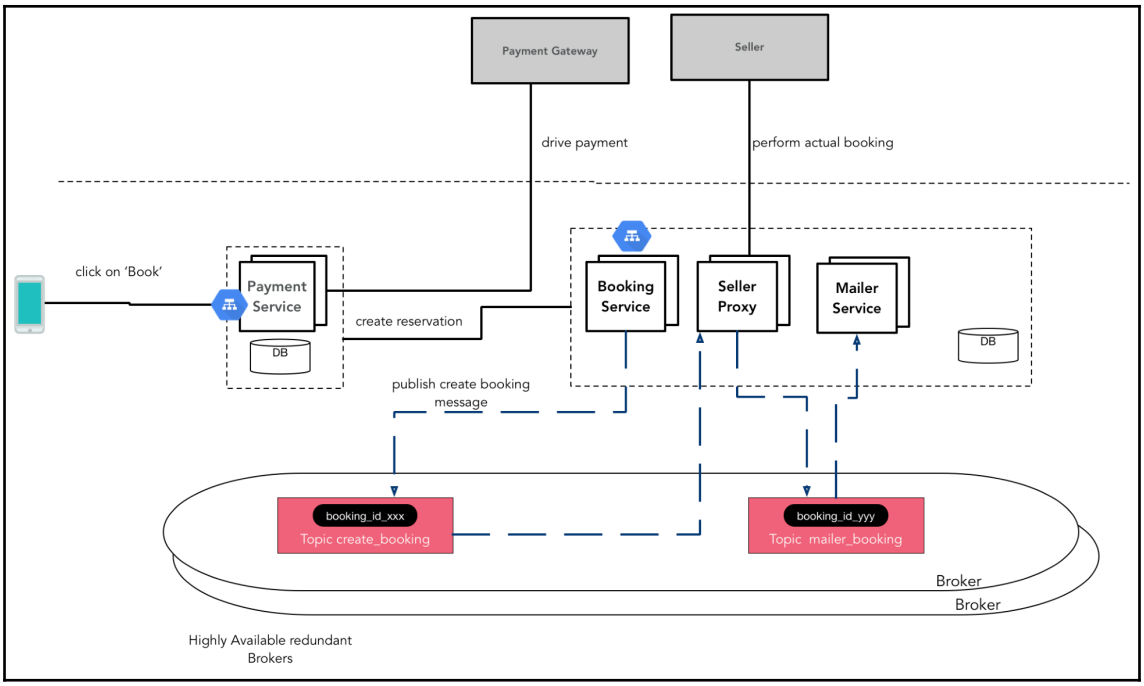

Flights Catalog
Catalog API
Listing Store Cluster
elastic
Ingestor
Listing Ingestor Lib
pull
Seller - A
Ingestor
Listing Ingestor Lib
push
Seller - B
transform data from various sources to canonical form. enrich with keywords to aid conceptual search
ML Models
word2vec

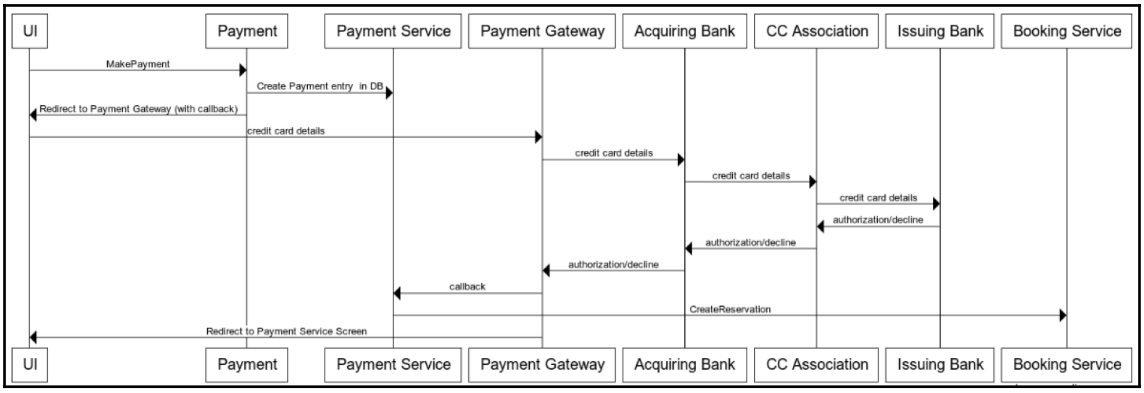

NY [0.6, 0.99]
New York [0.67, 0.9]
Washroom [0.83, 0.59]
Bathroom [0.81, 0.48]


Wallet Service
Flights Search Service
Flights Catalog
Seller - A
Seller - B
worktree : in-memory workflow graph of goroutines, each doing a specific function
rules engine
redis
key = src_dst_date
value = aggregated prices + listing
kafka message : explicit invalidation of cache
Booking Service


Service X
goroutine
Service Y
goroutine
channel
goroutine
aggregate final result
Service Z
goroutine


canonical listings
(static data)
Wallet Service
Catalog : Listing Store
Cluster
elastic
ML Models
word2vec
Ingestor
Listing Ingestor Lib
Price Ingestor Lib
pull
Seller - A
Hotels Search Service
Ingestor
Listing Ingestor Lib
Price Ingestor Lib
push
Seller - B
precomputed prices for various
check-in and check-out
combinations
worktree : in-memory
workflow graph of goroutines,
each doing a specific function
rules engine
Price Store Cluster
cassandra

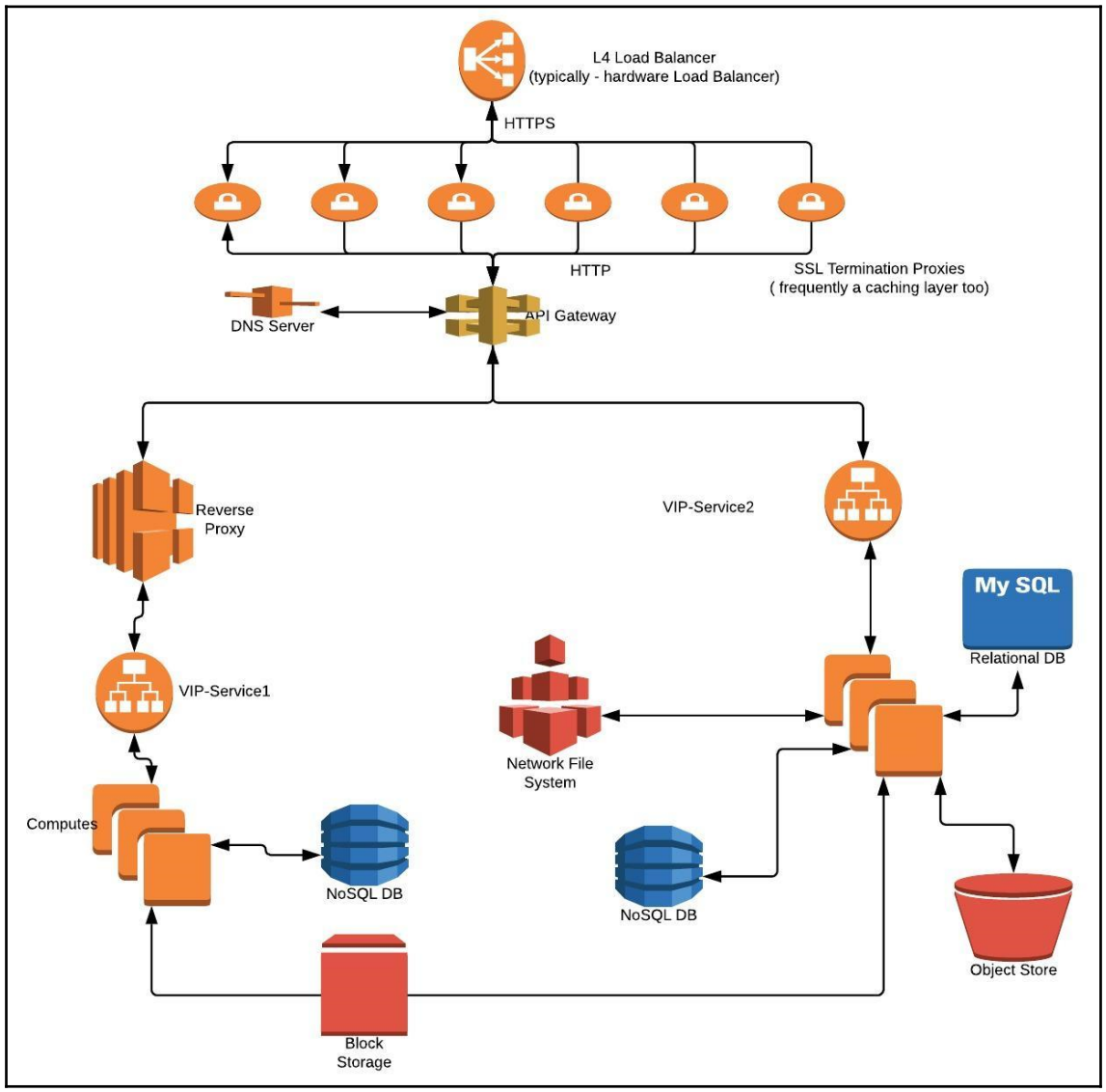

Payment Gateway
Seller
drive payment
perform actual booking
click on 'Book'
Payment Service
DB
create reservation
Booking Service
Seller Proxy
Mailer Service
DB
publish create booking message
booking_id_xxx
Topic create_booking
booking_id_yyy
Topic mailer_booking
Broker
Broker
Highly Available redundant Brokers

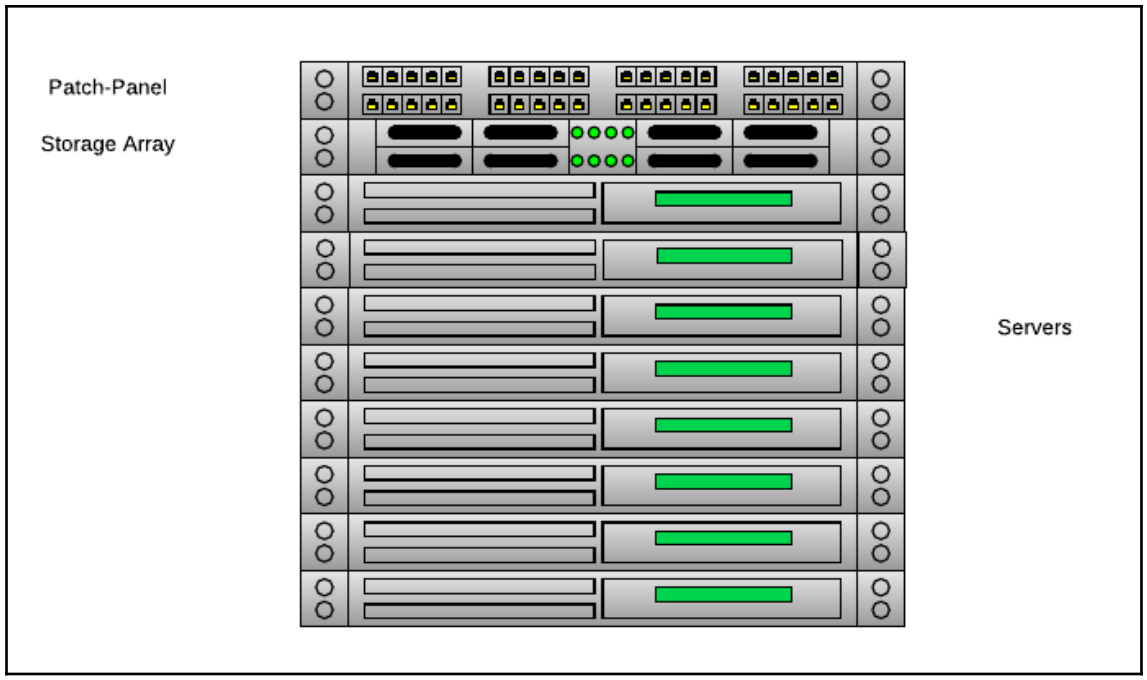

UI
Payment
Payment Service
Payment Gateway
Acquiring Bank
CC Association
Issuing Bank
Booking Service
MakePayment
Create Payment entry in DB
Redirect to Payment Gateway (with callback)
credit card details
credit card details
credit card details
credit card details
authorization/decline
authorization/decline
authorization/decline
callback
CreateReservation
Redirect to Payment Service Screen
UI
Payment
Payment Service
Payment Gateway
Acquiring Bank
CC Association
Issuing Bank
Booking Service

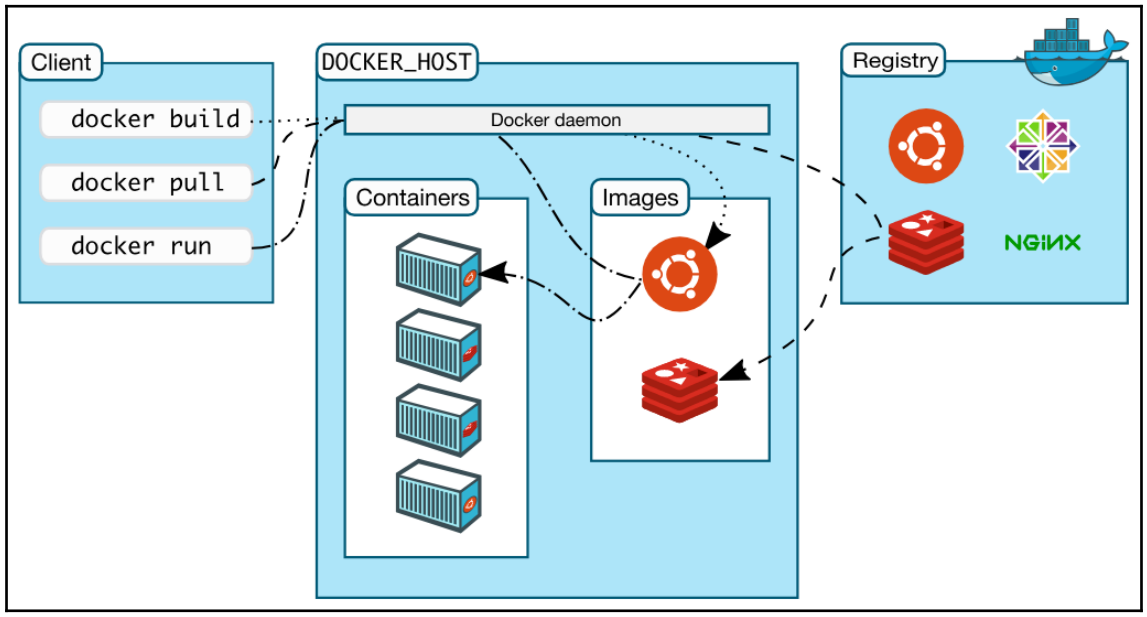
```
+--------------------+--------------+------+-----+---------+-------+
| Field              | Type         | Null | Key | Default | Extra |
+--------------------+--------------+------+-----+---------+-------+
| customer_id        | int(11)      | YES  |     | NULL    |       |
| payment_identifier | int(11)      | YES  |     | NULL    |       |
| entity_id          | int(11)      | YES  |     | NULL    |       |
| room_id            | int(11)      | YES  |     | NULL    |       |
| check_in           | datetime     | YES  |     | NULL    |       |
| check_out          | datetime     | YES  |     | NULL    |       |
| status             | int(10)      | YES  |     | NULL    |       |
| id                 | varchar(100) | NO   | PRI | NULL    |       |
+--------------------+--------------+------+-----+---------+-------+
```

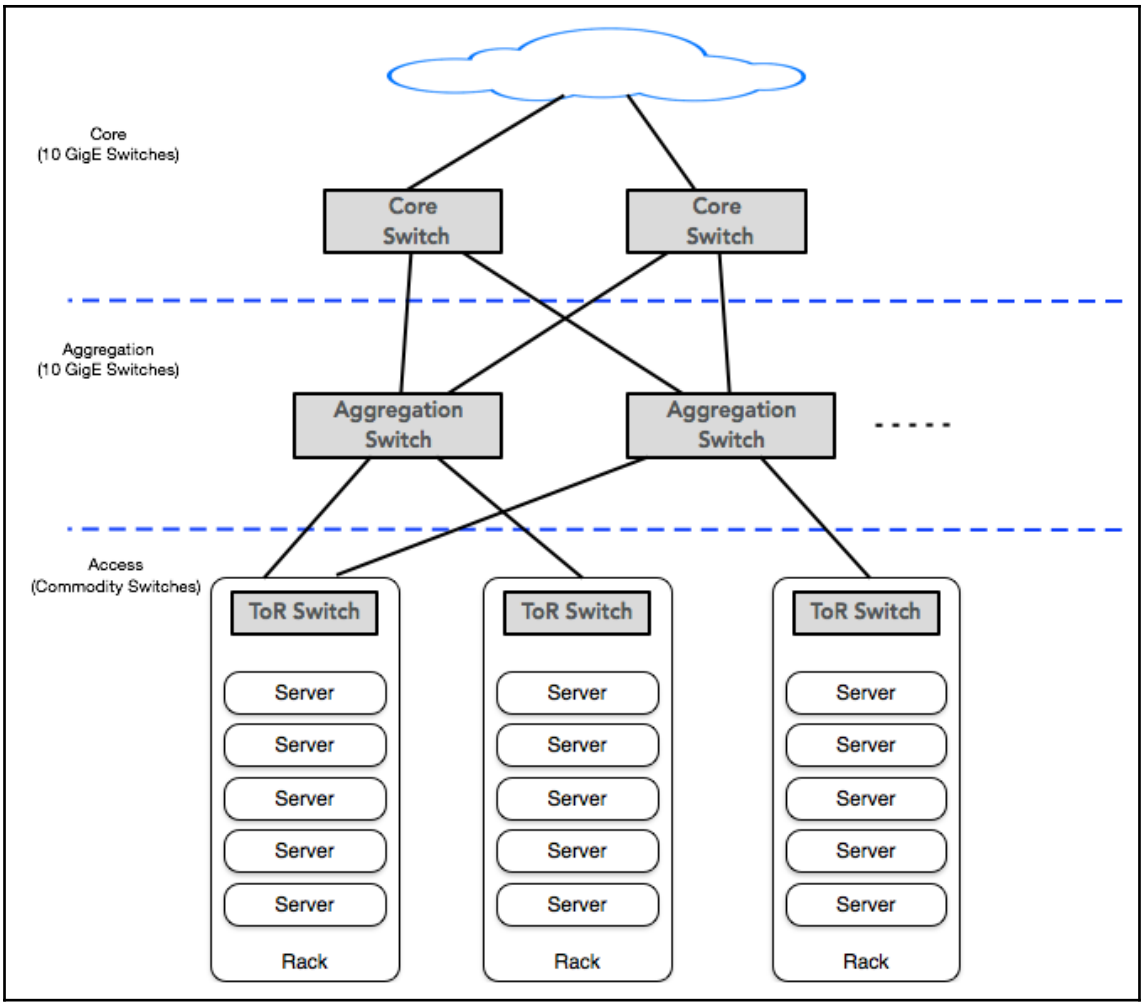
```
+-------------+-----------------+------+-----+---------+----------------+
| Field       | Type            | Null | Key | Default | Extra          |
+-------------+-----------------+------+-----+---------+----------------+
| id          | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| created_at  | timestamp       | YES  |     | NULL    |                |
| updated_at  | timestamp       | YES  |     | NULL    |                |
| deleted_at  | timestamp       | YES  | MUL | NULL    |                |
| entity_id   | int(10) unsigned | YES  |     | NULL    |                |
| room_id     | int(10) unsigned | YES  |     | NULL    |                |
| availability | int(11)         | YES  |     | NULL    |                |
+-------------+-----------------+------+-----+---------+----------------+
```

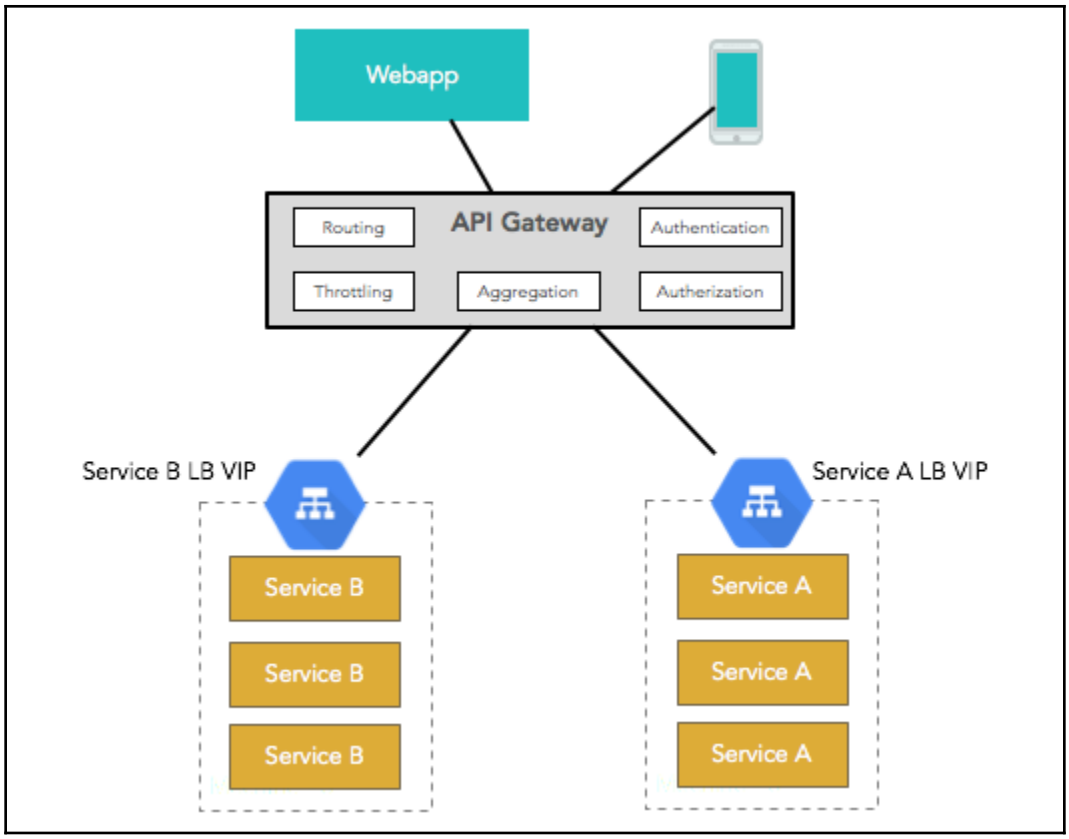
Chapter 11: Planning for Deployment

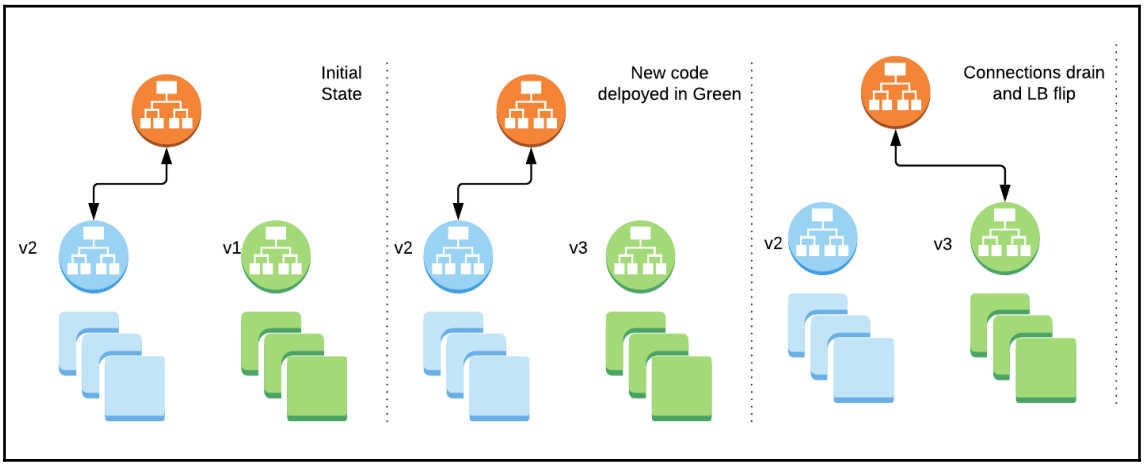


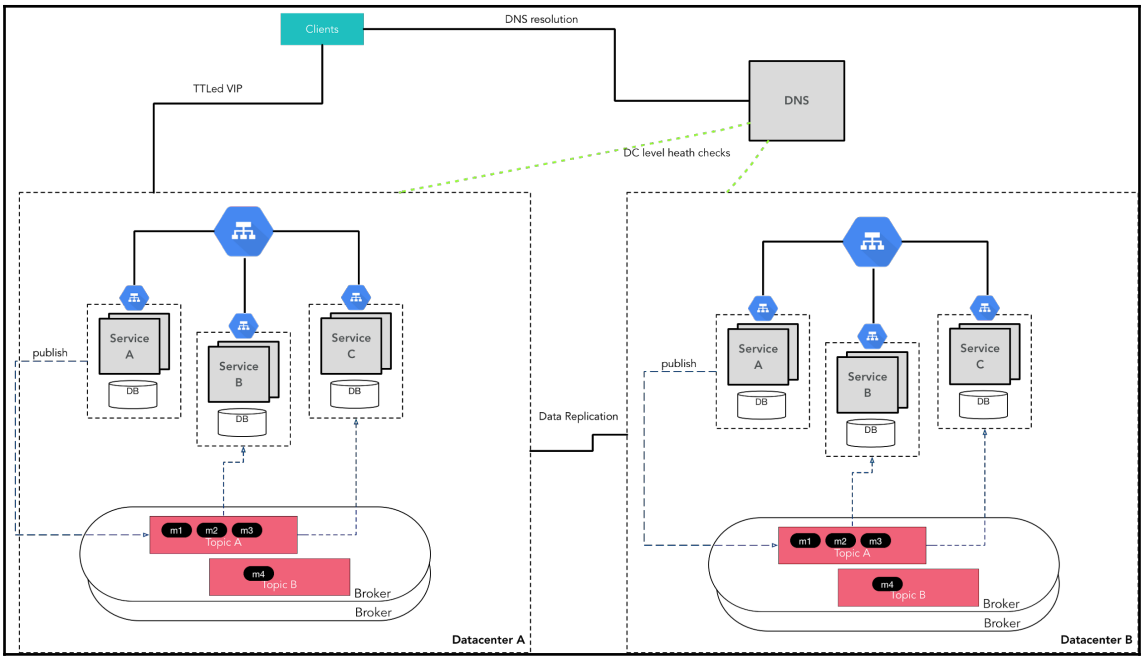

Patch-Panel
Storage Array
Servers

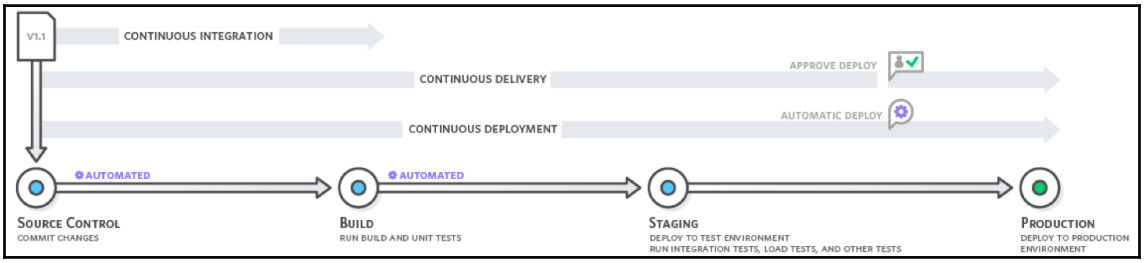

Client
docker build
docker pull
docker run
DOCKER_HOST
Docker daemon
Containers
Images
Registry
NGINX

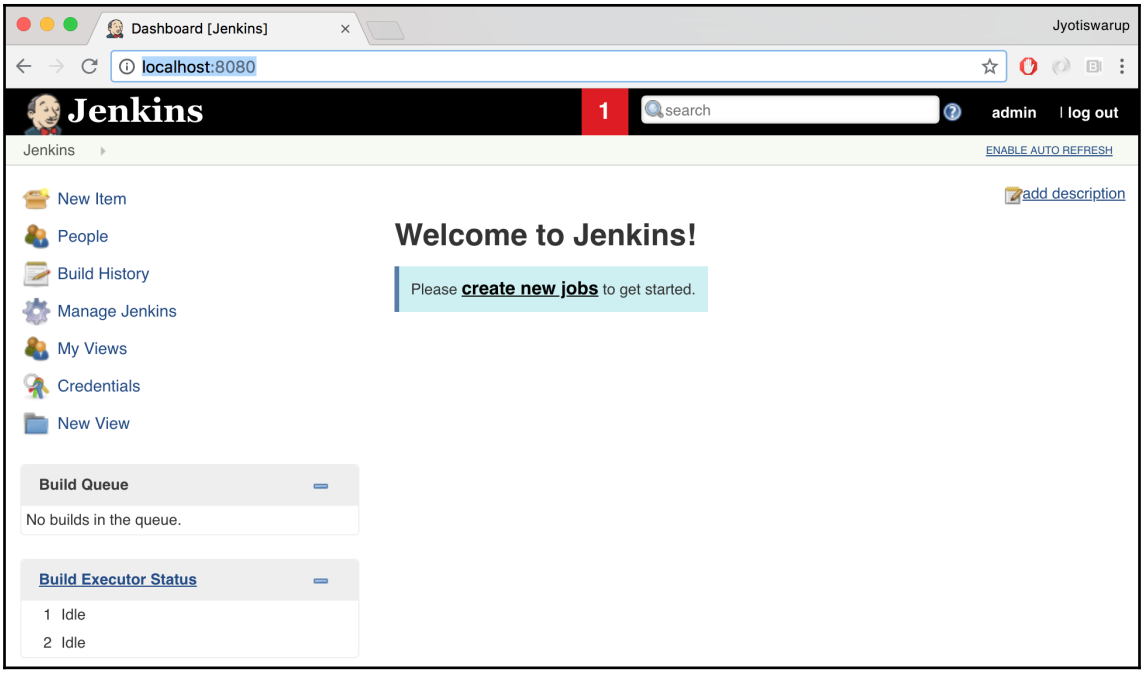

Core
(10 GigE Switches)
Core Switch
Core Switch
Aggregation
(10 GigE Switches)
Aggregation Switch
Aggregation Switch
Access
(Commodity Switches)
ToR Switch
ToR Switch
ToR Switch
Server
Server
Server
Server
Server
Server
Server
Server
Server
Server
Server
Server
Server
Server
Rack
Rack
Rack

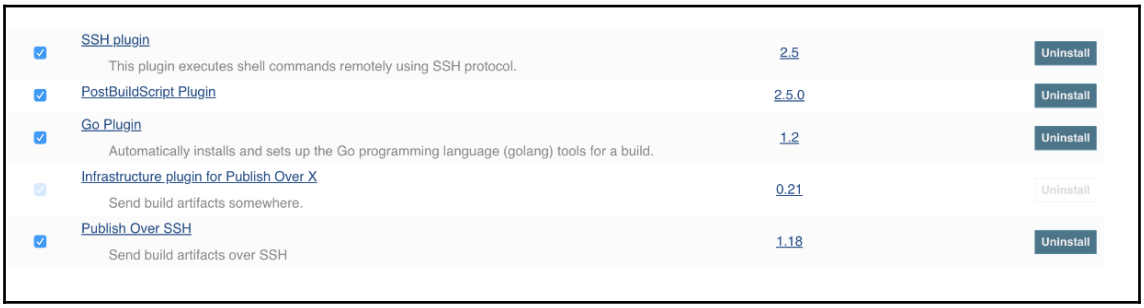

Webapp
API Gateway
Routing
Authentication
Throttling
Aggregation
Autherization
Service B LB VIP
Service A LB VIP
Service B
Service B
Service B
Service A
Service A
Service A

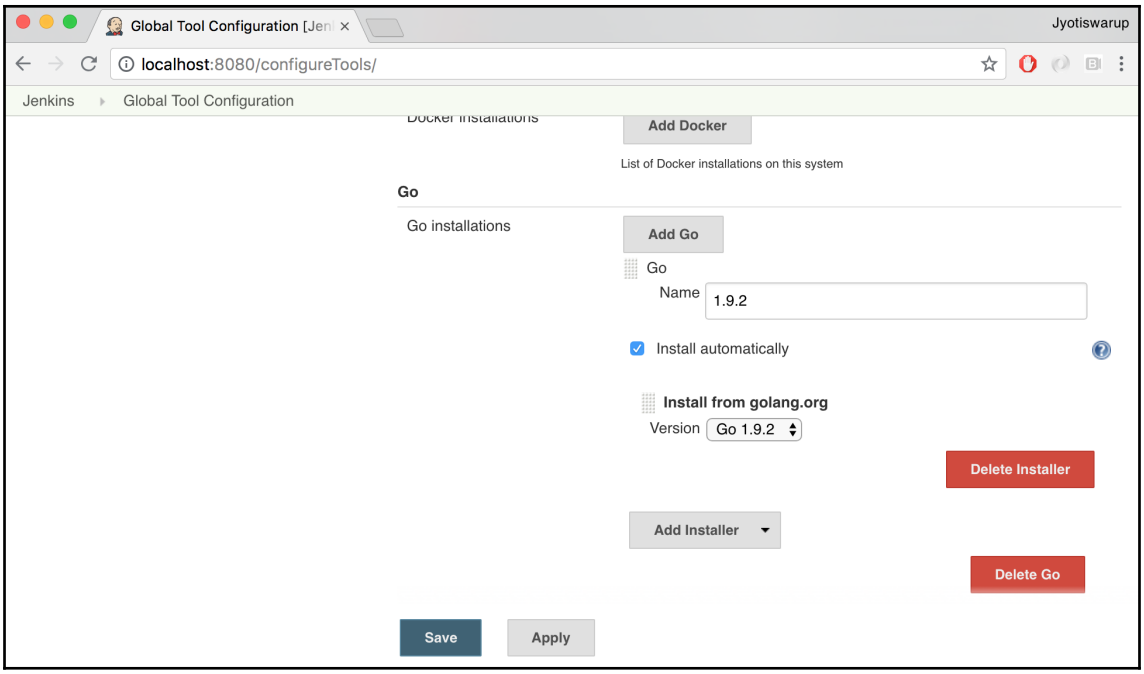

Initial State
v2
v1
New code delpoyed in Green
v2
v3
Connections drain and LB flip
v2
v3

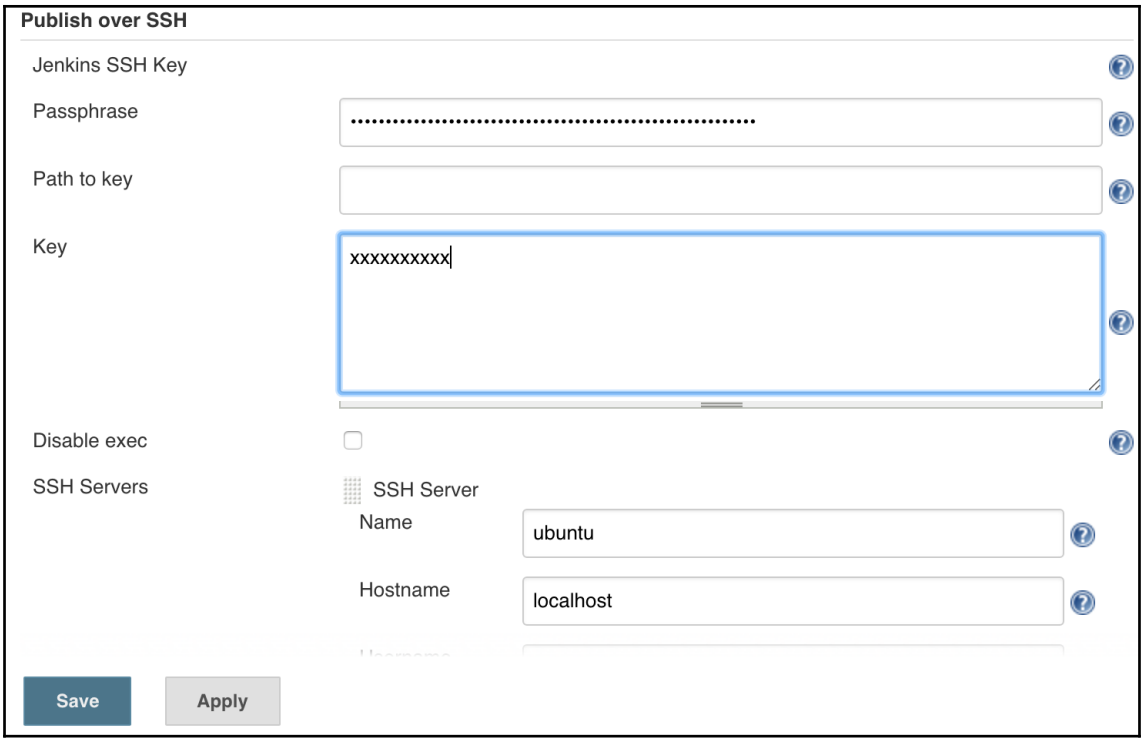

Clients
DNS resolution
TTLed VIP
DNS
DC level heath checks
publish
Service A
Service B
Service C
DB
Data Replication
m1
m2
m3
Topic A
m4
Topic B
Broker
Broker
Datacenter A
Datacenter B

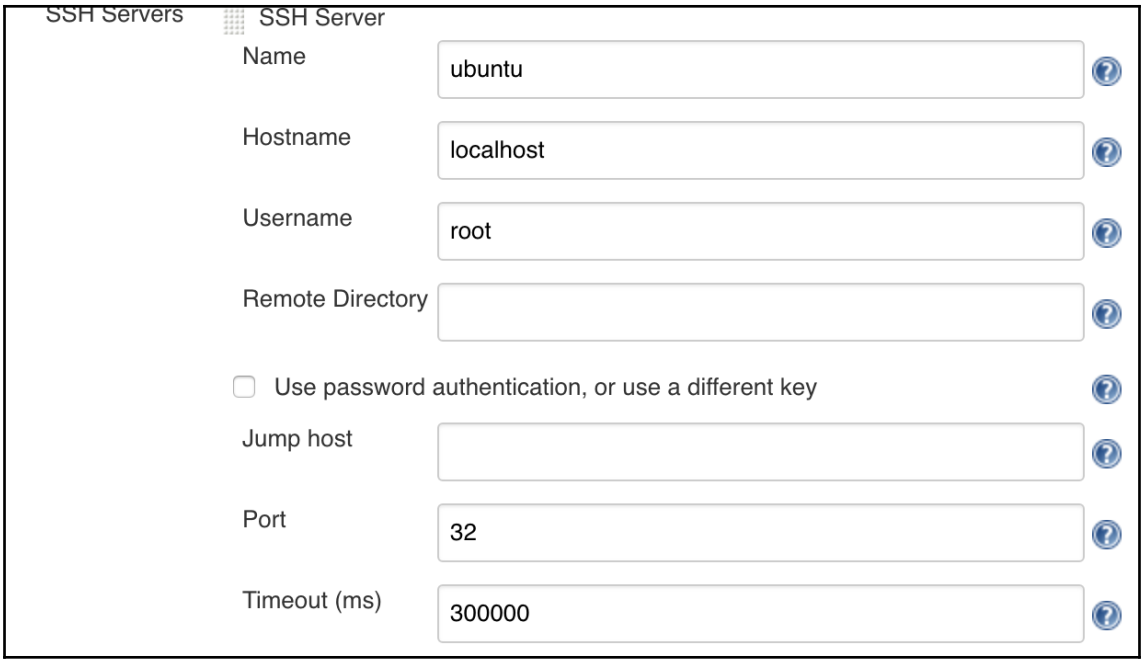

V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
AUTOMATED
AUTOMATED
SOURCE CONTROL
COMMIT CHANGES
BUILD
RUN BUILD AND UNIT TESTS
STAGING
DEPLOY TO TEST ENVIRONMENT
RUN INTEGRATION TESTS, LOAD TESTS, AND OTHER TESTS
PRODUCTION
DEPLOY TO PRODUCTION ENVIRONMENT

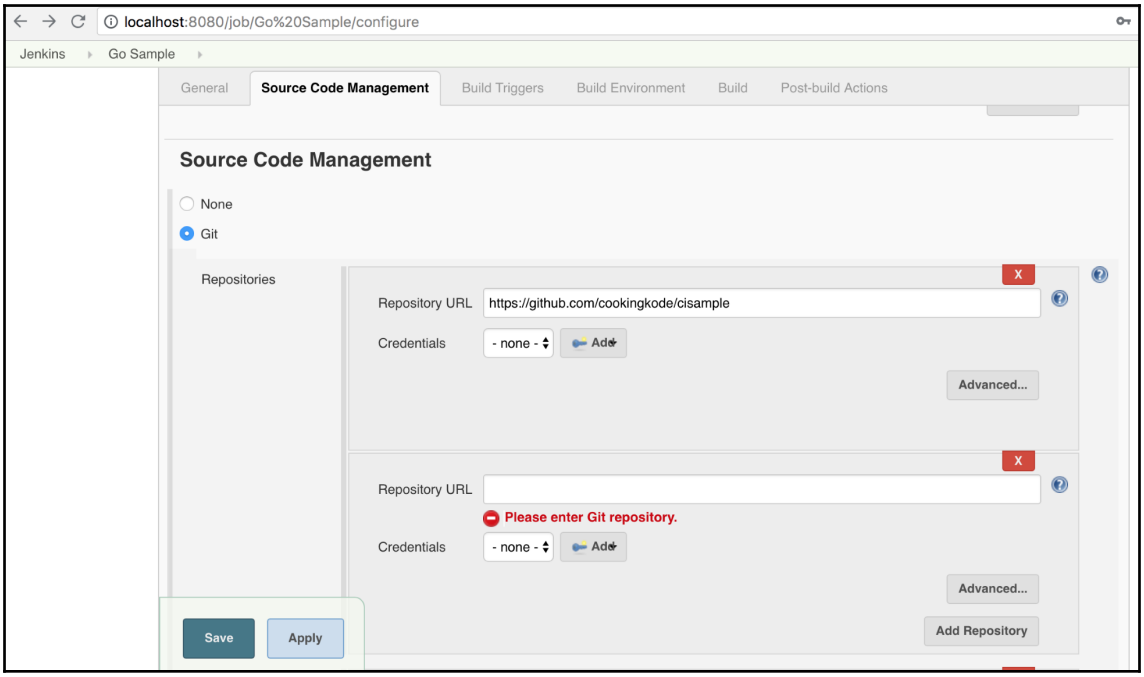

Dashboard [Jenkins]
Jyotiswarup
localhost:8080
Jenkins
1
search
admin
log out
Jenkins
ENABLE AUTO REFRESH
New Item
People
Build History
Manage Jenkins
My Views
Credentials
New View
add description
Welcome to Jenkins!
Please create new jobs to get started.
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle

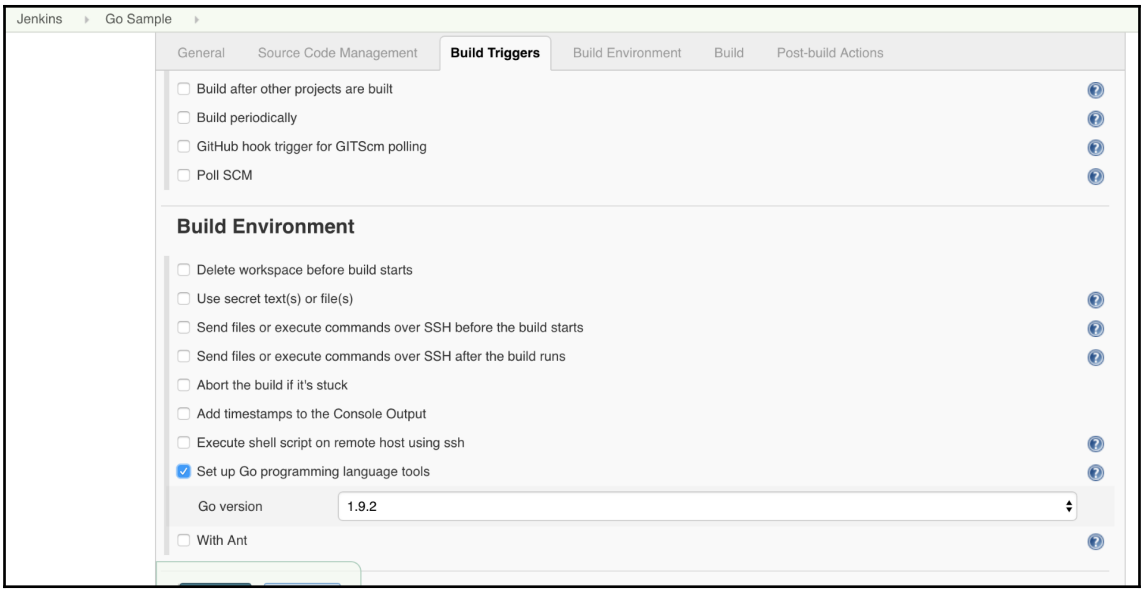

SSH plugin
This plugin executes shell commands remotely using SSH protocol.
2.5
Uninstall
PostBuildScript Plugin
2.5.0
Uninstall
Go Plugin
Automatically installs and sets up the Go programming language (golang) tools for a build.
1.2
Uninstall
Infrastructure plugin for Publish Over X
Send build artifacts somewhere.
0.21
Uninstall
Publish Over SSH
Send build artifacts over SSH
1.18
Uninstall


Global Tool Configuration [Jen
Jyotiswarup
localhost:8080/configureTools/
Jenkins
Global Tool Configuration
Add Docker
List of Docker installations on this system
Go
Go installations
Add Go
Go
Name
1.9.2
Install automatically
Install from golang.org
Version
Go 1.9.2
Delete Installer
Add Installer
Delete Go
Save
Apply

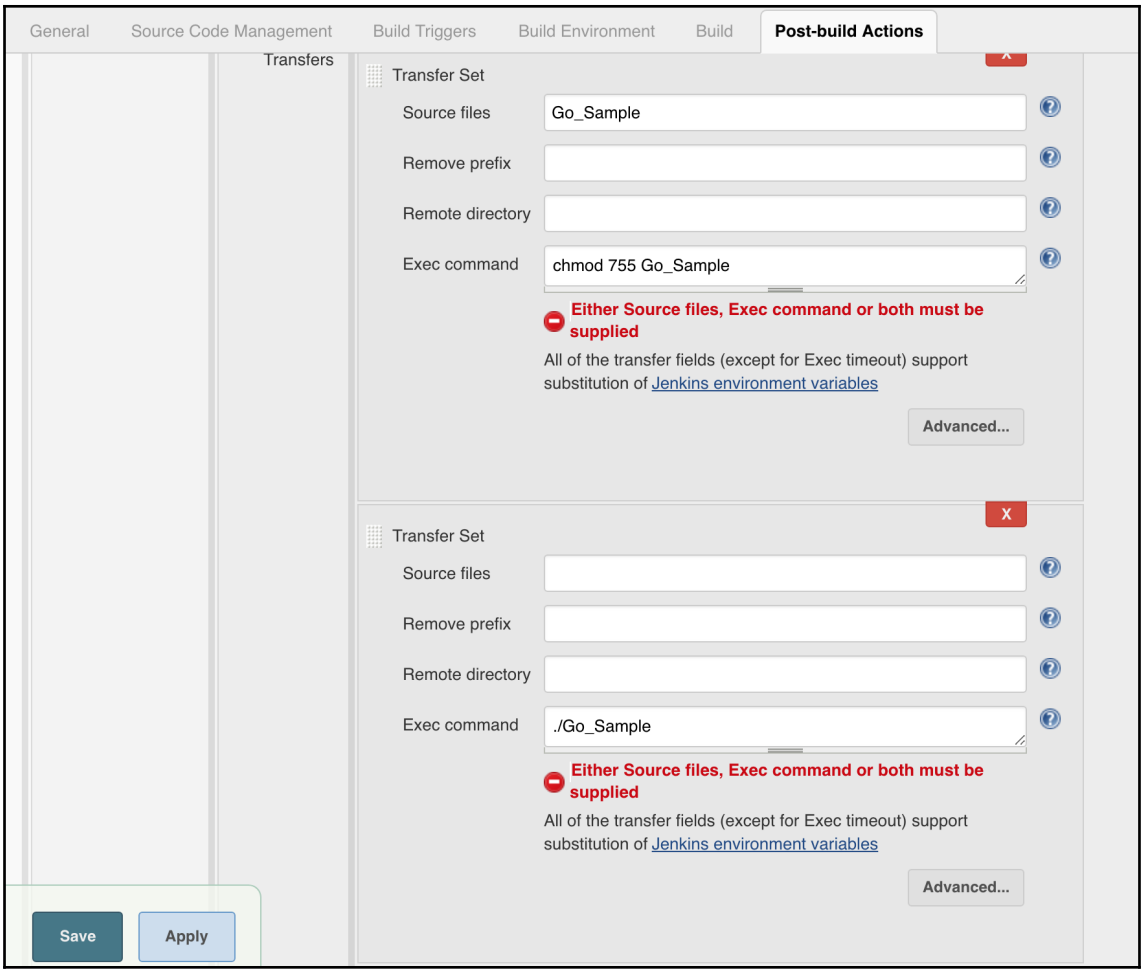

Publish over SSH
Jenkins SSH Key
Passphrase
Path to key
Key
xxxxxxxxxx
Disable exec
SSH Servers
SSH Server
Name
ubuntu
Hostname
localhost
Save
Apply

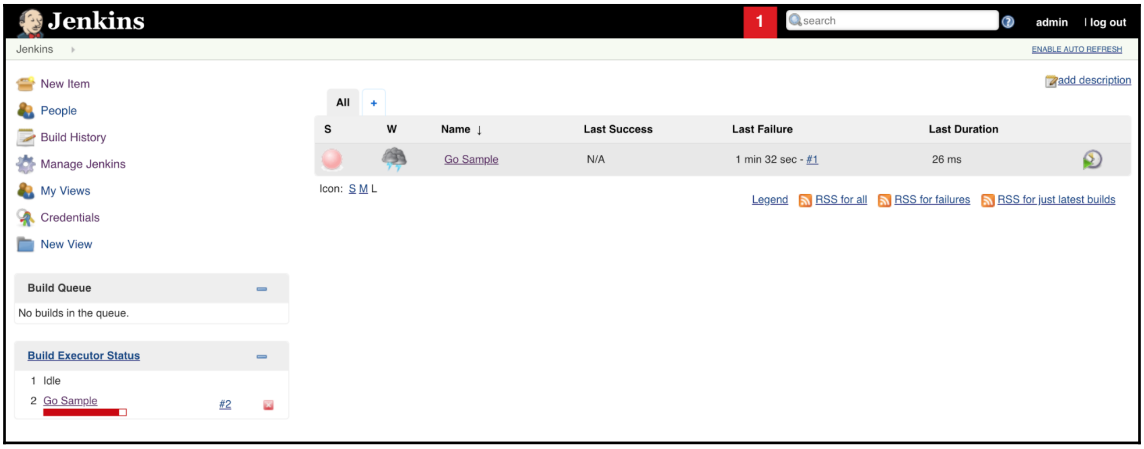

SSH Servers
SSH Server
Name
ubuntu
Hostname
localhost
Username
root
Remote Directory
Use password authentication, or use a different key
Jump host
Port
32
Timeout (ms)
300000

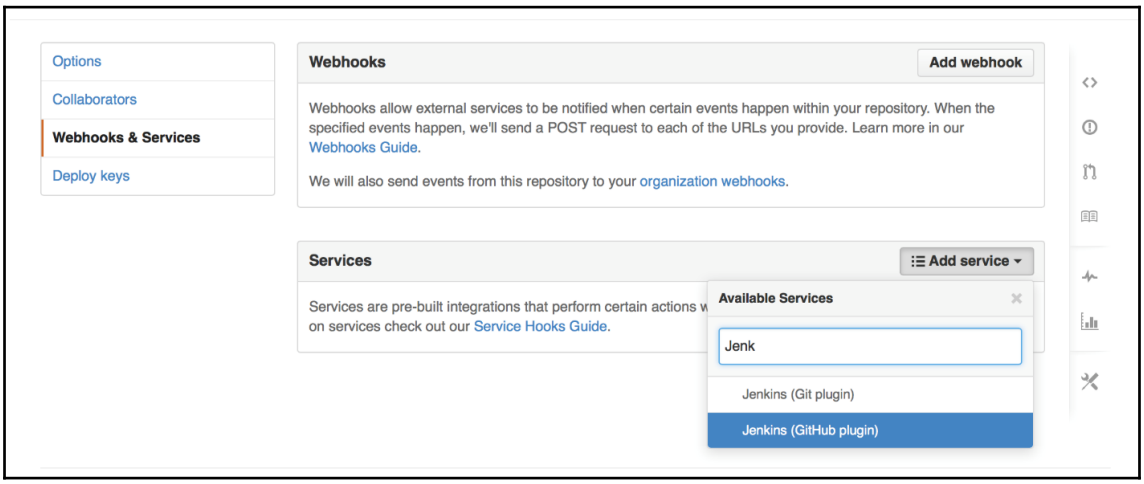

localhost:8080/job/Go%20Sample/configure
Jenkins
Go Sample
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Source Code Management
None
Git
Repositories
Repository URL
https://github.com/cookingkode/cisample
Credentials
- none -
Add
Advanced...
Repository URL
Please enter Git repository.
Credentials
- none -
Add
Advanced...
Add Repository
Save
Apply

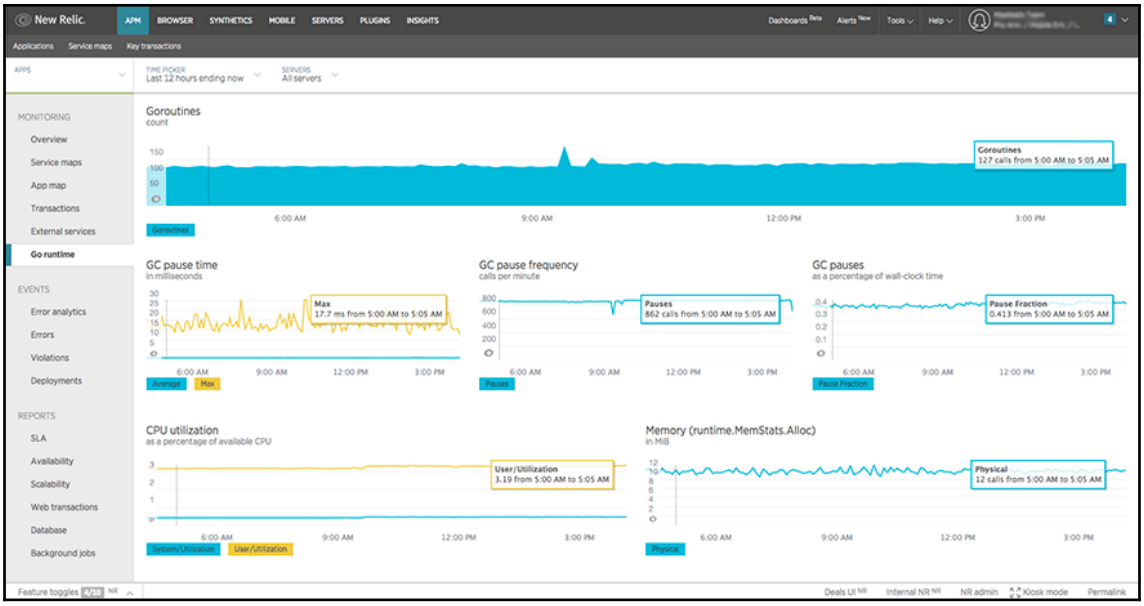

Jenkins
Go Sample
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Send files or execute commands over SSH before the build starts
Send files or execute commands over SSH after the build runs
Abort the build if it's stuck
Add timestamps to the Console Output
Execute shell script on remote host using ssh
Set up Go programming language tools
Go version
1.9.2
With Ant


Build
Execute shell
Command
go get -u github.com/gin-gonic/gin
See the list of available environment variables
Advanced...
Execute shell
Command
GOOS=linux GOARCH=386 CGO_ENABLED=0 go build -o Go_Sample
See the list of available environment variables
Advanced...
Add build step
Save
Apply

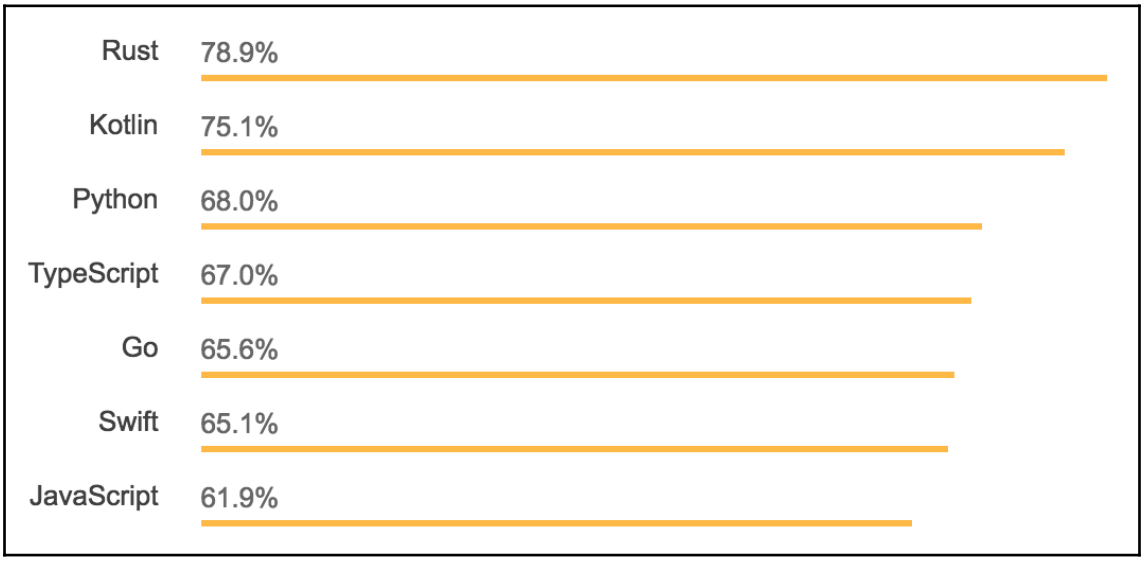

General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Transfers
Transfer Set
Source files
Go_Sample
Remove prefix
Remote directory
Exec command
chmod 755 Go_Sample
Either Source files, Exec command or both must be supplied
All of the transfer fields (except for Exec timeout) support substitution of Jenkins environment variables
Advanced...
X
Transfer Set
Source files
Remove prefix
Remote directory
Exec command
./Go_Sample
Either Source files, Exec command or both must be supplied
All of the transfer fields (except for Exec timeout) support substitution of Jenkins environment variables
Advanced...
Save
Apply

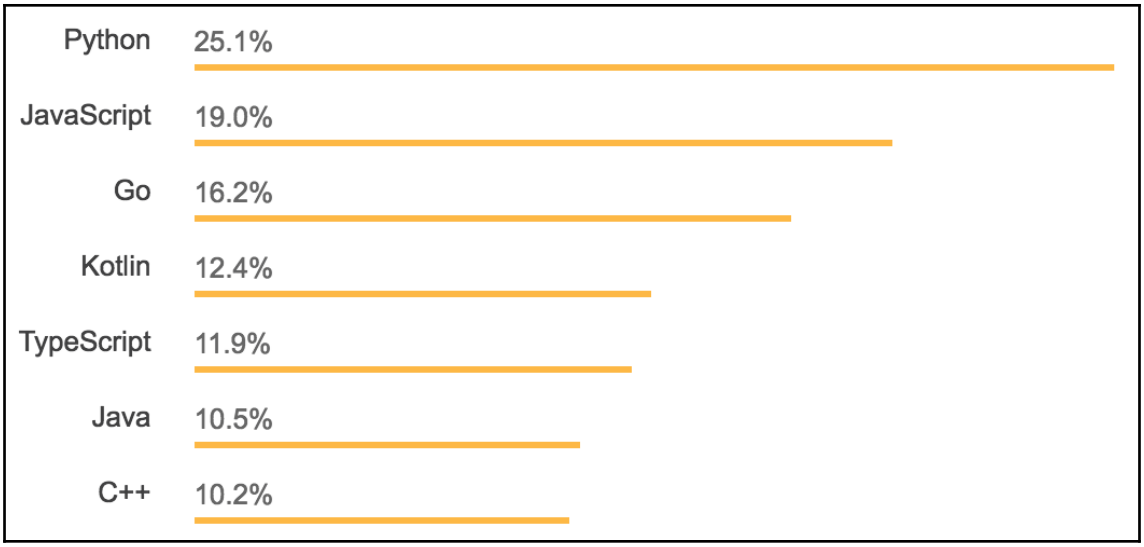

Jenkins
1
search
admin
log out
Jenkins
ENABLE AUTO REFRESH
New Item
People
Build History
Manage Jenkins
My Views
Credentials
New View
add description
All
+
S
W
Name ↓
Last Success
Last Failure
Last Duration
Go Sample
N/A
1 min 32 sec - #1
26 ms
Icon: S M L
Legend
RSS for all
RSS for failures
RSS for just latest builds
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Go Sample
#2

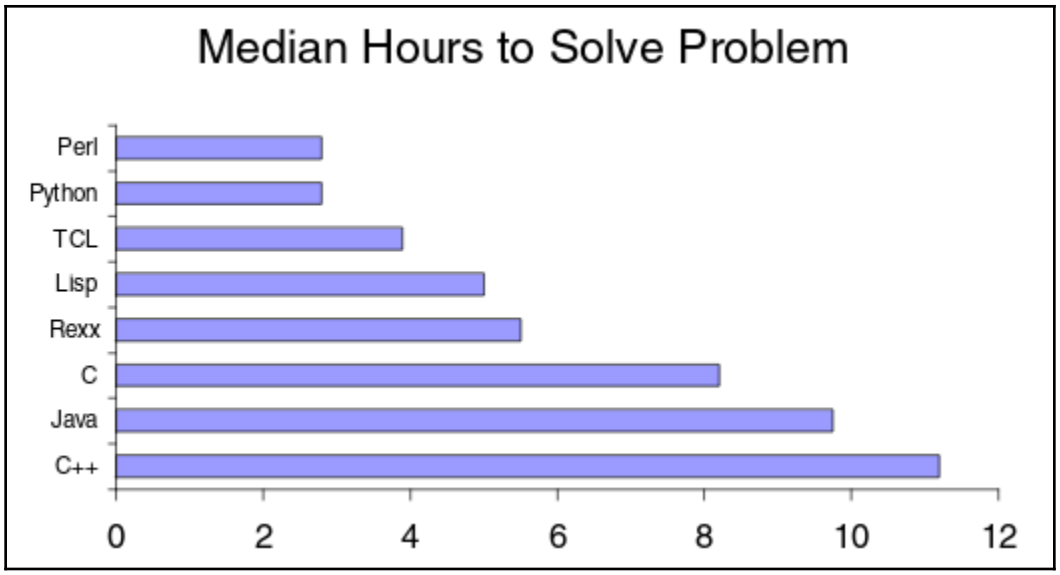

Options
Collaborators
Webhooks & Services
Deploy keys
Webhooks
Add webhook
Webhooks allow external services to be notified when certain events happen within your repository. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.
We will also send events from this repository to your organization webhooks.
Services
Add service
Services are pre-built integrations that perform certain actions w
on services check out our Service Hooks Guide.
Available Services
Jenk
Jenkins (Git plugin)
Jenkins (GitHub plugin)

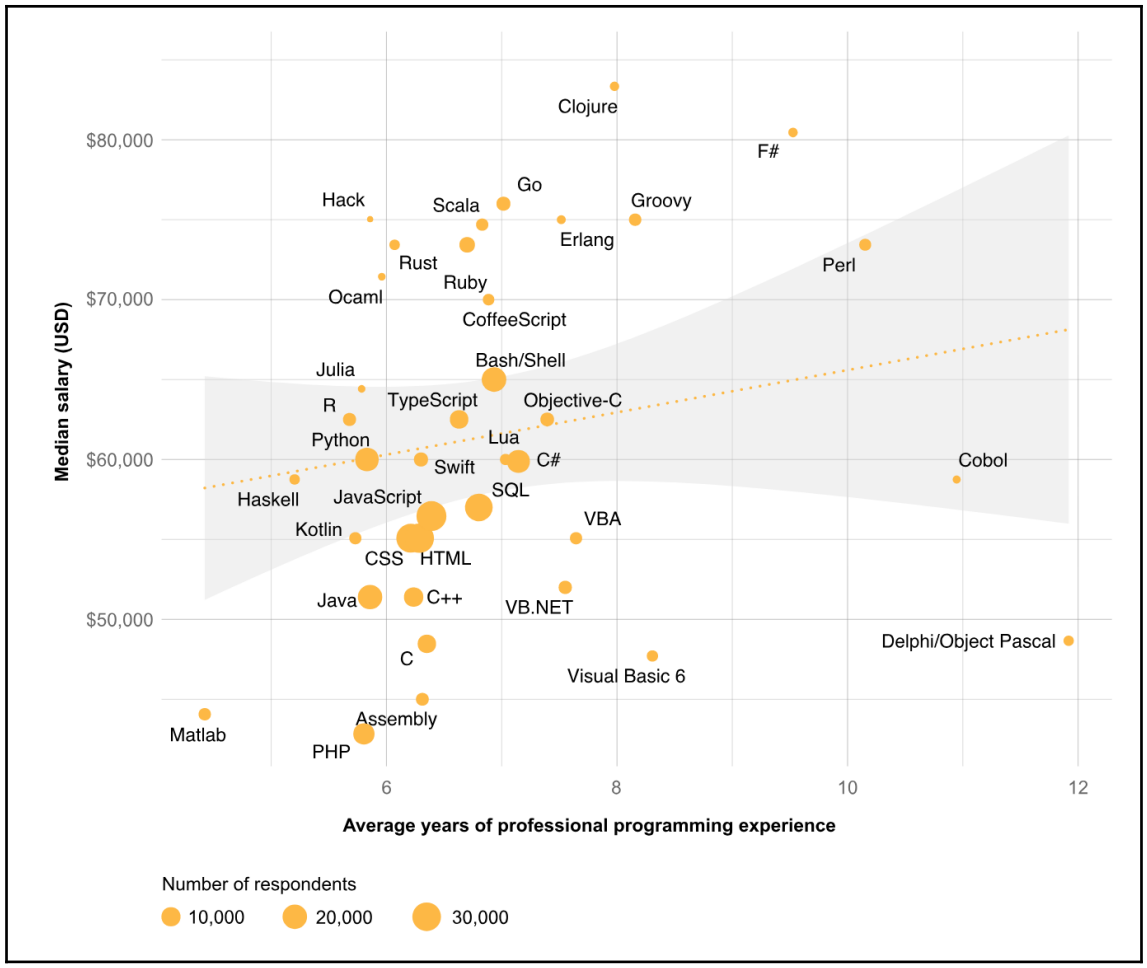

New Relic.
APM
BROWSER
SYNTHETICS
MOBILE
SERVERS
PLUGINS
INSIGHTS
Dashboards Beta
Alerts New
Tools
Help
Applications
Service maps
Key transactions
APPS
TIME PICKER
Last 12 hours ending now
SERVERS
All servers
MONITORING
Overview
Service maps
App map
Transactions
External services
Go runtime
EVENTS
Error analytics
Errors
Violations
Deployments
REPORTS
SLA
Availability
Scalability
Web transactions
Database
Background jobs
Goroutines
count
Goroutines
127 calls from 5:00 AM to 5:05 AM
6:00 AM
9:00 AM
12:00 PM
3:00 PM
GC pause time
in milliseconds
Max
17.7 ms from 5:00 AM to 5:05 AM
Average
Max
GC pause frequency
calls per minute
Pauses
862 calls from 5:00 AM to 5:05 AM
GC pauses
as a percentage of wall-clock time
Pause Fraction
0.413 from 5:00 AM to 5:05 AM
CPU utilization
as a percentage of available CPU
User/Utilization
3.19 from 5:00 AM to 5:05 AM
System/Utilization
Memory (runtime.MemStats.Alloc)
in MiB
Physical
12 calls from 5:00 AM to 5:05 AM
Feature toggles
Deals UI NR
Internal NR NR
NR admin
Kiosk mode
Permalink

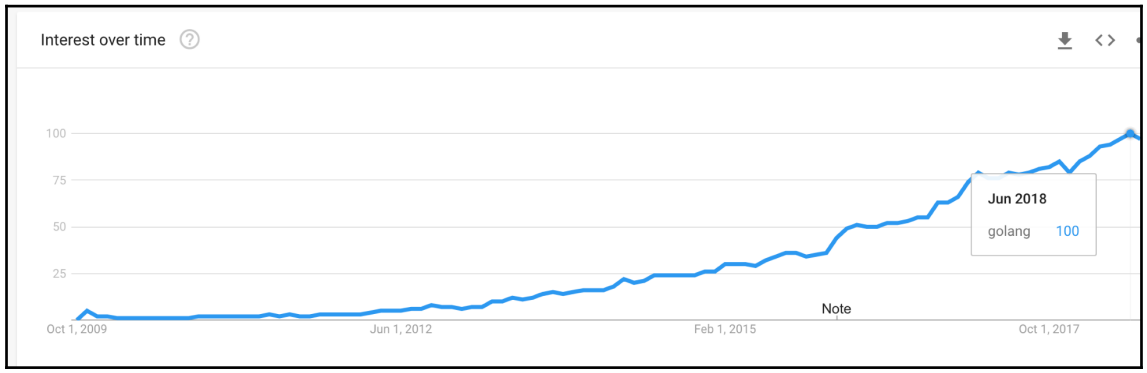
Chapter 12: Migrating Applications






Median Hours to Solve Problem
Perl
Python
TCL
Lisp
Rexx
C
Java
C++
0
2
4
6
8
10
12


Clojure
F#
$80,000
Go
Hack
Scala
Groovy
Erlang
Perl
Rust
Ruby
Ocaml
$70,000
CoffeeScript
Julia
Bash/Shell
R
TypeScript
Objective-C
Python
Lua
$60,000
Swift
C#
Cobol
Haskell
JavaScript
SQL
VBA
Kotlin
CSS
HTML
Java
C++
VB.NET
$50,000
Delphi/Object Pascal
C
Visual Basic 6
Matlab
Assembly
PHP
Median salary (USD)
6
8
10
12
Average years of professional programming experience
Number of respondents
10,000
20,000
30,000


Interest over time
100
75
50
25
Jun 2018
golang 100
Note
Oct 1, 2009
Jun 1, 2012
Feb 1, 2015
Oct 1, 2017