THE NEW GENERATION OF ENTERPRISE JAVA & .NET

DESIGNING FOR THE NEXT BIG THING

Jyoti Bansal, Founder and CEO

AppDynamics

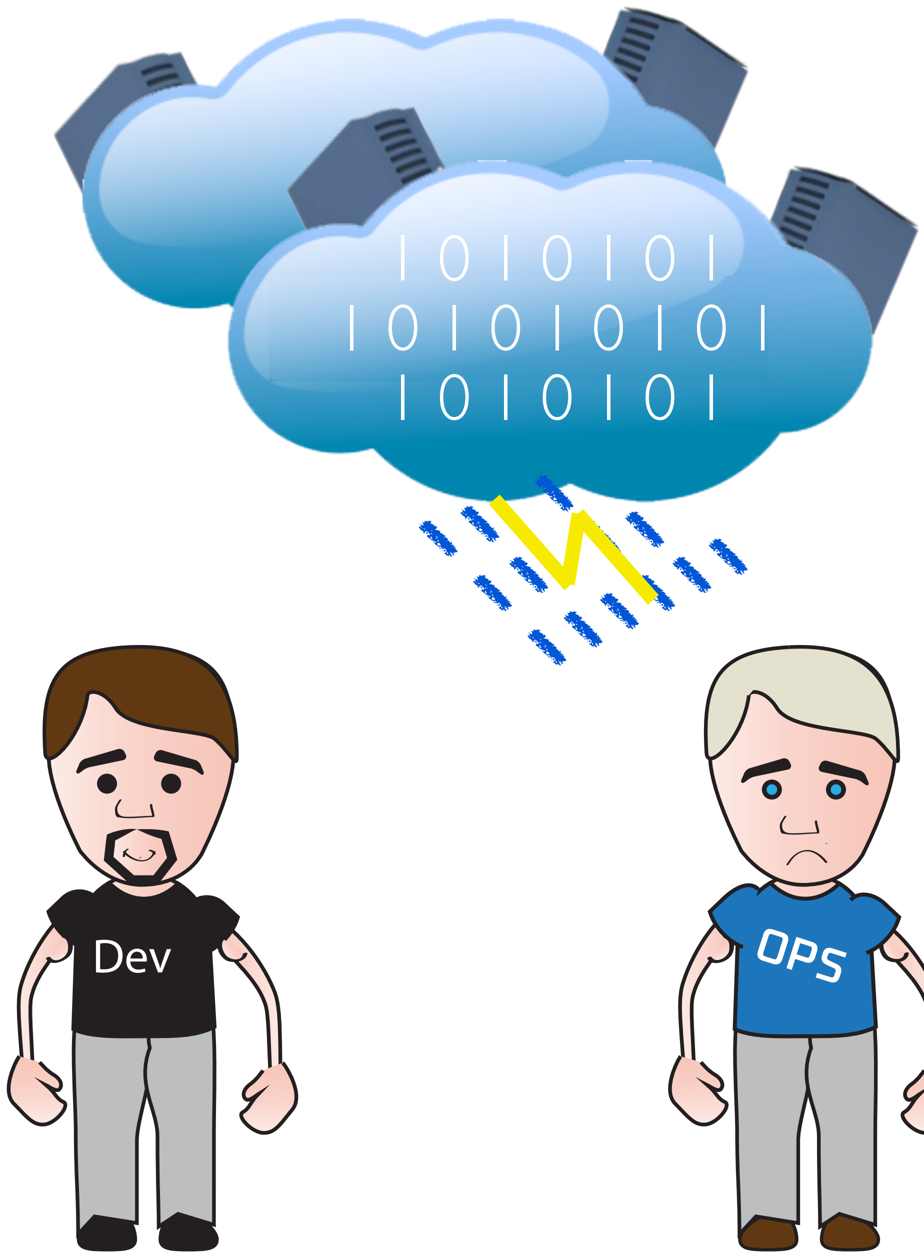
AGENDA

Cloud

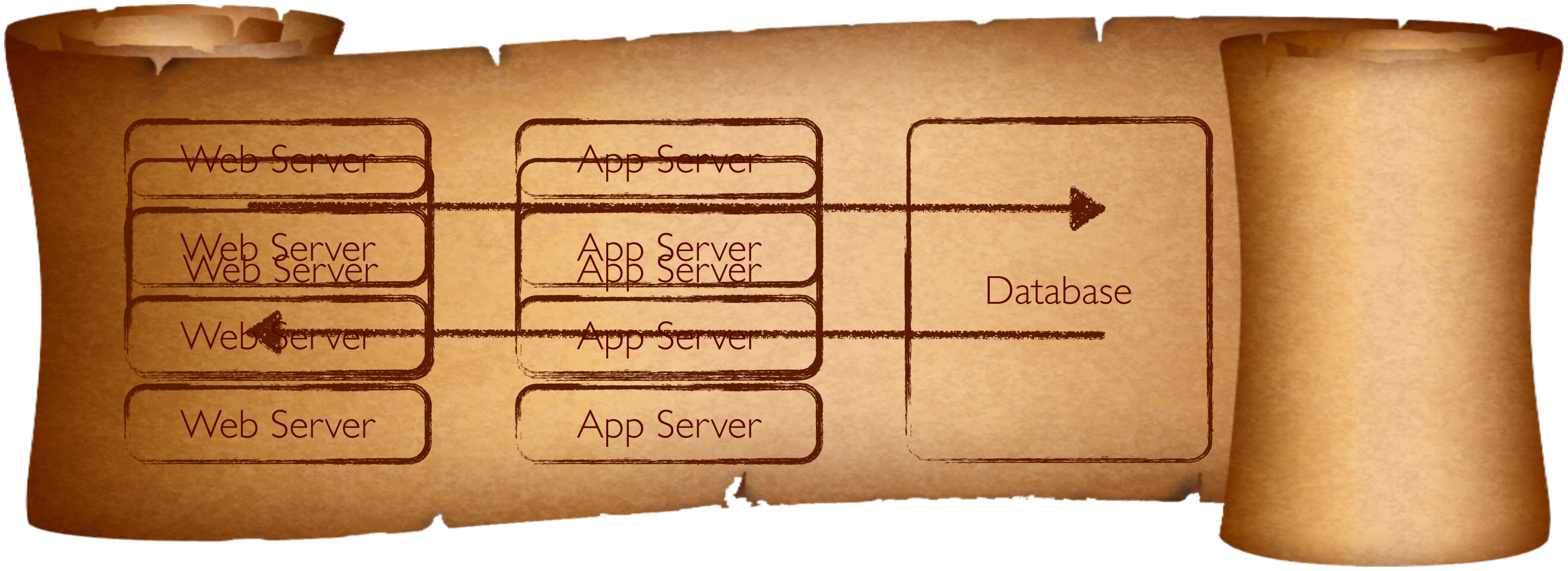
Big Data

DevOps

Managing Failure

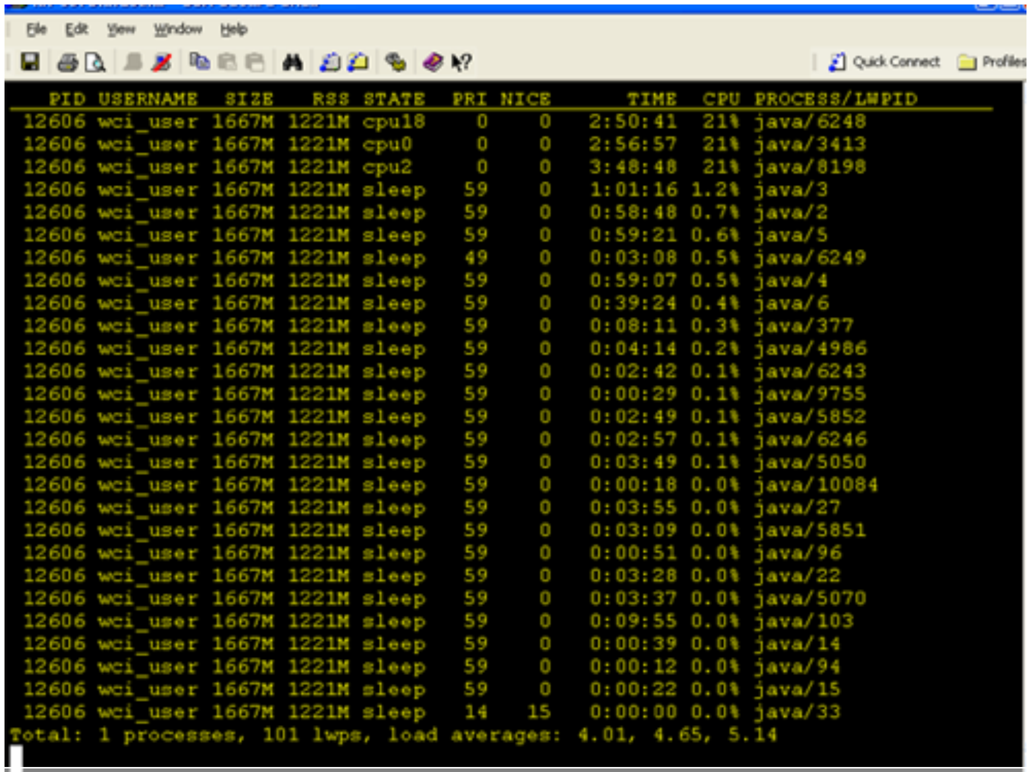


MONOLITHIC JAVA APPS



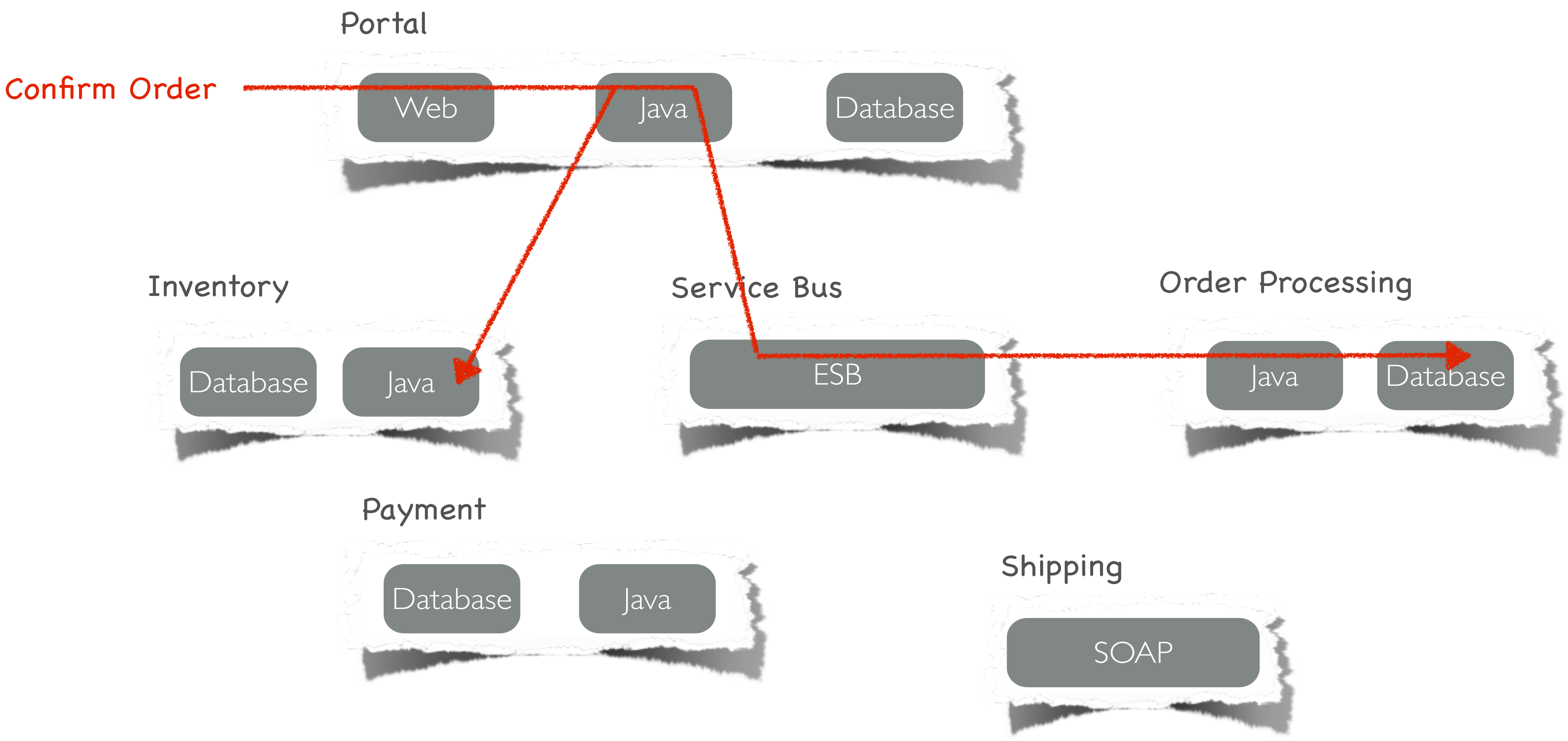
MONOLITHIC #FAIL

- Large Retail Organization
- Apache, Tomcat & Oracle
- Peak Season
- Zero Fault Tolerance



```
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/LWPID
 12606 wci_user 1667M 1221M cpu18    0    0   2:50:41  21% java/6248
 12606 wci_user 1667M 1221M cpu0     0    0   2:56:57  21% java/3413
 12606 wci_user 1667M 1221M cpu2     0    0   3:48:48  21% java/8198
 12606 wci_user 1667M 1221M sleep   59    0   1:01:16 1.2% java/3
 12606 wci_user 1667M 1221M sleep   59    0   0:58:48 0.7% java/2
 12606 wci_user 1667M 1221M sleep   59    0   0:59:21 0.6% java/5
 12606 wci_user 1667M 1221M sleep   49    0   0:03:08 0.5% java/6249
 12606 wci_user 1667M 1221M sleep   59    0   0:59:07 0.5% java/4
 12606 wci_user 1667M 1221M sleep   59    0   0:39:24 0.4% java/6
 12606 wci_user 1667M 1221M sleep   59    0   0:08:11 0.3% java/377
 12606 wci_user 1667M 1221M sleep   59    0   0:04:14 0.2% java/4986
 12606 wci_user 1667M 1221M sleep   59    0   0:02:42 0.1% java/6243
 12606 wci_user 1667M 1221M sleep   59    0   0:00:29 0.1% java/9755
 12606 wci_user 1667M 1221M sleep   59    0   0:02:49 0.1% java/5852
 12606 wci_user 1667M 1221M sleep   59    0   0:02:57 0.1% java/6246
 12606 wci_user 1667M 1221M sleep   59    0   0:03:49 0.1% java/5050
 12606 wci_user 1667M 1221M sleep   59    0   0:00:18 0.0% java/10084
 12606 wci_user 1667M 1221M sleep   59    0   0:03:55 0.0% java/27
 12606 wci_user 1667M 1221M sleep   59    0   0:03:09 0.0% java/5851
 12606 wci_user 1667M 1221M sleep   59    0   0:00:51 0.0% java/96
 12606 wci_user 1667M 1221M sleep   59    0   0:03:28 0.0% java/22
 12606 wci_user 1667M 1221M sleep   59    0   0:03:37 0.0% java/5070
 12606 wci_user 1667M 1221M sleep   59    0   0:09:55 0.0% java/103
 12606 wci_user 1667M 1221M sleep   59    0   0:00:39 0.0% java/14
 12606 wci_user 1667M 1221M sleep   59    0   0:00:12 0.0% java/94
 12606 wci_user 1667M 1221M sleep   59    0   0:00:22 0.0% java/15
 12606 wci_user 1667M 1221M sleep   14   15   0:00:00 0.0% java/33
Total: 1 processes, 101 lwps, load averages: 4.01, 4.65, 5.14
```

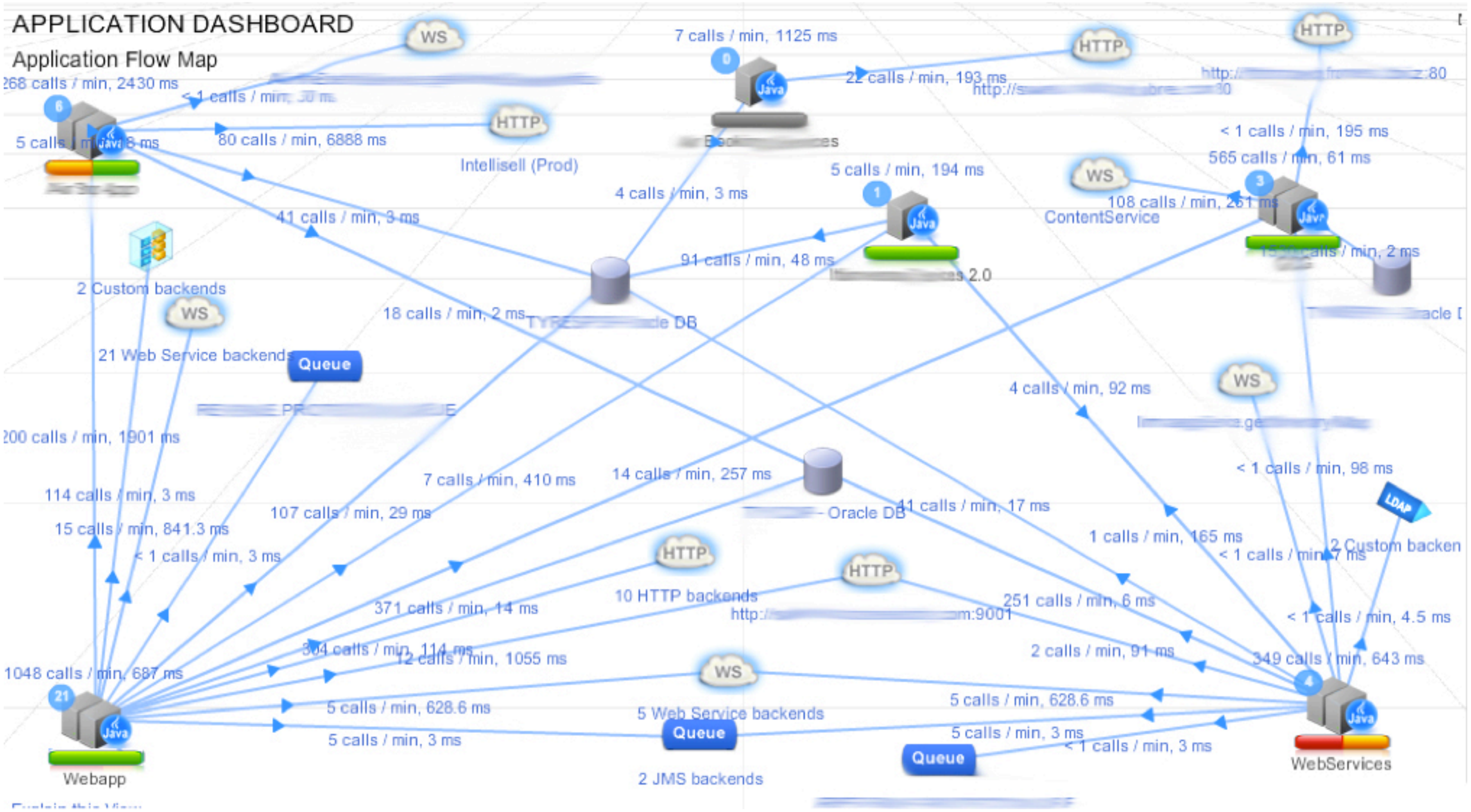
MODERN DISTRIBUTED SERVICES



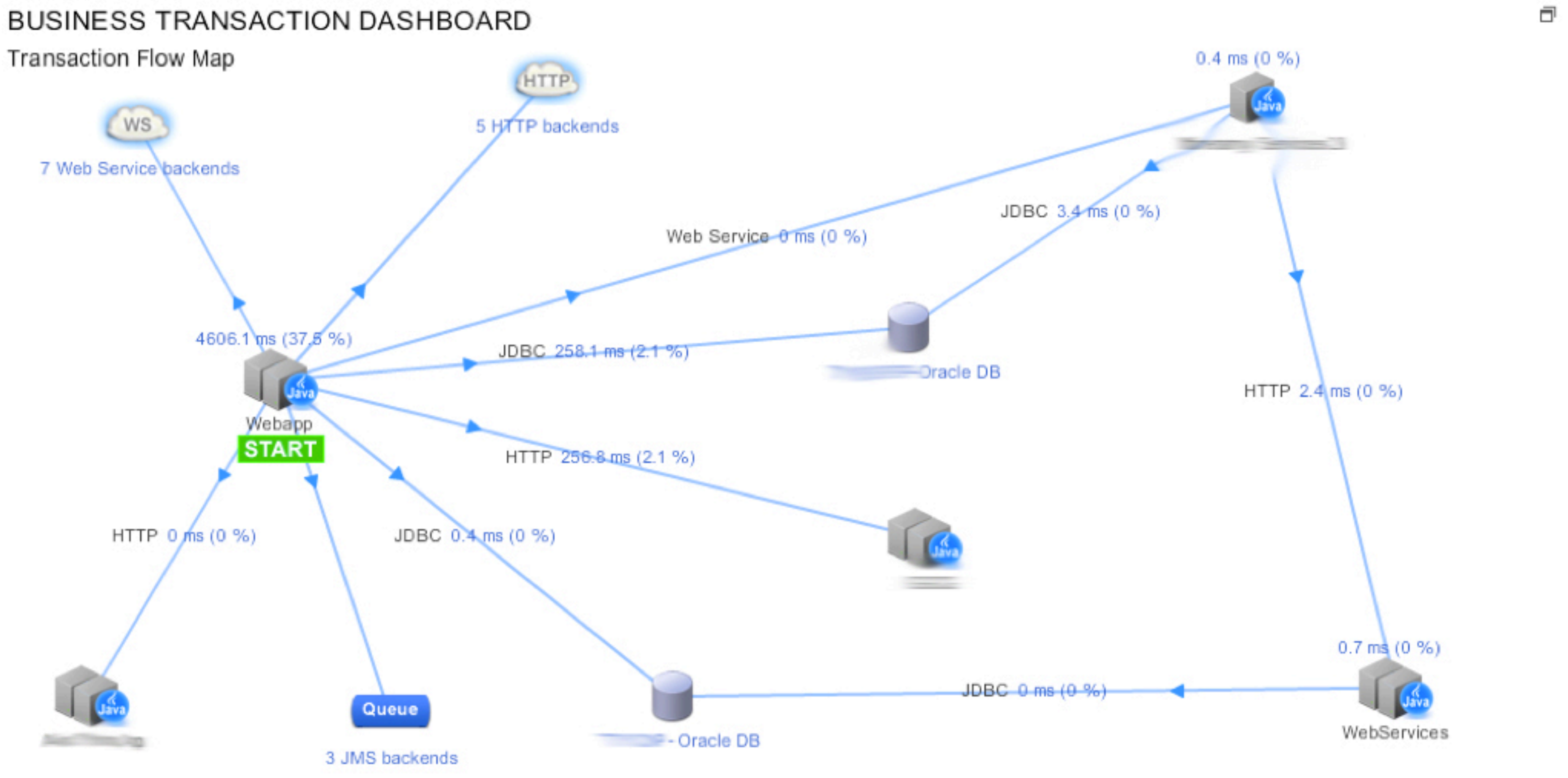
DISTRIBUTED #FAIL

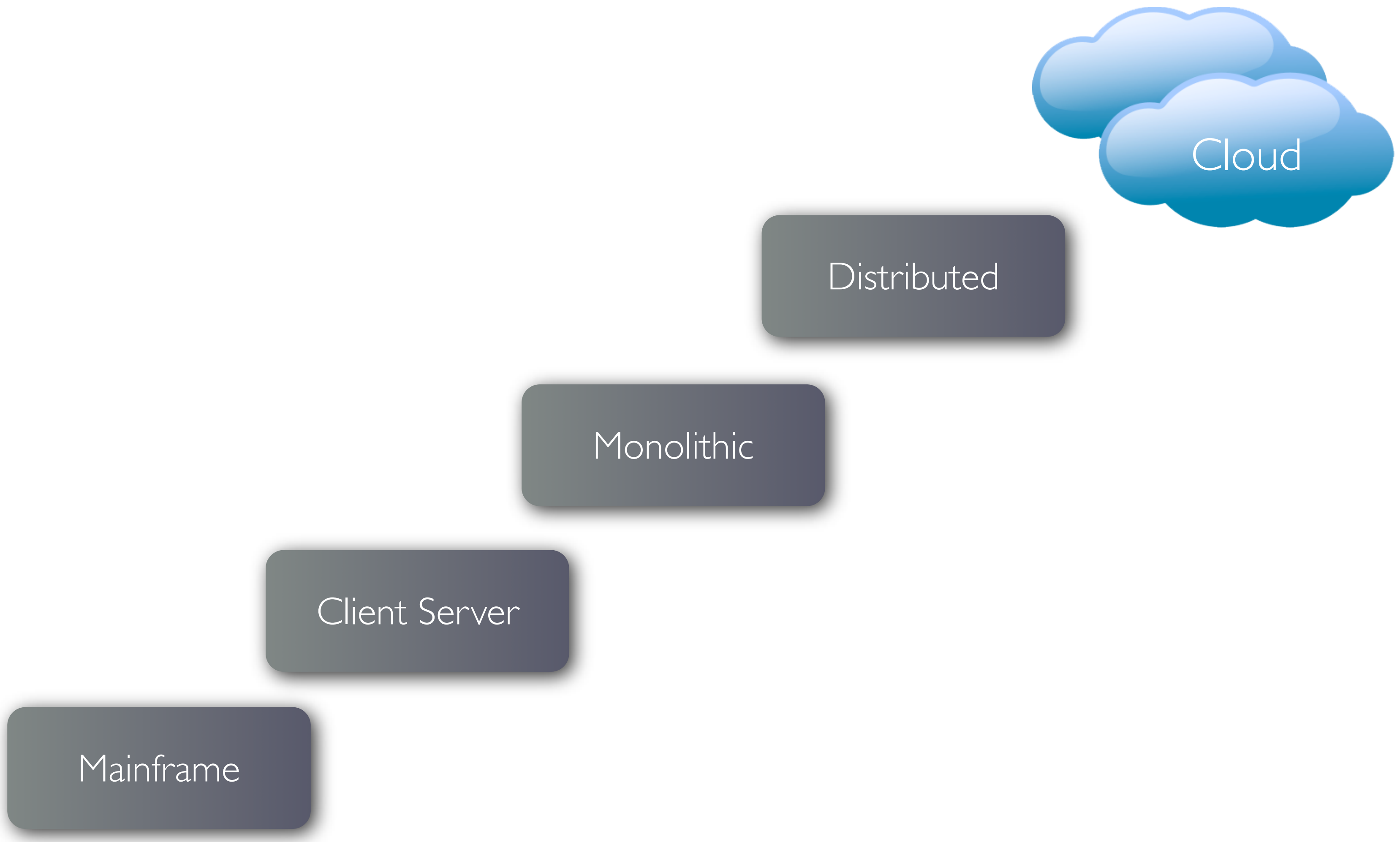
- Large Telco Organization
- Apache, Weblogic, Web Services & Oracle
- New Product Launch
- Shared Service couldn't handle traffic

DISTRIBUTED LOOKS NICE!



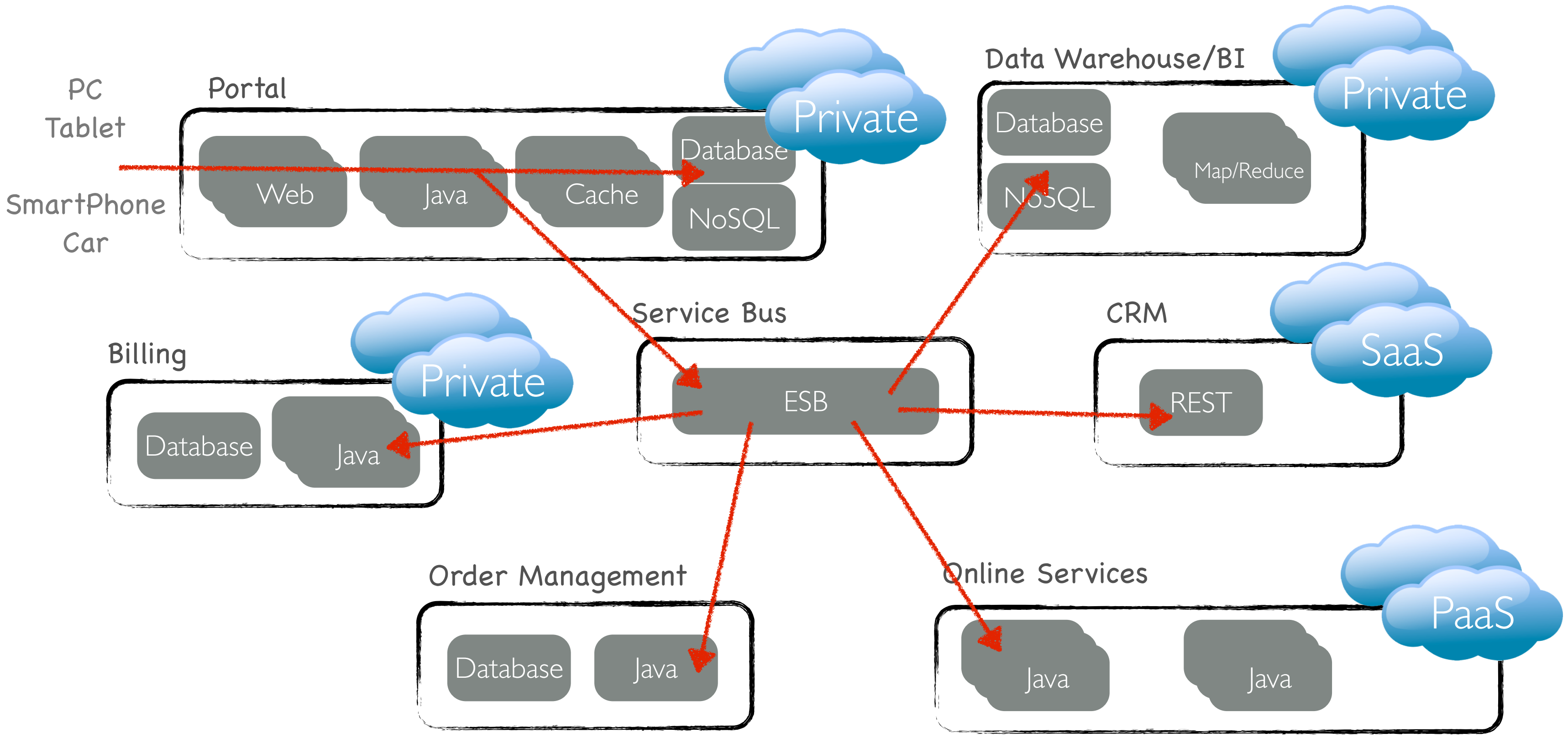
COMPLEX TO MANAGE





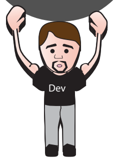
STAIRWAY TO HEAVEN OR HELL?

NEXT GEN DISTRIBUTED JAVA APPS



BIG DATA

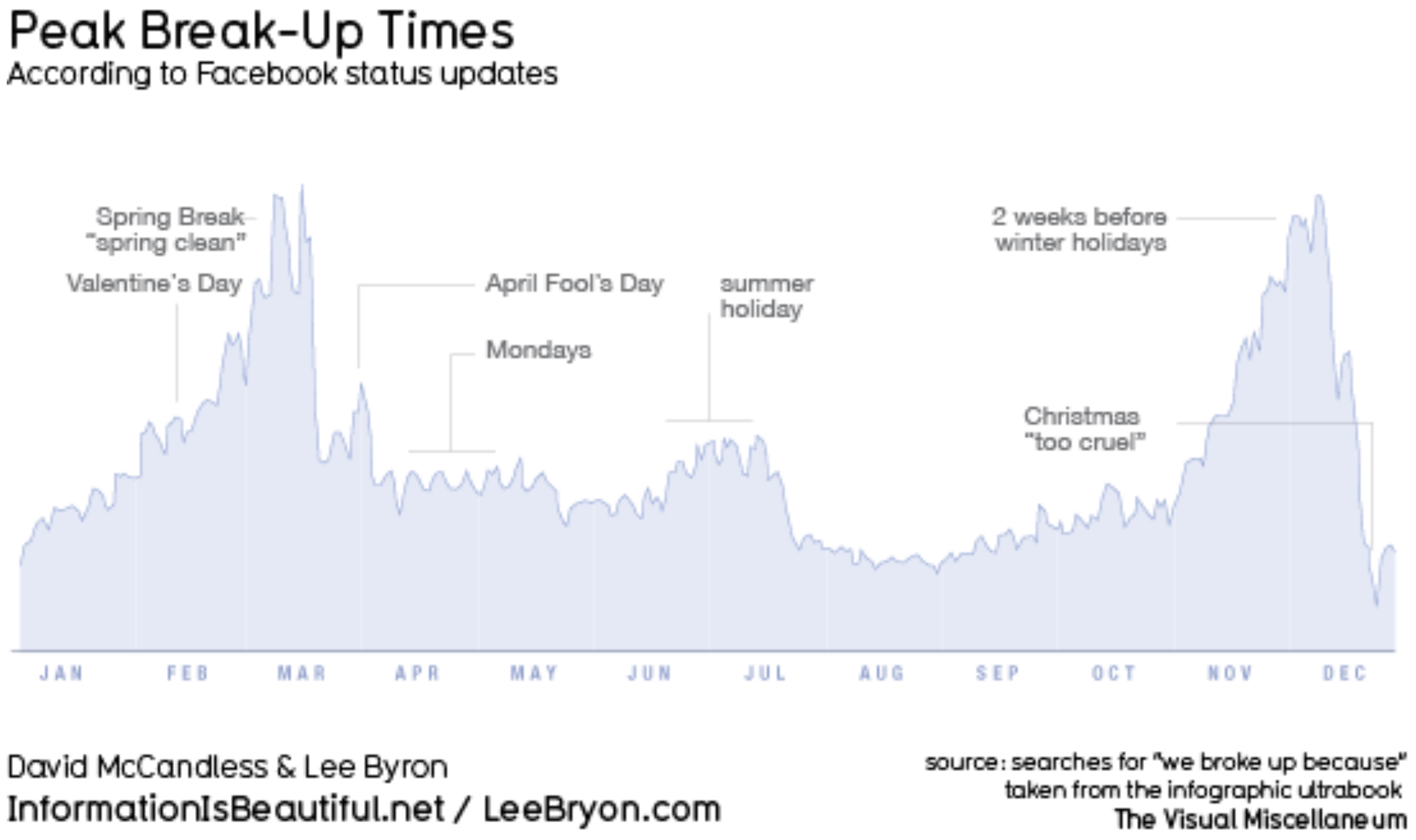
0 1 0 1 0 1 1 0



WHAT IS BIG DATA?

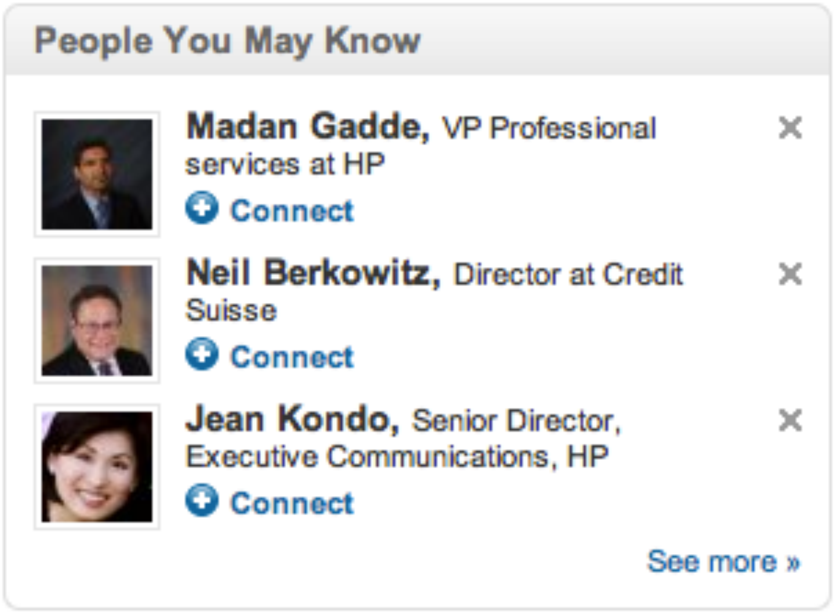
- Too big to store, organize and analyze
- GB's, TB's or PB's
- Parallel Processing of raw data
- Make sense of unstructured data
- Competitive edge for the business & apps

FINDING PATTERNS



WHO AND WHAT?

- Linkedin.com - people you might know
- AOL.com - behavioral analysis & targeting
- Beebler - matching people
- EBay - search optimization
- PokerTableStats - analyzing poker players history & stats



People You May Know

Madan Gadde, VP Professional services at HP
Connect

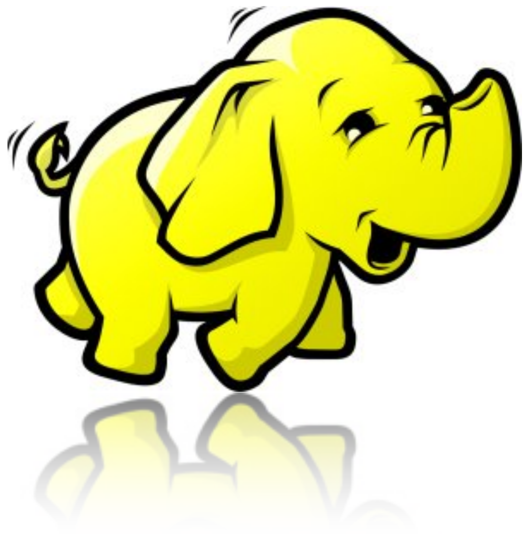
Neil Berkowitz, Director at Credit Suisse
Connect

Jean Kondo, Senior Director, Executive Communications, HP
Connect

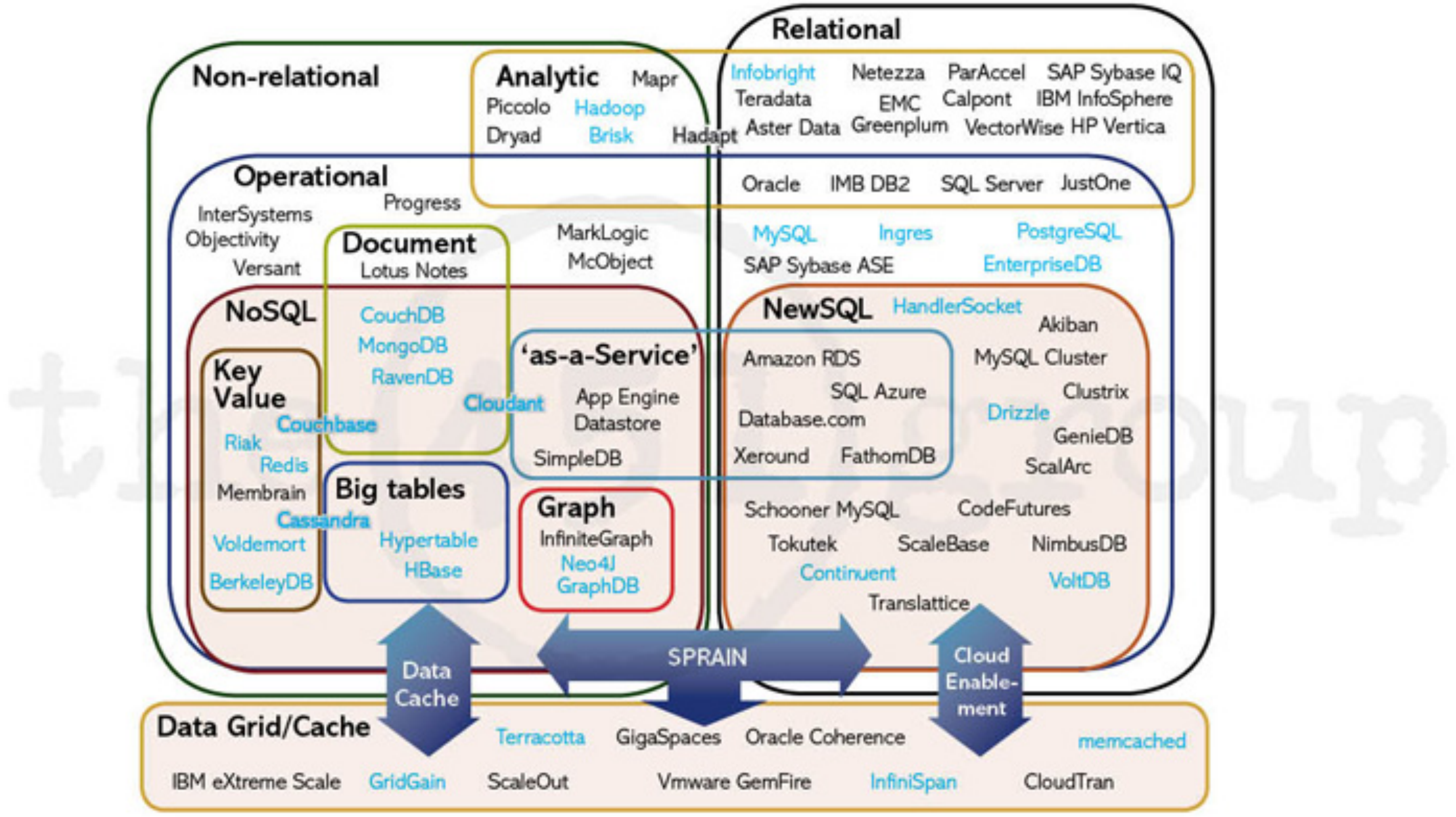
See more »

WHY NOW?

- Computing power is accessible and cheaper
- Technology like Map/Reduce (Hadoop) exist
- Possible to get answers in mins/hours vs. days
- Applications can exploit this intelligence

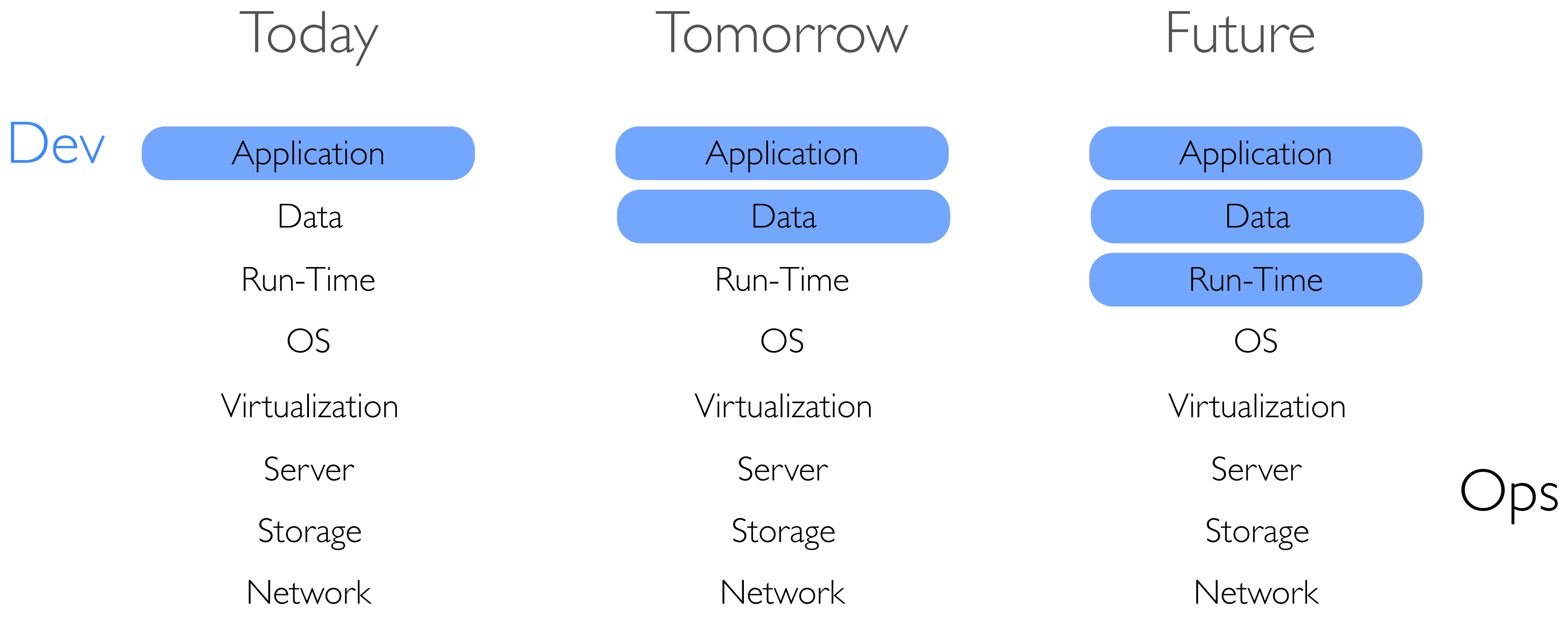


BIG DATA = BIG CHOICES



WHAT DOES THIS MEAN FOR DEV AND OPS?

DEVOPS RESPONSIBILITIES



DEVOPS RESPONSIBILITIES

Agile Dev creates Change

Ops wants less Change

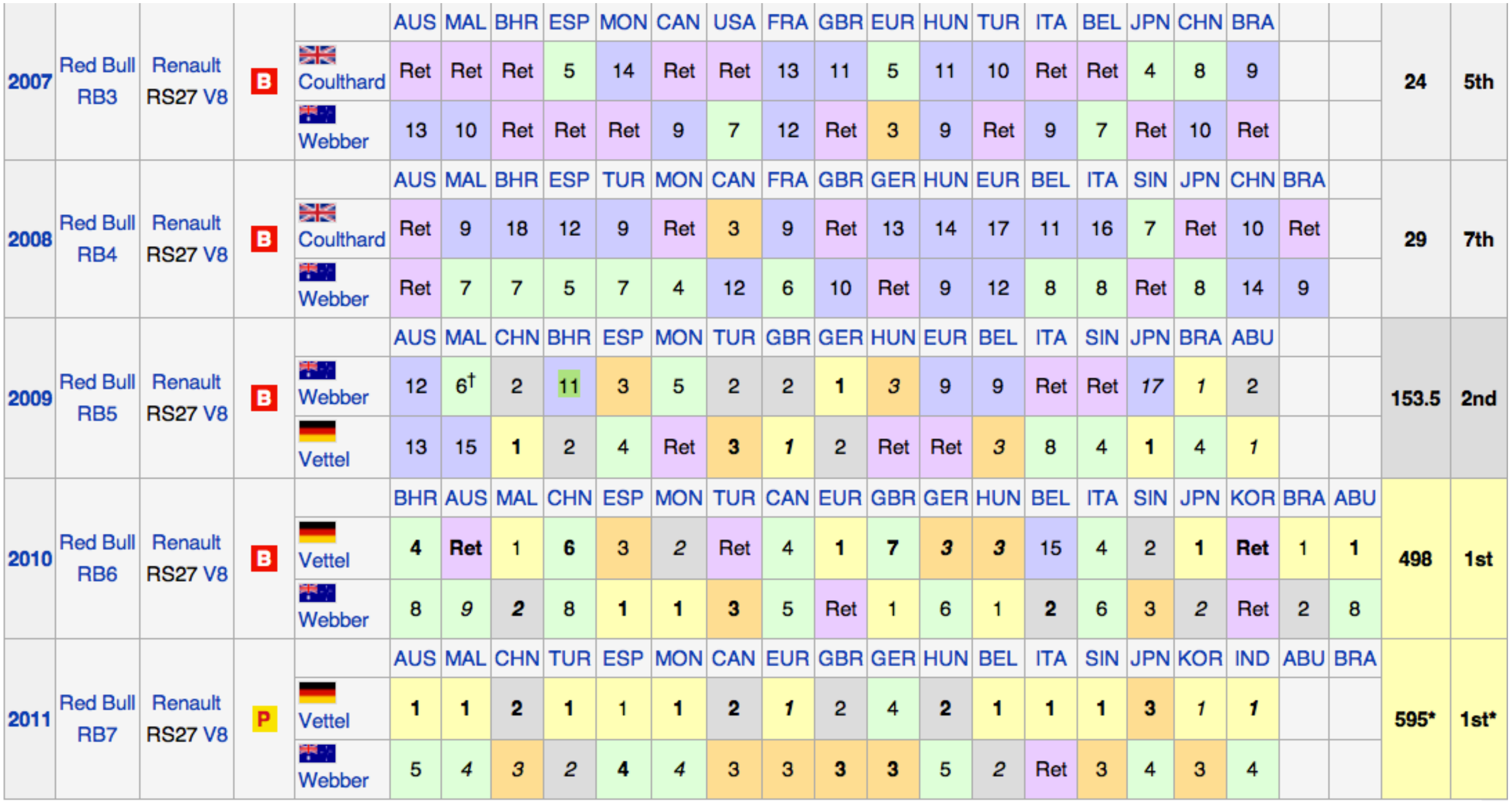
HOW DOES THIS WORK IN OTHER INDUSTRIES?

2011 FORMULA 1 WORLD CHAMPION SEBASTIAN VETTEL

FORMULA 1

Being **agile** and managing **change.**

CHANGE ISN'T EASY

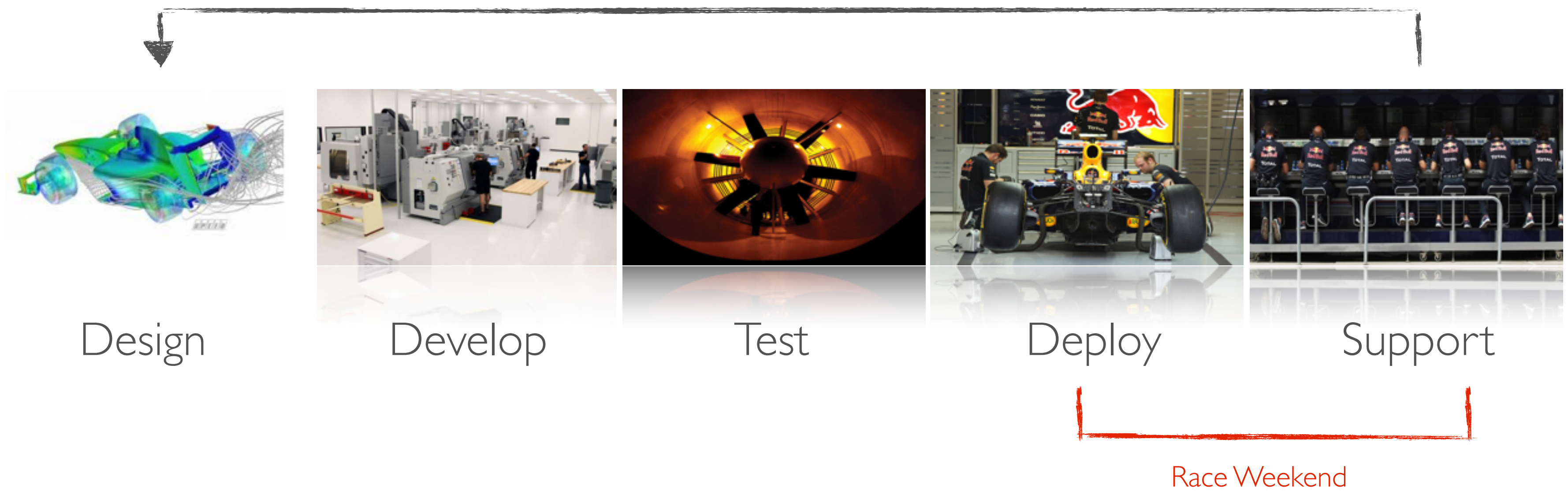


					AUS	MAL	BHR	ESP	MON	CAN	USA	FRA	GBR	EUR	HUN	TUR	ITA	BEL	JPN	CHN	BRA				
2007	Red Bull RB3	Renault RS27 V8	B	Coulthard	Ret	Ret	Ret	5	14	Ret	Ret	13	11	5	11	10	Ret	Ret	4	8	9			**24**	**5th**
				Webber	13	10	Ret	Ret	Ret	9	7	12	Ret	3	9	Ret	9	7	Ret	10	Ret				
					AUS	MAL	BHR	ESP	TUR	MON	CAN	FRA	GBR	GER	HUN	EUR	BEL	ITA	SIN	JPN	CHN	BRA			
2008	Red Bull RB4	Renault RS27 V8	B	Coulthard	Ret	9	18	12	9	Ret	3	9	Ret	13	14	17	11	16	7	Ret	10	Ret		**29**	**7th**
				Webber	Ret	7	7	5	7	4	12	6	10	Ret	9	12	8	8	Ret	8	14	9			
					AUS	MAL	CHN	BHR	ESP	MON	TUR	GBR	GER	HUN	EUR	BEL	ITA	SIN	JPN	BRA	ABU				
2009	Red Bull RB5	Renault RS27 V8	B	Webber	12	6†	2	11	3	5	2	2	**1**	*3*	9	9	Ret	Ret	*17*	*1*	2			**153.5**	**2nd**
				Vettel	13	15	**1**	2	4	Ret	**3**	***1***	2	Ret	Ret	*3*	8	4	**1**	4	*1*				
					BHR	AUS	MAL	CHN	ESP	MON	TUR	CAN	EUR	GBR	GER	HUN	BEL	ITA	SIN	JPN	KOR	BRA	ABU		
2010	Red Bull RB6	Renault RS27 V8	B	Vettel	**4**	**Ret**	1	**6**	3	*2*	Ret	4	**1**	**7**	***3***	***3***	15	4	2	**1**	**Ret**	1	**1**	**498**	**1st**
				Webber	8	*9*	**2**	8	**1**	**1**	**3**	5	Ret	1	6	1	**2**	6	3	*2*	Ret	2	8		
					AUS	MAL	CHN	TUR	ESP	MON	CAN	EUR	GBR	GER	HUN	BEL	ITA	SIN	JPN	KOR	IND	ABU	BRA		
2011	Red Bull RB7	Renault RS27 V8	P	Vettel	**1**	**1**	**2**	**1**	1	**1**	**2**	***1***	2	4	**2**	**1**	**1**	**1**	**3**	*1*	***1***			**595***	**1st***
				Webber	5	*4*	*3*	*2*	**4**	*4*	3	3	**3**	**3**	5	*2*	Ret	3	4	3	4				

AGILE TEAM WORK

- Cars Evolve
    - Up to 30 new parts per race
- Engineering
    - Aero, Engine, Transmission, ...
- Operations
    - Mechanics, Telemetry, Pit Crew

FI DEV LIFECYCLE



Engineers work hand in hand with Operations.

MONITOR & MANAGE CHANGE

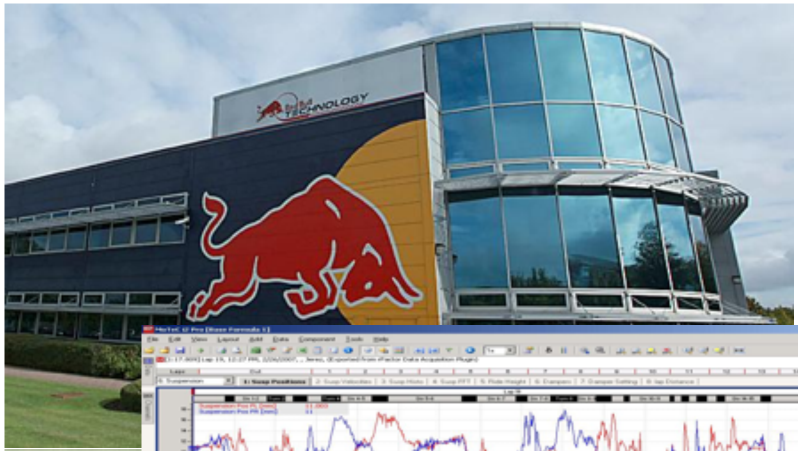
Fast

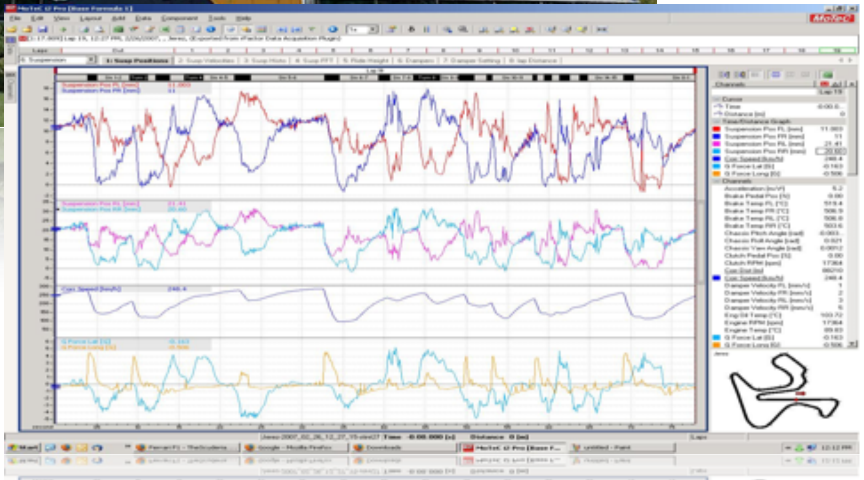
Slow

Fail

MEASURE CHANGE

- Why were we fast/slow/ useless?
- What new parts worked/ didn't work?
- Where did we find time?
- What areas can we improve?





WHAT CAN WE LEARN FROM F1?

#1 Teamwork & Communication

#2 Monitor & Manage Change

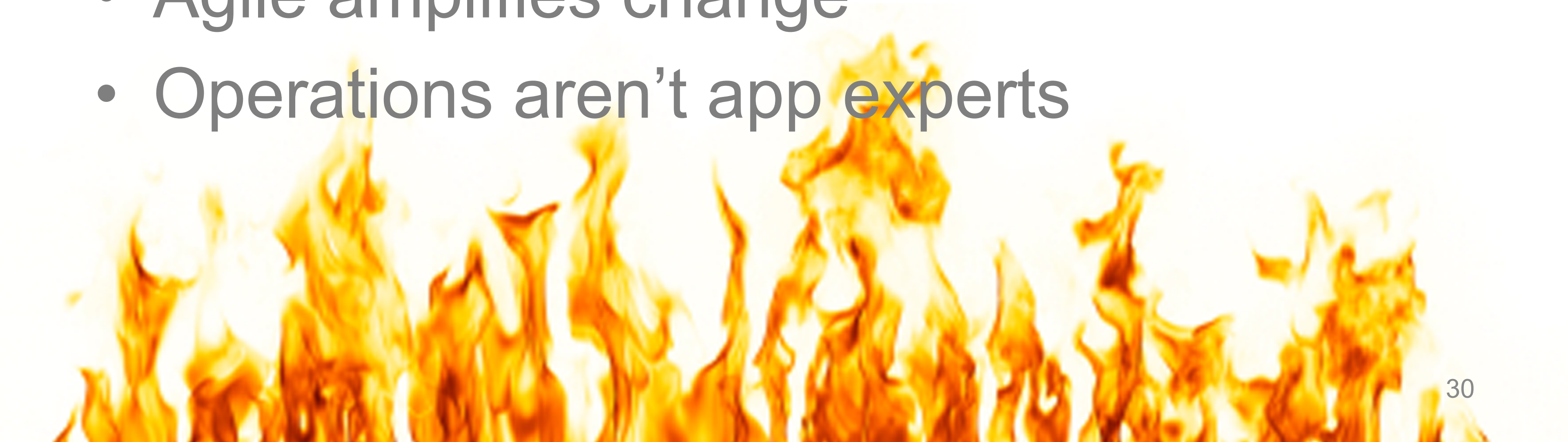
#3 Measure Success

DEVOPS COLLABORATION

- Change isn’t the Enemy
- Lack of Alignment is
- Monitor Change to Manage It
- Innovate > Fail > Learn > Succeed

Applications will Fail

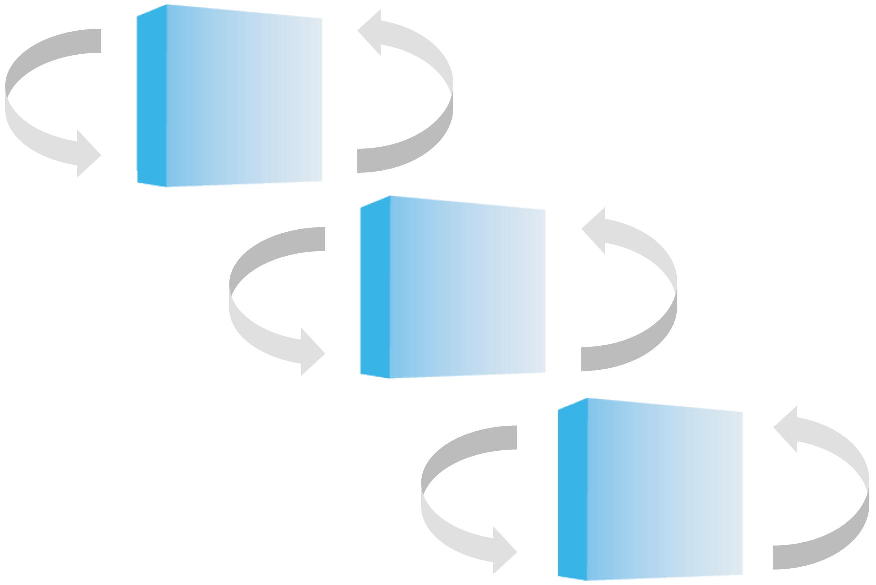
- Written by humans
- Get it to work then make it fast
- QA is dull and painful
- Too many points of failure
- Agile amplifies change
- Operations aren’t app experts



Current Methodology

If Application is Down or Hung

1. Reboot the JVM (Ops)
2. Check JVM console logs (Ops)
3. Analyze thread dumps (Dev)



Current Methodology

If Application is Slow

1. Check OS processes & resource (Ops)
2. Check JVM Metrics (Dev)
3. Check application logs (Dev)
4. Try to re-produce in Test (Dev)
5. Optimize everything (Dev)

Failure is Not a JVM Problem

When End Users complain they don't say:

I think my threads were suffering from synchronization, can you check that for me?

I'm a little worried about my objects and that damn garbage collection

Failure is a Business Problem

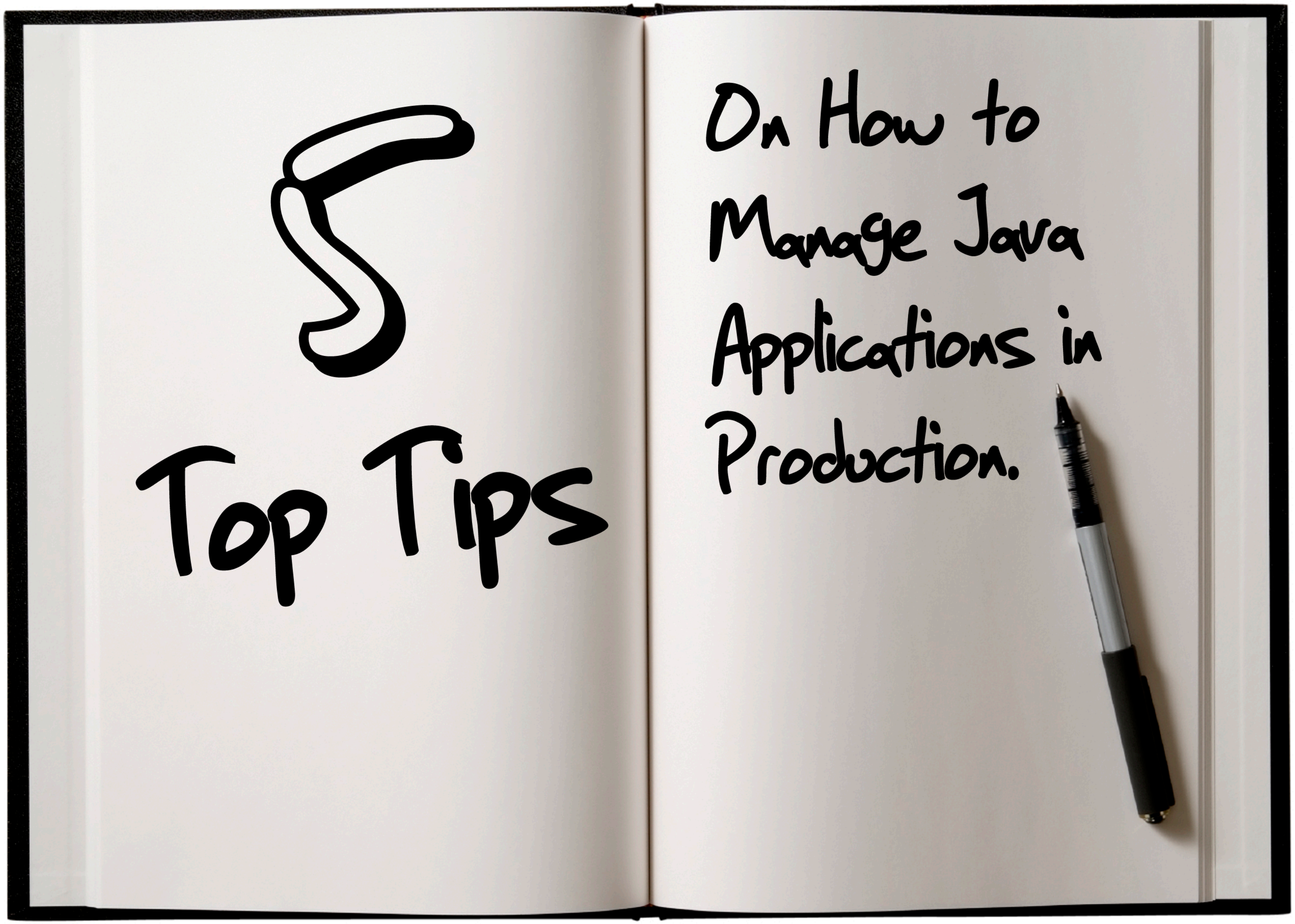
End Users normally say something like:

“My **order confirmation failed**”

“I can’t **retrieve customer records**”

“My **credit card payment** timed out”

It’s about Business Activity

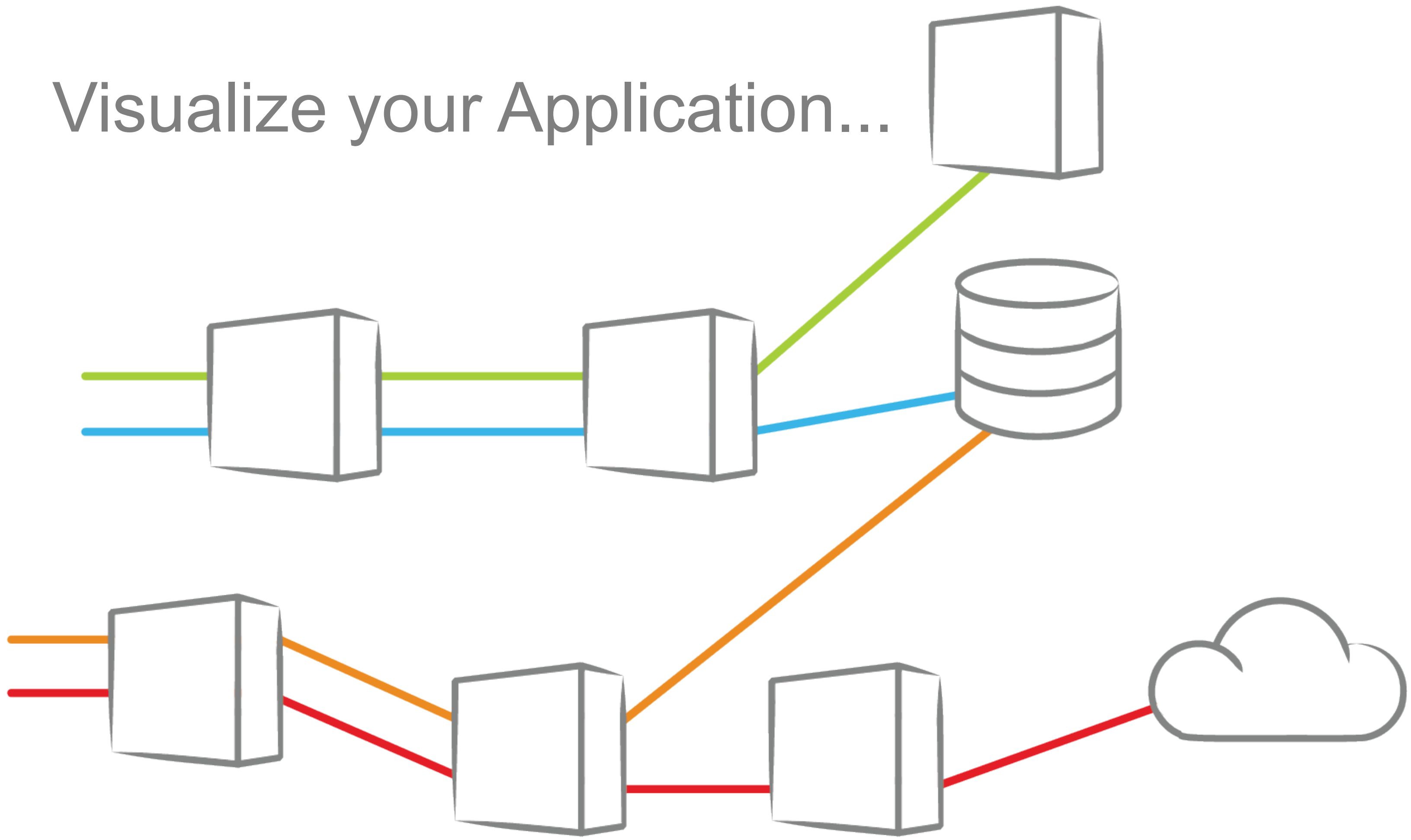


5 Top Tips

On How to Manage Java Applications in Production.

Tip #1

Visualize your Application...



...and Business Transactions.

Tip #2

Identify what is abnormal.

	Current	Baseline	Abnormal
Login	431 ms	402 ms	No
Browse Product	843 ms	856 ms	No
Search	201 ms	231 ms	No
Order Confirmation	5778 ms	3453 ms	Yes
Get Customer Details	1249 ms	1191 ms	No
Retrieve Order History	324 ms	333 ms	No
Authorize Payment	7141 ms	7662 ms	No

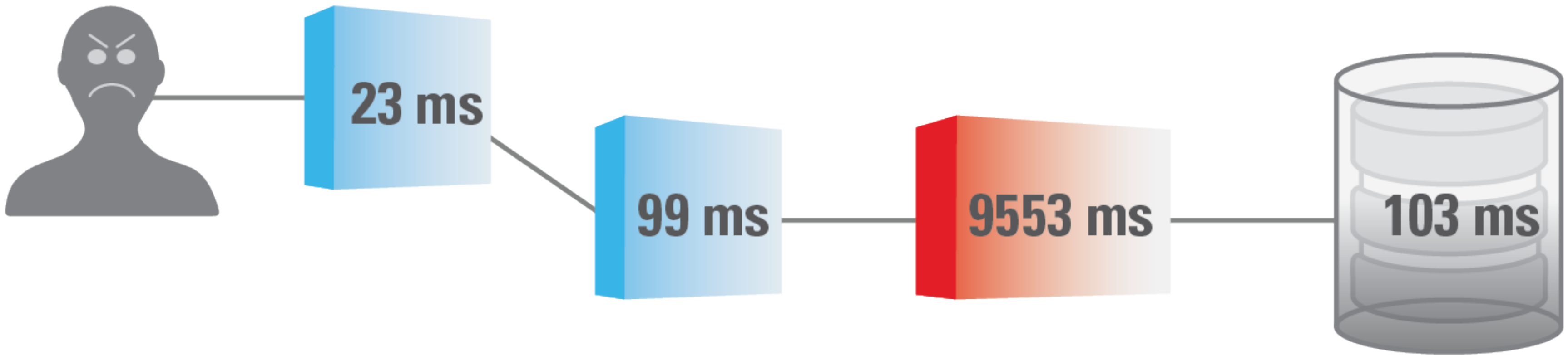
Focus on what to optimize.

(3% rule)

Tip #3

Track the Business Transaction flow.

Order Confirmation 9.778 ms



Isolate where to optimize.

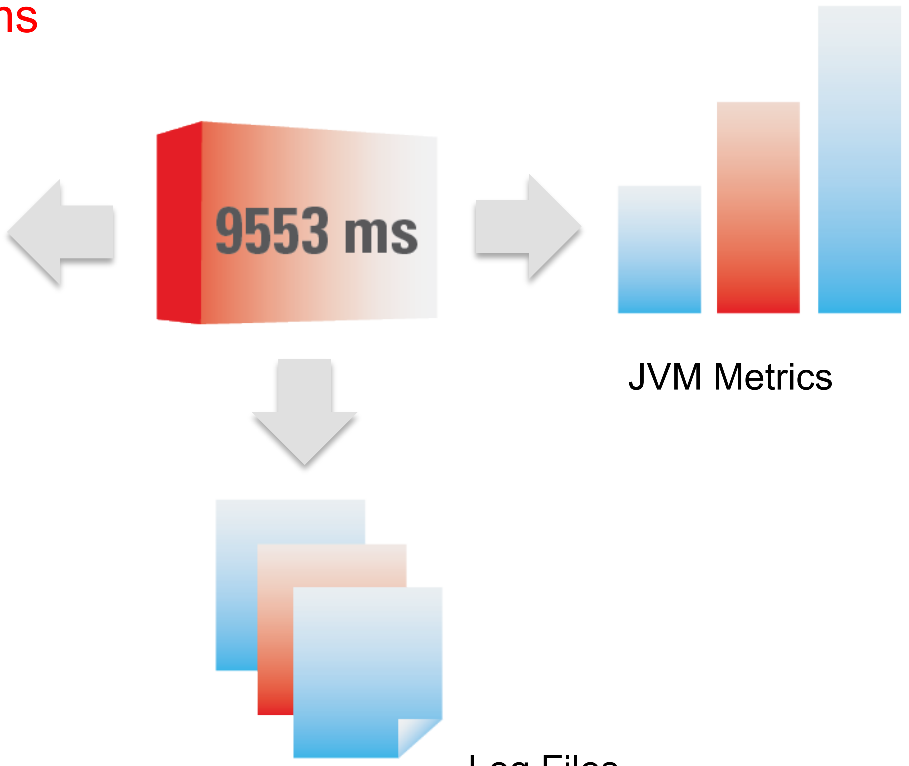
Tip #4

See Diagnostics for Slow Transaction. Optimize!

Order Confirmation 9.778 ms

```
com.organization.class.method()                 0ms
        com.organization.class.method()         6ms
        com.organization.class.method()         22ms
com.organization.class.method()                 0ms
com.organization.class.method()                 56ms
        com.organization.class.method()         10ms
        com.organization.class.method()         6ms
                com.organization.class.method() 9553ms
com.organization.class.method()                 3ms
```

Code Call Graph



Tip #5

Before

Verify Optimization.

After

9.778 ms

Order Confirmation

3.345 ms

```
com.organization.class.method()                   0ms
        com.organization.class.method()           6ms
        com.organization.class.method()           22ms
com.organization.class.method()                   0ms
com.organization.class.method()                   56ms
        com.organization.class.method()           10ms
        com.organization.class.method()           6ms
                com.organization.class.method()   9553ms
com.organization.class.method()                   3ms
```

```
com.organization.class.method()                   0ms
        com.organization.class.method()           6ms
        com.organization.class.method()           22ms
com.organization.class.method()                   0ms
com.organization.class.method()                   56ms
        com.organization.class.method()           10ms
        com.organization.class.method()           6ms
                com.organization.class.method()   2ms
com.organization.class.method()                   3ms
```

Stop Optimization!

SUMMARY

- Design for Failure
- Use Cloud for agility not cost
- Exploit Big Data & Real-time analytics
- Monitor, Manage and Measure Change