Pivotal

SpringOne Platform by Pivotal

re:Cap Seoul 2019

2019년 12월 17일(화) • 한화 드림플러스 강남

Pivotal

SpringOne Platform by Pivotal
re:Cap Seoul 2019

Apache Geode Summit re:Cap

신혜원 (Haewon Shin)
Pivotal Platform Architect

Apache Geode Summit

https://www.youtube.com/playlist?list=PLgCUiLdOC2pi9SYptnVamh6xxoeRVZx8s

Geode Sessions

- **Breaking Open Apache Geode: How It Works and Why**
- Introducing the Geode Native Client
- **Performance in Geode: How Fast Is It, How Is It Measured, and How Can It Be Improved?**
- Using Apache Geode: Lessons Learned at Southwest Airlines
- A Fireside Chat with Apache Geode Committers
- **Visualize Your Geode Metrics**
- **Reactive Event Processing with Apache Geode**
- Data Serialization and CI/CD Techniques for Apache Geode
- High-Performance Data Processing with Spring Cloud Data Flow and Geode
- Scaling Beyond a Billion Transactions Per Day with Sub-Second Responses
- Scalable, Cloud-Native Data Applications by Example
- Simple Data Movement Patterns: Legacy Application to Cloud-Native Environment and Apache Geode

Agenda

- What is Geode?
- How to Monitor: Visualize Geode Metrics
- Performance in Geode
- Reactive Programming with Geode

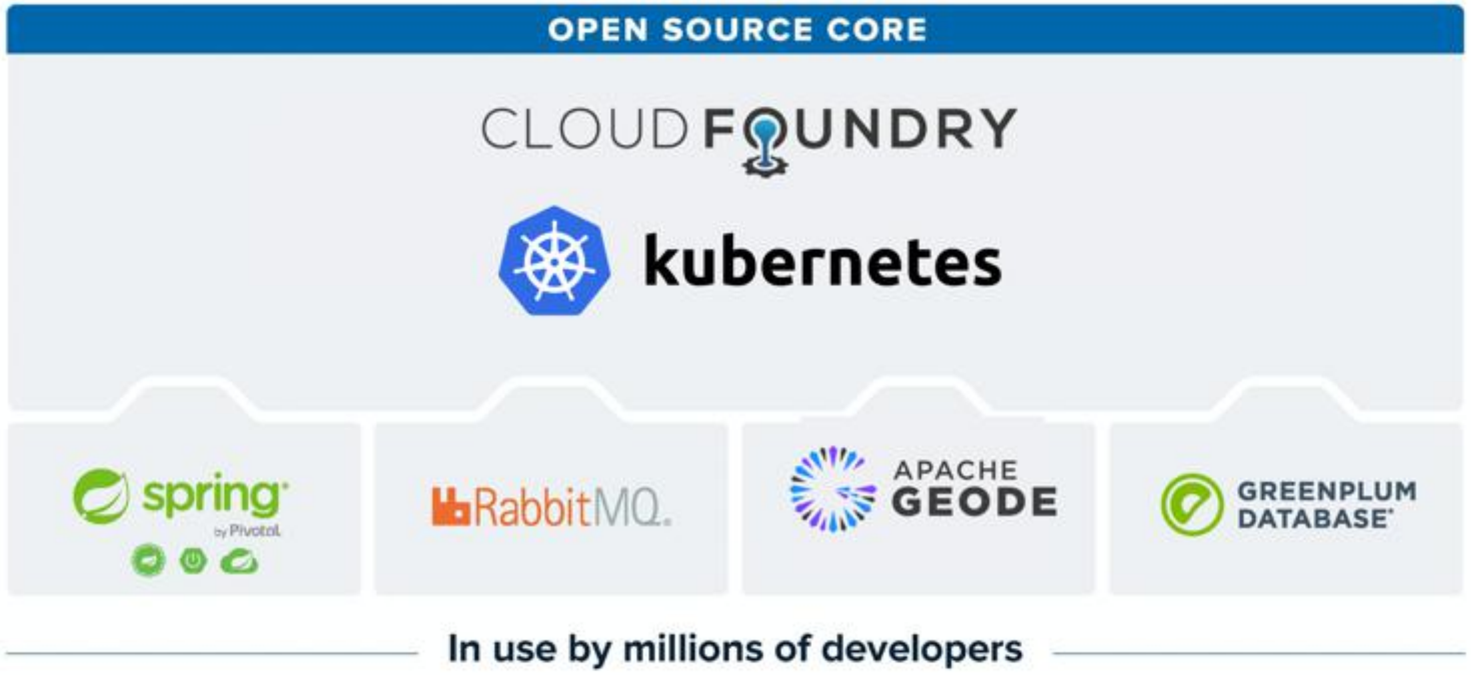
Pivotal

SpringOne Platform by Pivotal

re:Cap Seoul 2019

What is Geode?

피보탈이 투자하고 있는 오픈소스 생태계



Geode / GemFire / Pivotal Cloud Cache (PCC)

Continuous investment in data R&D

What is good for one is good for the other

Pivotal
Cloud Cache

On-platform & clouds

High-performance caching & data acceleration for microservices

https://pivotal.io/pivotal-cloud-cache

Pivotal
GemFire®

Off-platform & native clients

In-memory data grid enabling event-driven architecture and fast-data access patterns

https://pivotal.io/pivotal-gemfire

APACHE
GEODE

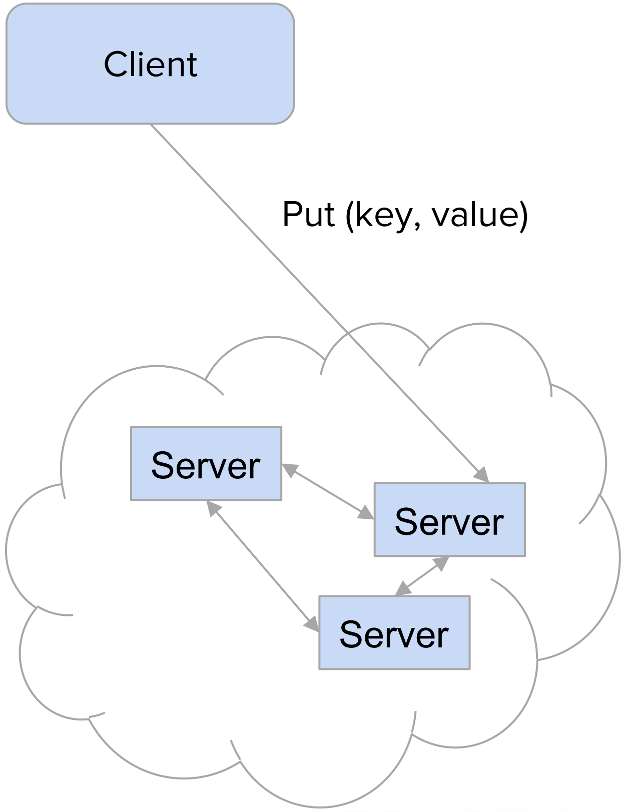
Open source - Performance is key. Consistency is a must.

Reliable transaction processing and shared-nothing architecture for very low latency performance with high concurrency processing

https://content.pivotal.io/pivotal-gemfire/scaling-data-services-with-pivotal-gemfire

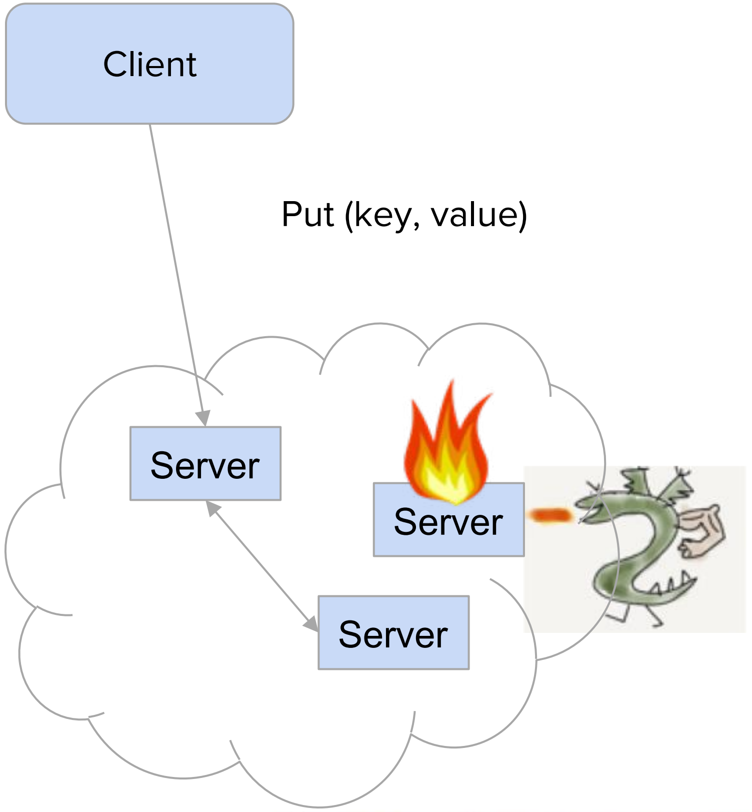
What is Geode?

- Distributed key-value store



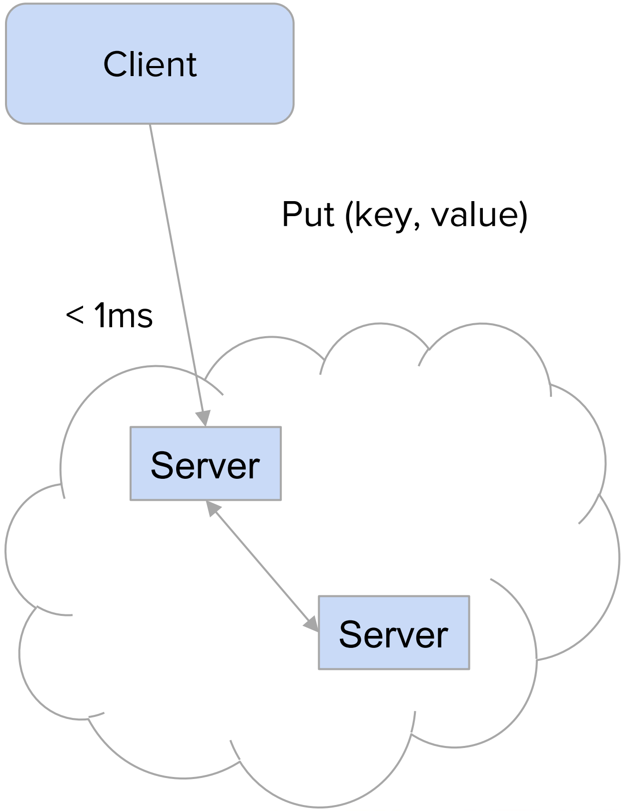
What is Geode?

- Distributed key-value store
- High available



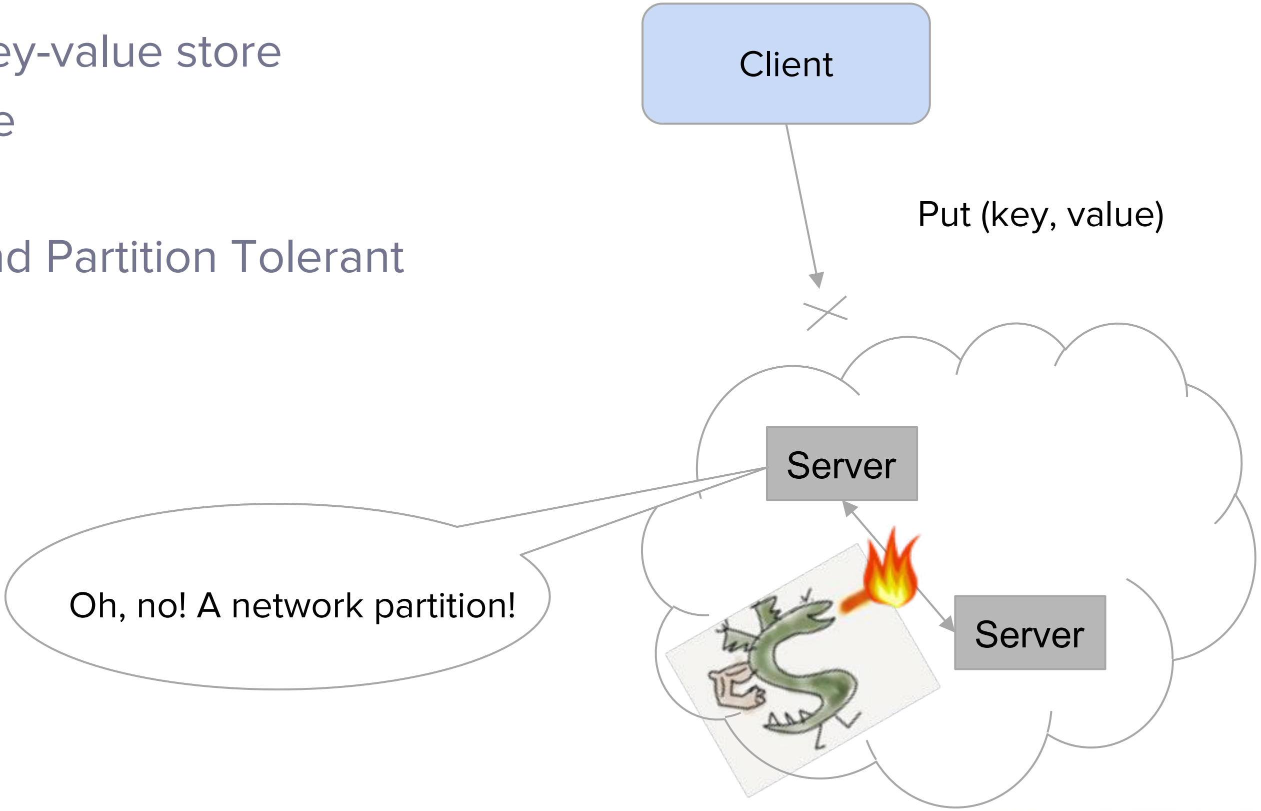
What is Geode?

- Distributed key-value store
- High available
- Low Latency



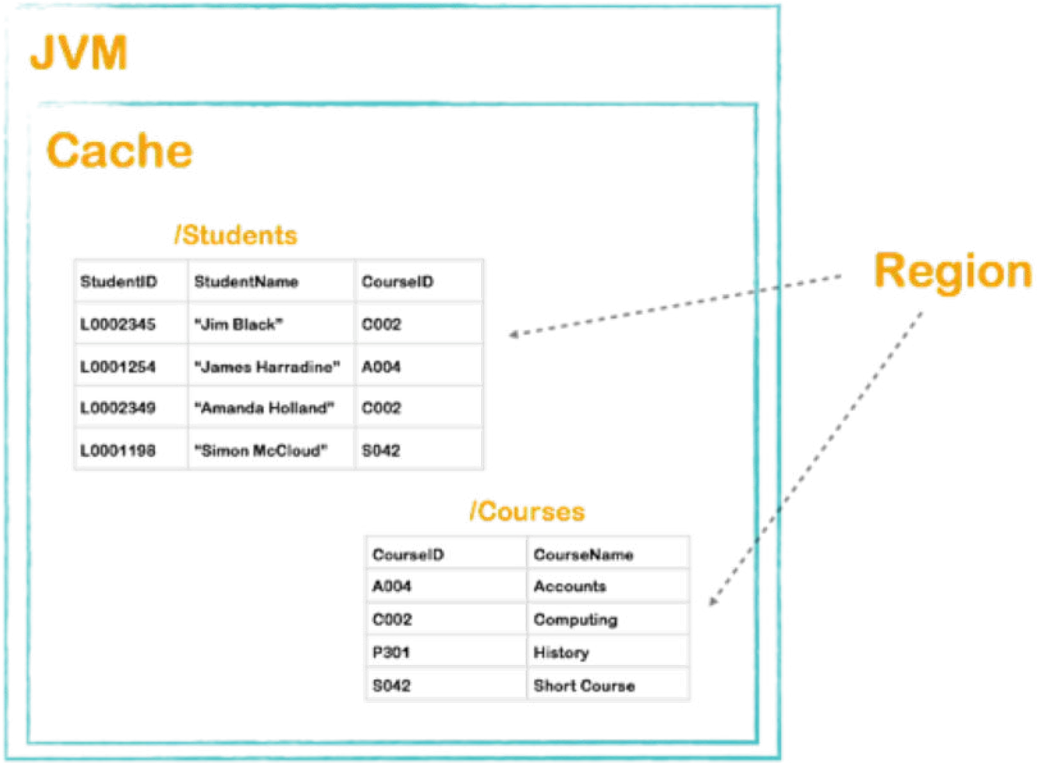
What is Geode?

- Distributed key-value store
- High available
- Low Latency
- Consistent and Partition Tolerant



Regions

- Synonymous to a Table in NoSQL terminology
- Stores Data in **<Key,Value>** pairs with unique Keys
- Divided into buckets across Cache Members



/Students

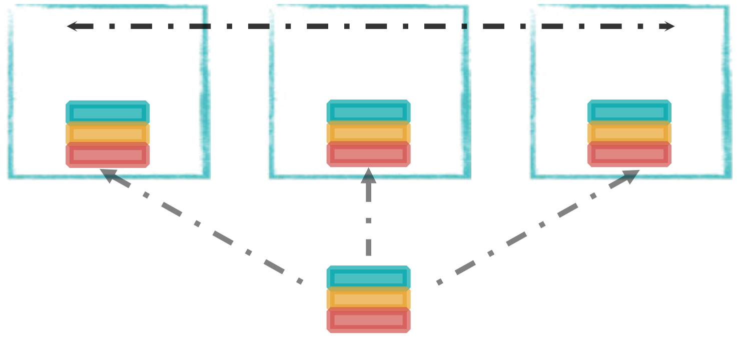
StudentID	StudentName	CourseID
L0002345	"Jim Black"	C002
L0001254	"James Harradine"	A004
L0002349	"Amanda Holland"	C002
L0001198	"Simon McCloud"	S042

/Courses

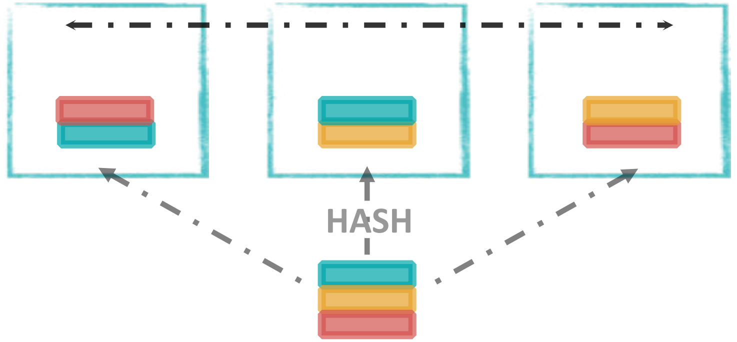
CourseID	CourseName
A004	Accounts
C002	Computing
P301	History
S042	Short Course

Regions Data Placement Design

Replicated



Partitioned



Pivotal

SpringOne Platform by Pivotal re:Cap Seoul 2019

Pivotal

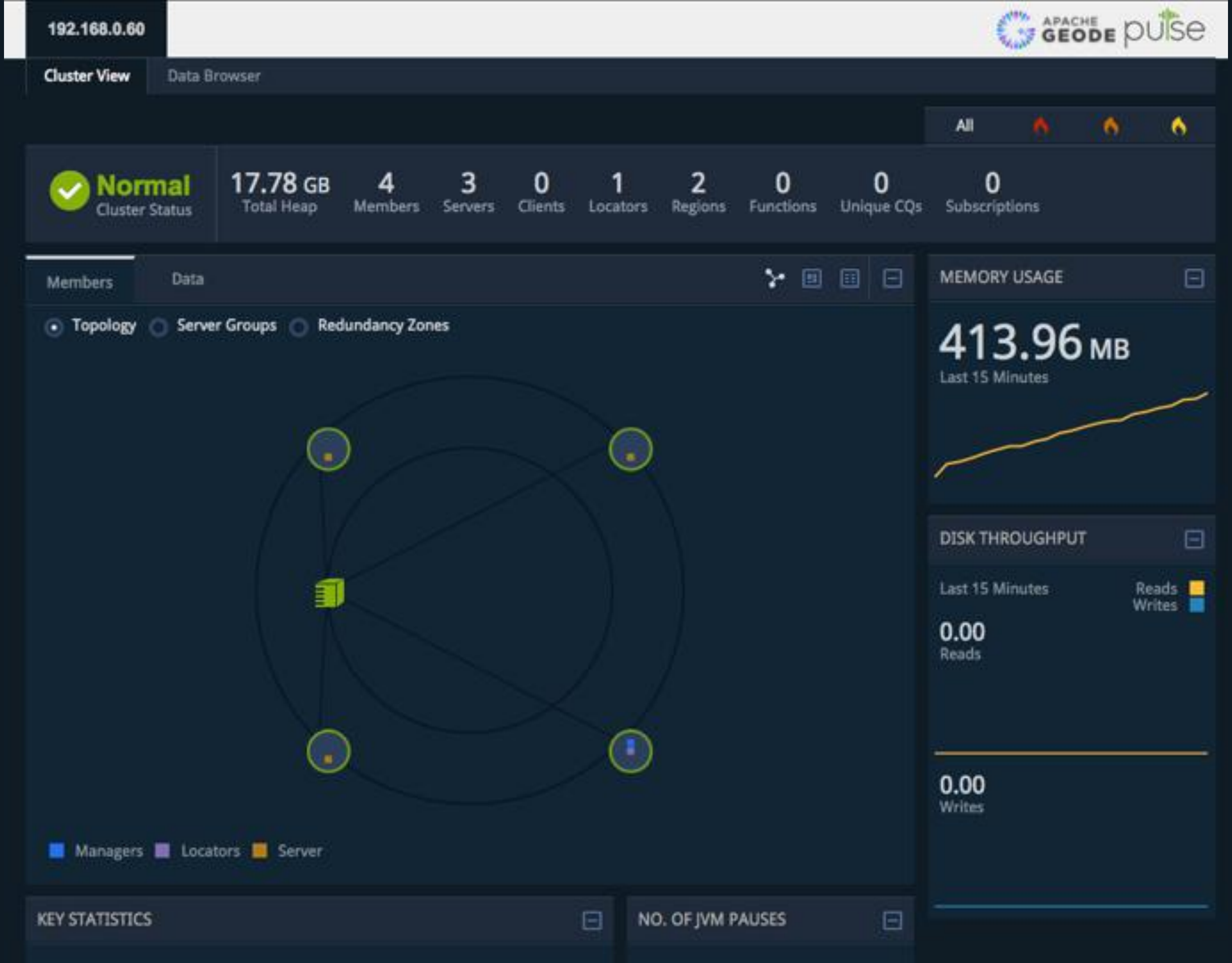
SpringOne Platform by Pivotal

re:Cap Seoul 2019

Visualize your Geode Metrics

Viewing Geode Metrics Before Now

- Internal metrics written to a local file in a proprietary format
  - Viewable with a custom viewing tool
  - Generally viewed after the events happened
- A subset published via JMX


192.168.0.60

Cluster View   Data Browser

All

Normal
Cluster Status

17.78 GB Total Heap
4 Members
3 Servers
0 Clients
1 Locators
2 Regions
0 Functions
0 Unique CQs
0 Subscriptions

Members   Data

[✓ Topology]   Server Groups   Redundancy Zones

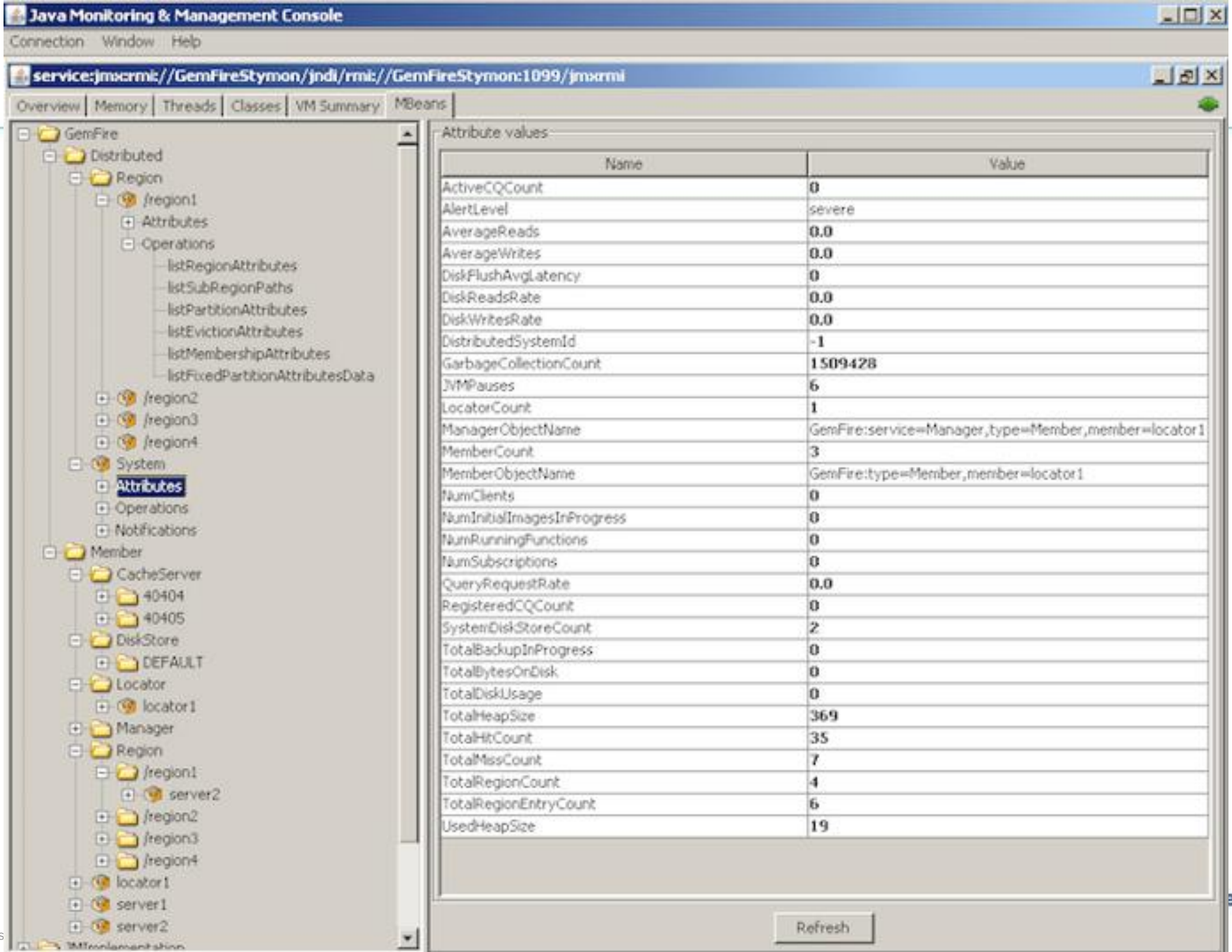
🟦 Managers 🟪 Locators 🟧 Server

MEMORY USAGE
413.96 MB
Last 15 Minutes

DISK THROUGHPUT
Last 15 Minutes
Reads 🟨
Writes 🟦
0.00
Reads
0.00
Writes

KEY STATISTICS

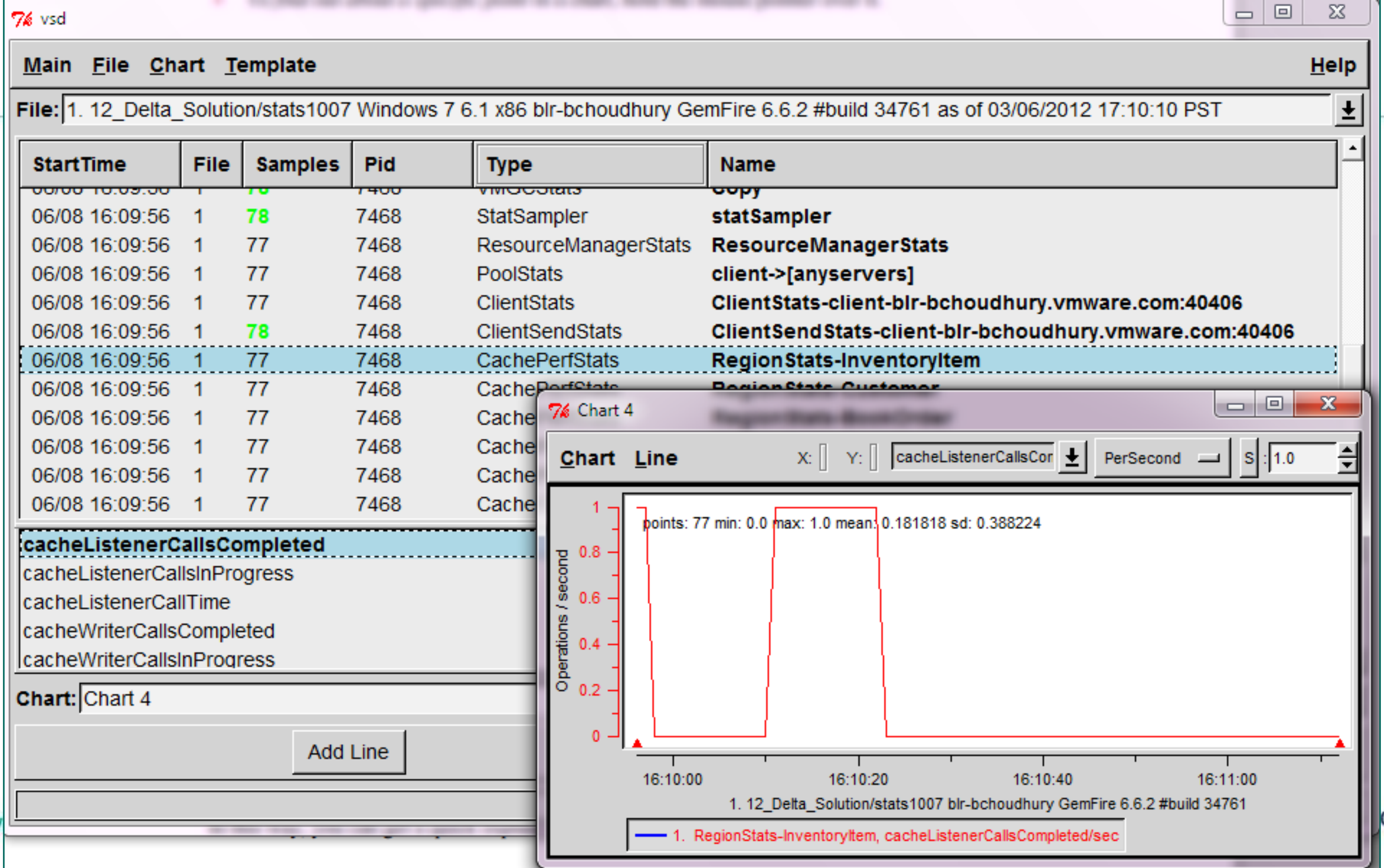
NO. OF JVM PAUSES


Java Monitoring & Management Console
Connection
Window
Help
service:jmxrmi://GemFireStymon/jndi/rmi://GemFireStymon:1099/jmxrmi
Overview
Memory
Threads
Classes
VM Summary
MBeans

```
GemFire
  Distributed
    Region
      /region1
        Attributes
        Operations
          listRegionAttributes
          listSubRegionPaths
          listPartitionAttributes
          listEvictionAttributes
          listMembershipAttributes
          listFixedPartitionAttributesData
      /region2
      /region3
      /region4
    System
      Attributes
      Operations
      Notifications
  Member
    CacheServer
      40404
      40405
    DiskStore
      DEFAULT
    Locator
      locator1
    Manager
    Region
      /region1
        server2
      /region2
      /region3
      /region4
    locator1
    server1
    server2
JMImplementation
```

Attribute values

Name	Value
ActiveCQCount	0
AlertLevel	severe
AverageReads	0.0
AverageWrites	0.0
DiskFlushAvgLatency	0
DiskReadsRate	0.0
DiskWritesRate	0.0
DistributedSystemId	-1
GarbageCollectionCount	1509428
JVMPauses	6
LocatorCount	1
ManagerObjectName	GemFire:service=Manager,type=Member,member=locator1
MemberCount	3
MemberObjectName	GemFire:type=Member,member=locator1
NumClients	0
NumInitialImagesInProgress	0
NumRunningFunctions	0
NumSubscriptions	0
QueryRequestRate	0.0
RegisteredCQCount	0
SystemDiskStoreCount	2
TotalBackupInProgress	0
TotalBytesOnDisk	0
TotalDiskUsage	0
TotalHeapSize	369
TotalHitCount	35
TotalMissCount	7
TotalRegionCount	4
TotalRegionEntryCount	6
UsedHeapSize	19



vsd

Main File Chart Template Help

File: 1. 12_Delta_Solution/stats1007 Windows 7 6.1 x86 blr-bchoudhury GemFire 6.6.2 #build 34761 as of 03/06/2012 17:10:10 PST

StartTime	File	Samples	Pid	Type	Name
06/08 16:09:56	1	78	7468	VMGCStats	Copy
06/08 16:09:56	1	78	7468	StatSampler	statSampler
06/08 16:09:56	1	77	7468	ResourceManagerStats	ResourceManagerStats
06/08 16:09:56	1	77	7468	PoolStats	client->[anyservers]
06/08 16:09:56	1	77	7468	ClientStats	ClientStats-client-blr-bchoudhury.vmware.com:40406
06/08 16:09:56	1	78	7468	ClientSendStats	ClientSendStats-client-blr-bchoudhury.vmware.com:40406
06/08 16:09:56	1	77	7468	CachePerfStats	RegionStats-InventoryItem
06/08 16:09:56	1	77	7468	CachePerfStats	RegionStats-Customer
06/08 16:09:56	1	77	7468	Cache	
06/08 16:09:56	1	77	7468	Cache	
06/08 16:09:56	1	77	7468	Cache	
06/08 16:09:56	1	77	7468	Cache	

cacheListenerCallsCompleted
cacheListenerCallsInProgress
cacheListenerCallTime
cacheWriterCallsCompleted
cacheWriterCallsInProgress

Chart: Chart 4

Add Line

Chart 4

Chart Line X: Y: cacheListenerCallsCor PerSecond S 1.0

points: 77 min: 0.0 max: 1.0 mean: 0.181818 sd: 0.388224

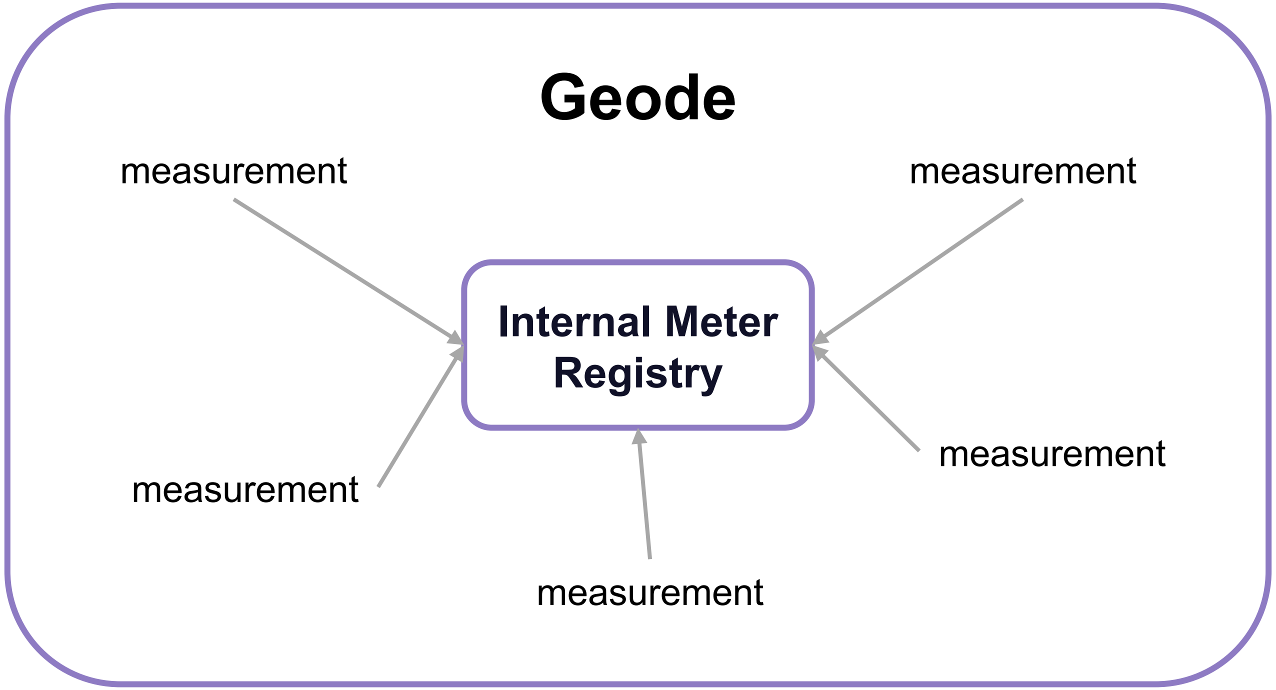
1. 12_Delta_Solution/stats1007 blr-bchoudhury GemFire 6.6.2 #build 34761

1. RegionStats-InventoryItem, cacheListenerCallsCompleted/sec

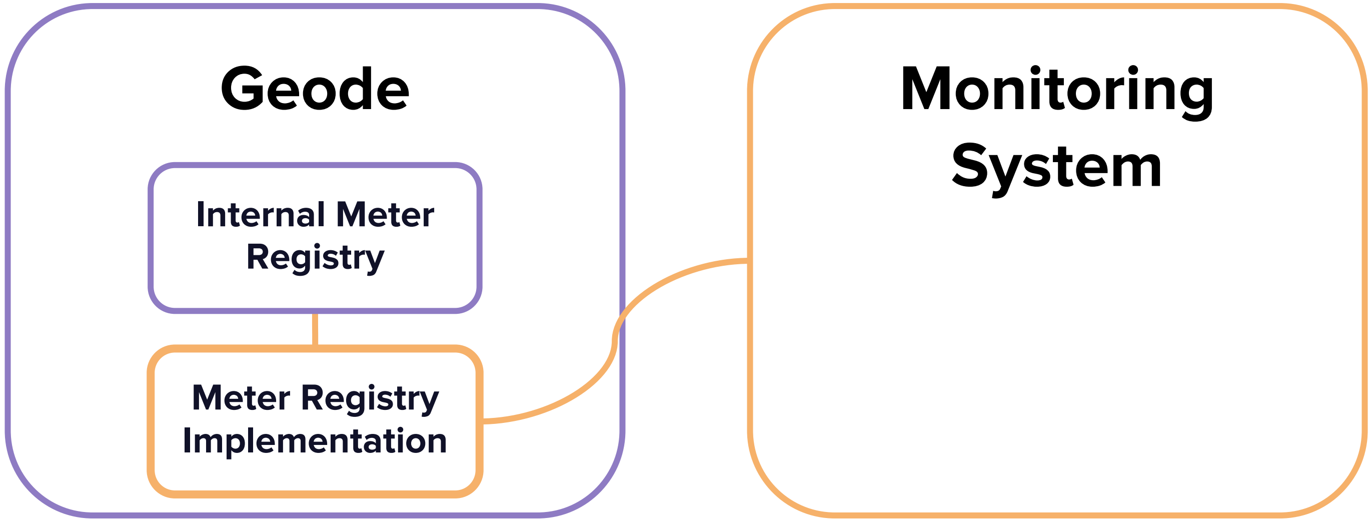
Micrometer in Geode

- Make key metrics visible in external monitoring systems
- View metrics while the system is running
- Augment metrics with details to aid understanding

Meter Registries



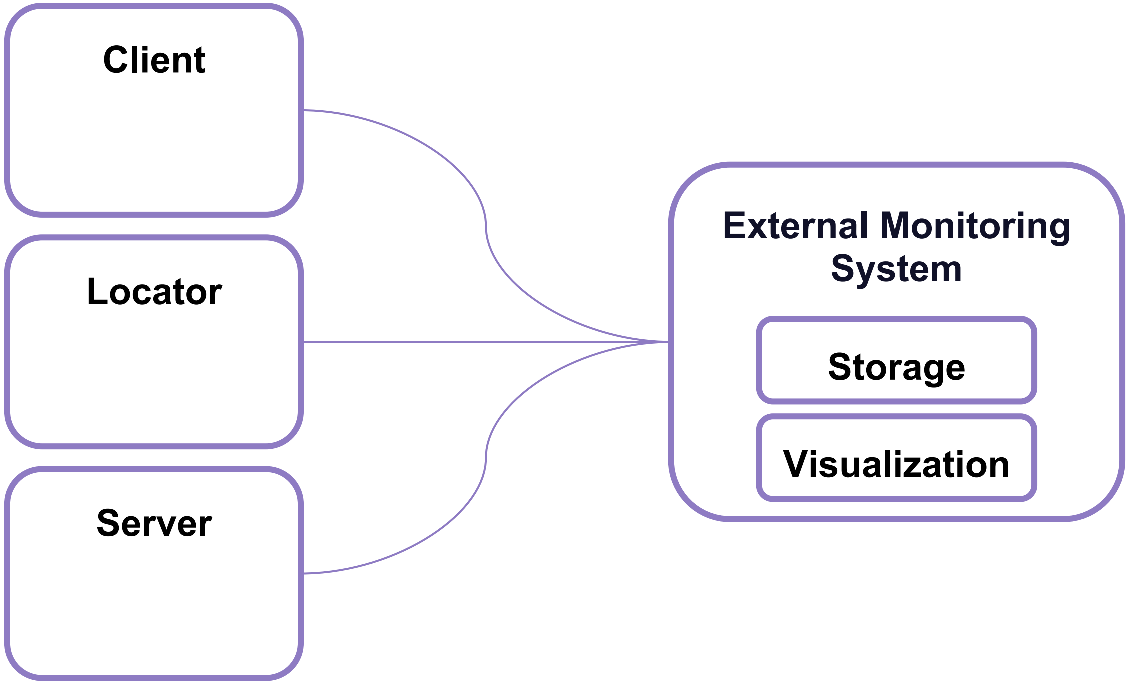
Connecting to Monitoring Systems

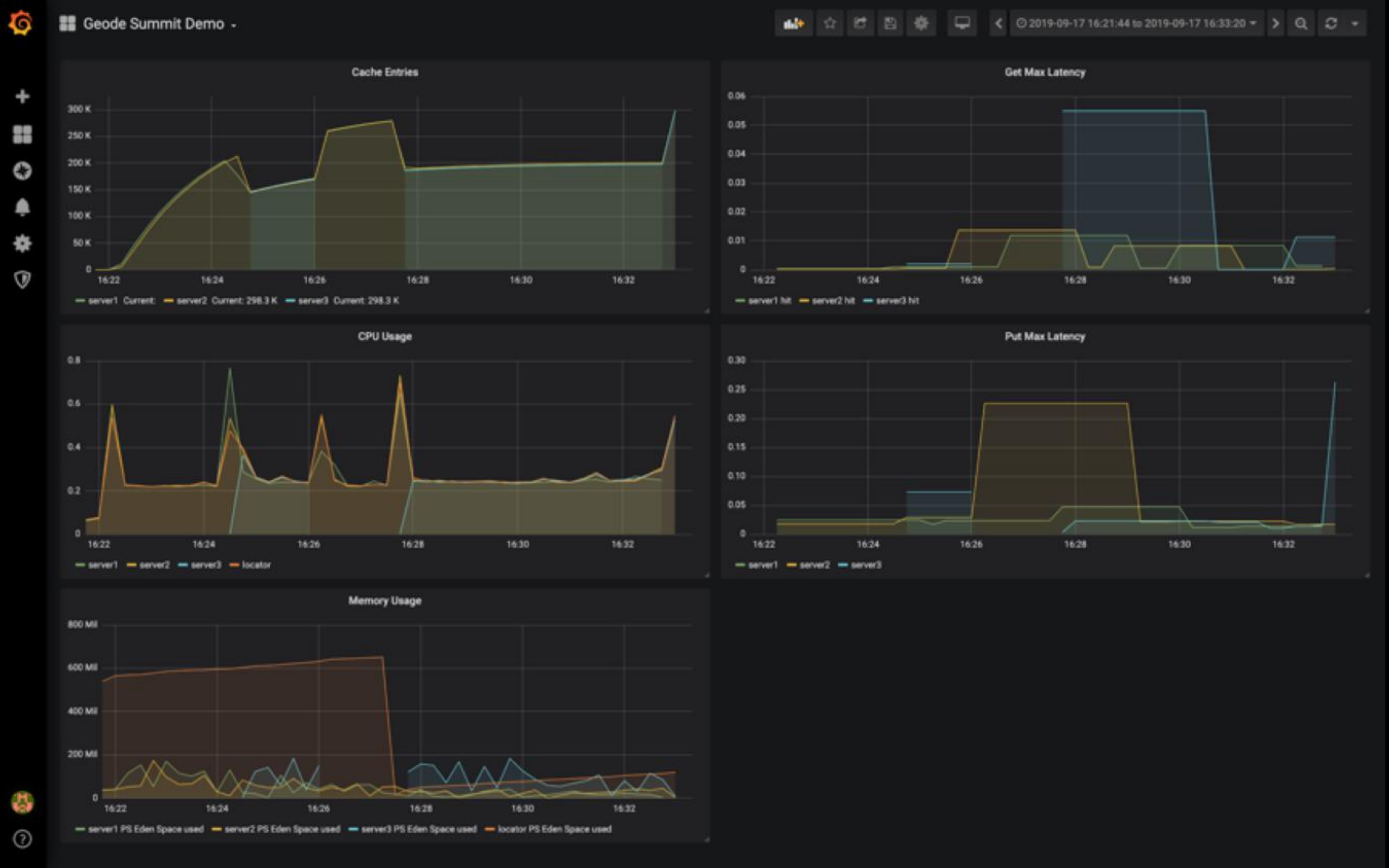


Connecting to Monitoring Systems

- Chose a Micrometer meter registry implementation that publishes to your monitoring system
  - For a list of implementations, see: http://micrometer.io/docs
- Create a MetricsPublishingService that adds your registry to Geode

Live View of Metrics




Geode Summit Demo
2019-09-17 16:21:44 to 2019-09-17 16:33:20
Cache Entries
server1 Current:
server2 Current: 298.3 K
server3 Current: 298.3 K
Get Max Latency
server1 hit
server2 hit
server3 hit
CPU Usage
server1
server2
server3
locator
Put Max Latency
server1
server2
server3
Memory Usage
server1 PS Eden Space used
server2 PS Eden Space used
server3 PS Eden Space used
locator PS Eden Space used

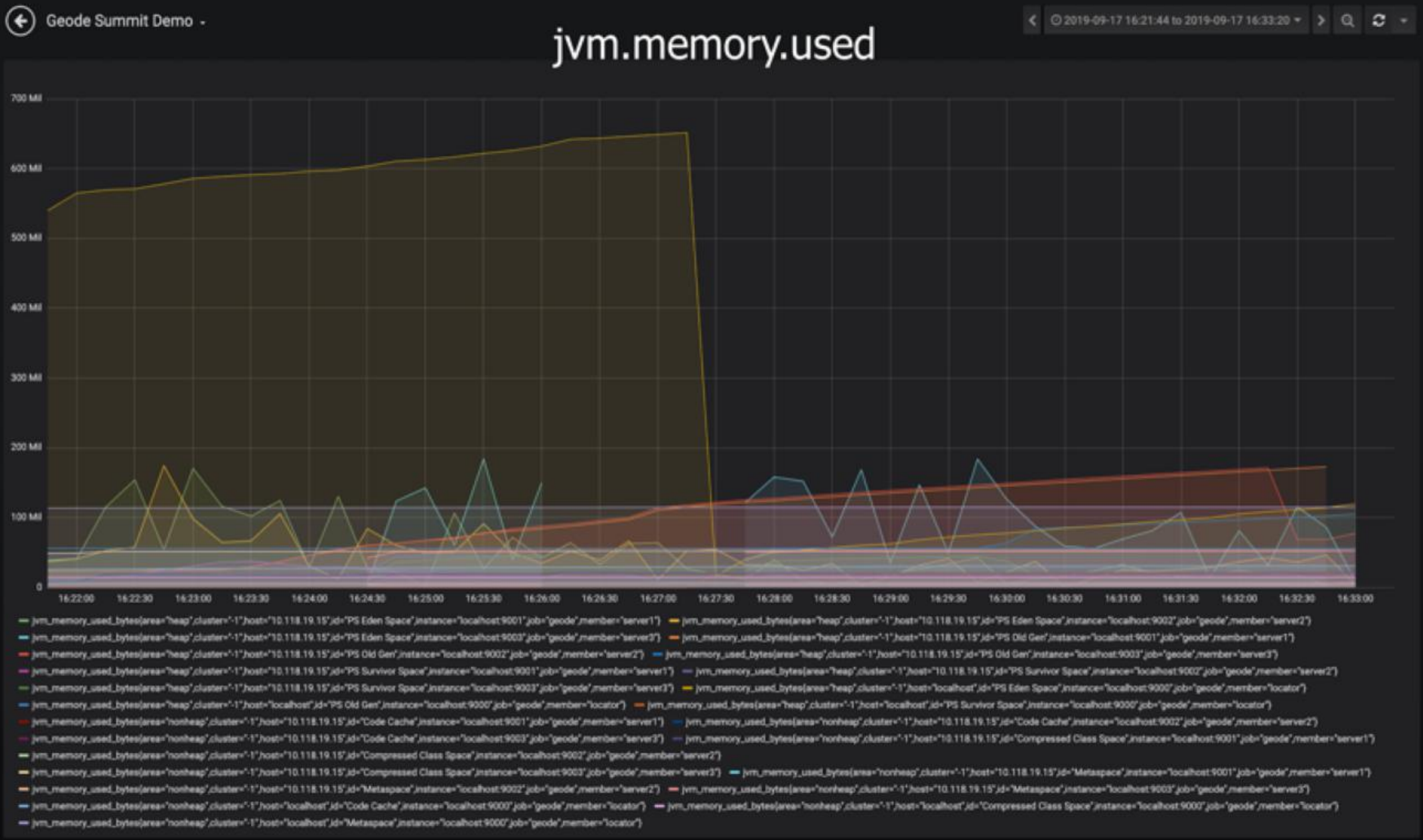
How Geode Records Measurements

- Uses Micrometer ***meters*** to record measurements
  - **eventsReceivedCounter.increment(delta);**
- Collects the meters in its ***meter registry***

Current Geode Meters

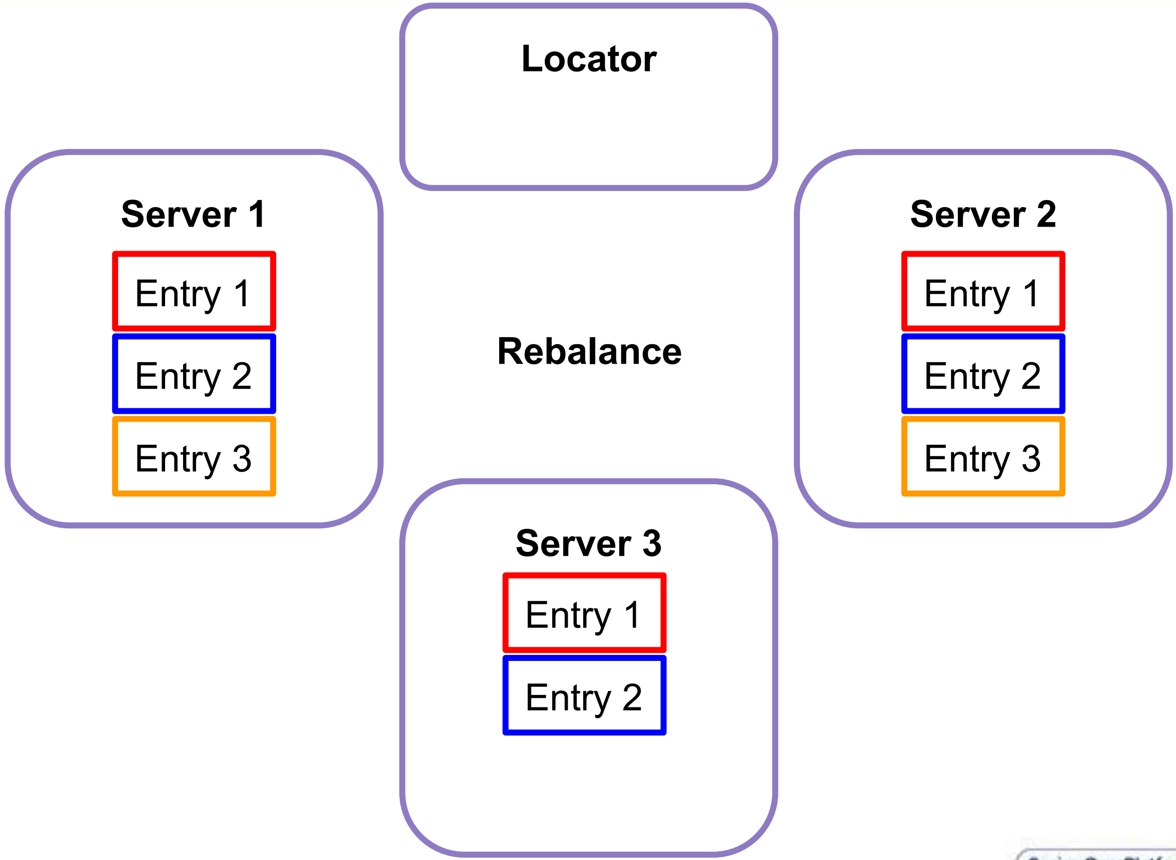
- System metrics
  - Processors
  - File descriptors
- JVM metrics
  - Process
  - Memory
  - Garbage collection
  - Threads
- Cache Entries

jvm.memory.used

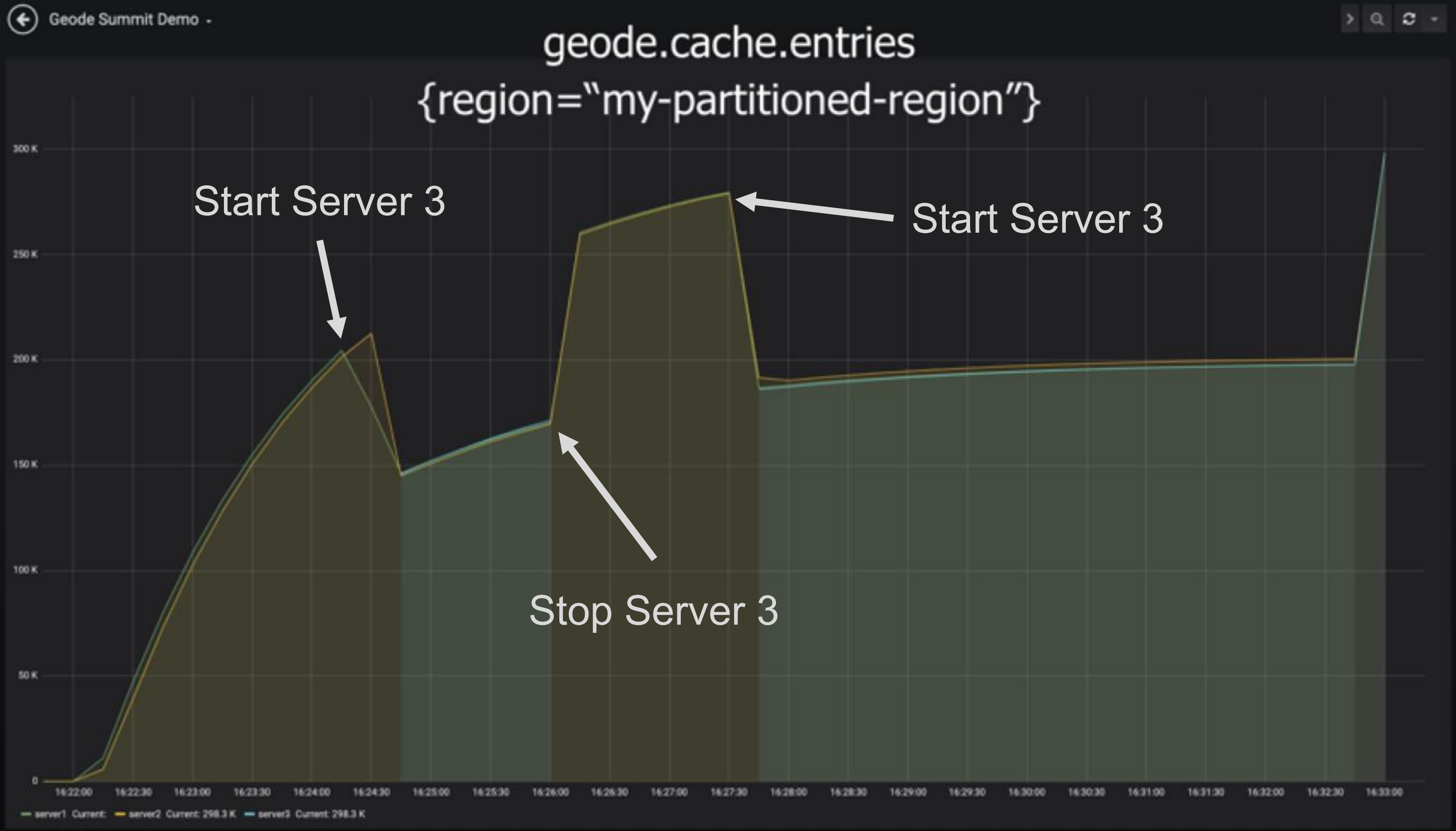


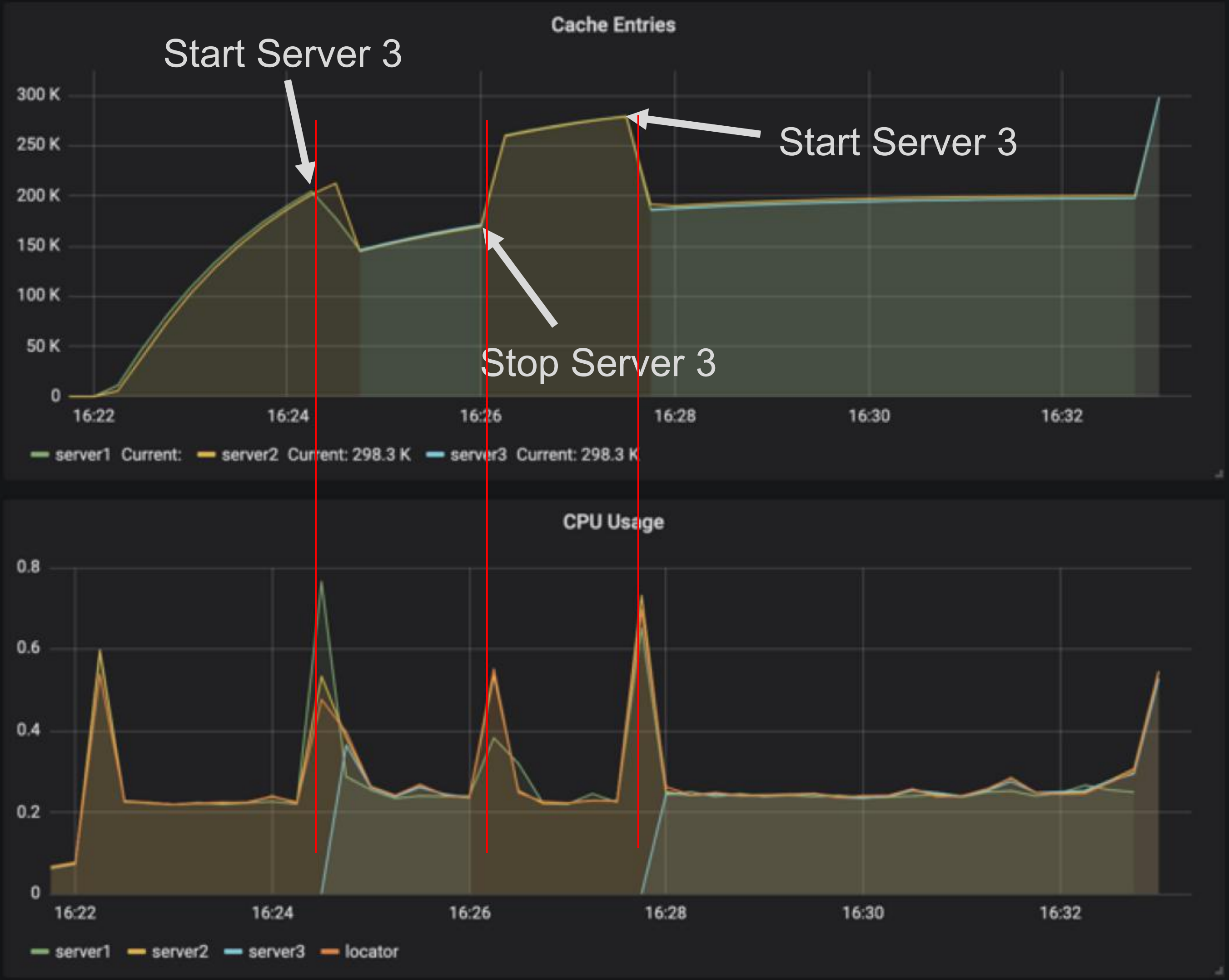
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Eden Space",instance="localhost:9001",job="geode",member="server1"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Eden Space",instance="localhost:9002",job="geode",member="server2"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Eden Space",instance="localhost:9003",job="geode",member="server3"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Old Gen",instance="localhost:9001",job="geode",member="server1"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Old Gen",instance="localhost:9002",job="geode",member="server2"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Old Gen",instance="localhost:9003",job="geode",member="server3"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Survivor Space",instance="localhost:9001",job="geode",member="server1"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Survivor Space",instance="localhost:9002",job="geode",member="server2"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="10.118.19.15",id="PS Survivor Space",instance="localhost:9003",job="geode",member="server3"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="localhost",id="PS Eden Space",instance="localhost:9000",job="geode",member="locator"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="localhost",id="PS Old Gen",instance="localhost:9000",job="geode",member="locator"}
- jvm_memory_used_bytes{area="heap",cluster="-1",host="localhost",id="PS Survivor Space",instance="localhost:9000",job="geode",member="locator"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Code Cache",instance="localhost:9001",job="geode",member="server1"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Code Cache",instance="localhost:9002",job="geode",member="server2"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Code Cache",instance="localhost:9003",job="geode",member="server3"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Compressed Class Space",instance="localhost:9001",job="geode",member="server1"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Compressed Class Space",instance="localhost:9002",job="geode",member="server2"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Compressed Class Space",instance="localhost:9003",job="geode",member="server3"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Metaspace",instance="localhost:9001",job="geode",member="server1"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Metaspace",instance="localhost:9002",job="geode",member="server2"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="10.118.19.15",id="Metaspace",instance="localhost:9003",job="geode",member="server3"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="localhost",id="Code Cache",instance="localhost:9000",job="geode",member="locator"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="localhost",id="Compressed Class Space",instance="localhost:9000",job="geode",member="locator"}
- jvm_memory_used_bytes{area="nonheap",cluster="-1",host="localhost",id="Metaspace",instance="localhost:9000",job="geode",member="locator"}

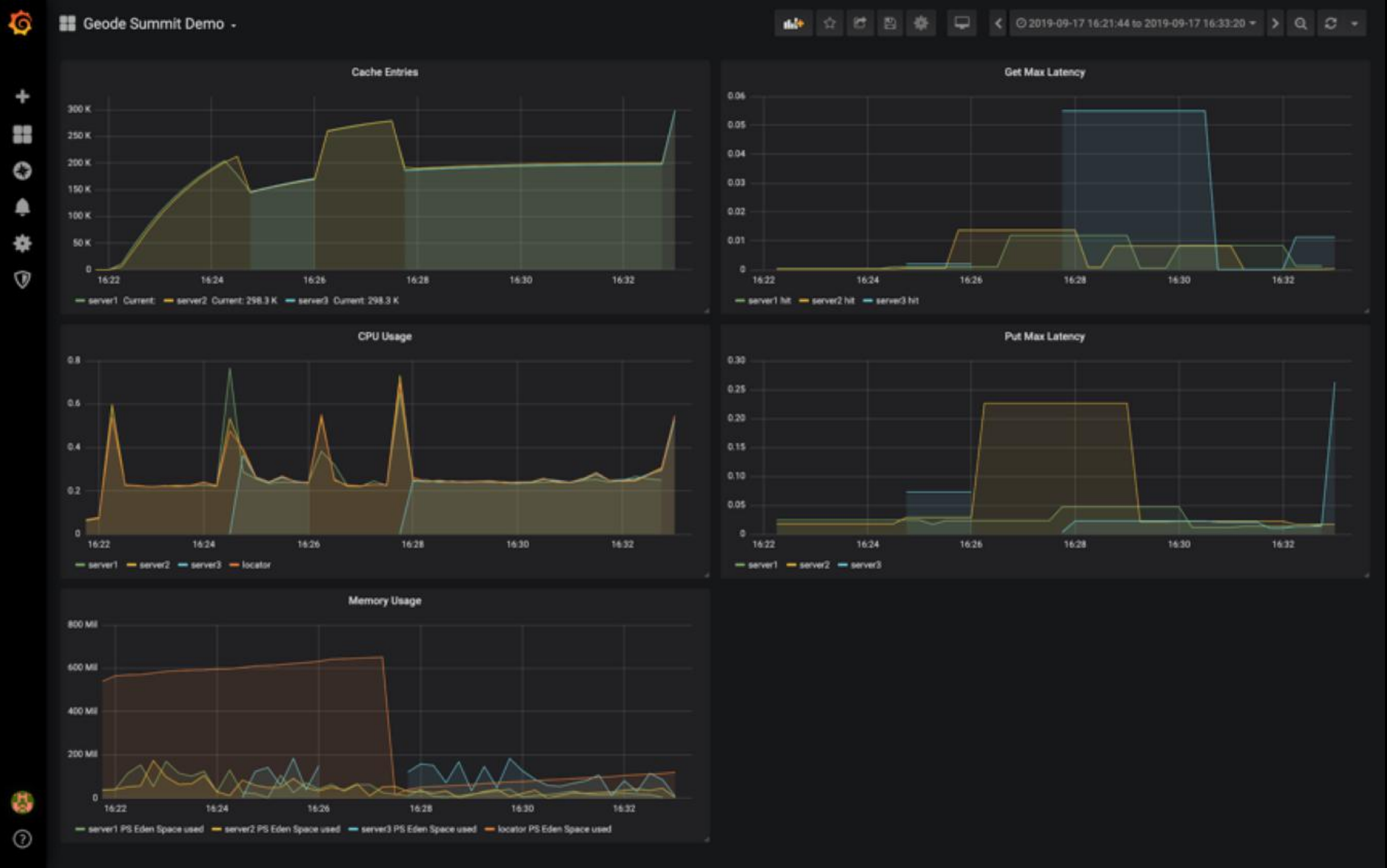
Scenario – Partition Region Redundancy 1



geode.cache.entries {region=“my-partitioned-region”}




Cache Entries
Start Server 3
Start Server 3
Stop Server 3
300 K
250 K
200 K
150 K
100 K
50 K
0
16:22
16:24
16:26
16:28
16:30
16:32
server1 Current:
server2 Current: 298.3 K
server3 Current: 298.3 K
CPU Usage
0.8
0.6
0.4
0.2
0
16:22
16:24
16:26
16:28
16:30
16:32
server1
server2
server3
locator


Geode Summit Demo
2019-09-17 16:21:44 to 2019-09-17 16:33:20
Cache Entries
server1 Current:
server2 Current: 298.3 K
server3 Current: 298.3 K
Get Max Latency
server1 hit
server2 hit
server3 hit
CPU Usage
server1
server2
server3
locator
Put Max Latency
server1
server2
server3
Memory Usage
server1 PS Eden Space used
server2 PS Eden Space used
server3 PS Eden Space used
locator PS Eden Space used

The Future Depends on You

- Send Geode's metrics to your favorite monitoring system
- What works well for you?
- What additional metrics would best help you manage your Geode systems?

For Further Information

- Micrometer: http://micrometer.io/
- Prometheus: https://prometheus.io
- Grafana: https://grafana.com
- Publishing Geode Metrics to External Monitoring Systems:
- https://cwiki.apache.org/confluence/display/GEODE/Publishing+Geode+Metrics+to+External+Monitoring+Systems
- Example code to publish metrics to Prometheus:
- https://github.com/moleske/geode-registry-example

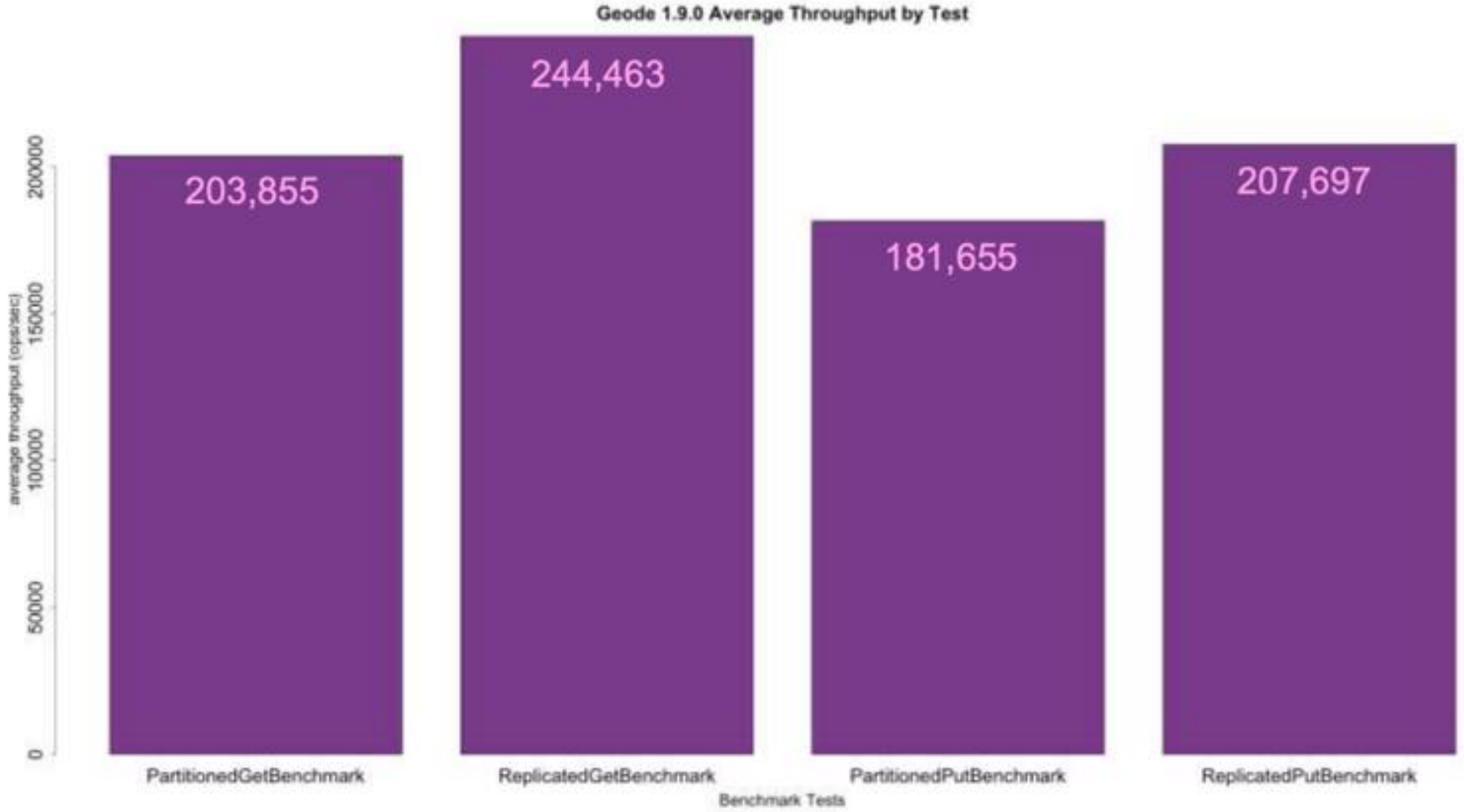
Pivotal

SpringOne Platform by Pivotal

re:Cap Seoul 2019

Performance in Geode

Performance of Geode 1.9.0



Pivotal

Seoul 2019

Creating the Geode Benchmark – Features

- Run by anyone interested in Geode
- Have others create benchmarks
- Visualize benchmark results over time
- Increase benchmark coverage of Geode

Creating the Geode Benchmark – Goals

- On demand
- Against any revision of Geode
- On AWS cluster deployment of Geode
- On any dev machine in the office
- From Concourse CI pipeline
- With a profiler attached
- Compare two runs of benchmarks for performance changes

Tests Currently in the Benchmarks

- **ReplicatedGetBenchmark**
- ReplicatedGetLongBenchmark
- **ReplicatedPutBenchmark**
- ReplicatedPutLongtBenchmark
- ReplicatedPutAllBenchmark
- ReplicatedPutAllLongBenchmark
- ReplicatedFunctionExecutionBenchmark
- ReplicatedFunctionExecutionWithArgumentsBenchmark
- ReplicatedFunctionExecutionWithFiltersBenchmark

- **PartitionedGetBenchmark**
- PartitionedGetLongBenchmark
- **PartitionedPutBenchmark**
- PartitionedPutLongBenchmark
- PartitionedPutAllBenchmark
- PartitionedPutAllLongBenchmark
- PartitionedIndexedQueryBenchmark
- PartitionedFunctionExecutionWithArgumentsBenchmark
- PartitionedFunctionExecutionWithFiltersBenchmark

Tests Currently in the Benchmarks

https://github.com/apache/geode-benchmarks

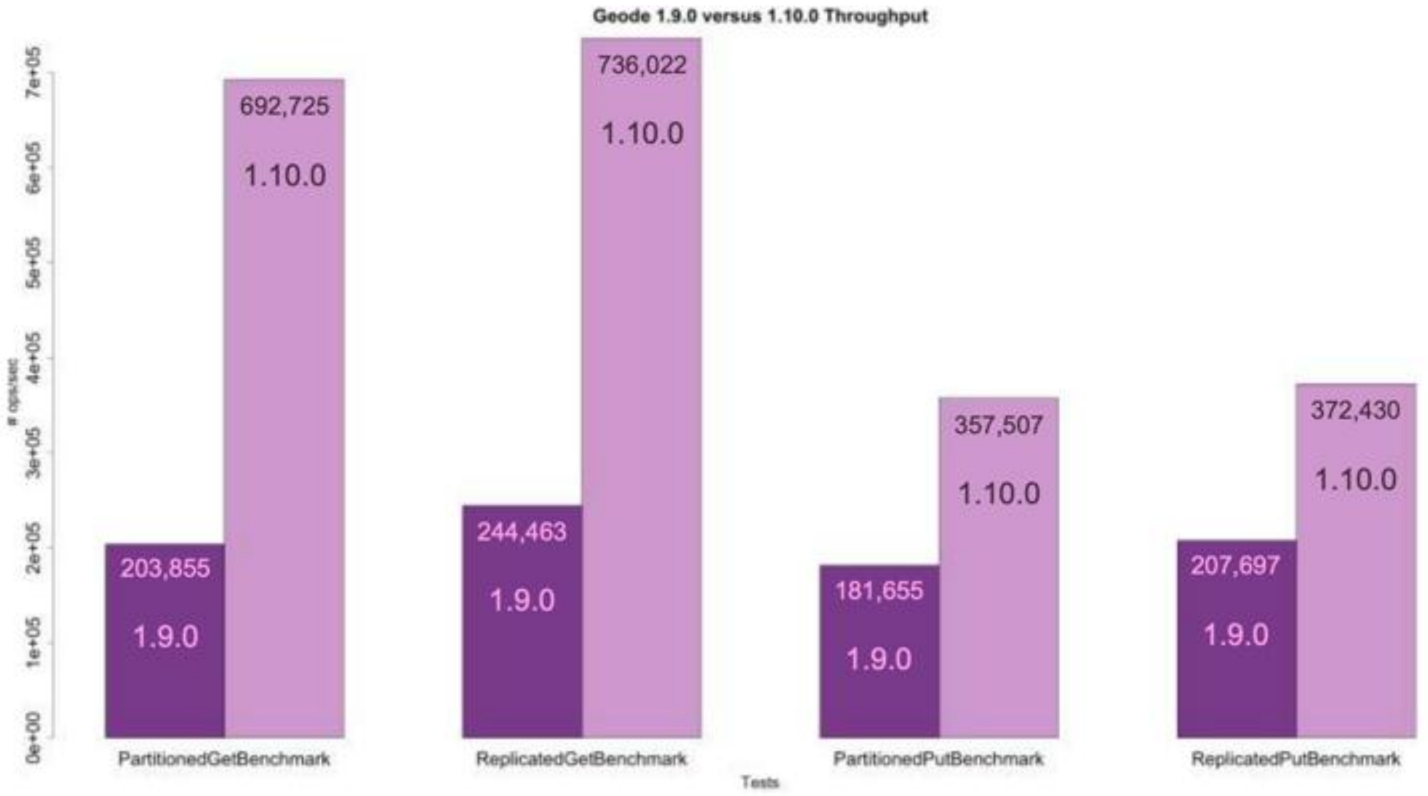
Other Tested Configurations

- With SSL
- With JDKs: 8, 11, 12, 13
- With Security Manager
- With Garbage Collectors:
    - CMS
    - G1
    - Z
    - Shenandoah
- Adjustable max heap size

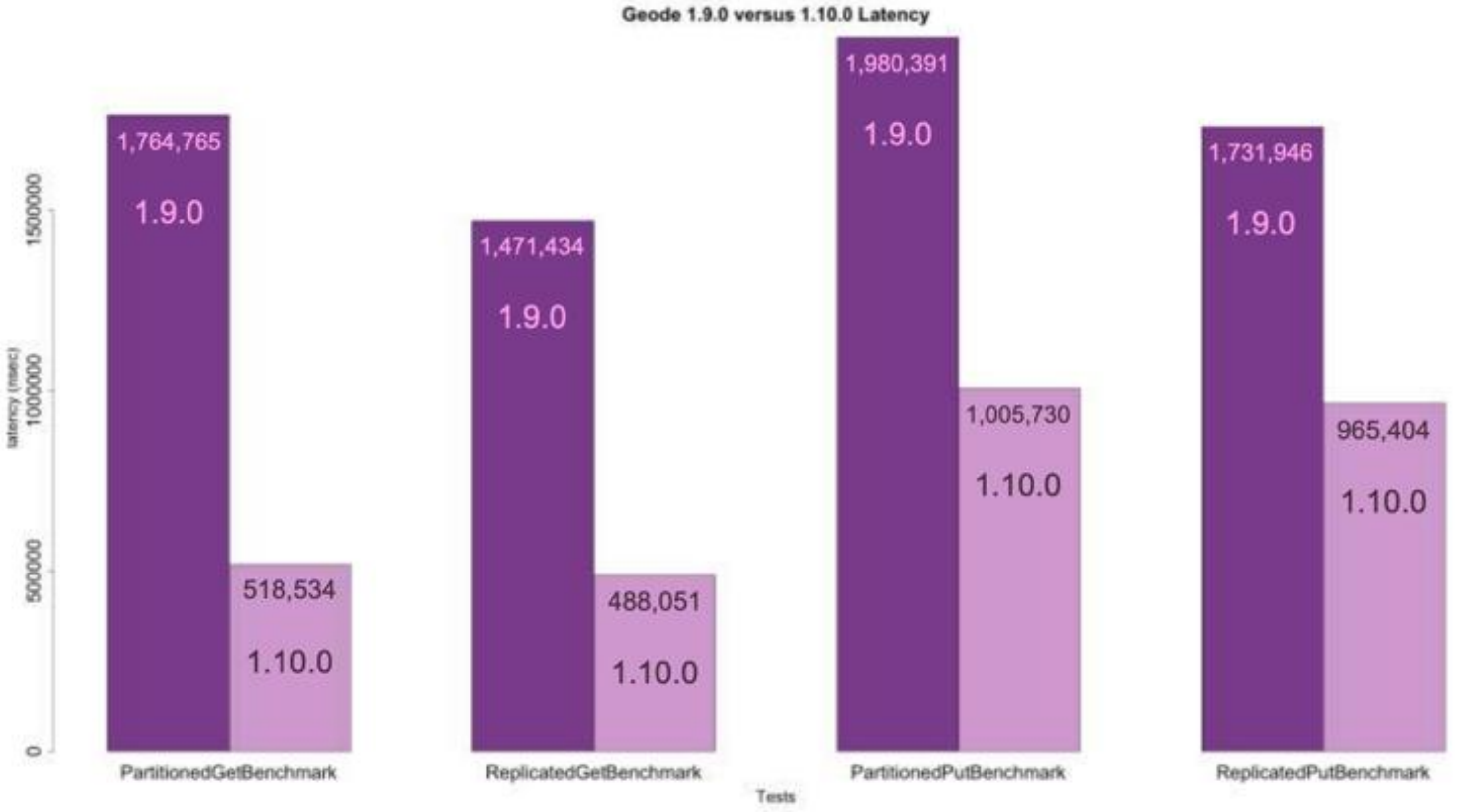
Finding Performance Bottleneck

- Monitor locks
- Thread Park/Unpark Reentrant Locks
- Allocations/GC
- Overuse of synchronization
- Getting a system property in a hot path
- Lazy initialization of objects in a hot path
- Synchronization on a container (ex. Hash map)

Comparing Performance of 1.9.0 & 1.10.0



Comparing Performance of 1.9.0 & 1.10.0



Pivotal

SpringOne Platform by Pivotal

re:Cap Seoul 2019

Reactive Event Processing with Geode

reactive programming is about...

“
non-blocking, event-driven applications that scale with a small number of threads with backpressure as a key ingredient that aims to ensure producers do not overwhelm consumers”

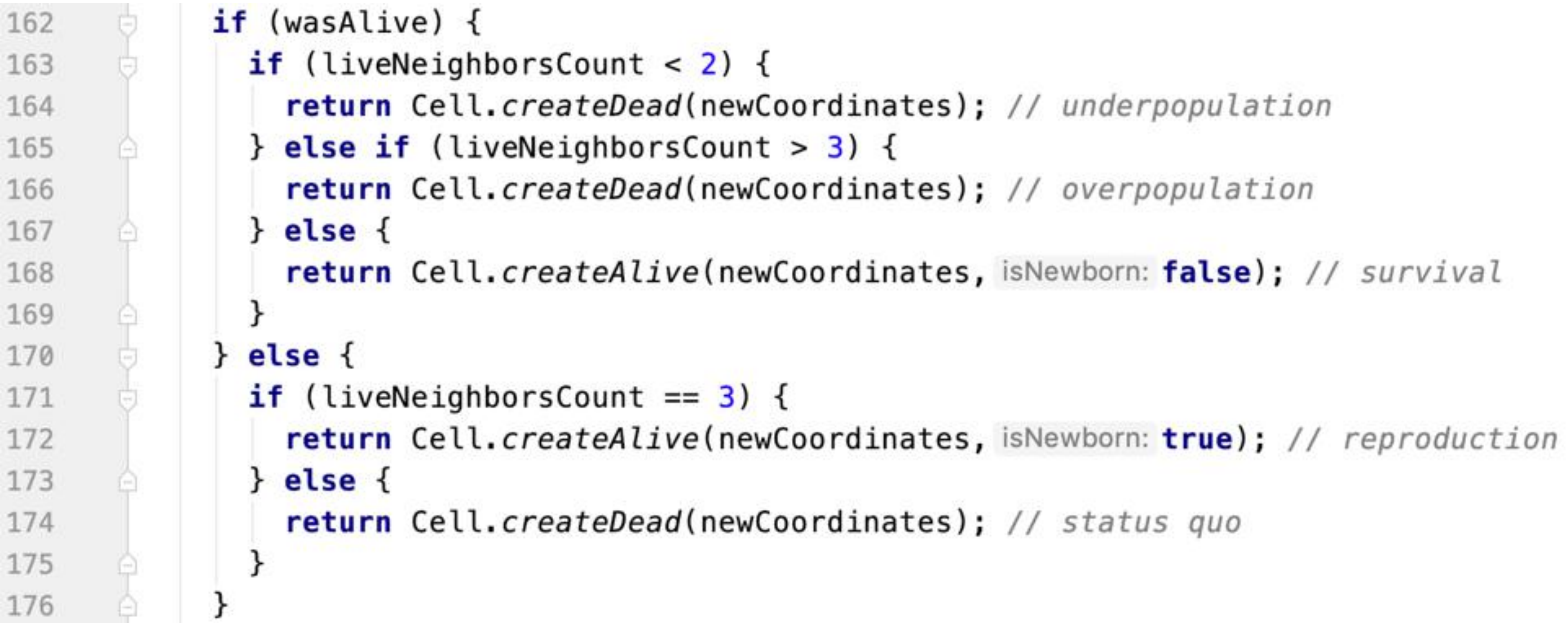
- Rossen Stoyanchev
spring.io blog July 28, 2016

Questions

- Geode as reactive consumer (Subscriber)
  - How do active APIs fit with Geode's can Geode's load shedding policy be adapted to reactive back-pressure

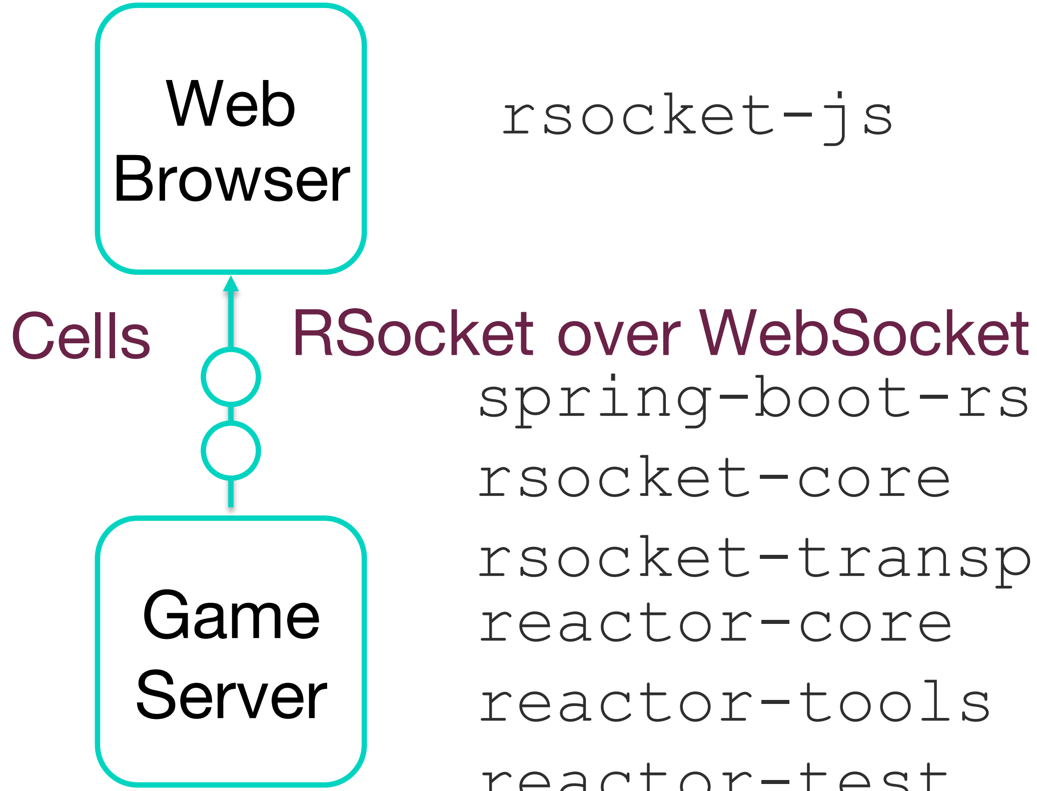
- Geode as reactive producer (Publisher)
  - Again, how do reactive APIs fit Geode's
  - Can Geode produce long data streams incrementally

Conway’s Game of Life



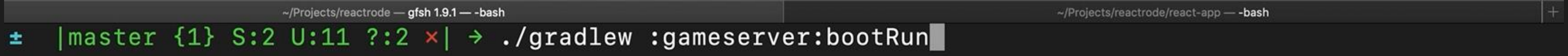
```java
if (wasAlive) {
  if (liveNeighborsCount < 2) {
    return Cell.createDead(newCoordinates); // underpopulation
  } else if (liveNeighborsCount > 3) {
    return Cell.createDead(newCoordinates); // overpopulation
  } else {
    return Cell.createAlive(newCoordinates, isNewborn: false); // survival
  }
} else {
  if (liveNeighborsCount == 3) {
    return Cell.createAlive(newCoordinates, isNewborn: true); // reproduction
  } else {
    return Cell.createDead(newCoordinates); // status quo
  }
}
```

A Reactive Toy



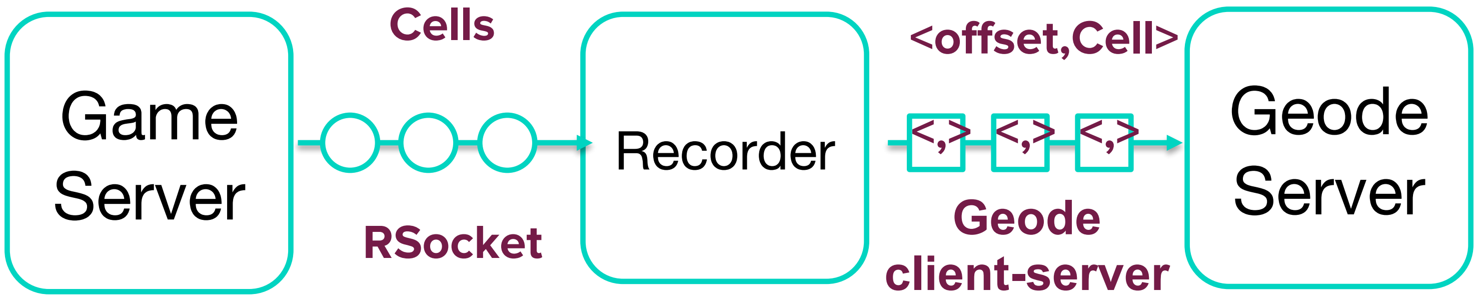
```
rsocket-js
```

```
spring-boot-rsocket
rsocket-core
rsocket-transport-netty
reactor-core
reactor-tools
reactor-test
```



```
~/Projects/reactrode — gfsh 1.9.1 — -bash        ~/Projects/reactrode/react-app — -bash
±  |master {1} S:2 U:11 ?:2 ×| → ./gradlew :gameserver:bootRun
```

Record

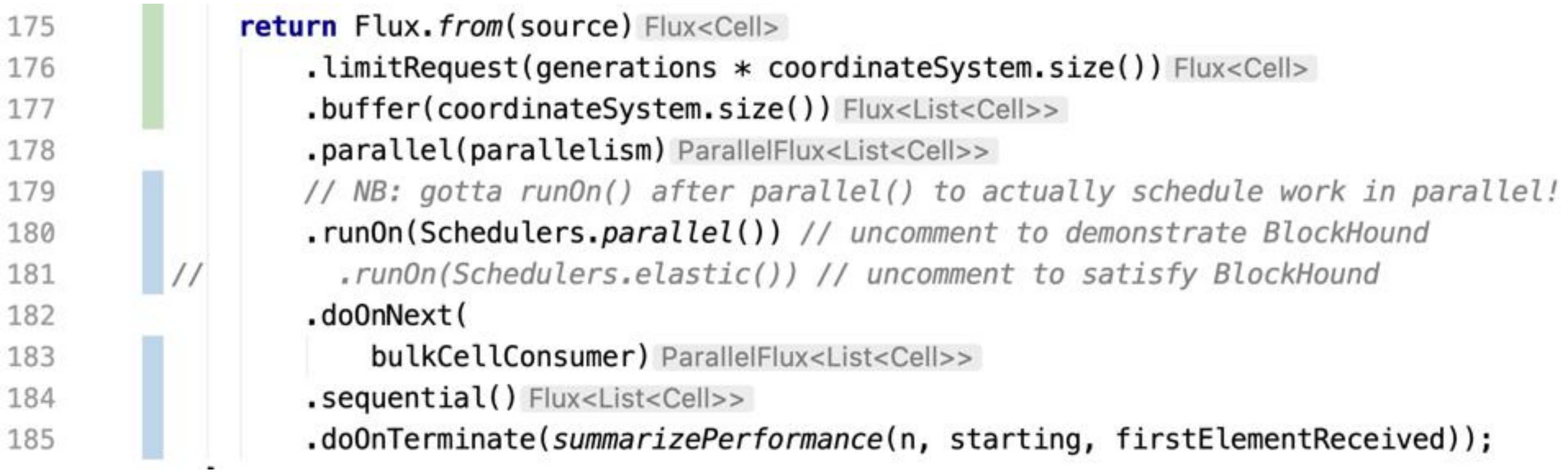


```
spring-boot-rsocket
rsocket-core
rsocket-transport-netty

reactor-core
reactor-tools
reactor-test
```

… +

```
spring-geode-starter
spring-data-geode-test
spring-boot-starter-test
```

Parallel putAll



```
return Flux.from(source) Flux<Cell>
    .limitRequest(generations * coordinateSystem.size()) Flux<Cell>
    .buffer(coordinateSystem.size()) Flux<List<Cell>>
    .parallel(parallelism) ParallelFlux<List<Cell>>
    // NB: gotta runOn() after parallel() to actually schedule work in parallel!
    .runOn(Schedulers.parallel()) // uncomment to demonstrate BlockHound
//      .runOn(Schedulers.elastic()) // uncomment to satisfy BlockHound
    .doOnNext(
        bulkCellConsumer) ParallelFlux<List<Cell>>
    .sequential() Flux<List<Cell>>
    .doOnTerminate(summarizePerformance(n, starting, firstElementReceived));
```

Pivotal

SpringOne Platform by Pivotal re:Cap Seoul 2019

{put,putAll} x {serial,parallel}

	put	**putAll**
serial	8	23
parallel	21	124

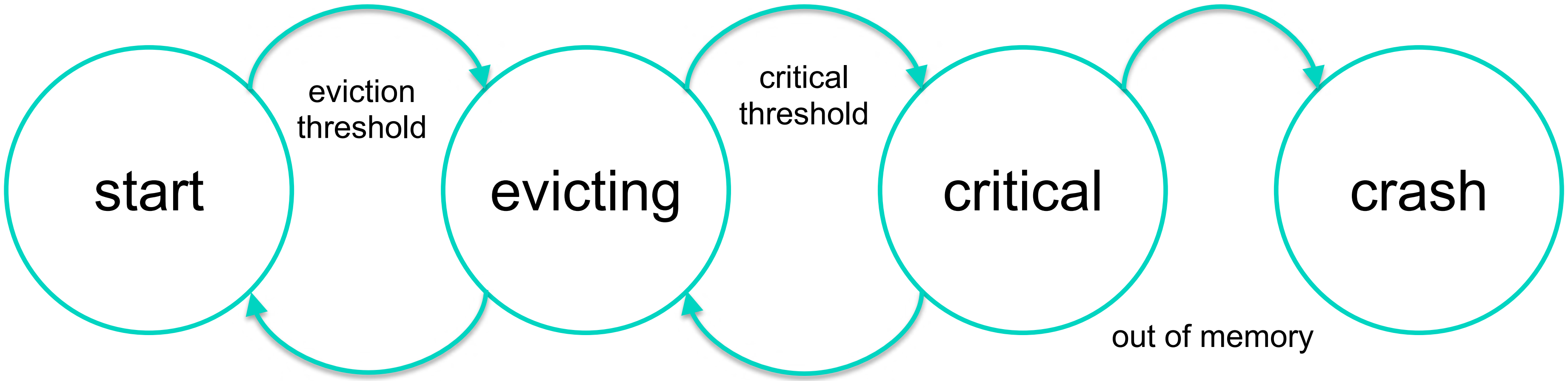
thousands of Cells per second
MacBook Pro; 8-way parallelism
de-tuned—for *relative* comparison only

But remember...

“reactive programming is about…
non-blocking…”

- Rossen Stoyanchev
spring.io blog July 28, 2016

Geode ResourceManager

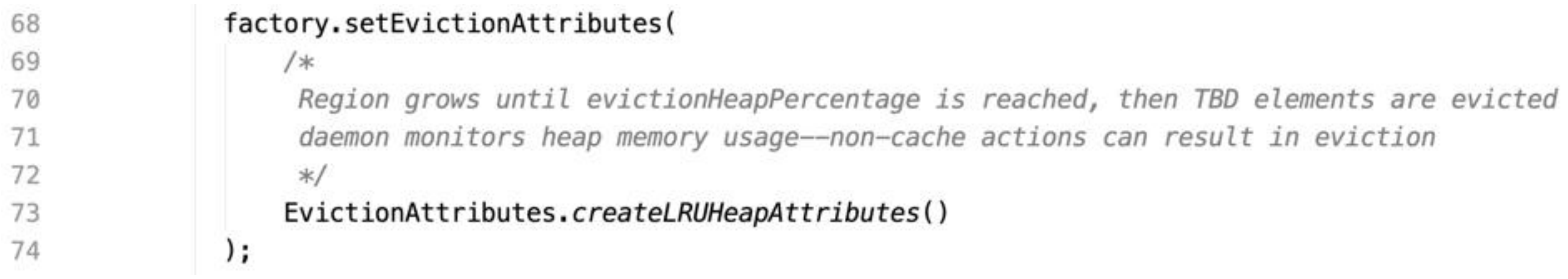


evictions happening

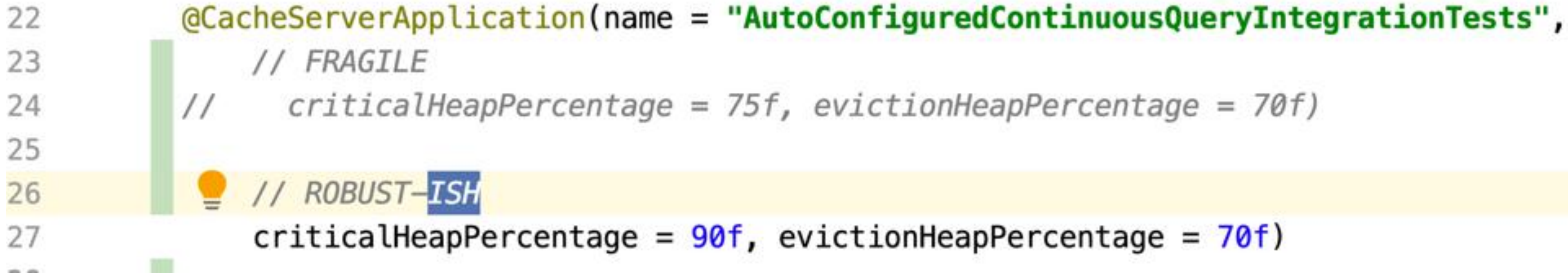
puts + queries serviced

LOAD SHEDDING

Eviction Config (purposely fragile)

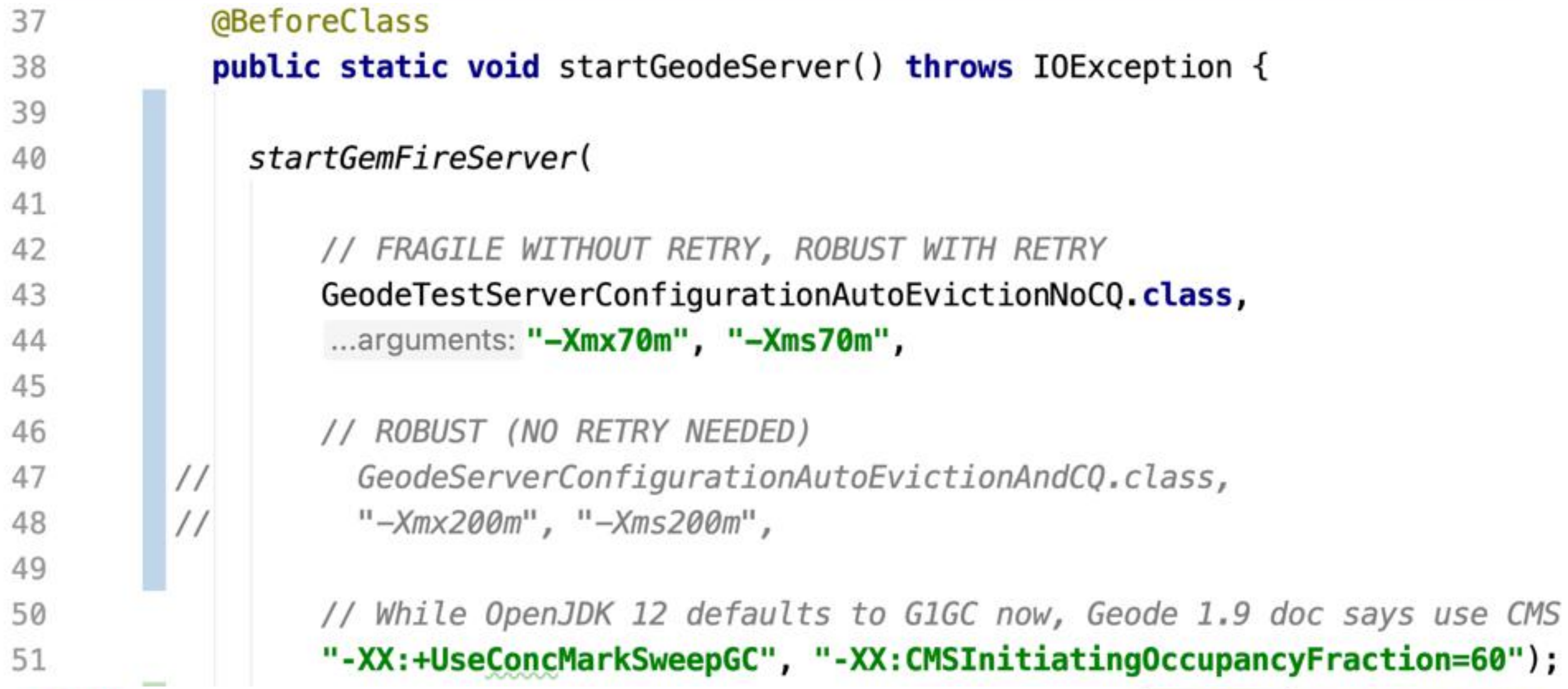


```
        factory.setEvictionAttributes(
            /*
             Region grows until evictionHeapPercentage is reached, then TBD elements are evicted
             daemon monitors heap memory usage--non-cache actions can result in eviction
             */
            EvictionAttributes.createLRUHeapAttributes()
        );
```



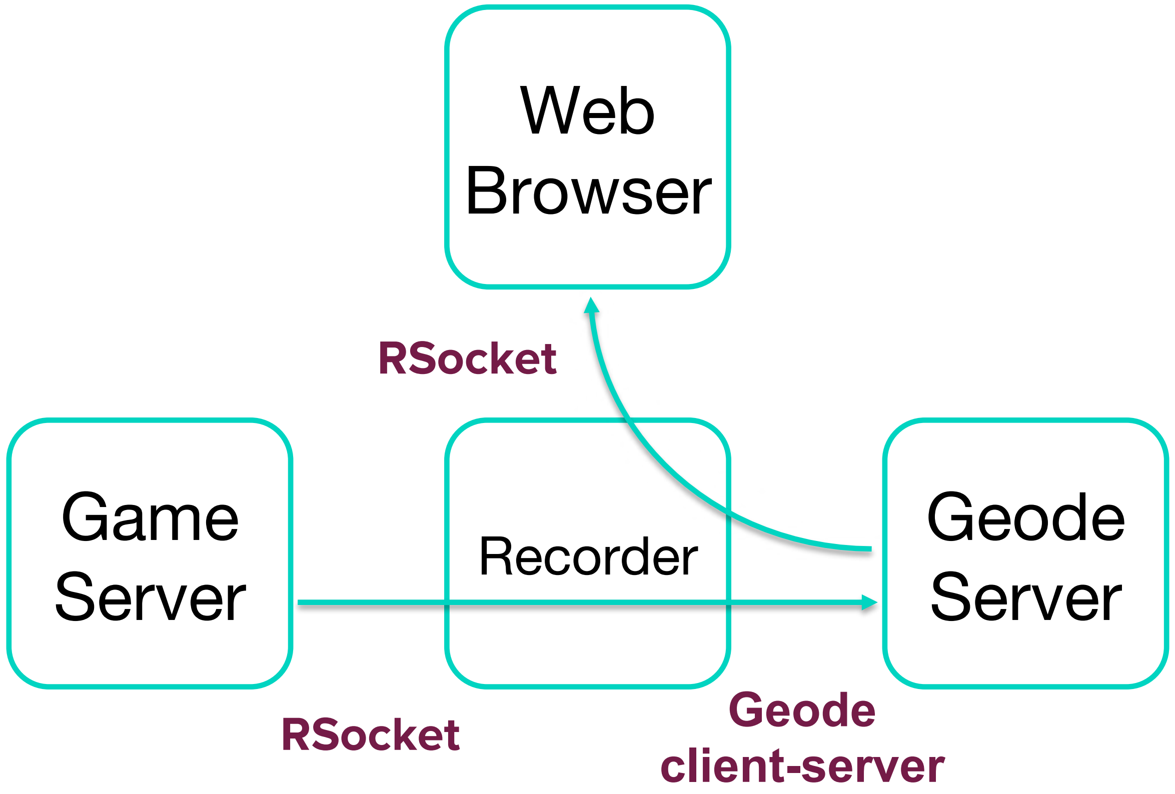
```
@CacheServerApplication(name = "AutoConfiguredContinuousQueryIntegrationTests",
    // FRAGILE
//      criticalHeapPercentage = 75f, evictionHeapPercentage = 70f)

    // ROBUST-ISH
    criticalHeapPercentage = 90f, evictionHeapPercentage = 70f)
```

Eviction Config (purposely fragile)

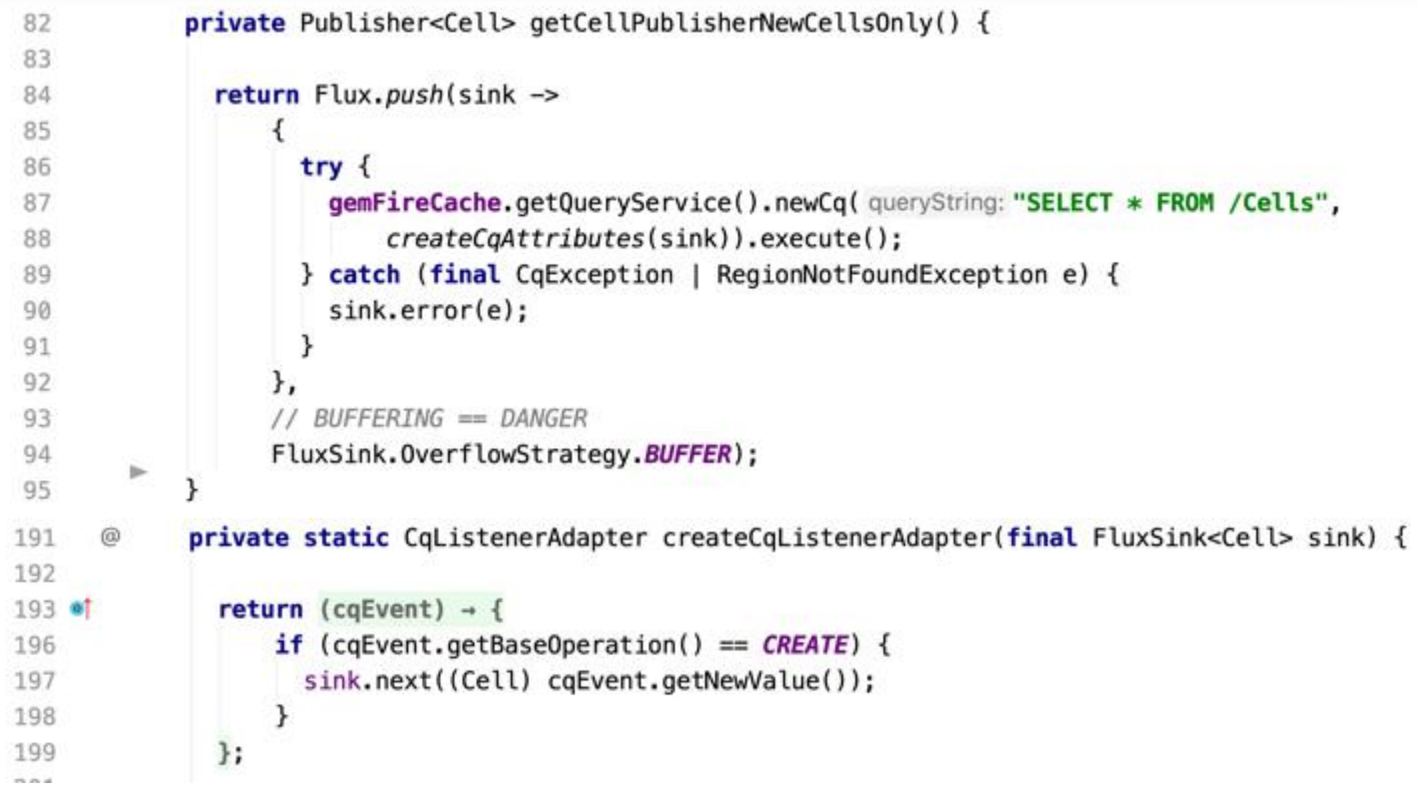


```java
    @BeforeClass
    public static void startGeodeServer() throws IOException {

      startGemFireServer(

          // FRAGILE WITHOUT RETRY, ROBUST WITH RETRY
          GeodeTestServerConfigurationAutoEvictionNoCQ.class,
          ...arguments: "-Xmx70m", "-Xms70m",

          // ROBUST (NO RETRY NEEDED)
//          GeodeServerConfigurationAutoEvictionAndCQ.class,
//          "-Xmx200m", "-Xms200m",

          // While OpenJDK 12 defaults to G1GC now, Geode 1.9 doc says use CMS
          "-XX:+UseConcMarkSweepGC", "-XX:CMSInitiatingOccupancyFraction=60");
```

Playback


Web Browser
RSocket
Game Server
Recorder
Geode Server
RSocket
Geode client-server

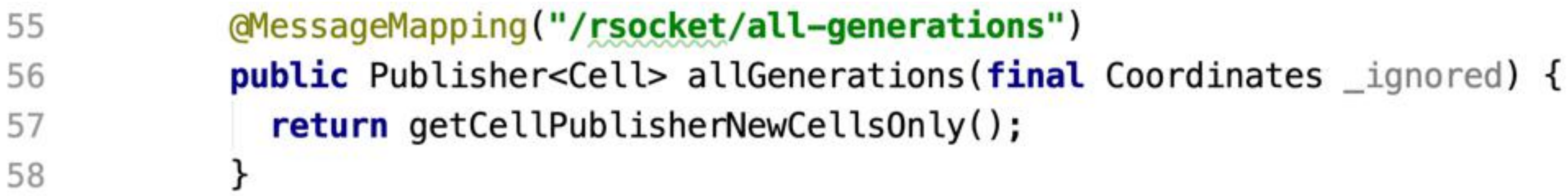
Hot Flux



```java
    private Publisher<Cell> getCellPublisherNewCellsOnly() {

      return Flux.push(sink ->
          {
            try {
              gemFireCache.getQueryService().newCq( queryString: "SELECT * FROM /Cells",
                  createCqAttributes(sink)).execute();
            } catch (final CqException | RegionNotFoundException e) {
              sink.error(e);
            }
          },
          // BUFFERING == DANGER
          FluxSink.OverflowStrategy.BUFFER);
    }
```

```java
    private static CqListenerAdapter createCqListenerAdapter(final FluxSink<Cell> sink) {

      return (cqEvent) → {
          if (cqEvent.getBaseOperation() == CREATE) {
            sink.next((Cell) cqEvent.getNewValue());
          }
      };
```

Endpoint



```java
    @MessageMapping("/rsocket/all-generations")
    public Publisher<Cell> allGenerations(final Coordinates _ignored) {
      return getCellPublisherNewCellsOnly();
    }
```

Results I

- Geode as reactive consumer (Subscriber)
  - How do active APIs fit with Geode's can Geode's load shedding policy be adapted to reactive back-pressure
    - Saw serial, parallel, `put`, `putAll`
    - BlockHound + `Schedulers.boundedElastic()`
  - Can Geode's load shedding policy be adapted to reactive back0pressure
    - App design, capacity planning, testing as usual
    - Back off on `LowMemoryException`

Results II

- Geode as reactive producer (Publisher)
  - Again, how do reactive APIs *fit* with Geode's
    - Demonstrated a simple CQ -> hot Flux
  - Can Geode produce long data streams incrementally
    - Query results chunked cold/cold-ish Fluxes present some challenges
    - CQ ordering of initial results would be nice

Pivotal

SpringOne Platform by Pivotal

re:Cap Seoul 2019

2019년 12월 17일(화) • 한화 드림플러스 강남