



EFFICIENT JAVA MEMORY

Copyright 2020 Kirk Pepperdine

OUR MARKETING SLIDE

- ▶ Kirk Pepperdine
 - ▶ Author of jPDM, a performance diagnostic model
 - ▶ Author of the original Java Performance Tuning workshop
- ▶ Co-founded jClarity
 - ▶ Building the smart generation of performance diagnostic tooling
 - ▶ Bring predictability into the diagnostic process
- ▶ Co-founded JCrete
 - ▶ The hottest unconference on the planet
- ▶ Java Champion

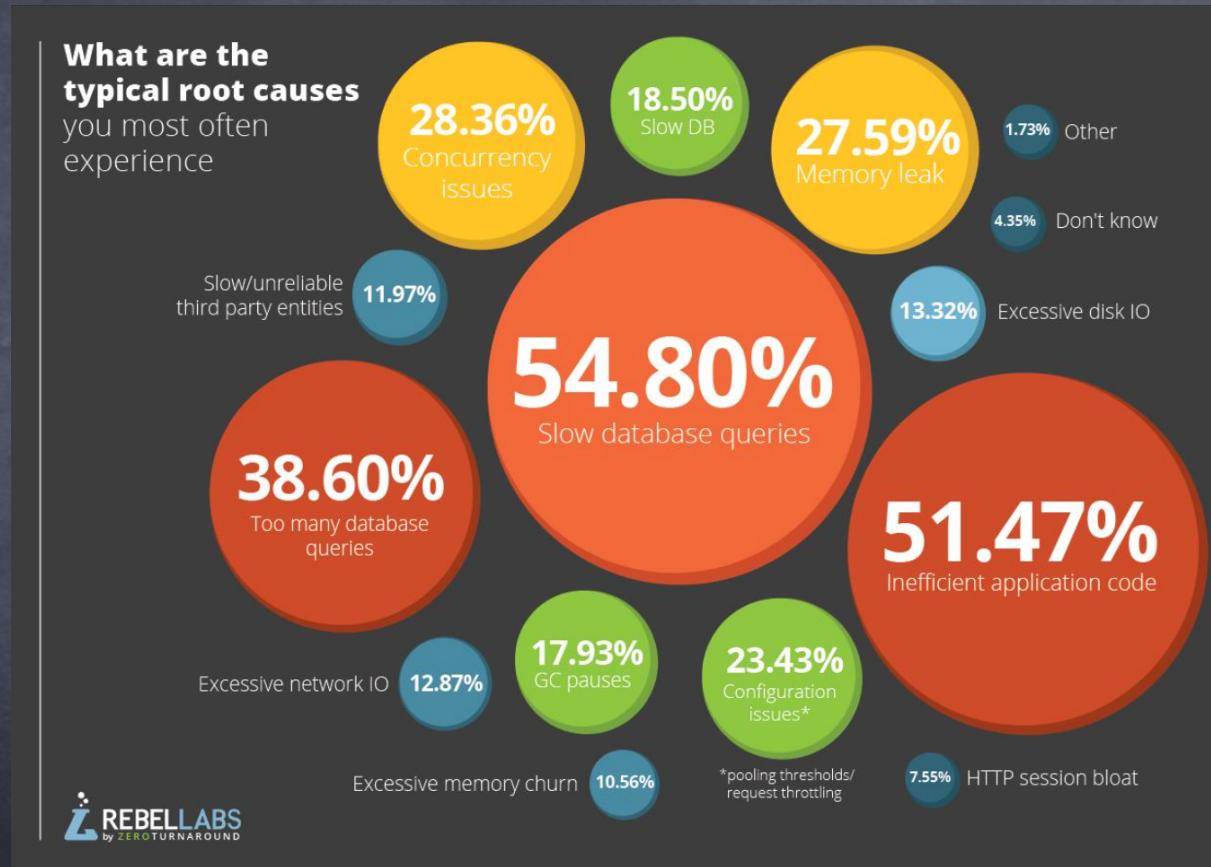


Demo

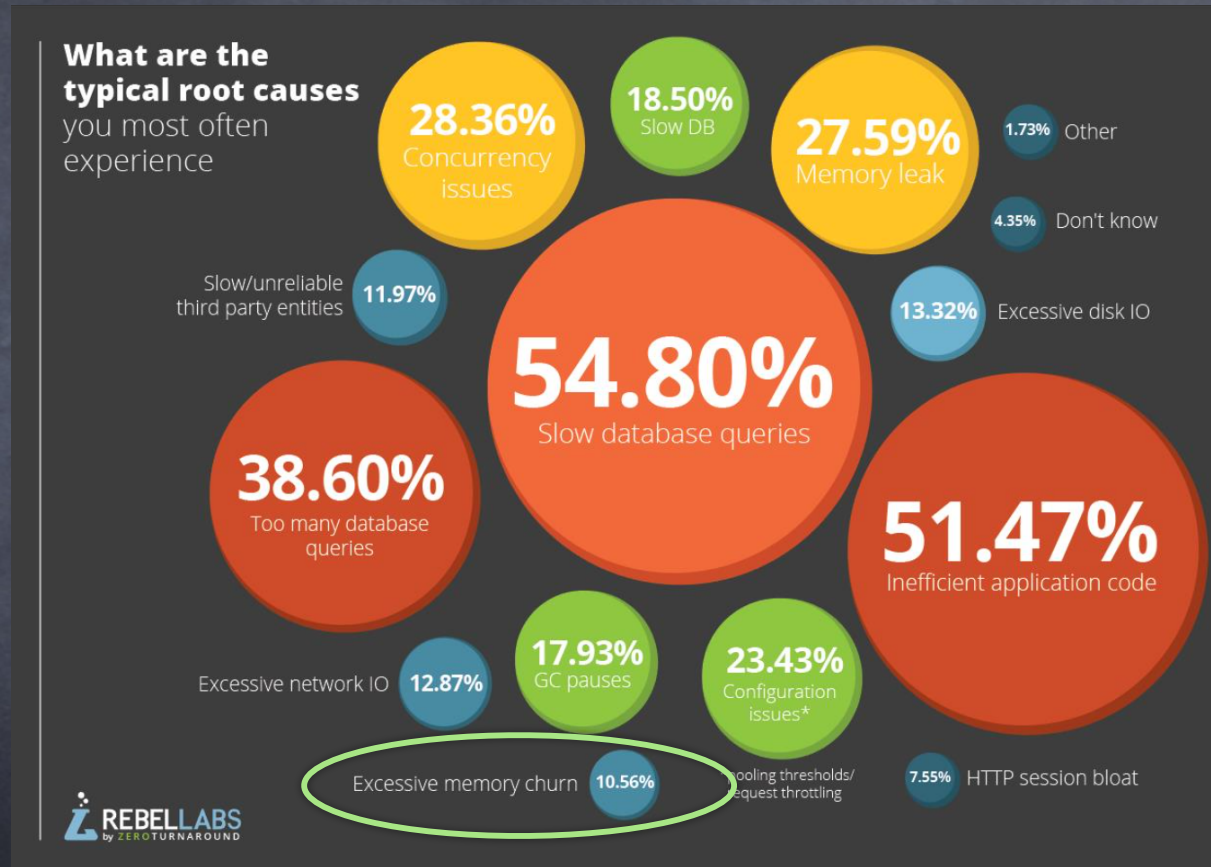


What is your performance trouble spot

INDUSTRY SURVEY




INDUSTRY SURVEY





> 70% of all applications are bottlenecked
on memory



and no,
Garbage Collection
is not a fault!!!!

DO YOU USE



DO YOU USE



Spring Boot

DO YOU USE



Cassandra

Copyright 2020 Kirk Pepperdine

DO YOU USE

Cassandra
or any big nosql solution

Copyright 2020 Kirk Pepperdine

DO YOU USE



Apache Spark

Copyright 2020 Kirk Pepperdine

DO YOU USE

Apache Spark
or any big data framework

Copyright 2020 Kirk Pepperdine

DO YOU USE



Log4J

Copyright 2020 Kirk Pepperdine

DO YOU USE

The background of the slide is a dark, textured surface resembling a chalkboard. Faintly visible in the background is a chalk drawing of a person's head in profile, facing right, with a large thought bubble above it. The text is centered on the slide.

Log4J
or any Java logging framework

Copyright 2020 Kirk Pepperdine

DO YOU USE



JSON

Copyright 2020 Kirk Pepperdine

DO YOU USE

JSON

With almost any Marshalling protocol

DO YOU USE

A faint, light-colored chalkboard-style drawing of a person's head and shoulders. The person has a large, oval-shaped thought bubble above their head. Inside the thought bubble, there are three smaller circles of varying sizes, suggesting a sequence of thoughts or a process. The drawing is done in a simple, sketchy style with visible lines.

ECom caching products

Copyright 2020 Kirk Pepperdine

DO YOU USE

ECom caching products
Hibernate

Copyright 2020 Kirk Pepperdine

DO YOU USE

ECom caching products
Hibernate
and so on

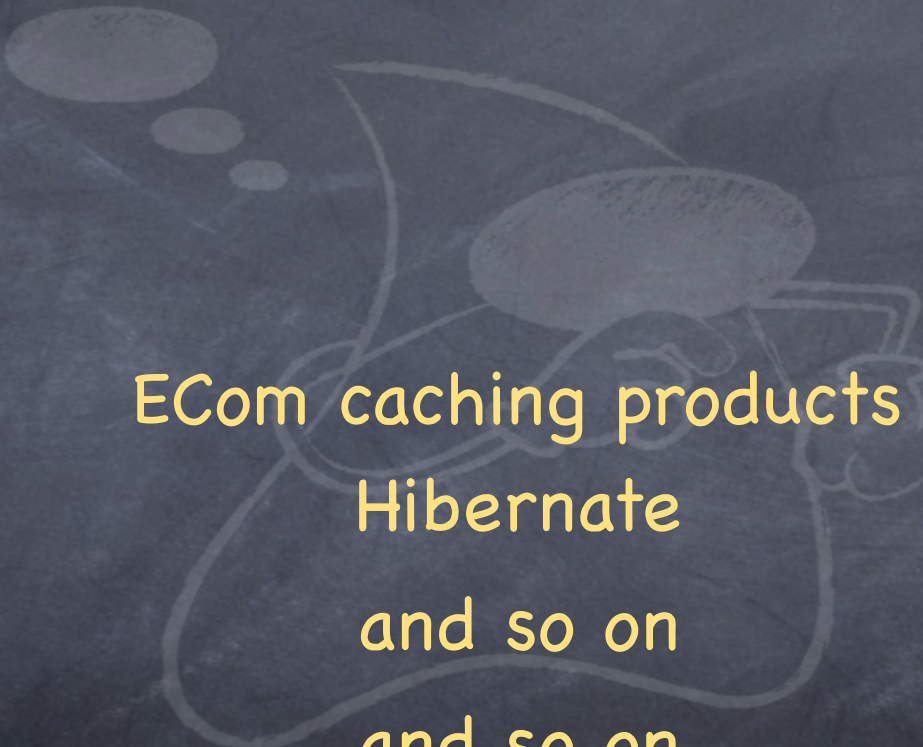
Copyright 2020 Kirk Pepperdine

DO YOU USE

ECom caching products
Hibernate
and so on
and so on

Copyright 2020 Kirk Pepperdine

DO YOU USE



ECom caching products
Hibernate
and so on
and so on
and so on



then you are very likely in this 70%



Copyright 2020 Kirk Pepperdine

WAR STORIES

- ▶ Reduced allocation rates from 1.8gb/sec to 0
 - ▶ tps jumped from 400,000 to 25,000,000!!!
- ▶ Stripped all logging out of a transactional engine
 - ▶ Throughput jumped by a factor of 4x
- ▶ Wrapped 2 logging statements in a web socket framework
 - ▶ Memory churn reduced by a factor of 2
- ▶ and

ALLOCATION SITE

```
Foo foo = new Foo();
```

forms an allocation site

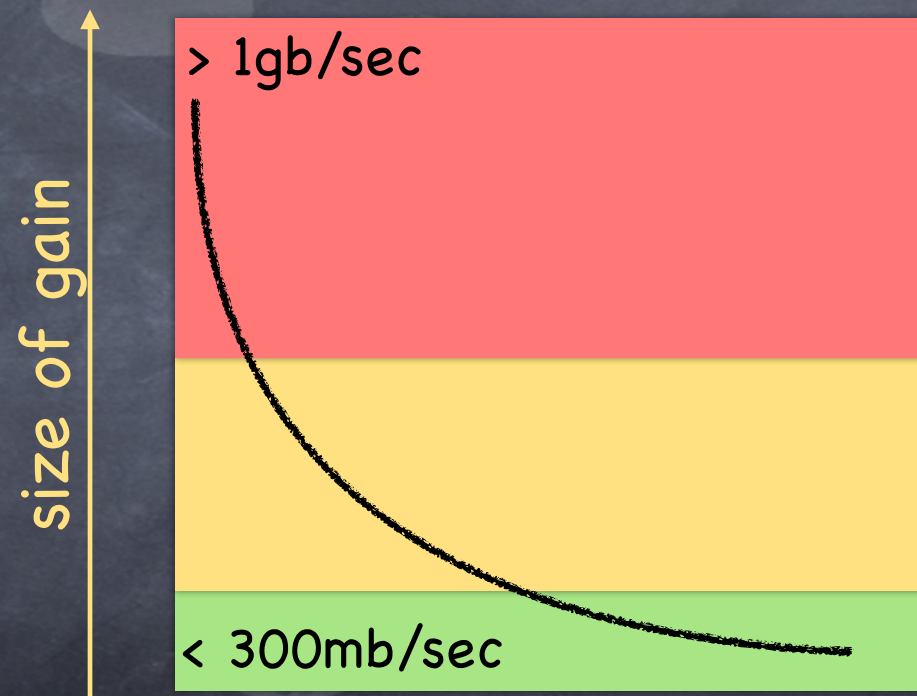
```
0: new          #2    // class java/lang/Object  
2: dup  
4: invokespecial #1    // Method java/lang/Object."<init>":()V
```

- ▶ Allocation will (mostly) occur in Java heap
 - ▶ fast path
 - ▶ slow path
 - ▶ small objects maybe optimized to an scalar allocation

ALLOCATIONS

- ▶ Size vs. Frequency
 - ▶ cost of allocating large objects mostly equal to cost smaller objects
 - ▶ cost of allocations is mostly inexpensive
 - ▶ cheap * a lot == expensive
- ▶ Allocation rate is an approximation of allocation frequency
 - ▶ use allocation rate as a proxy measure for allocation frequency

REDUCING ALLOCATIONS



OTHER PROBLEMS

- ▶ High memory churn rates
 - ▶ many temporary objects
- ▶ Quickly fill Eden
 - ▶ frequent young gc cycles
 - ▶ speeds up aging
 - ▶ premature promotion
 - ▶ more frequent tenured cycles
 - ▶ increased copy costs
 - ▶ increased heap fragmentation
 - ▶ Allocation is quick
 - ▶ quick * large number = slow

SIDE-EFFECTS

- ▶ High memory churn rates
- ▶ many temporary objects

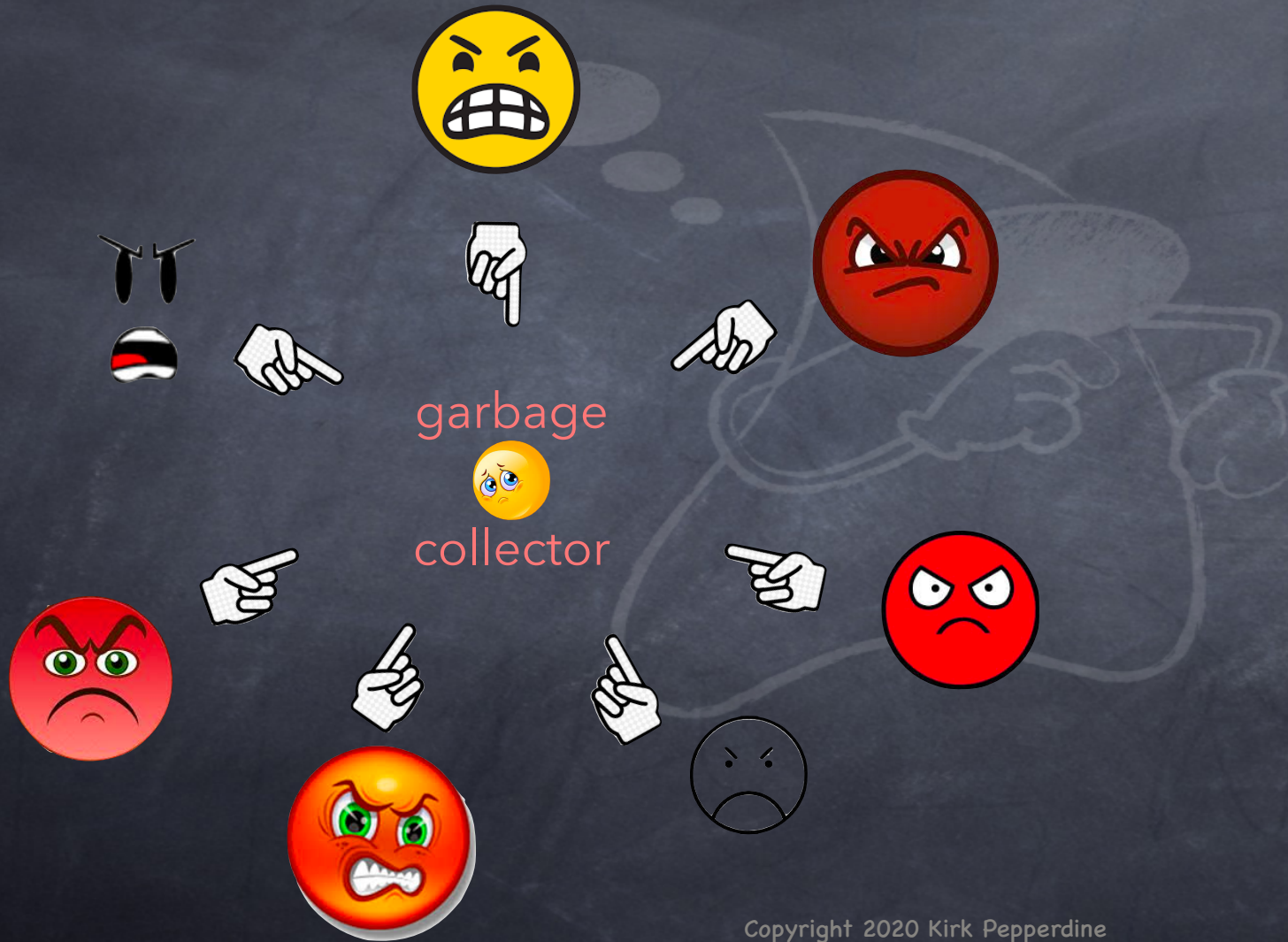


- ▶ Quickly fill Eden
- ▶ **frequent young gc cycles**
- ▶ speeds up aging
- ▶ premature promotion
- ▶ **more frequent tenured cycles**
- ▶ increased copy costs
- ▶ increased heap fragmentation
- ▶ Allocation is quick
- ▶ quick * large number = slow

Hyper active
garbage collector



BLAME GAME



Copyright 2020 Kirk Pepperdine

SERIOUSLY????



garbage

collector

DEFENSELESS

garbage



collector





What about Mastermind?



 @kcpeppe

 kirk@kodewerk.com

*Ask me about our
Java Performance Tuning Workshops*