Bob Brown

Microservices

Transentia Pty. Ltd.
bob@transentia.com.au
http://www.transentia.com.au

Do YOU suffer from MONOLITHS?

Do YOU suffer from MONOLITHS?
Do YOU have trouble satisfying your boss?

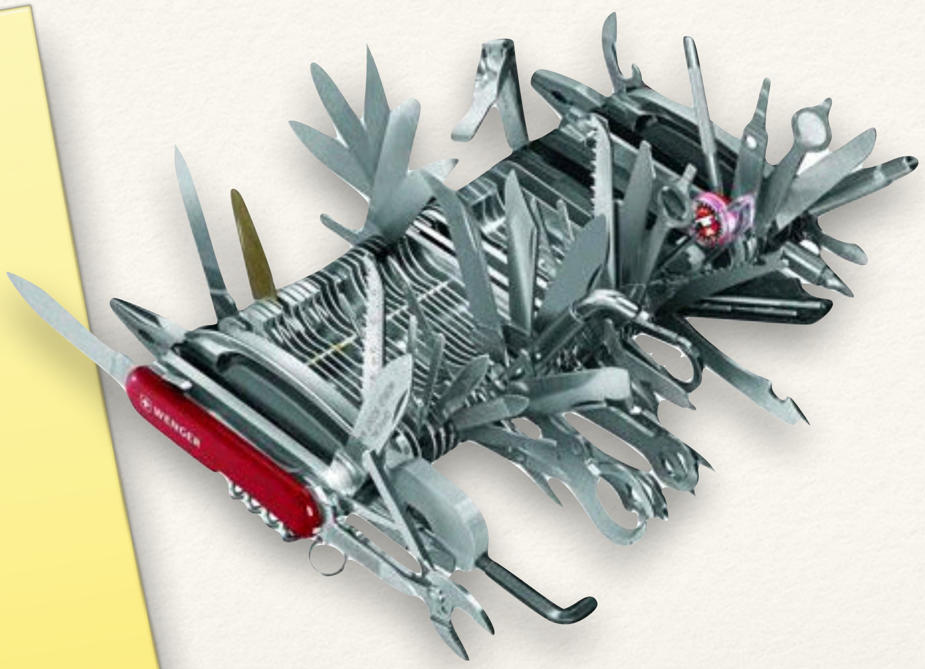
Do YOU suffer from MONOLITHS?
Do YOU have trouble satisfying your boss?
Do YOU have trouble **scaling up** sometimes?

Do YOU suffer from MONOLITHS?
Do YOU have trouble satisfying your boss?
Do YOU have trouble scaling up sometimes?
Are you **TIRED** of all those clever salesmen, with their silver bullets
and their **EXPENSIVE** patent nostrums?

YOU NEED ALL NEW Microservices!

PREMISE

The current crop of "Enterprise Solutions" have not delivered bang-for-buck, while being so large & complex that not even the best developers can produce high-quality software within time and budget.



PROMISE

A "back to basics" approach with modern technologies can get us to where we need to go easier & faster, with far nicer programmer ergonomics and with a better end result.

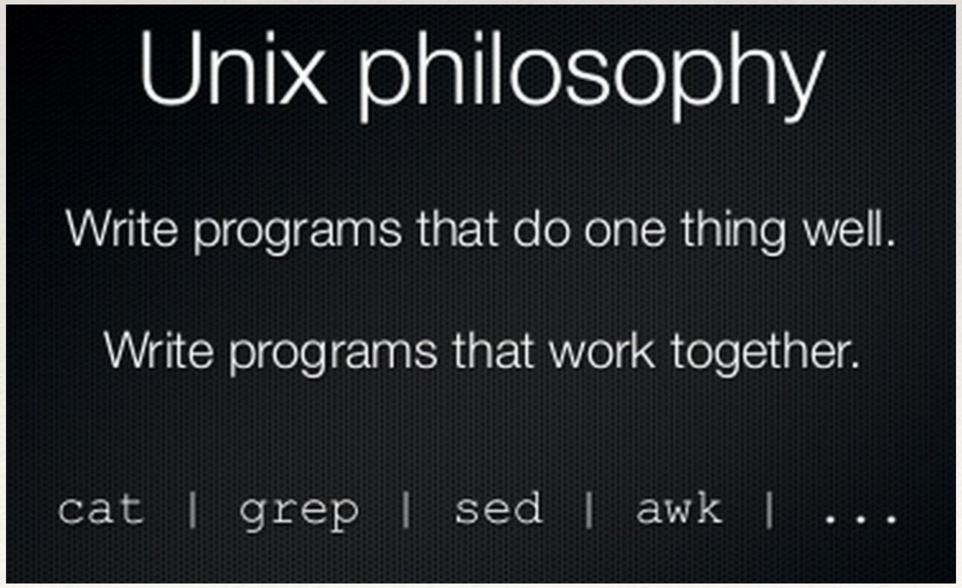
What Is?

"An approach to developing a single service application as a **suite of small services**, each usually running in its **own process** and communicating with **lightweight mechanisms**, usually an HTTP resource API. These services are **built around business capabilities**, may be written in **different programming languages** and use **different data storage technologies**. They are typically **highly reliable**, adhere to fundamental **DevOps principles** for their runtime management and are **deployed via highly automated processes**."

Adapted from: http://www.thoughtworks.com/insights/blog/microservices-nutshell

Assertions/Statements

- fine grained SOA architecture done the UNIX way
- it's NOT about size
- it's not always about processes; shared libraries sometimes rule harder…
- human comprehension…small enough to fit in your head
- distributed objects for hipsters
- it's just SOA done {right,wrong}
- SOA without the vendor bulls4!t


Unix philosophy

Write programs that do one thing well.

Write programs that work together.

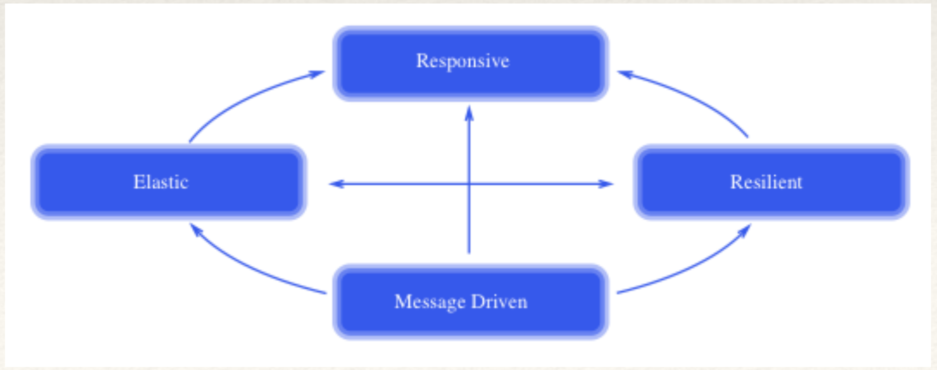
```
cat | grep | sed | awk | ...
```

Assertions/Statements...

- Microservices are not strictly defined and that's the beauty. It is a lightweight style of implementing SOA that works.
- the word "monolith" that's dreaded by architects has a very positive air from the customers perspective
- ...a triangulation on ideal practices for app development, paying particular attention to the dynamics of the organic growth of an app over time, the dynamics of collaboration between developers working on the app's codebase, and avoiding the cost of software erosion
- for every person who thinks they are doing micro-services, I bet I can find someone who would argue they should have split it up more (or less).
- Any piece of functionality that is in danger of being built more than once in an organization (think authentication, user management, etc.) in a classic stovepipe architecture is a candidate for a micro-service (or set of micro-services as the case may be).

Reactive Manifesto 2.0

- Today's demands are simply not met by yesterday's software architectures.
- Reactive Systems are:
  - Responsive: The system responds in a timely manner if at all possible.
  - Resilient: The system stays responsive in the face of failure.
  - Elastic: The system stays responsive under varying workload.
  - Message Driven: Reactive Systems rely on asynchronous message-passing to establish a boundary between components that ensures loose coupling, isolation, location transparency, and provides the means to delegate errors as messages.



http://www.reactivemanifesto.org/

Livin' The Dream

comparethe market.com.au™

NETFLIX

SOUNDCLOUD

yammer

The Enterprise Social Network

realestate.com.au®

Australia's No.1 property site™

ebay

amazon.com®

KARMA

Still Dreamin'

“Netflix , which is a very popular video streaming service that’s responsible for up to 30% of internet traffic, has a large scale, service-oriented architecture. They handle over a billion calls per day to their video streaming API from over 800 different kinds of devices. Each API call fans out to an average of six calls to backend services.

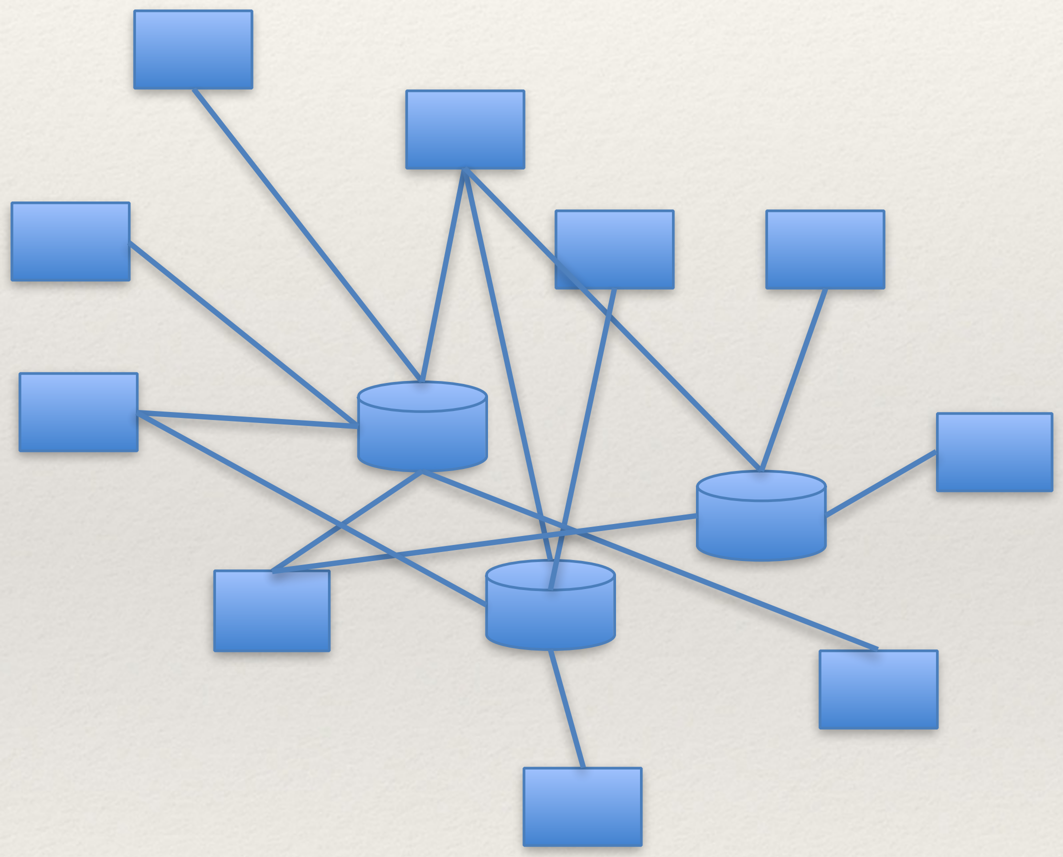
Amazon.com originally had a two-tier architecture. In order to scale they migrated to a service-oriented architecture consisting of hundreds of backend services. Several applications call these services including the applications that implement the Amazon.com website and the web service API. The Amazon.com website application calls 100-150 services to get the data that used to build a web page.

The auction site ebay.com also evolved from a monolithic architecture to a service-oriented architecture. The application tier consists of multiple independent applications. Each application implements the business logic for a specific function area such as buying or selling.”

http://microservices.io/patterns/microservices.html

Evolution

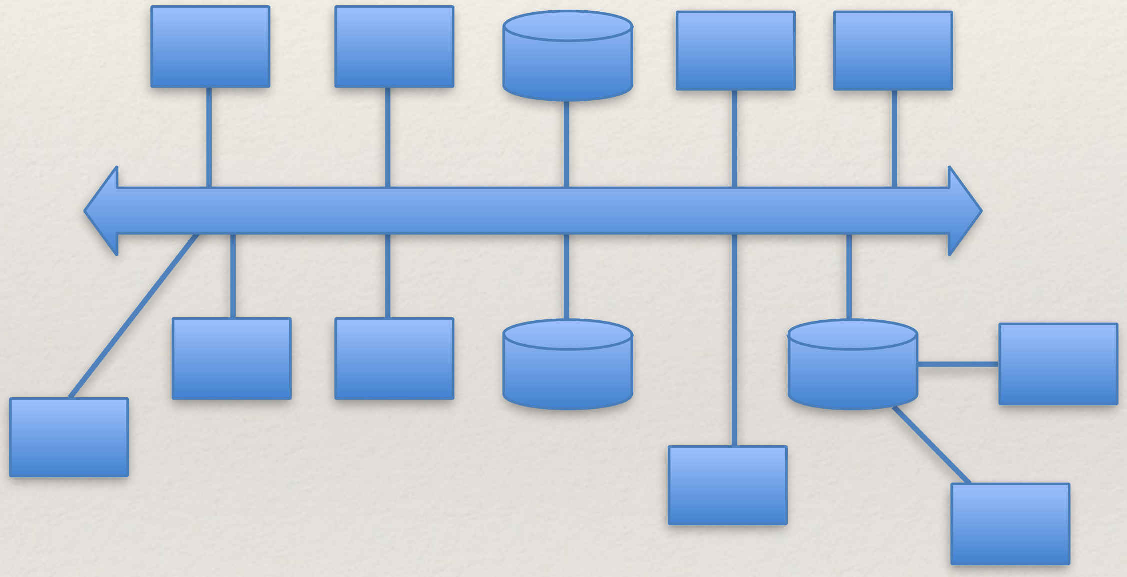
- we started with:



plain old 2-tier, DCE, CORBA, J(2)EE, .Net, …

Evolution.

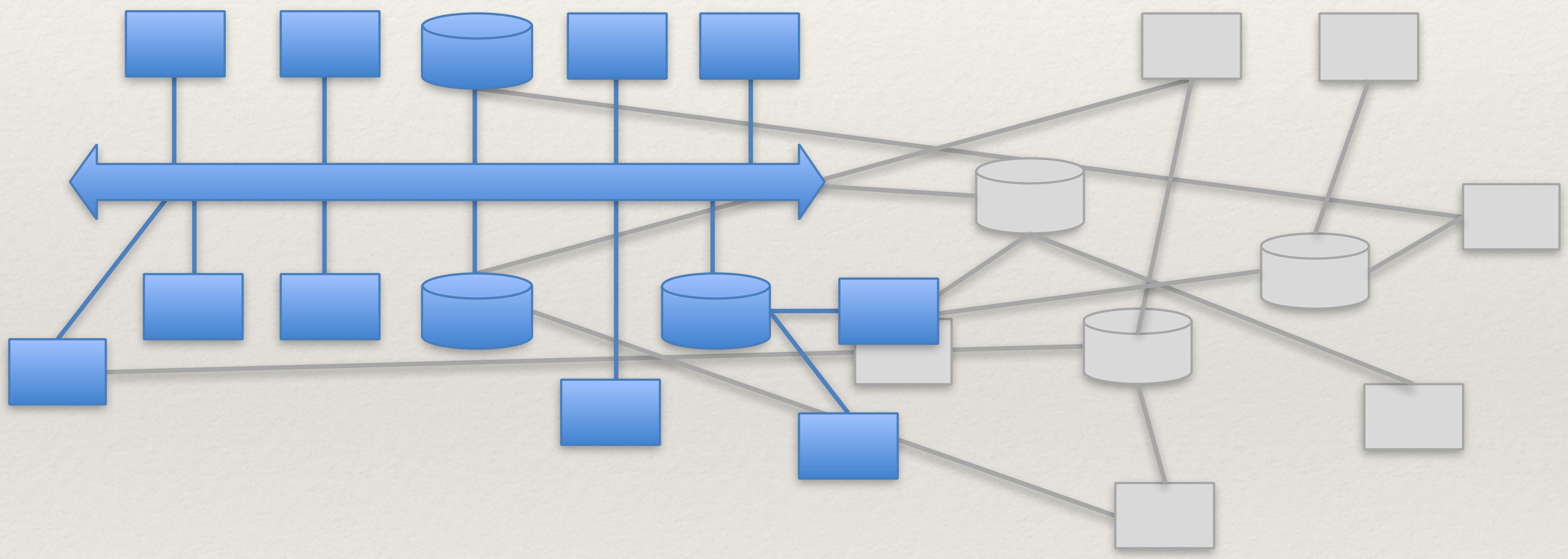
- then we paid lots of $ for the privilege of building this:



ESBs FTW…

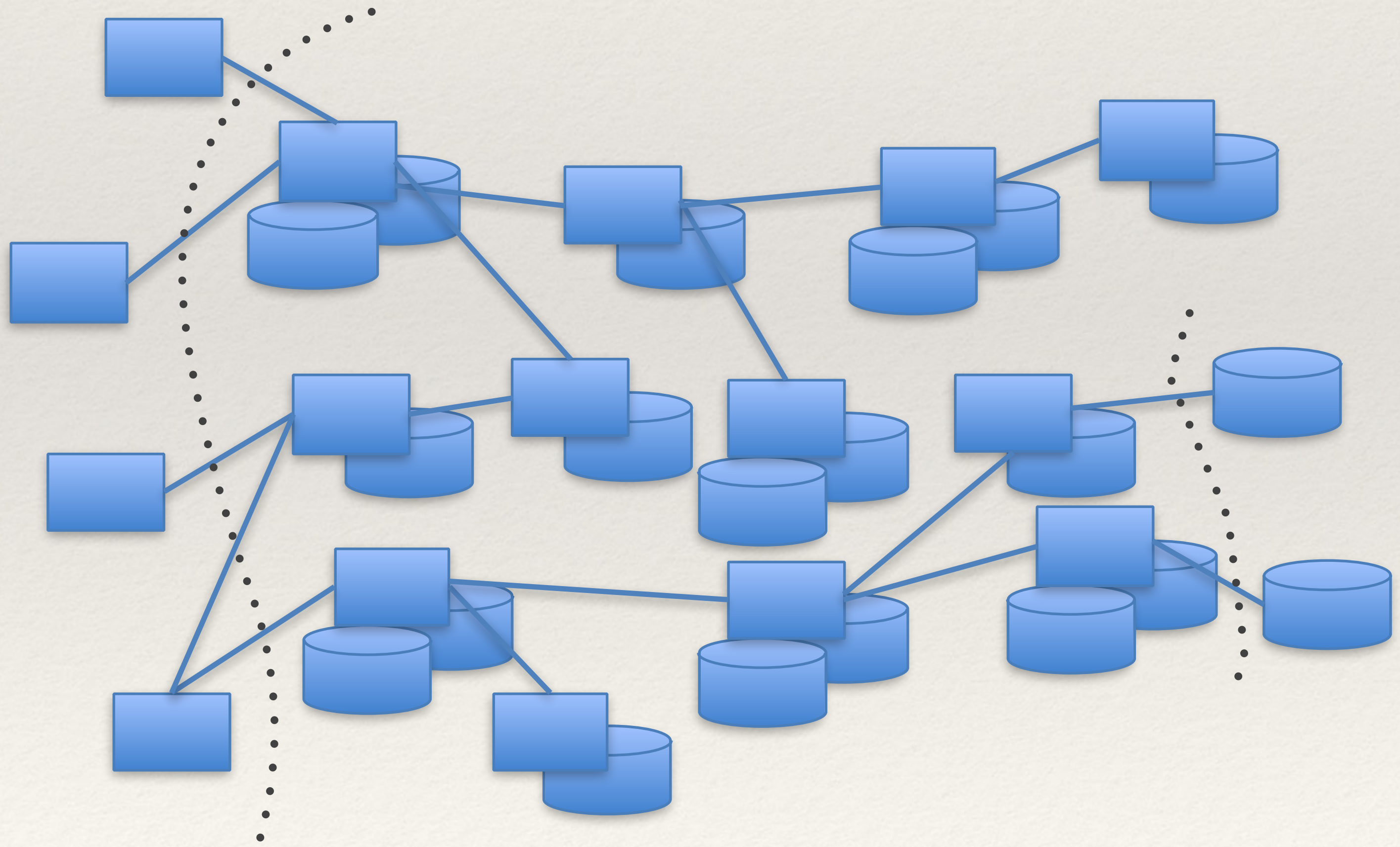
Evolution..

- ...and we probably really ended up with this:



Evolution…

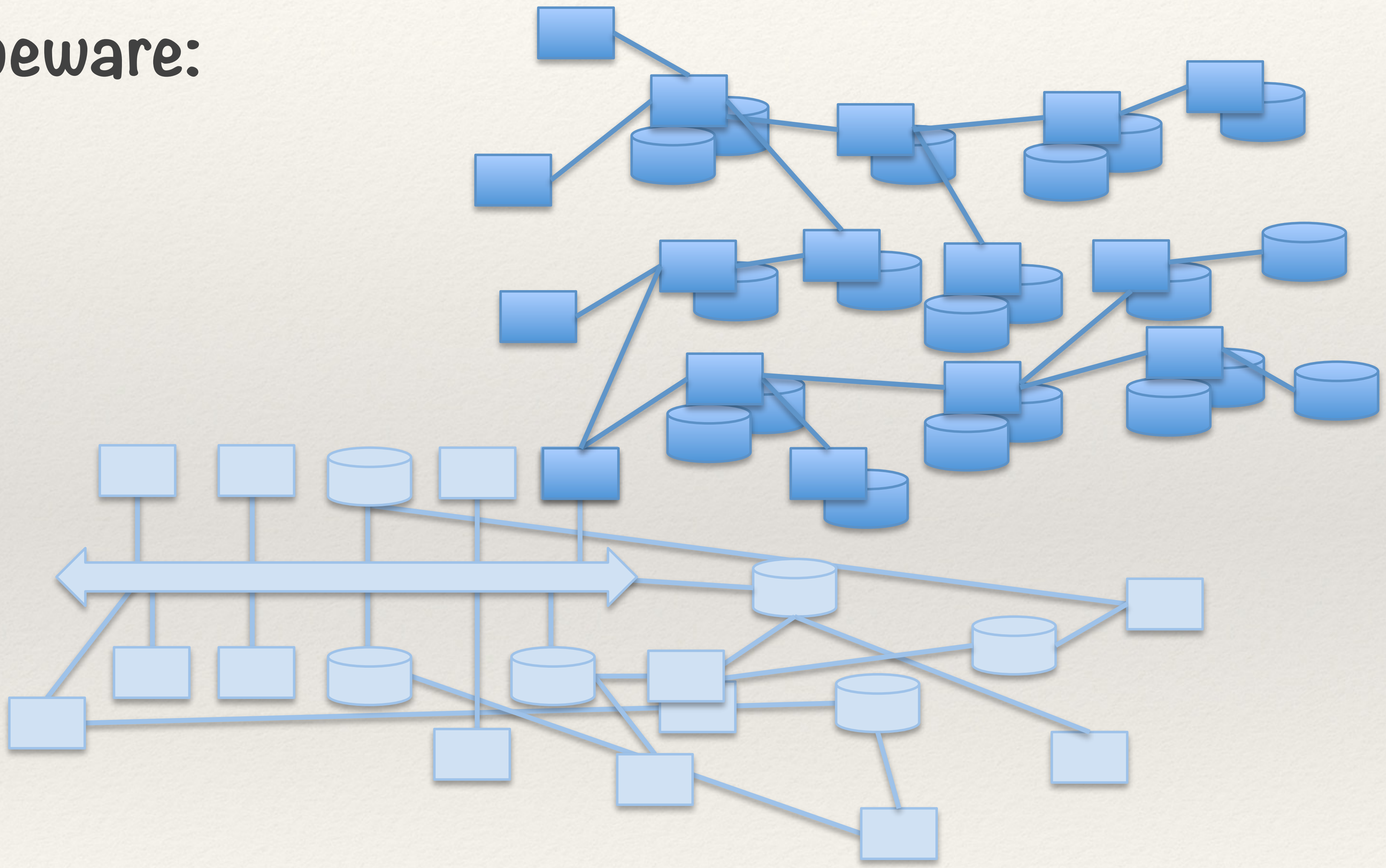
- so let's try this:
    - the Entity-Control-Boundary pattern



Ahhh…

Evolution….

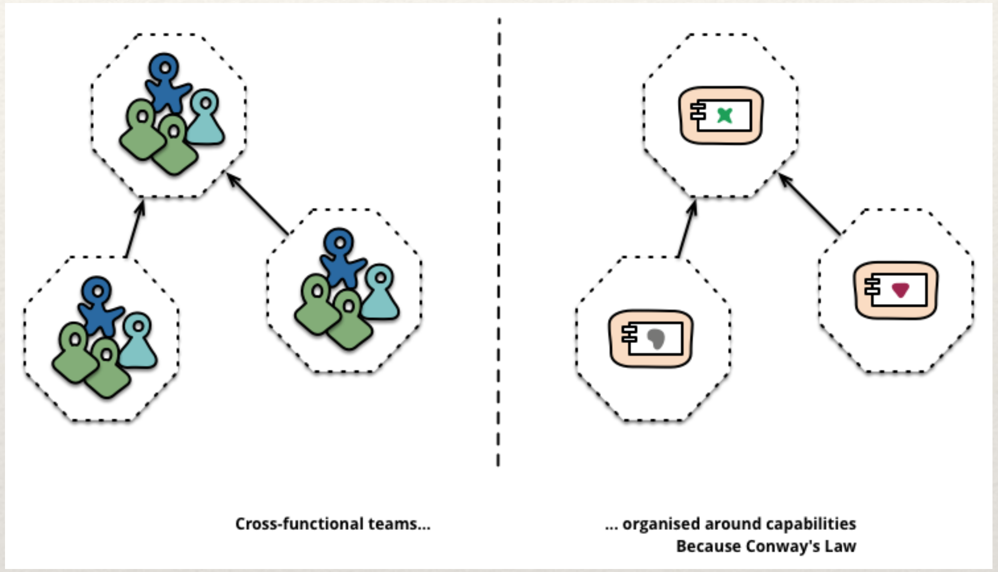
- but beware:



Ahhh…the march of progress!

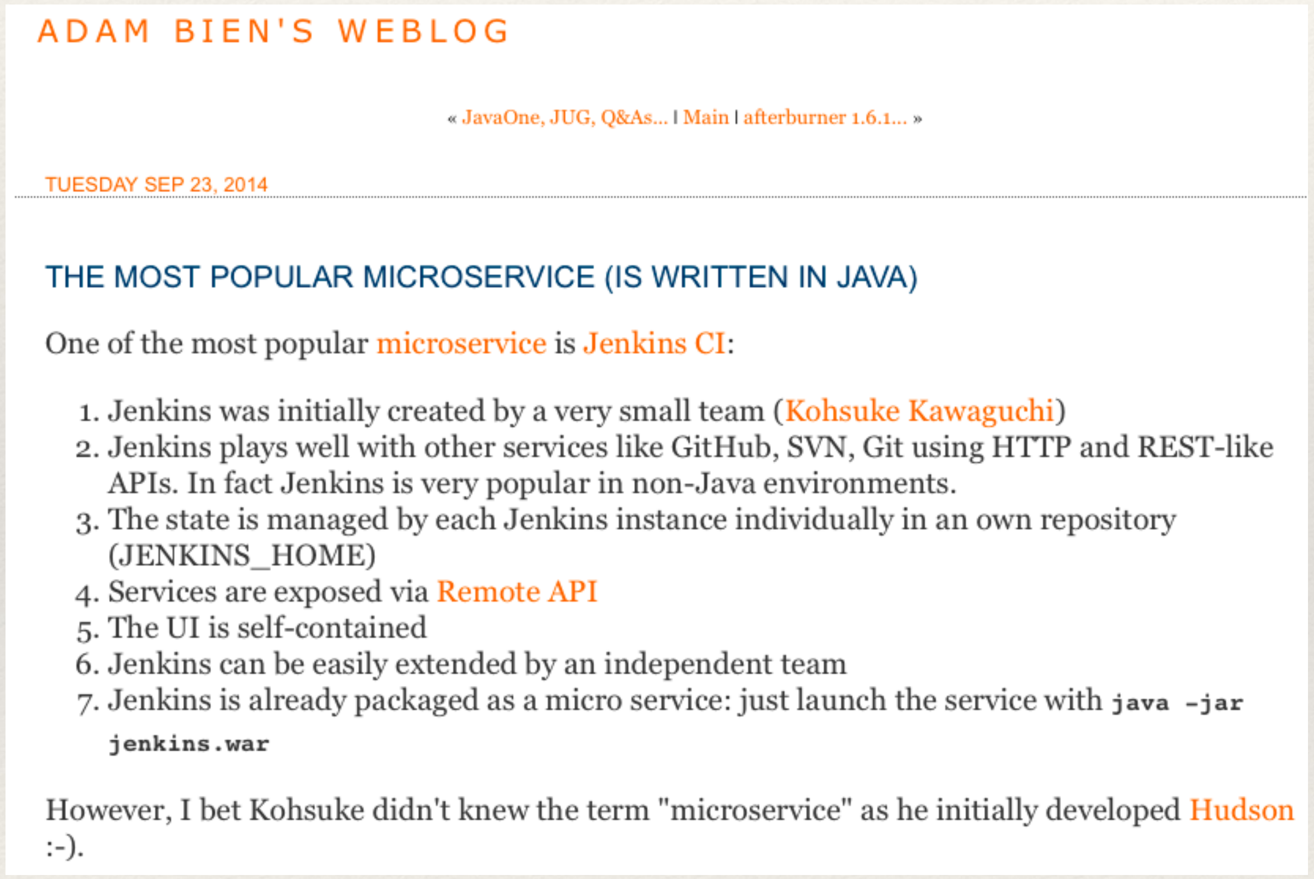
Is This For You(r Organisation)?

- a microservice may exist as/promote an organisational silo
  - silos! organisations like silos!
- conway's law
  - "...organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations."
- brooks' assertion
  - "...product quality is strongly affected by organization structure."
- houghson's warning
  - "Just don't buy too deeply into the idea that by getting the responsibilities of your software right, that you will somehow reduce the impact that all of that business dysfunction has on you as a software developer. Part of the maturation process for a company is cleaning up its business processes in parallel to cleaning up its software processes."
- THE uber-rant from Steve Yegge about Amazon/Google: https://plus.google.com/+RipRowan/posts/eVeouesvaVX



Yes!

- You're already using micro services



ADAM BIEN'S WEBLOG

« JavaOne, JUG, Q&As... | Main | afterburner 1.6.1... »

TUESDAY SEP 23, 2014

THE MOST POPULAR MICROSERVICE (IS WRITTEN IN JAVA)

One of the most popular microservice is Jenkins CI:

1. Jenkins was initially created by a very small team (Kohsuke Kawaguchi)
2. Jenkins plays well with other services like GitHub, SVN, Git using HTTP and REST-like APIs. In fact Jenkins is very popular in non-Java environments.
3. The state is managed by each Jenkins instance individually in an own repository (JENKINS_HOME)
4. Services are exposed via Remote API
5. The UI is self-contained
6. Jenkins can be easily extended by an independent team
7. Jenkins is already packaged as a micro service: just launch the service with `java -jar jenkins.war`

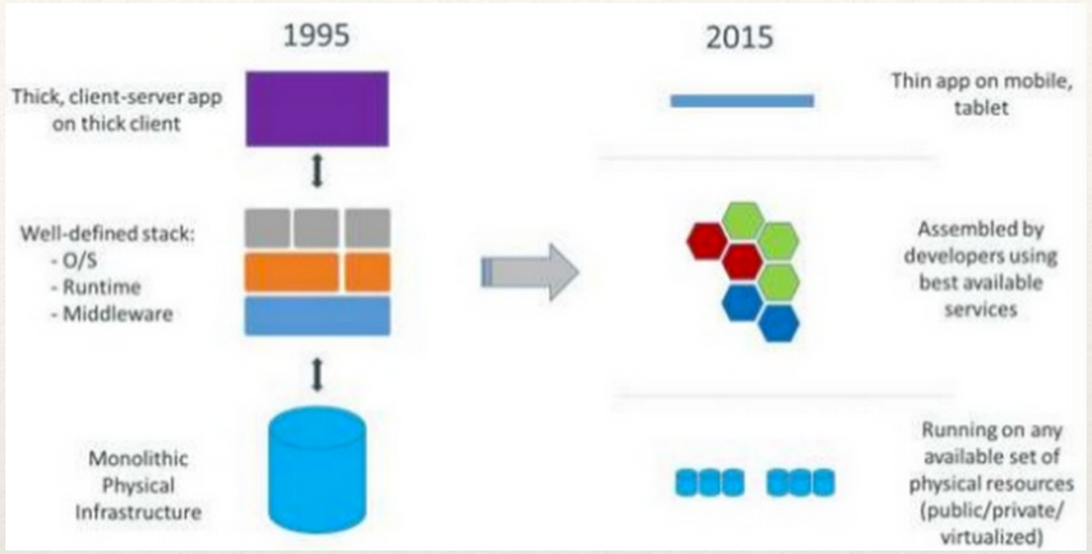
However, I bet Kohsuke didn't knew the term "microservice" as he initially developed Hudson :-).

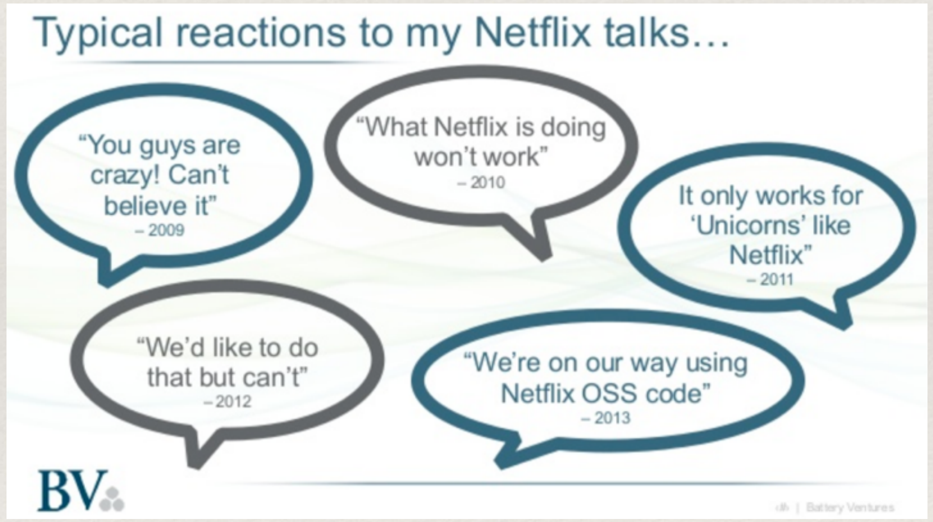
http://www.adam-bien.com/roller/abien/

- bitbucket, github, etc.

It's Time!

- cheaper/faster n/w and h/w
- the almighty Cloud
  - there's some amazing stuff out there now
- docker and like tools
- focus on developer ergonomics
  - adoption of DevOps
  - adoption of Agile and a new need for simplicity
  - ascendancy of REST/JSON
  - rise of micro frameworks…sinatra, spring boot, dropwizard, ratpack, node.js…
- pick the best developers { }, rather than the best available developer in scheme X


1995
2015
Thick, client-server app on thick client
Well-defined stack:
- O/S
- Runtime
- Middleware
Monolithic Physical Infrastructure
Thin app on mobile, tablet
Assembled by developers using best available services
Running on any available set of physical resources (public/private/ virtualized)


Typical reactions to my Netflix talks…

"You guys are crazy! Can't believe it" – 2009

"What Netflix is doing won't work" – 2010

It only works for 'Unicorns' like Netflix" – 2011

"We'd like to do that but can't" – 2012

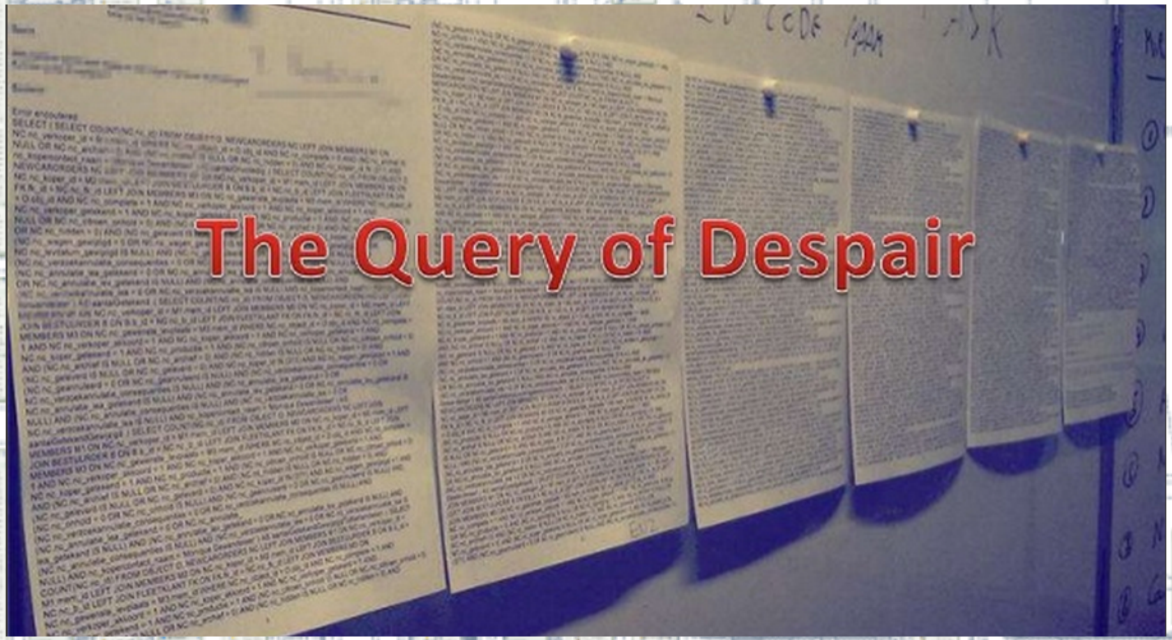
"We're on our way using Netflix OSS code" – 2013

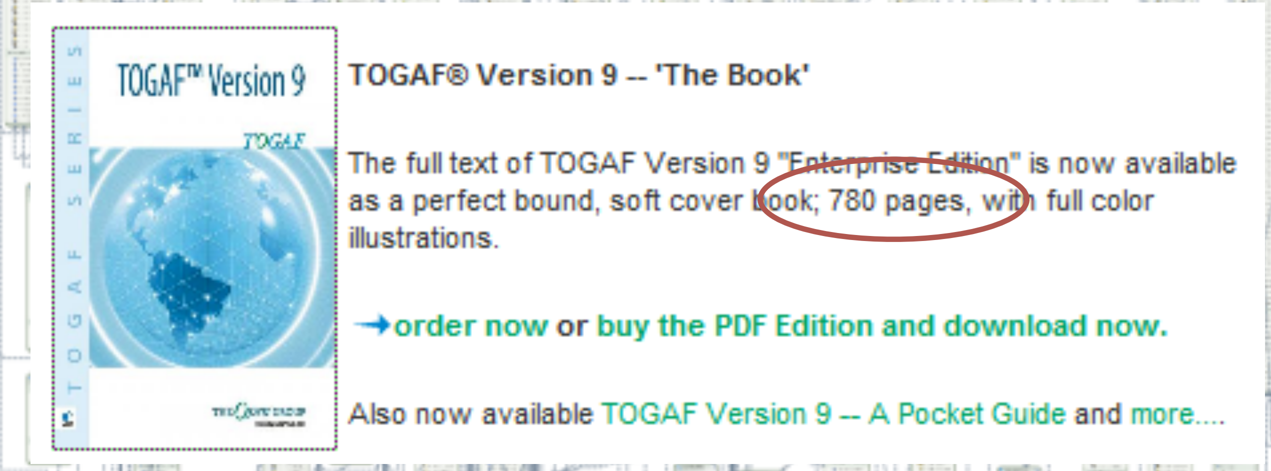
Adrian Crockford

It's Time!...

- IoT
    - lots of (permanently) connected devices
    - lots of small data packets
    - but resulting in big data
- rise of dynamic languages/features
    - ruby, groovy, python, etc. getting looked at more seriously
    - java, c# adopting more coolness
- but where are you on the scalability spectrum?
    - not everyone needs to be a netflix
    - not everyone can afford to be an amazon
    - affirmation therapy for the enterprise: sometimes monoliths are OK

Jumped Sharks?



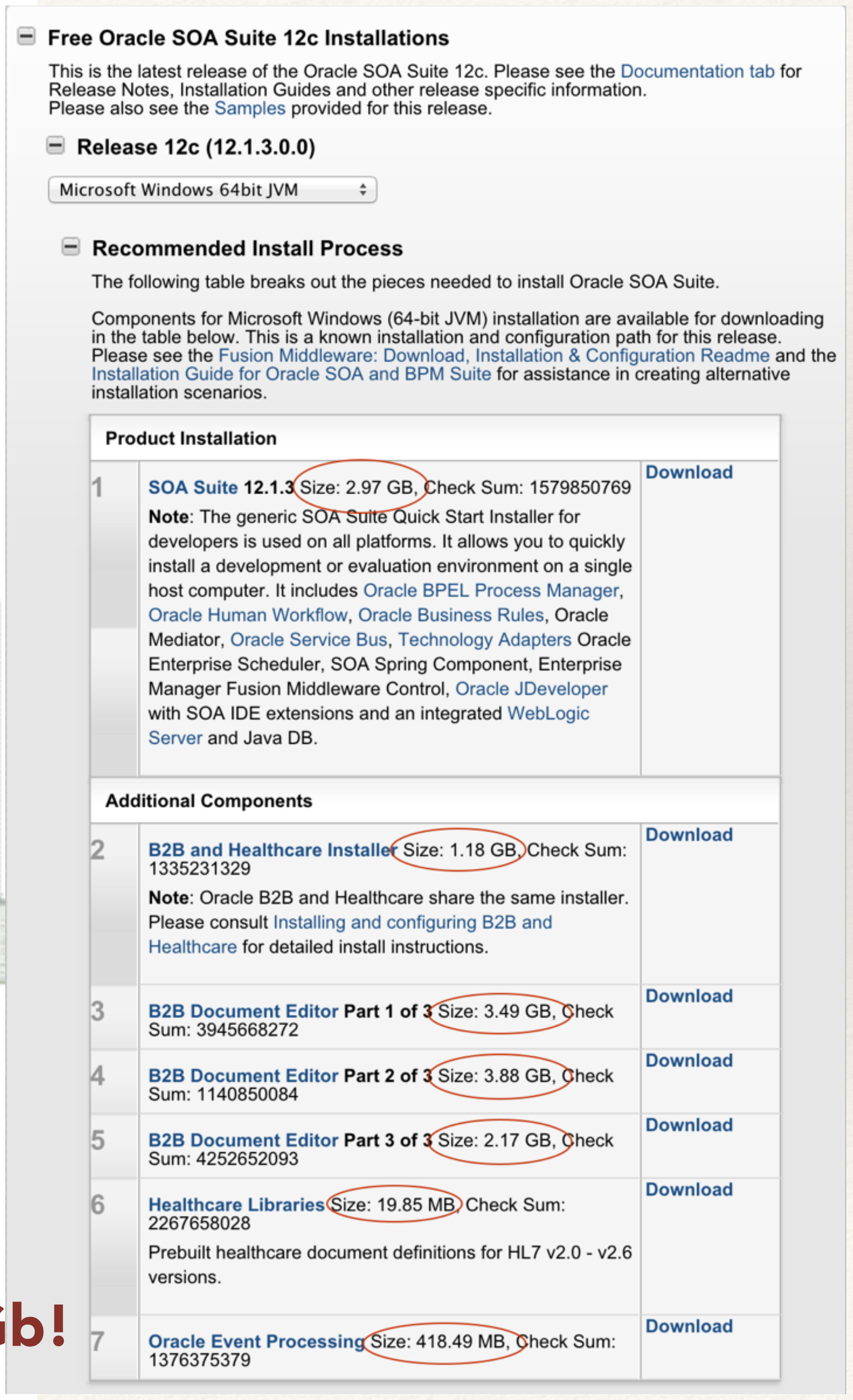


TOGAF® Version 9 -- 'The Book'

The full text of TOGAF Version 9 "Enterprise Edition" is now available as a perfect bound, soft cover book; 780 pages, with full color illustrations.

→order now or buy the PDF Edition and download now.

Also now available TOGAF Version 9 -- A Pocket Guide and more....



Free Oracle SOA Suite 12c Installations

This is the latest release of the Oracle SOA Suite 12c. Please see the Documentation tab for Release Notes, Installation Guides and other release specific information.
Please also see the Samples provided for this release.

Release 12c (12.1.3.0.0)

Microsoft Windows 64bit JVM

Recommended Install Process

The following table breaks out the pieces needed to install Oracle SOA Suite.

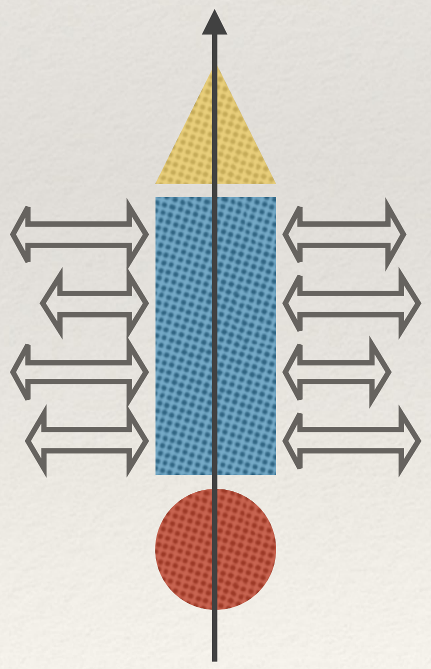
Components for Microsoft Windows (64-bit JVM) installation are available for downloading in the table below. This is a known installation and configuration path for this release. Please see the Fusion Middleware: Download, Installation & Configuration Readme and the Installation Guide for Oracle SOA and BPM Suite for assistance in creating alternative installation scenarios.

	Product Installation	
1	**SOA Suite 12.1.3** Size: 2.97 GB, Check Sum: 1579850769 **Note**: The generic SOA Suite Quick Start Installer for developers is used on all platforms. It allows you to quickly install a development or evaluation environment on a single host computer. It includes Oracle BPEL Process Manager, Oracle Human Workflow, Oracle Business Rules, Oracle Mediator, Oracle Service Bus, Technology Adapters Oracle Enterprise Scheduler, SOA Spring Component, Enterprise Manager Fusion Middleware Control, Oracle JDeveloper with SOA IDE extensions and an integrated WebLogic Server and Java DB.	Download
	Additional Components	
2	**B2B and Healthcare Installer** Size: 1.18 GB, Check Sum: 1335231329 **Note**: Oracle B2B and Healthcare share the same installer. Please consult Installing and configuring B2B and Healthcare for detailed install instructions.	Download
3	**B2B Document Editor Part 1 of 3** Size: 3.49 GB, Check Sum: 3945668272	Download
4	**B2B Document Editor Part 2 of 3** Size: 3.88 GB, Check Sum: 1140850084	Download
5	**B2B Document Editor Part 3 of 3** Size: 2.17 GB, Check Sum: 4252652093	Download
6	**Healthcare Libraries** Size: 19.85 MB, Check Sum: 2267658028 Prebuilt healthcare document definitions for HL7 v2.0 - v2.6 versions.	Download
7	**Oracle Event Processing** Size: 418.49 MB, Check Sum: 1376375379	Download

14.13Gb!

A Single Business Capability

- having a single business reason to change
- having minimal dependencies
- having minimal impact upon the rest of the estate
- accessing standardised facilities/cross-cutting concerns that are also micro services
    - security, configuration, health checks, caching, logging…



Data Ownership

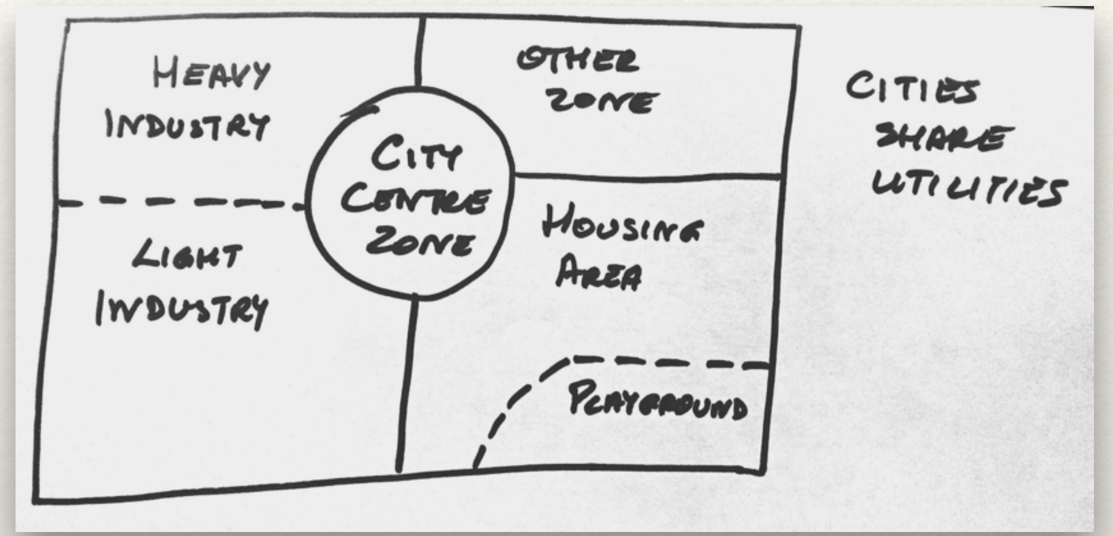
- encapsulates its own data: bounded data context
- look beyond ACID
  - become comfortable with eventual consistency
    - scheduled updates, event-driven propagation, caching
  - become comfortable with duplication
    - not a problem, per. se.: better to be more concerned with partitioning and amenability to substitution
    - but: hard to know which data is authoritative
- reporting, etc. made more troublesome
  - an opportunity to refactor the query of despair?

"There are two hard things in computer science: cache invalidation, naming things, and off-by-one errors."

Data Ownership...

- embrace multiple co-existent "canonical models"
  - canonical models ignore extant usecases and usage patterns
    - (esp. latter) can kill utilisation
  - There will never be only one. You will never control the whole world. Deal!
    - (shipping)product vs. (billing)product
    - (vendor-a)address vs. (vendor-b)address, (t_0)policy vs. (t_1)policy, etc.
- learn to love polyglot persistence

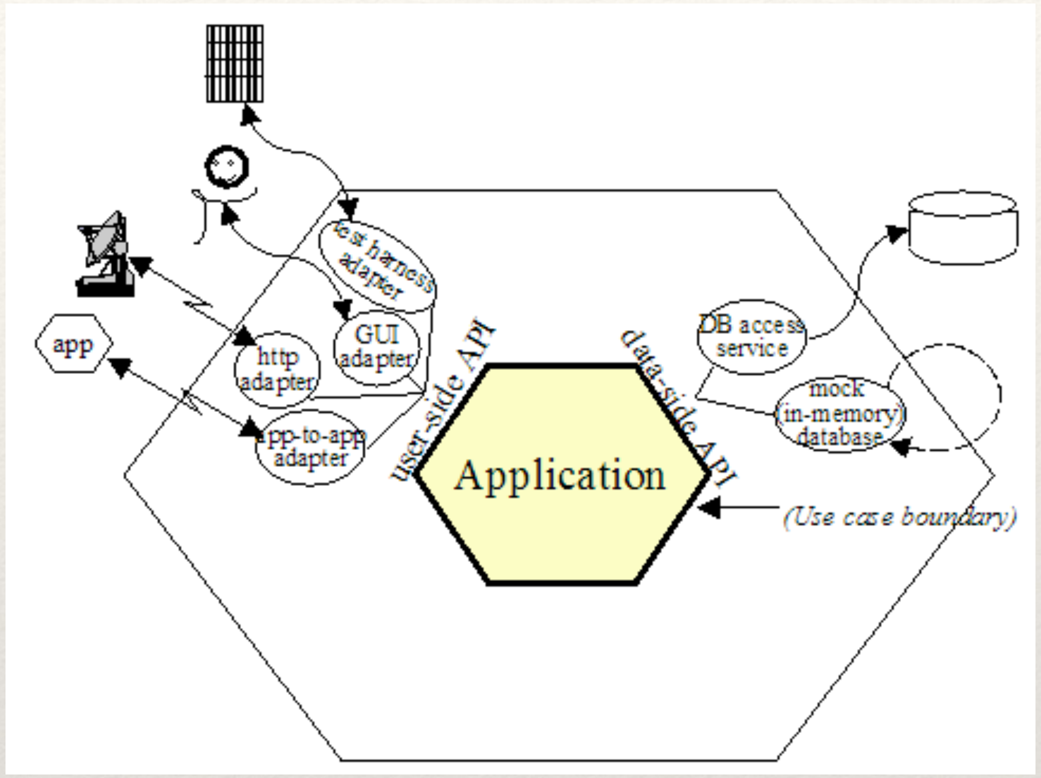
Service Interactions



- recognise common life cycles
- recognise locality of reference
  - "town planning" model
- asynchrony/event-driven important, say some
  - reduces blocking: increase throughput/power ratio
  - convert hard dependencies to soft ones: increases resilience
    - prevent 'dangelberries' leading to "slumbering herds"
  - contentious : eschew explicit choreography/orchestration
    - underlying feelings/fears: thar be vendors(== $$$, lock-in)/bottlenecks

Service Interactions...

- micro-level
  - strictly ports&adapters (Cockburn's 'hexagonal') architecture
  - service is oblivious to source or destination of request/response
- macro-level
  - adopt the Entity-Control-Boundary pattern



http://microservices.io/patterns/microservices.html

http://alistair.cockburn.us/Hexagonal+architecture

Be Of The Web, Not Behind The Web

- “standardise the gaps between the services”
  - standard protocols/APIs: HTTP, REST, simple MQ, protocol buffers, etc.
- technology of implementation is irrelevant
  - java, c#, PERL, PHP, ruby, groovy, RDBMS, NoSQL, flat files, etc., etc., etc.,...all OK
    - but: can does NOT imply should! don’t become a technology zoo...
  - small means easy to adopt “latest & greatest”
    - overcome fear
      - quick to deliver, quick to change
      - disposable services
      - fowler: design to be strangled out of existence once service is deemed ‘legacy’
    - ‘surgical’ updates; continuous delivery; YAGNI

Smart Endpoints & Dumb Pipes

- traditional: Enterprise Service Bus (ESB) products
  - Swiss army knife approach; sophisticated facilities for message routing, choreography, transformation, with complex protocols such as WS-Choreography or BPEL or orchestration by a central tool
  - Easy to sell: "just plug into this and all will be fine..."
  - Difficult to make perform
  - Difficult/expensive to push to the cloud
- microservices
  - aim to be as decoupled and as cohesive as possible—they own their own domain logic and act more as filters in the classical Unix sense: receiving a request, applying logic as appropriate and producing a response
  - no service locator furphy: simple choreography
  - no WS-* hell: simple RESTish protocols

KISS?

- a web application with an in-process back-end can be load-balanced much more simply than separate UI and service sites, without suffering the performance penalty of remote communication
    - also has fewer/easier failure modes, etc

http://genehughson.wordpress.com/2014/08/22/fears-for-tiers-do-you-need-a-service-layer/

Eight Fallacies Of Distributed Computing

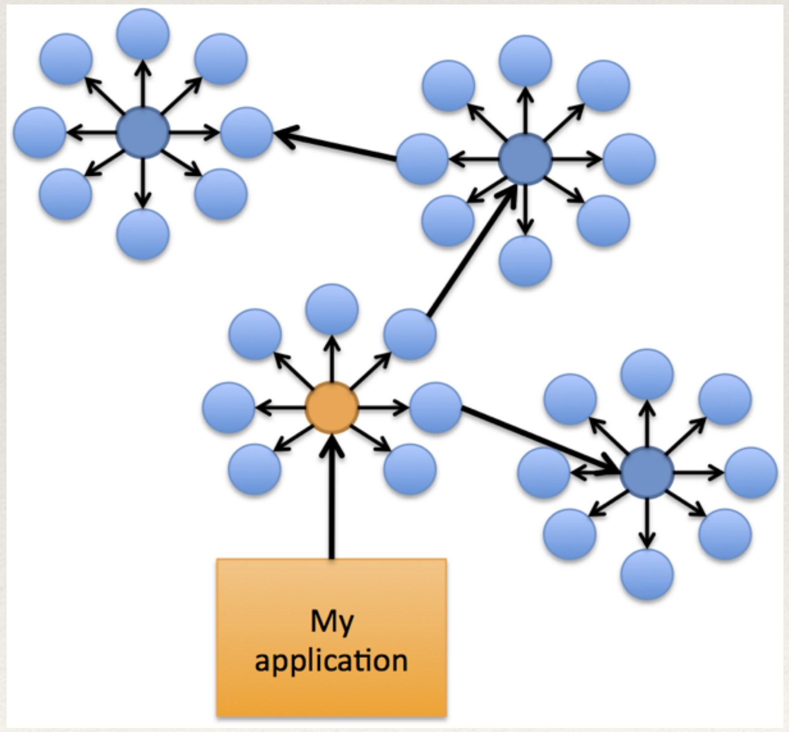
- L Peter Deutsch, Sun fellow, 1994
  - The network is reliable.
  - Latency is zero.
  - Bandwidth is infinite.
  - The network is secure.
  - Topology doesn't change.
  - There is one administrator.
  - Transport cost is zero.
  - The network is homogeneous.

These assumptions ultimately prove false, resulting either in the failure of the system, a substantial reduction in system scope, or in large, unplanned expenses required to redesign the system to meet its original goals.

http://www.rgoarchitects.com/Files/fallacies.pdf

Nanoservice Anti-Pattern

- immediate source of fear!
- cause: not accounting for all fallacies in problem space
  - making assumptions: infinite bandwidth, zero latency, no errors, perfect understanding, etc.…
- granularity so fine that overhead outweighs utility
  - communications, maintenance, monitoring, etc. all up
  - performance/utilisation down
- fragmented logic
  - complexity gets out of control



http://www.tigerteam.dk/2014/micro-services-its-not-only-the-size-that-matters-its-also-how-you-use-them-part-1/

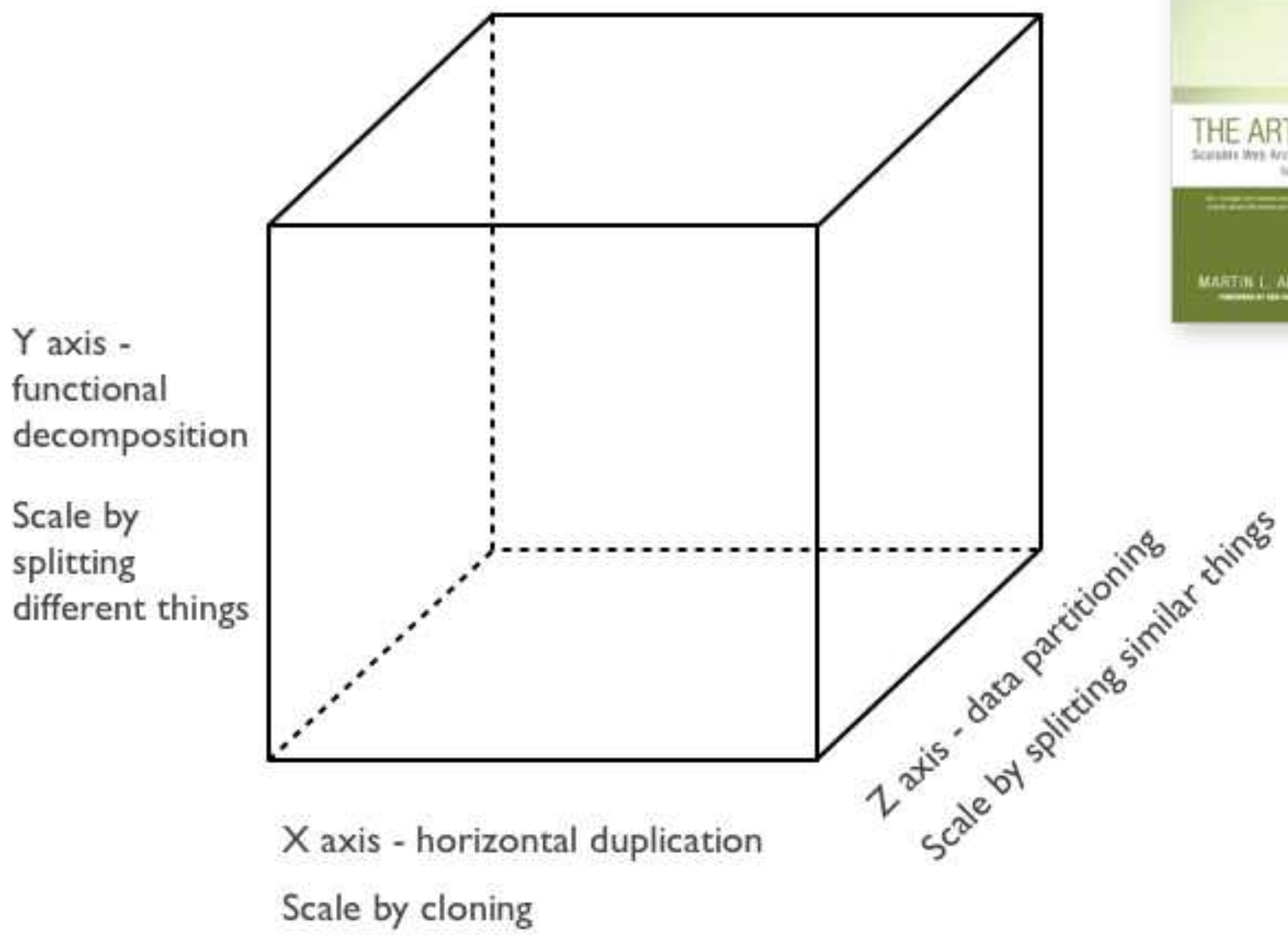
Fowler's first Law of Distributed Object Design: "don't distribute your objects."
http://martinfowler.com/articles/distributed-objects-microservices.html

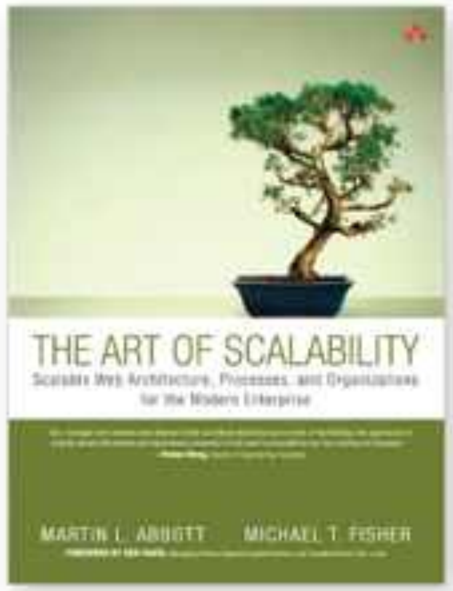
No! Just No!

```
String do(String arg)
```

Scalability Cube

3 dimensions to scaling

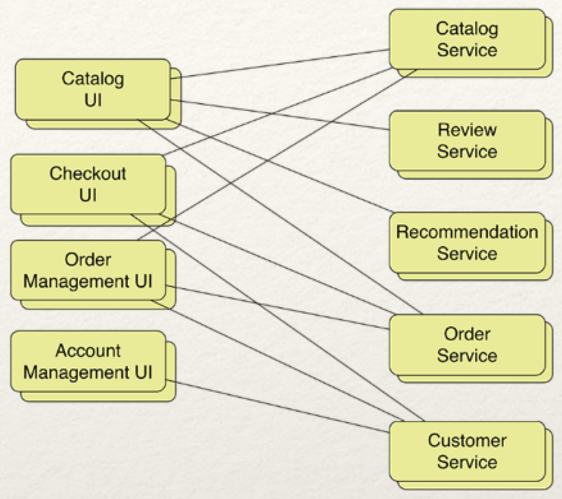

Y axis - functional decomposition
Scale by splitting different things
X axis - horizontal duplication
Scale by cloning
Z axis - data partitioning
Scale by splitting similar things


THE ART OF SCALABILITY

Scalable Web Architecture, Processes, and Organizations for the Modern Enterprise

MARTIN L. ABBOTT MICHAEL T. FISHER

Scalability Cube…

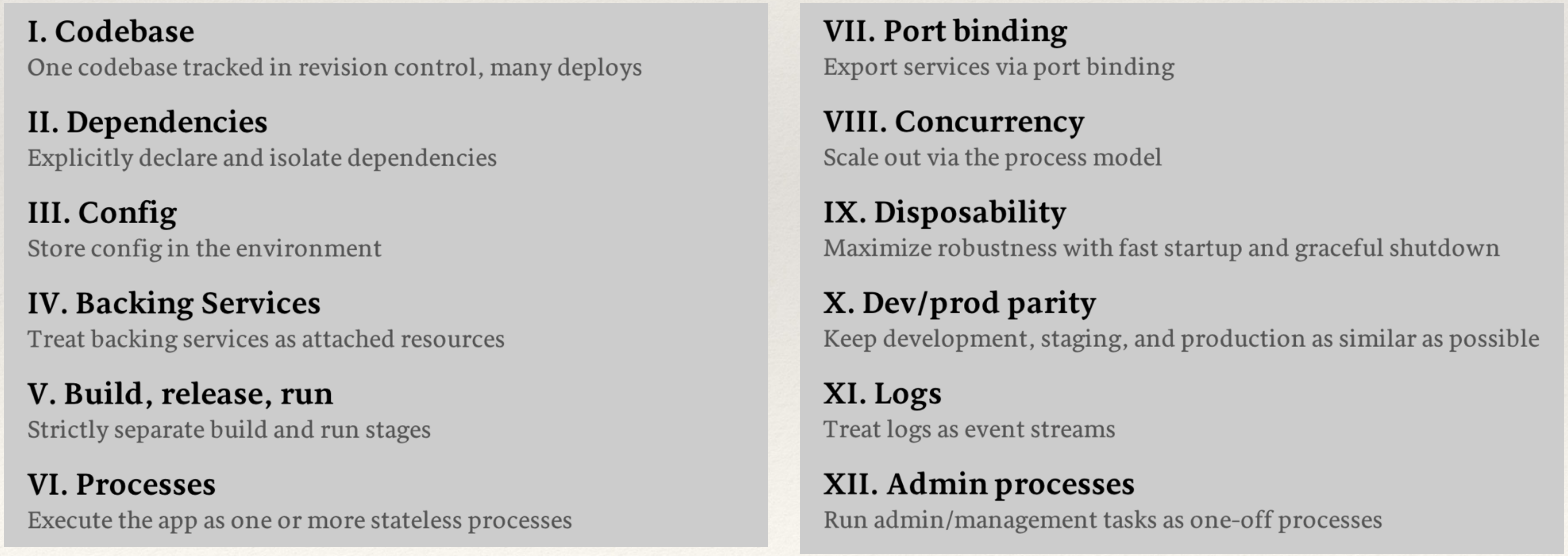


- X-axis scaling
  - run multiple copies of an application behind a load balancer
    - state, caching issues
- Y-axis scaling
  - splits the application into multiple, different services, each responsible for a single function
    - verb-based decomposition: define services that implement a single use case ('classic' SOA?)
    - decompose by noun: create services responsible for all operations related to a particular entity (REST SOA?)
- Z-axis scaling
  - each server runs an identical copy of the code against only a subset of the data
    - sharding/partitioning criteria, routing/aggregation issues
  - greater resiliency (tolerate partial failure), but greater complexity

"The key to being a cloud-native application is being scalable on all three of those axes", http://www.ekho.me/news/ekho-kent-langley-techops-article/

12-Factor App

- 12factor.net
    - liberate the micro services from your monoliths



I. Codebase
One codebase tracked in revision control, many deploys

II. Dependencies
Explicitly declare and isolate dependencies

III. Config
Store config in the environment

IV. Backing Services
Treat backing services as attached resources

V. Build, release, run
Strictly separate build and run stages

VI. Processes
Execute the app as one or more stateless processes

VII. Port binding
Export services via port binding

VIII. Concurrency
Scale out via the process model

IX. Disposability
Maximize robustness with fast startup and graceful shutdown

X. Dev/prod parity
Keep development, staging, and production as similar as possible

XI. Logs
Treat logs as event streams

XII. Admin processes
Run admin/management tasks as one-off processes

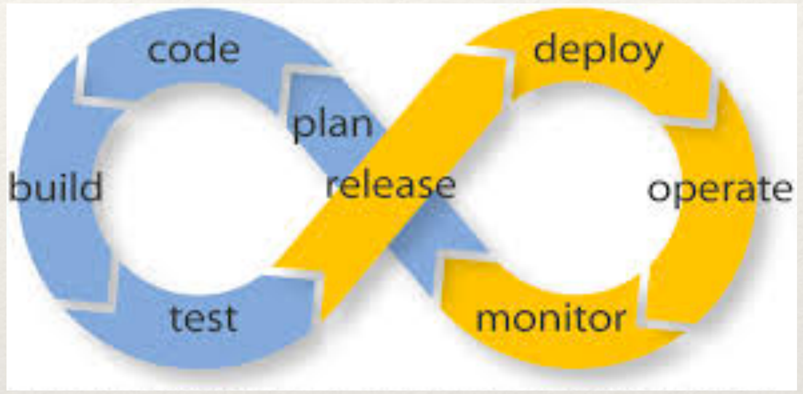
Danger! Will Robinson!

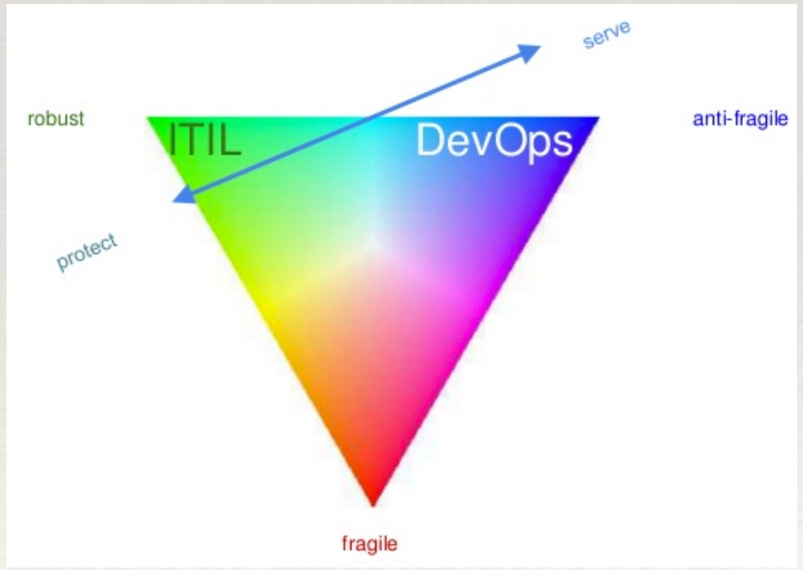
- duplication of effort
  - may need to be an Amazon to support overhead
- distributed systems are complex
  - proven too complex for some, historically
- asynchrony/choreography is difficult!
- need more sophisticated
  - infrastructure
  - (DevOps,management)teams

“There is a law of conservation of complexity in software. When we break up big things into small pieces we invariably push the complexity to their interaction.”

Dependent Upon DevOps

- substantial DevOps/cross-functional team skills required
  - continuous delivery
  - Amazon
    - "you build it, you run it"
    - "two-pizza team" projects
- templated deployments
  - make docker your new best friend
  - adopt to encapsulate learning, not to freeze stacks

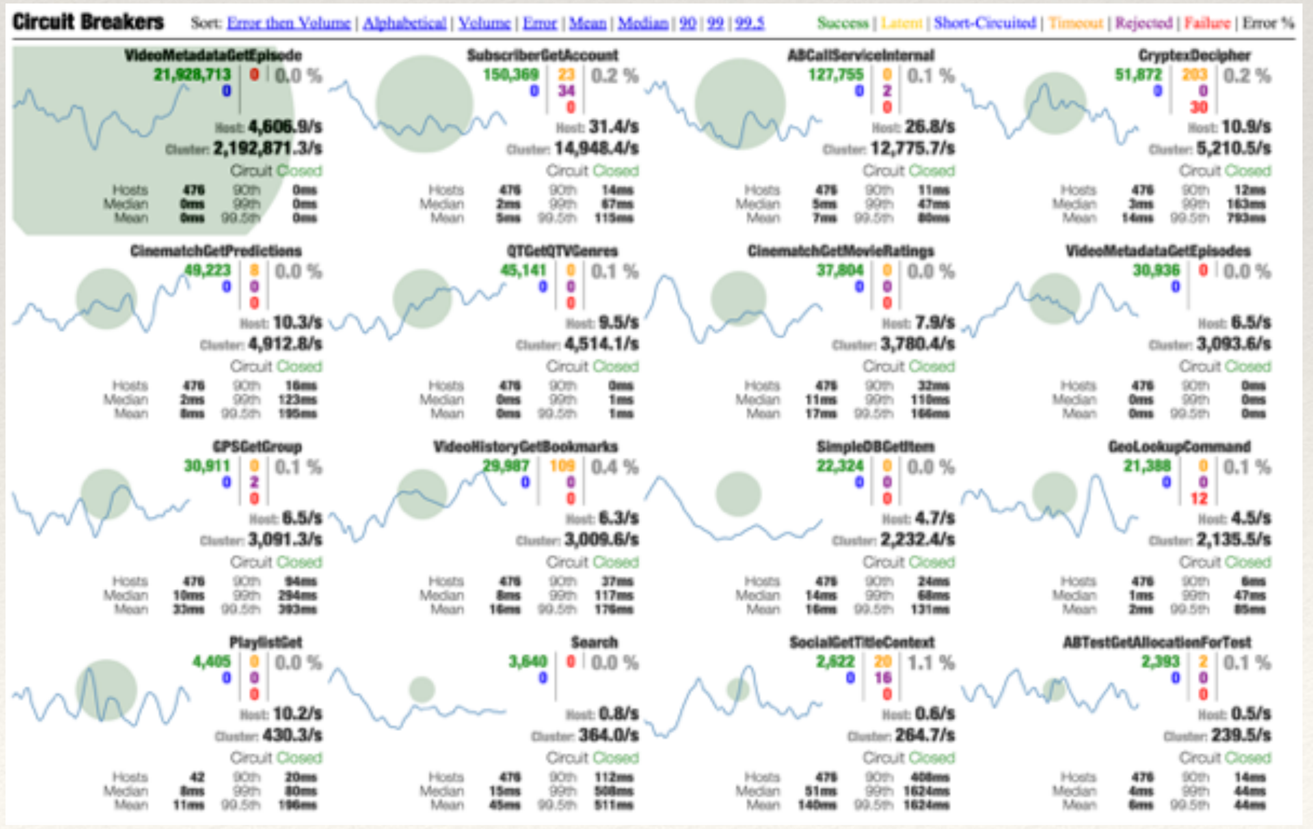




- **Fragile** = Humpty Dumpty
- **Robust** = A medieval castle
- **Anti-fragile** = The Borg collective

Monitoring, Not Testing

- you have to get MUCH better at monitoring...
    - Coda Hale: metrics, health checks, etc.
- ...and control; adaptive systems
    - Netflix: lots of services means having lots of 'canarys' and alternatives
        - "production is the best test environment"
    - Netflix: hysterix for "resilience engineering"
    - Netflix: circuit-breaker



```
Circuit Breakers   Sort: Error then Volume | Alphabetical | Volume | Error | Mean | Median | 90 | 99 | 99.5   Success | Latent | Short-Circuited | Timeout | Rejected | Failure | Error %

VideoMetadataGetEpisode
21,928,713 | 0 | 0.0 %
0
Host: 4,606.9/s
Cluster: 2,192,871.3/s
Circuit Closed
Hosts   476   90th   0ms
Median  0ms   99th   0ms
Mean    0ms   99.5th 0ms

SubscriberGetAccount
150,369 | 23 | 0.2 %
0 | 34
0
Host: 31.4/s
Cluster: 14,948.4/s
Circuit Closed
Hosts   476   90th   14ms
Median  2ms   99th   67ms
Mean    5ms   99.5th 115ms

ABCallServiceInternal
127,755 | 0 | 0.1 %
0 | 2
0
Host: 26.8/s
Cluster: 12,775.7/s
Circuit Closed
Hosts   476   90th   11ms
Median  5ms   99th   47ms
Mean    7ms   99.5th 80ms

CryptexDecipher
51,872 | 203 | 0.2 %
0 | 0
30
Host: 10.9/s
Cluster: 5,210.5/s
Circuit Closed
Hosts   476   90th   12ms
Median  3ms   99th   163ms
Mean    14ms  99.5th 793ms

CinematchGetPredictions
49,223 | 8 | 0.0 %
0 | 0
0
Host: 10.3/s
Cluster: 4,912.8/s
Circuit Closed
Hosts   476   90th   16ms
Median  2ms   99th   123ms
Mean    8ms   99.5th 195ms

QTGetQTVGenres
45,141 | 0 | 0.1 %
0 | 0
0
Host: 9.5/s
Cluster: 4,514.1/s
Circuit Closed
Hosts   476   90th   0ms
Median  0ms   99th   1ms
Mean    0ms   99.5th 1ms

CinematchGetMovieRatings
37,804 | 0 | 0.0 %
0 | 0
0
Host: 7.9/s
Cluster: 3,780.4/s
Circuit Closed
Hosts   476   90th   32ms
Median  11ms  99th   110ms
Mean    17ms  99.5th 166ms

VideoMetadataGetEpisodes
30,936 | 0 | 0.0 %
0
Host: 6.5/s
Cluster: 3,093.6/s
Circuit Closed
Hosts   476   90th   0ms
Median  0ms   99th   0ms
Mean    0ms   99.5th 0ms

GPSGetGroup
30,911 | 0 | 0.1 %
0 | 2
0
Host: 6.5/s
Cluster: 3,091.3/s
Circuit Closed
Hosts   476   90th   94ms
Median  10ms  99th   294ms
Mean    33ms  99.5th 393ms

VideoHistoryGetBookmarks
29,987 | 109 | 0.4 %
0 | 0
0
Host: 6.3/s
Cluster: 3,009.6/s
Circuit Closed
Hosts   476   90th   37ms
Median  8ms   99th   117ms
Mean    16ms  99.5th 176ms

SimpleDBGetItem
22,324 | 0 | 0.0 %
0 | 0
0
Host: 4.7/s
Cluster: 2,232.4/s
Circuit Closed
Hosts   476   90th   24ms
Median  14ms  99th   68ms
Mean    16ms  99.5th 131ms

GeoLookupCommand
21,388 | 0 | 0.1 %
0 | 0
12
Host: 4.5/s
Cluster: 2,135.5/s
Circuit Closed
Hosts   476   90th   6ms
Median  1ms   99th   47ms
Mean    2ms   99.5th 85ms

PlaylistGet
4,405 | 0 | 0.0 %
0 | 0
0
Host: 10.2/s
Cluster: 430.3/s
Circuit Closed
Hosts   42    90th   20ms
Median  8ms   99th   80ms
Mean    11ms  99.5th 196ms

Search
3,640 | 0 | 0.0 %
0
Host: 0.8/s
Cluster: 364.0/s
Circuit Closed
Hosts   476   90th   112ms
Median  15ms  99th   508ms
Mean    45ms  99.5th 511ms

SocialGetTitleContext
2,622 | 20 | 1.1 %
0 | 16
0
Host: 0.6/s
Cluster: 264.7/s
Circuit Closed
Hosts   476   90th   408ms
Median  51ms  99th   1624ms
Mean    140ms 99.5th 1624ms

ABTestGetAllocationForTest
2,393 | 2 | 0.1 %
0 | 0
0
Host: 0.5/s
Cluster: 239.5/s
Circuit Closed
Hosts   476   90th   14ms
Median  4ms   99th   44ms
Mean    6ms   99.5th 44ms
```

Monitoring, Not Testing…

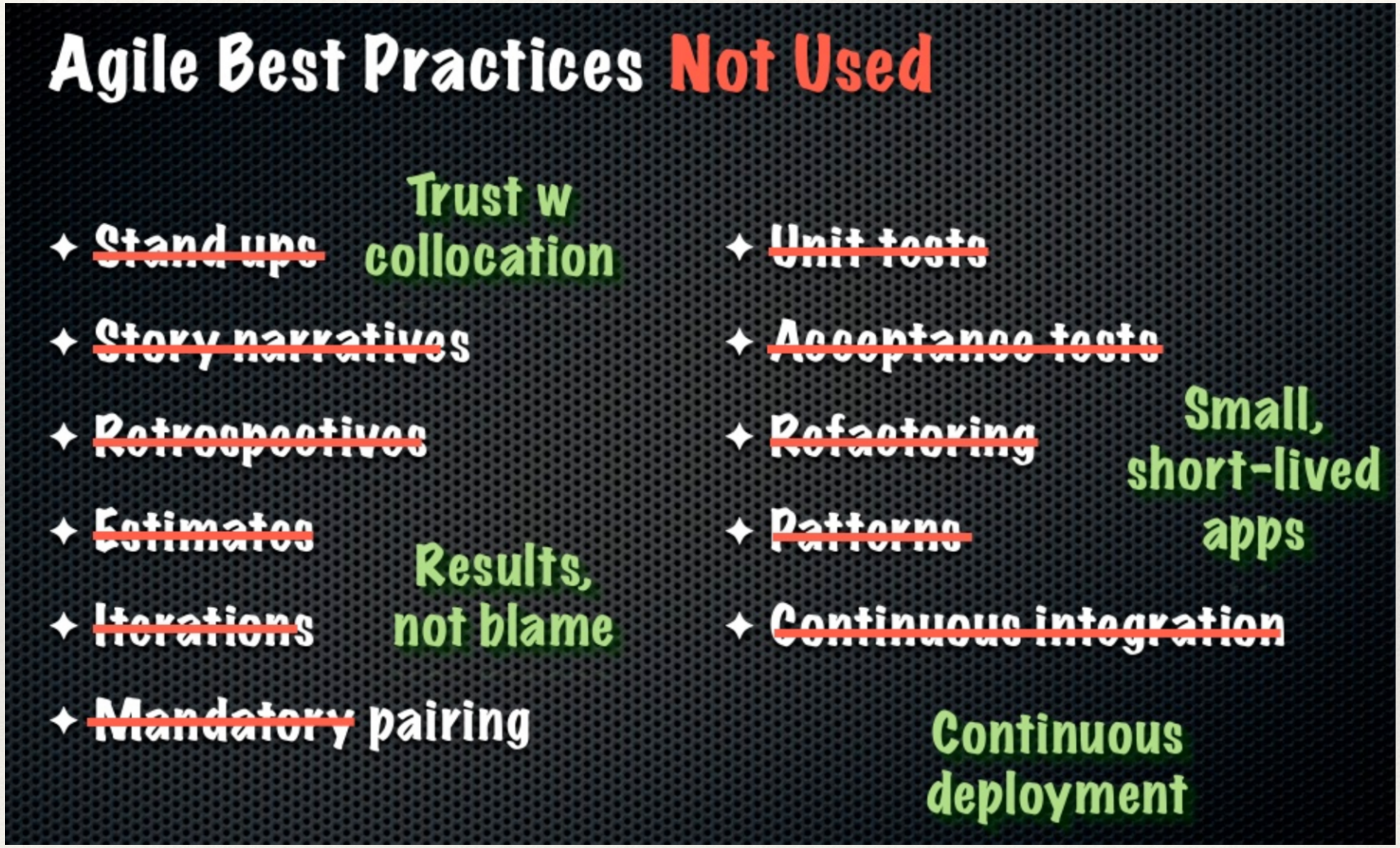
- design for failure, not to avoid it ('cos you can't avoid it)
  - Netflix: simian army/chaos monkey
    - and chaos Gorilla

"Failures happen, and they inevitably happen when least desired. If your application can't tolerate a system failure would you rather find out by being paged at 3am or after you are in the office having already had your morning coffee?"

— https://github.com/Netflix/SimianArmy/wiki/Chaos-Monkey

"…it may well be the case that the only thing still functioning in the server is the little component that knows how to say "I'm fine, roger roger, over and out" in a cheery droid voice." — Yegge

Whither Agile?



Agile Best Practices Not Used

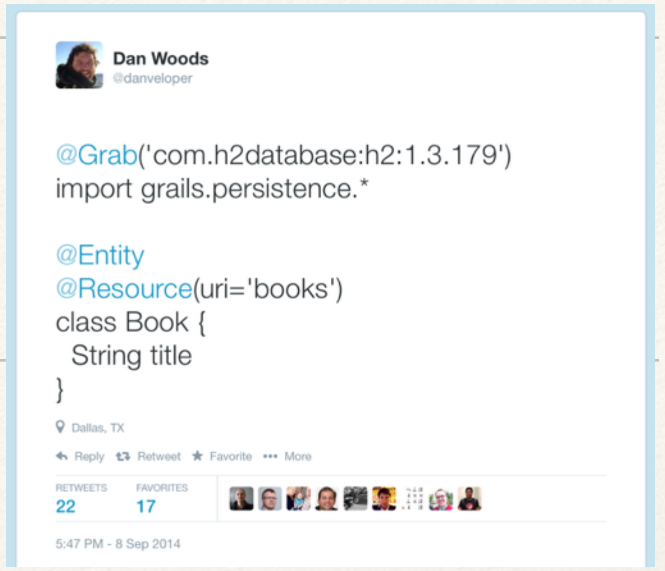
- ~~Stand ups~~ Trust w collocation
- ~~Story narratives~~
- ~~Retrospectives~~
- ~~Estimates~~
- ~~Iterations~~ Results, not blame
- ~~Mandatory~~ pairing
- ~~Unit tests~~
- ~~Acceptance tests~~
- ~~Refactoring~~ Small, short-lived apps
- ~~Patterns~~
- ~~Continuous integration~~

Continuous deployment

http://www.slideshare.net/fredgeorge/micro-service-architecure

"...bedrock principles of Agile have been rendered unnecessary, something that... surprised us."

Technology


Dan Woods
@danveloper

```
@Grab('com.h2database:h2:1.3.179')
import grails.persistence.*

@Entity
@Resource(uri='books')
class Book {
  String title
}
```

Dallas, TX

tweetable full data-driven rest application with Grails 3

- a plethora!
- frameworks
  - wrap your actual code in a just-good-enough communication layer, plus support tooling
    - ratpack, dropwizard, spring boot, vert.x, node.js, sinatra, gilliam, etc.
- deployment
  - flockport, docker, puppet, vagrant, ansible, etc.
  - fabric8, http://fabric8.io/gitbook/overview.html

The Rise of the Full-Stack Architect
http://dejanglozic.com/2014/05/12/the-rise-of-the-full-stack-architect/

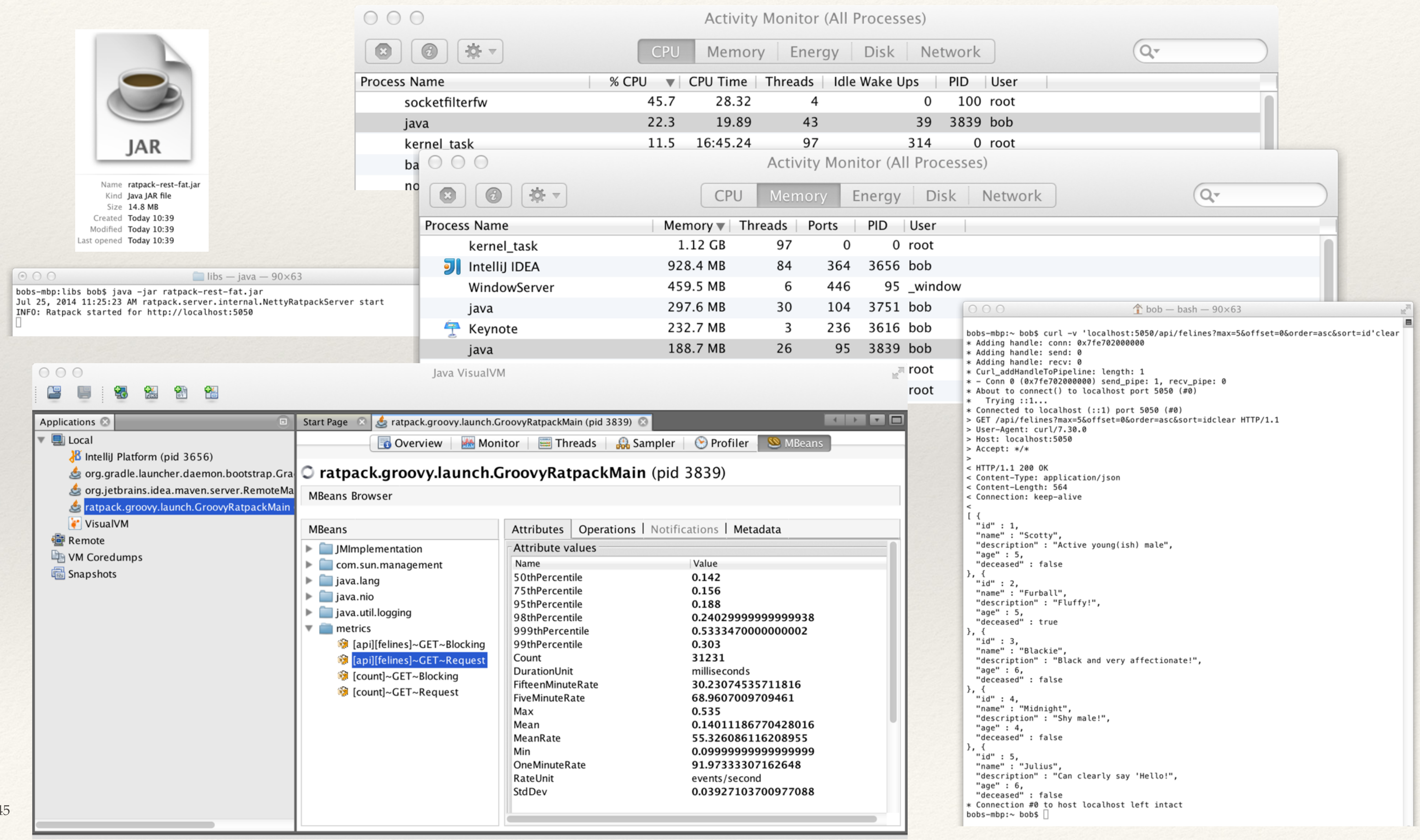
Ratpack REST-Style MicroService

```
ratpack {
    bindings {
        add new JacksonModule(); add new AbstractModule() { … bind(FelineStore).in(SINGLETON) … };
        add new CodaHaleMetricsModule().jmx().console()

       init { FelineStore felineStore -> … }
    }

    handlers { FelineStore datastore ->
        get("api/felines/count") {
            blocking { datastore.size() }
                .then { render json(count: it) }
        }
        handler("api/felines/:id?") {
            def id = pathTokens.id?.safeParseAsLong()
            byMethod {
                get {
                    blocking { id ? datastore.get(id) : datastore.list(request.queryParams) }
                       .then { if (it != null) render json(it) else clientError(404) }
                    }
                }
                post {
                    blocking { def f = parse Feline; datastore.add(f) }
                        .then { render json(it) }
                }
                delete {
                    blocking { id ? datastore.delete(id) : null }
                        .then { clientError(it ? 204 : 404) }
                }
                put {
                    blocking { def f = parse Feline; f.id = id; f.id ? datastore.update(f) : null }
                    .then { clientError(it ? 204 : 404) }
                }
            }
        }
        get {
            render groovyTemplate("grid.html", title: "AngularJS + Ng-grid + Bootstrap + Ratpack REST")
        }

        assets "public"
    }
}
```

Ratpack

Simple, lean & powerful HTTP apps

Ratpack MicroService...



Name ratpack-rest-fat.jar
Kind Java JAR file
Size 14.8 MB
Created Today 10:39
Modified Today 10:39
Last opened Today 10:39

libs — java — 90×63

```
bobs-mbp:libs bob$ java -jar ratpack-rest-fat.jar
Jul 25, 2014 11:25:23 AM ratpack.server.internal.NettyRatpackServer start
INFO: Ratpack started for http://localhost:5050
```

Activity Monitor (All Processes)

Process Name	% CPU	CPU Time	Threads	Idle Wake Ups	PID	User
socketfilterfw	45.7	28.32	4	0	100	root
java	22.3	19.89	43	39	3839	bob
kernel_task	11.5	16:45.24	97	314	0	root
ba						
no						

Activity Monitor (All Processes)

Process Name	Memory	Threads	Ports	PID	User
kernel_task	1.12 GB	97	0	0	root
IntelliJ IDEA	928.4 MB	84	364	3656	bob
WindowServer	459.5 MB	6	446	95	_window
java	297.6 MB	30	104	3751	bob
Keynote	232.7 MB	3	236	3616	bob
java	188.7 MB	26	95	3839	bob
					root
					root

bob — bash — 90×63

```
bobs-mbp:~ bob$ curl -v 'localhost:5050/api/felines?max=5&offset=0&order=asc&sort=id'clear
* Adding handle: conn: 0x7fe702000000
* Adding handle: send: 0
* Adding handle: recv: 0
* Curl_addHandleToPipeline: length: 1
* - Conn 0 (0x7fe702000000) send_pipe: 1, recv_pipe: 0
* About to connect() to localhost port 5050 (#0)
*   Trying ::1...
* Connected to localhost (::1) port 5050 (#0)
> GET /api/felines?max=5&offset=0&order=asc&sort=idclear HTTP/1.1
> User-Agent: curl/7.30.0
> Host: localhost:5050
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: application/json
< Content-Length: 564
< Connection: keep-alive
<
[ {
  "id" : 1,
  "name" : "Scotty",
  "description" : "Active young(ish) male",
  "age" : 5,
  "deceased" : false
}, {
  "id" : 2,
  "name" : "Furball",
  "description" : "Fluffy!",
  "age" : 5,
  "deceased" : true
}, {
  "id" : 3,
  "name" : "Blackie",
  "description" : "Black and very affectionate!",
  "age" : 6,
  "deceased" : false
}, {
  "id" : 4,
  "name" : "Midnight",
  "description" : "Shy male!",
  "age" : 4,
  "deceased" : false
}, {
  "id" : 5,
  "name" : "Julius",
  "description" : "Can clearly say 'Hello!",
  "age" : 6,
  "deceased" : false
* Connection #0 to host localhost left intact
bobs-mbp:~ bob$ 
```

Java VisualVM

Applications
- Local
  - IntelliJ Platform (pid 3656)
  - org.gradle.launcher.daemon.bootstrap.Gra
  - org.jetbrains.idea.maven.server.RemoteMa
  - ratpack.groovy.launch.GroovyRatpackMain
  - VisualVM
- Remote
- VM Coredumps
- Snapshots

Start Page | ratpack.groovy.launch.GroovyRatpackMain (pid 3839)

Overview | Monitor | Threads | Sampler | Profiler | MBeans

ratpack.groovy.launch.GroovyRatpackMain (pid 3839)

MBeans Browser

MBeans
- JMImplementation
- com.sun.management
- java.lang
- java.nio
- java.util.logging
- metrics
  - [api][felines]~GET~Blocking
  - [api][felines]~GET~Request
  - [count]~GET~Blocking
  - [count]~GET~Request

Attributes | Operations | Notifications | Metadata

Attribute values

Name	Value
50thPercentile	0.142
75thPercentile	0.156
95thPercentile	0.188
98thPercentile	0.24029999999999938
999thPercentile	0.5333470000000002
99thPercentile	0.303
Count	31231
DurationUnit	milliseconds
FifteenMinuteRate	30.23074535711816
FiveMinuteRate	68.9607007009461
Max	0.535
Mean	0.14011186770428016
MeanRate	55.326086116208955
Min	0.099999999999999999
OneMinuteRate	91.97333307162648
RateUnit	events/second
StdDev	0.03927103700977088

Cool Places To Visit

- Tools
  - Netflix Open Source Software Center: http://netflix.github.io/
  - Soundcloud developer site: https://developers.soundcloud.com/
  - Twitter blogs: https://blog.twitter.com/developer
  - Amazon developer tools: https://aws.amazon.com/developertools/
- Links
  - http://wayfinder.co/pathways/53536427f7040a11002ae407/a-field-guide-to-microservices-april-2014-edition
  - http://blog.arkency.com/2014/07/microservices-72-resources/
  - http://www.mattstine.com/microservices
  - http://microservices.io/
  - http://blog.devopsguys.com/2013/07/17/devops-antifragility-and-the-borg-collective/
  - https://blog.yourkarma.com/building-microservices-at-karma
  - http://www.tigerteam.dk/2014/micro-services-its-not-only-the-size-that-matters-its-also-how-you-use-them-part-1/

Final Thought

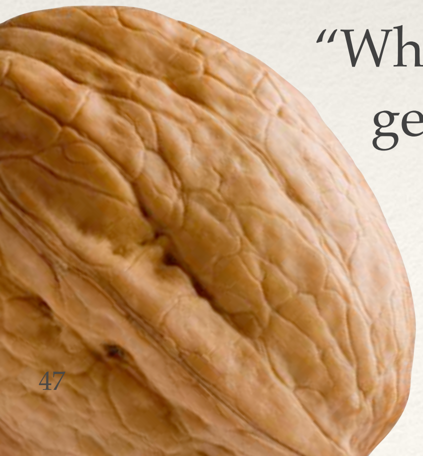
"The Empire has always been a realm of colossal resources. They've calculated everything in planets, in stellar systems, in whole sectors of the Galaxy. Their generators are gigantic because they thought in gigantic fashion.

"But we—we, our little Foundation, our single world almost without metallic resources—have had to work with brute economy. Our generators have had to be the size of our thumb, because it was all the metal we could afford. We had to develop new techniques and new methods—techniques and methods the Empire can't follow because they have degenerated past the stage where they can make any vital scientific advance.

"With all their nuclear shields, large enough to protect a ship, a city, an entire world; they could never build one to protect a single man. To supply light and heat to a city, they have motors six stories high—I saw them—where ours could fit into this room. And when I told one of their nuclear specialists that a lead container the size of a walnut contained a nuclear generator, he almost choked with indignation on the spot.

"Why, they don't even understand their own colossi any longer. The machines work from generation to generation automatically and the caretakers are a hereditary caste who would be helpless if a single D-tube in all that vast structure burnt out."

— Isaac Asimov, Foundation



The End.

(Of my session...the beginning of your microservices journey?)

Media Acknowledgements

- Slide 6: http://monashlss.com/sites/default/files/2014/page/new.png
  http://www.dgsimports.net.au/images/detailed/1/dgs_new_cat.png
- Slide 7: http://thecollegestartup.com/wp-content/uploads/2012/08/feature-bloat.png
  http://www.knighton-tools.co.uk/acatalog/06900.jpg
- Slide 8: http://static.comicvine.com/uploads/original/8/80292/3711653-why_not_zoidberg__by_nogard00-d5523pl.jpg
- Slide 9: http://www.Slideshare.net/mohitthatte/microservices-rubyconf2013
- Slide 11: logos of respective organisations
- Slide 18: http://martinfowler.com/articles/microservices.html
- Slide 20: http://www.Slideshare.net/pini4/microservices-and-the-future-on-infrastructure?related=5a
  http://www.Slideshare.net/adriancockcroft/qcon-new-york-speed-and-scale
- Slide 22: http://www.Slideshare.net/jeppec/soa-and-event-driven-architecture-soa-20
  http://www.campusmvp.net/wp-content/uploads/2013/02/large-model.png
  http://www.oracle.com/us/products/middleware/soa/overview/index.html
  http://www.opengroup.org/togaf/
- Slide 26: http://oskarkorczak.blogspot.com.au/2014/03/growing-applications-handled-by-micro.html
- Slide 27: http://alistair.cockburn.us/Hexagonal+architecture
- Slide 32: http://www.tigerteam.dk/2014/micro-services-its-not-only-the-size-that-matters-its-also-how-you-use-them-part-1/
- Slide 34: http://microservices.io/articles/scalecube.html
- Slide 35: http://www.infoq.com/articles/microservices-intro
- Slide 37: http://www.zerohedge.com/article/danger-danger-will-robinson
- Slide 38: http://www.virtualizationpractice.com/topics/agile-cloud-development/
  http://blog.devopsguys.com/2013/07/17/devops-antifragility-and-the-borg-collective/
- Slide 39: http://techblog.netflix.com/2012/11/hystrix.html
- Slide 40: http://www.slideshare.net/fredgeorge/micro-service-architecure slide 38:
- Slide 43: http://www.ratpack.io/
- Slide 47: http://ynaija.com/11-amazing-benefits-and-uses-of-walnuts/
- Slide 48: http://simpsons.wikia.com/wiki/File:Best-simpsons-gifs-world-without-lawyers.gif