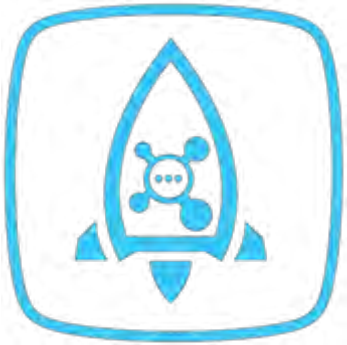
阿里中间件
Alibaba
Middleware

万亿级数据洪峰下的消息引擎
Apache RocketMQ

誓嘉

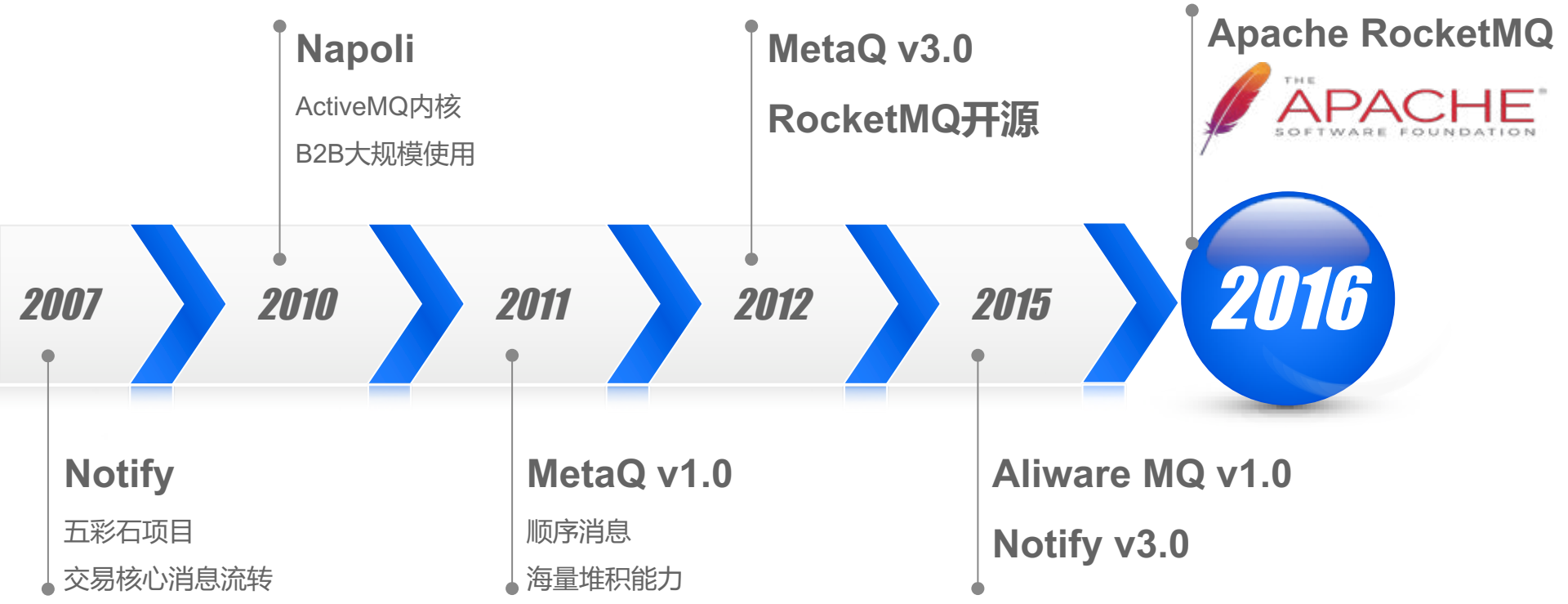


自我介绍

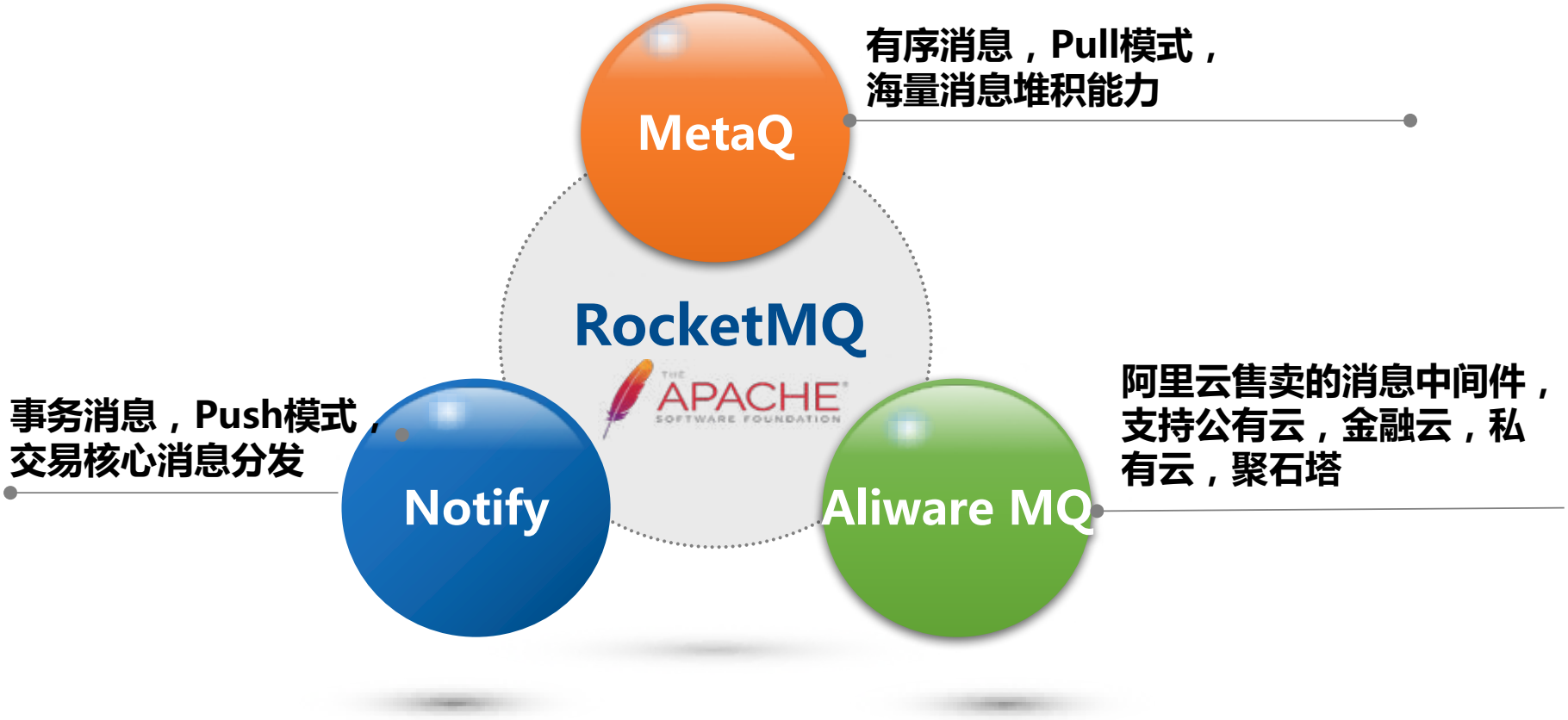
- 花名：誓嘉
- 真名：王小瑞
- vintagewang@apache.org
- @阿里巴巴-中间件
- Apache RocketMQ 创始人，PPMC Member，Committer
- Open-Messaging创始人

CONTENTS

阿里消息中间件演变历史

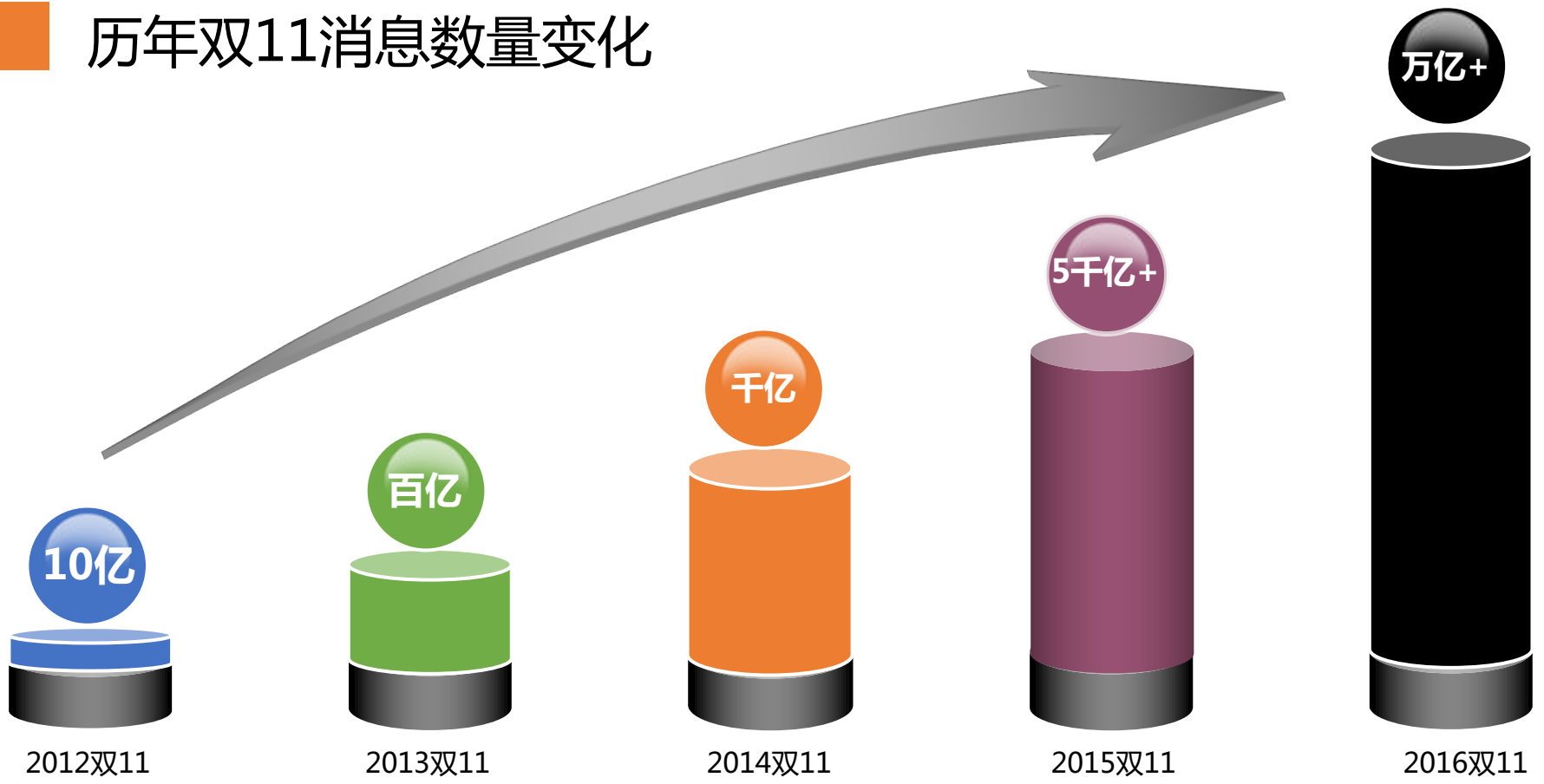


阿里消息中间件现状

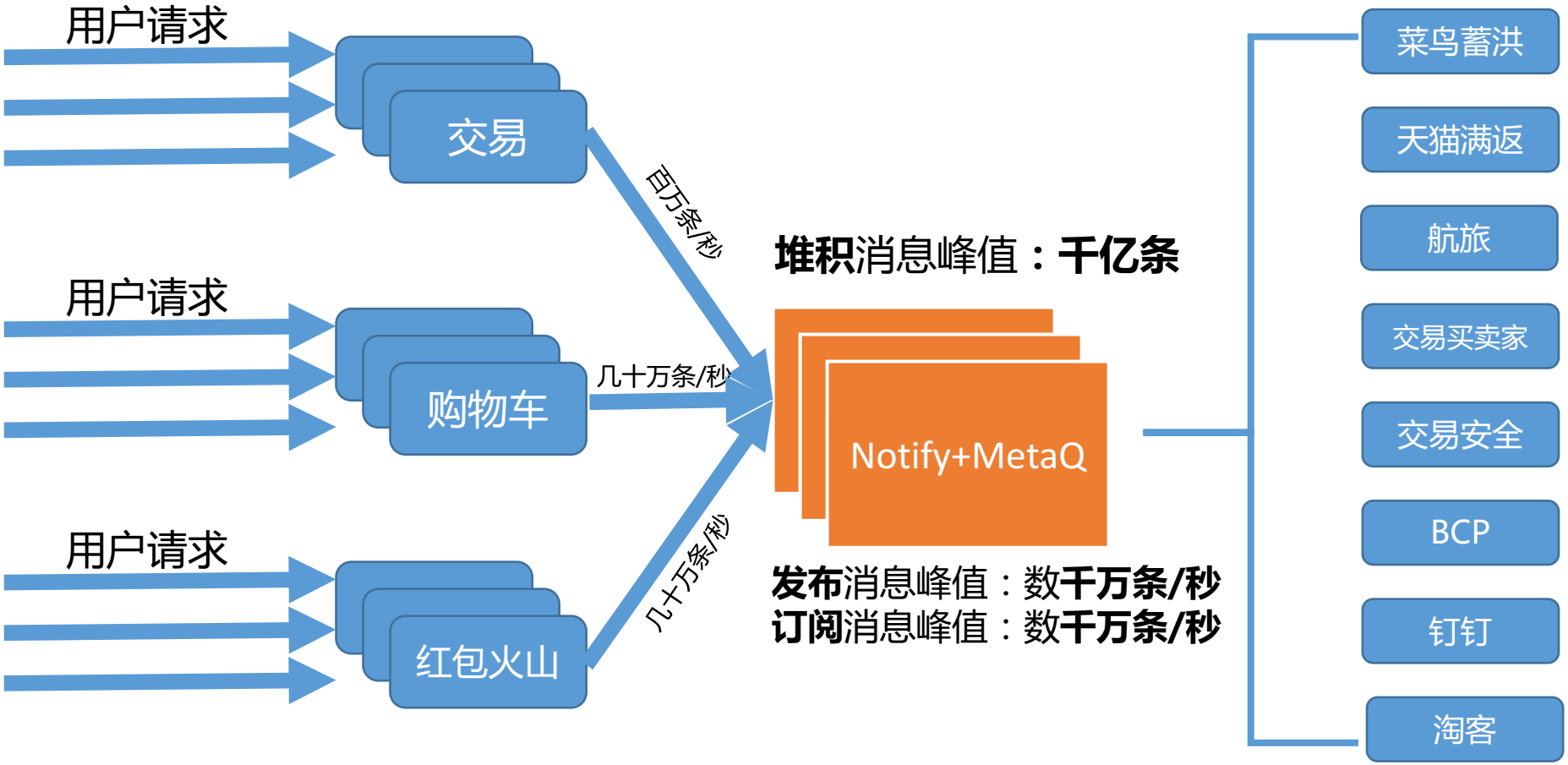

MetaQ
有序消息，Pull模式，
海量消息堆积能力
RocketMQ
THE APACHE SOFTWARE FOUNDATION
事务消息，Push模式，
交易核心消息分发
Notify
Aliware MQ
阿里云售卖的消息中间件，
支持公有云，金融云，私
有云，聚石塔

CONTENTS

历年双11消息数量变化



消息中间件核心链路


用户请求
交易
用户请求
购物车
用户请求
红包火山
百万条/秒
几十万条/秒
几十万条/秒
堆积消息峰值：千亿条
Notify+MetaQ
发布消息峰值：数千万条/秒
订阅消息峰值：数千万条/秒
菜鸟蓄洪
天猫满返
航旅
交易买卖家
交易安全
BCP
钉钉
淘客

万亿洪峰下有哪些问题

问题的本质：

可用性无限接近100%

可靠性无限接近100%

可用性 > 可靠性

双十一当天高可用要求 ~~ 100%

$$Availability = \frac{MTBF}{(MTBF + MTTR)}$$

$MTBF$: Mean time between failure.

$MTTR$: Mean time to recover.

$MTTR$ = 1 seconds

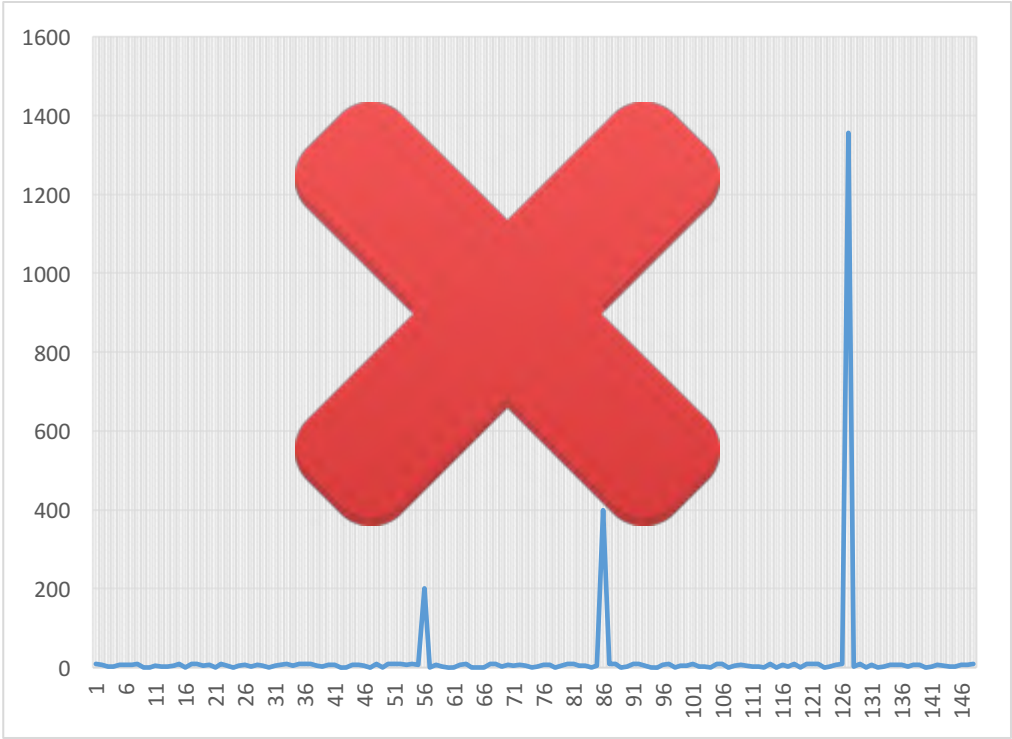
$$Availability = \frac{60*60*24 - 1}{(60*60*24)} = 99.999\%$$

双十一当天高可用标准

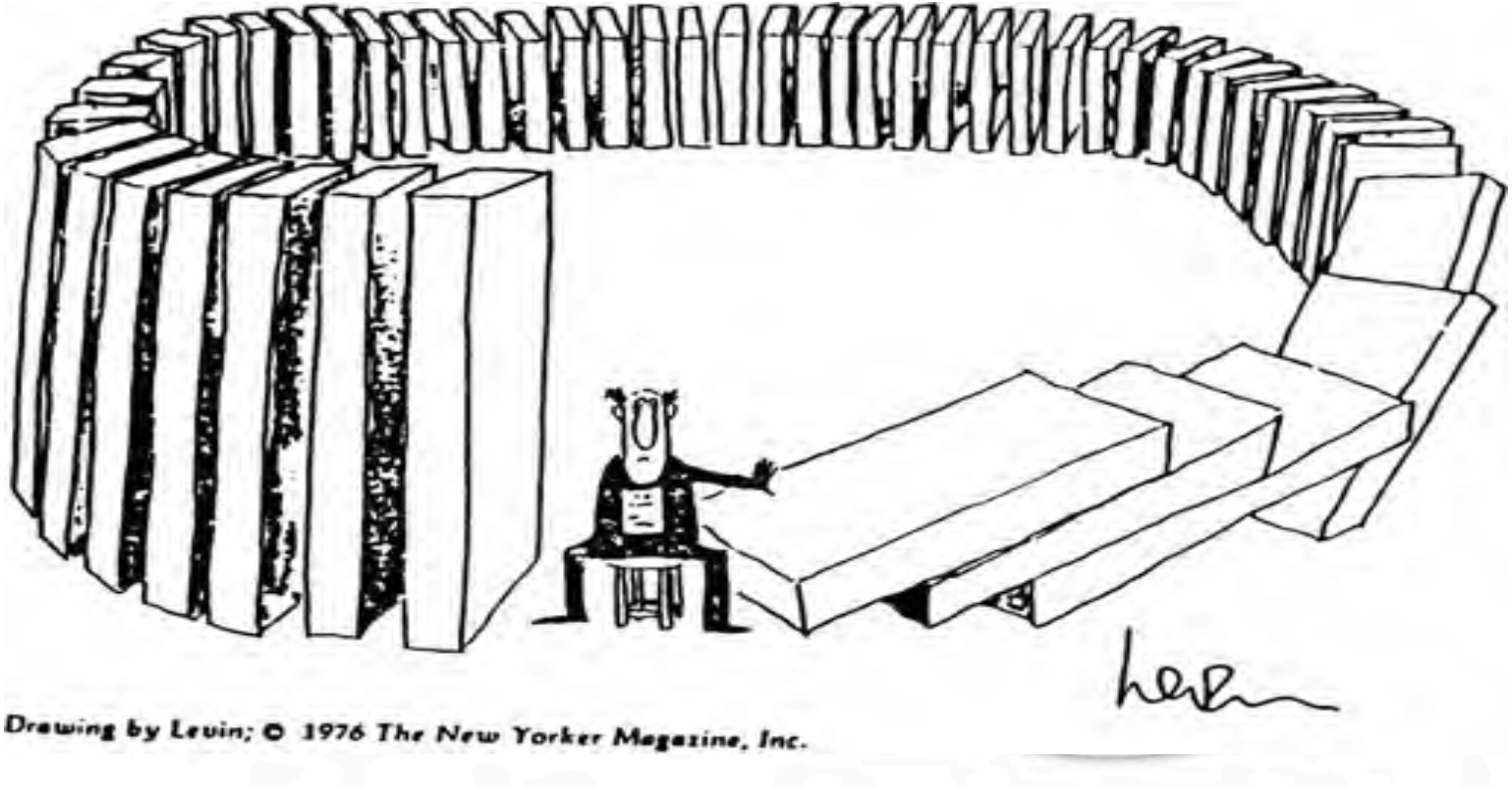
1. 每秒支撑千万级消息发布
2. 每条消息发布最大响应时间

   不超过20ms
3. 每条消息发布平均响应时间

   不超过3ms



分布式慢请求带来的挑战



Drawing by Levin; © 1976 The New Yorker Magazine, Inc.

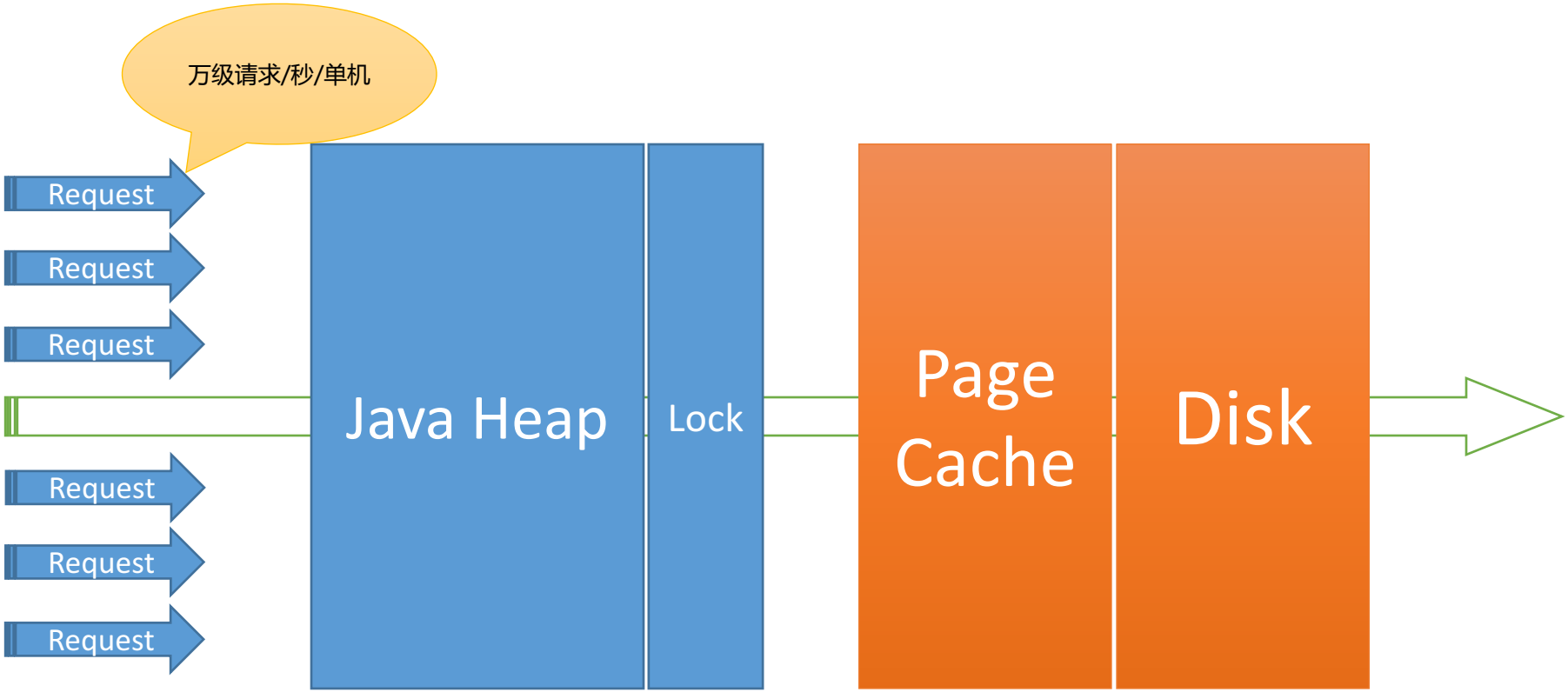
消息中间件分布式慢请求解法

01 低延迟分布式存储系统

02 在线熔断机制，秒级隔离

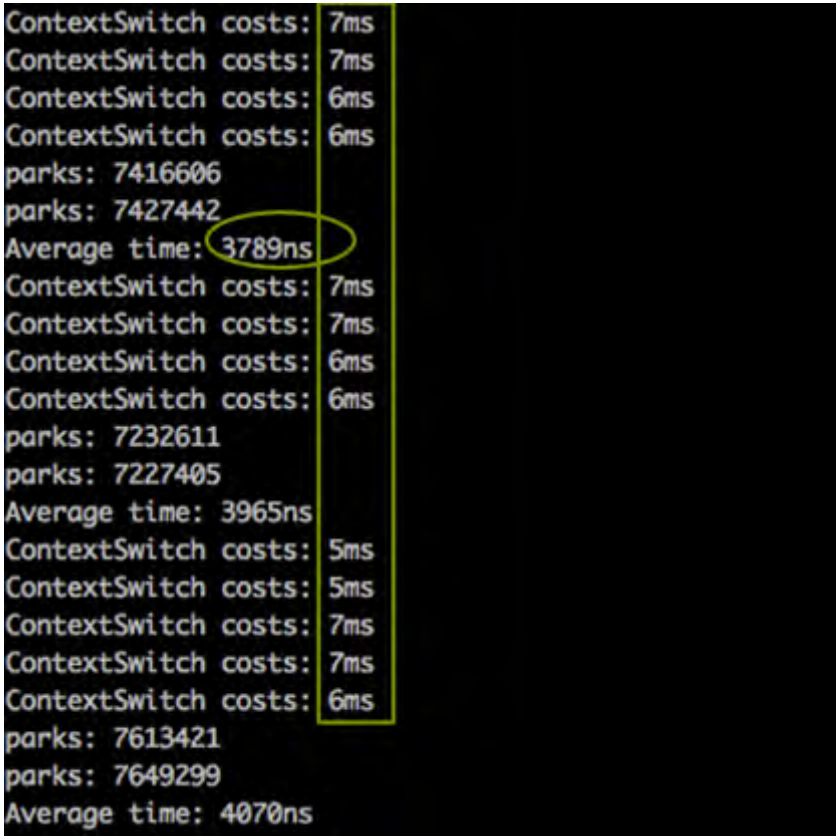
03 容量保障，限流

低延迟分布式存储系统 – RocketMQ存储

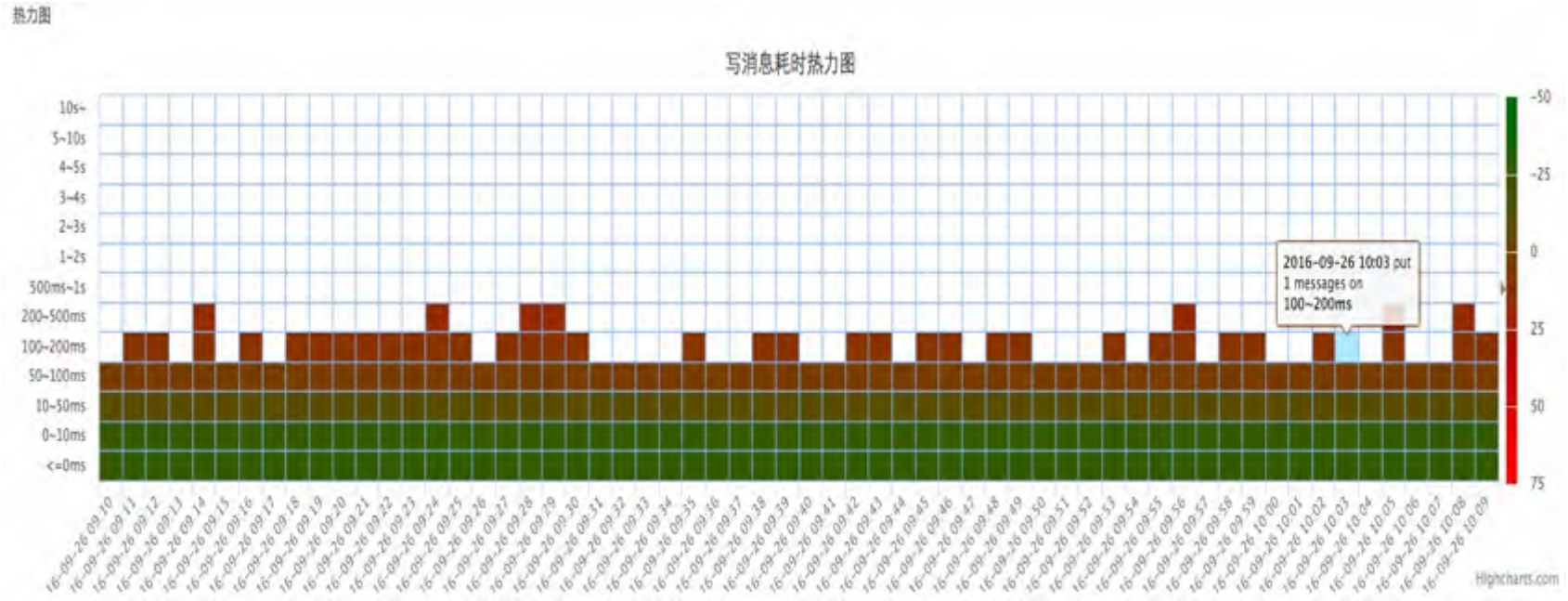


低延迟分布式存储系统 – 并发锁的开销

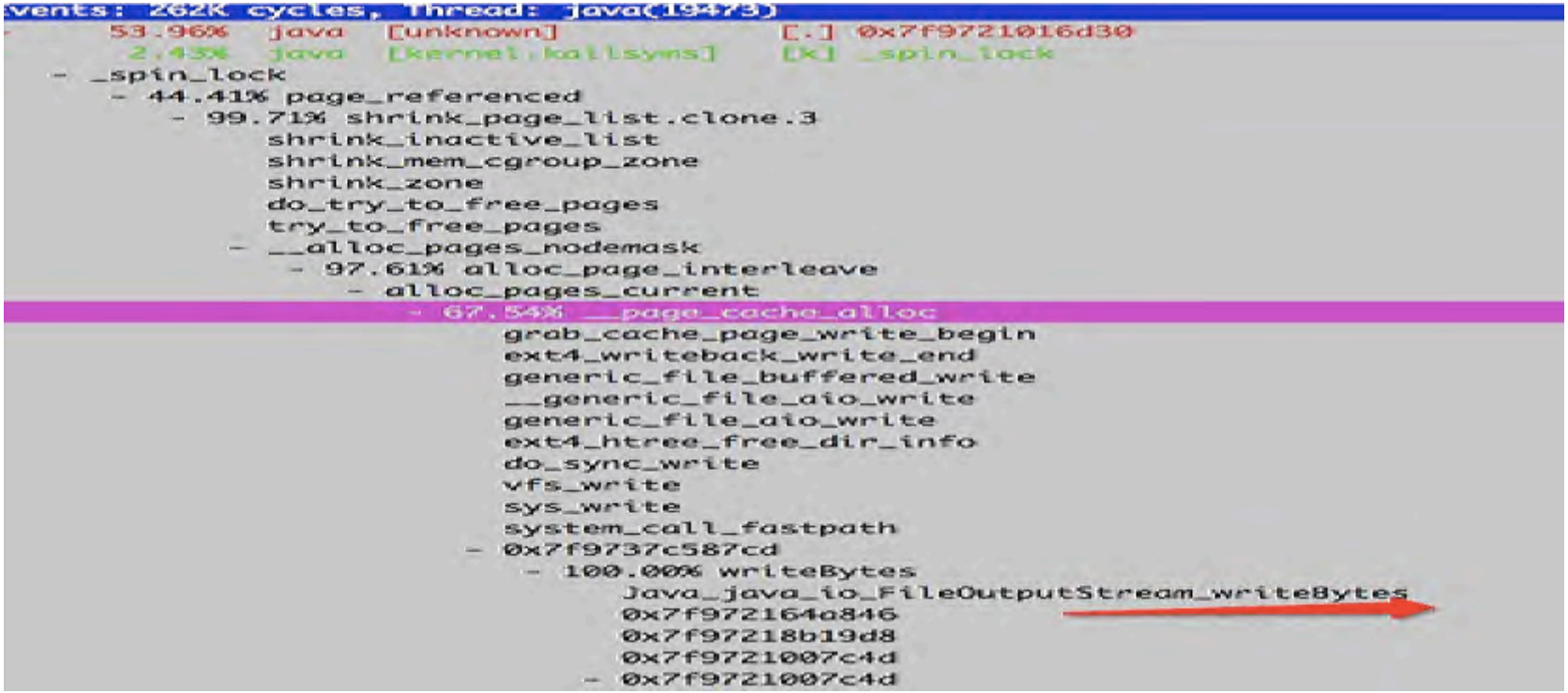
- ReentrantLock/synchronized
  - Fair
  - Unfair
- LockSupport.*unpark/park*



低延迟分布式存储系统 – PageCache真的那么快吗？

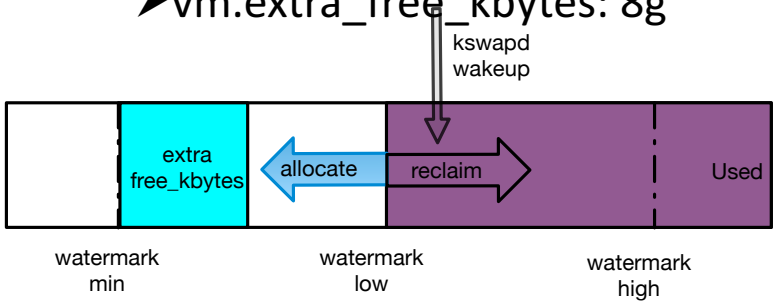


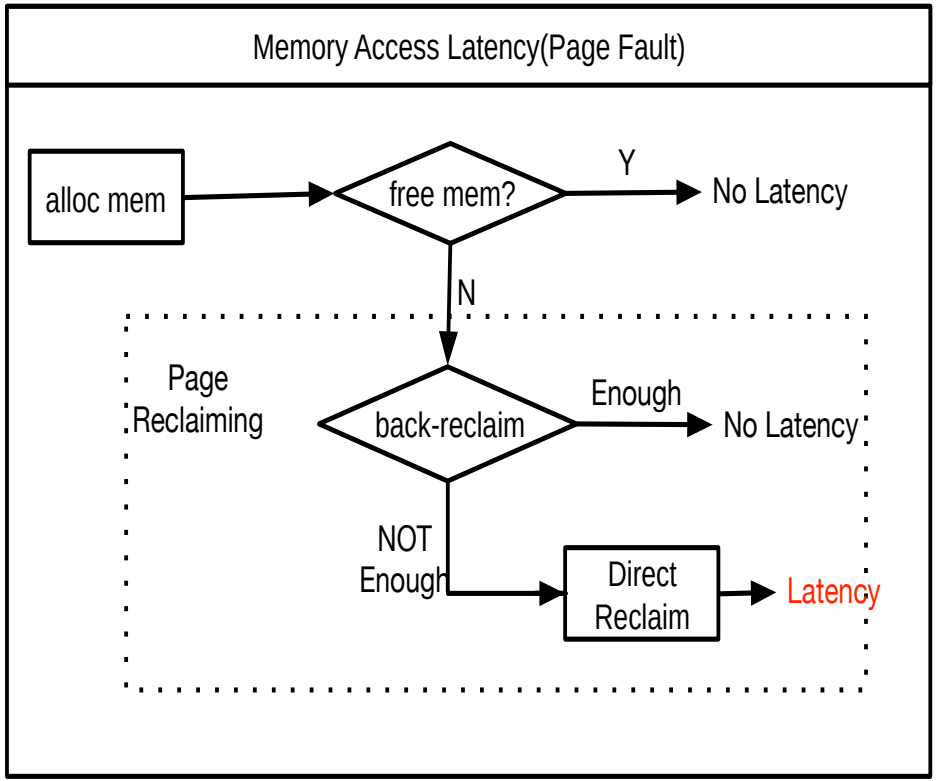
低延迟分布式存储系统 – PageCache的毛刺现象分析



低延迟分布式存储系统 – PageCache的毛刺现象分析

- Memory access latency issues:
  - Direct reclaim
    - Background reclaim (kswapd)
    - Foreground reclaim (direct reclaim)
  - Page fault
    - Major page fault will produce latency
- Direct reclaim
  - vm.min_free_kbytes: 3g
  - vm.extra_free_kbytes: 8g





低延迟分布式存储系统 – PageCache的毛刺内核源码分析

自旋锁-

```
struct address_space {
    struct inode        *host;      /* owner: inode, block_device */
    struct radix_tree_root  page_tree;  /* radix tree of all pages */
    spinlock_t      tree_lock;  /* and lock protecting it */
    unsigned int        i_mmap_writable;/* count VM_SHARED mappings */
    struct prio_tree_root   i_mmap;     /* tree of private and shared mappings */
    struct list_head    i_mmap_nonlinear;/*list VM_NONLINEAR mappings */
    spinlock_t      i_mmap_lock;    /* protect tree, count, list */
    unsigned int        truncate_count; /* Cover race condition with truncate */
    unsigned long       nrpages;    /* number of total pages */
    pgoff_t         writeback_index;/* writeback starts here */
    const struct address_space_operations *a_ops;   /* methods */
    unsigned long       flags;      /* error bits/gfp mask */
    struct backing_dev_info *backing_dev_info; /* device readahead, etc */
    spinlock_t      private_lock;   /* for use by the address_space */
    struct list_head    private_list;   /* ditto */
    struct address_space    *assoc_mapping; /* ditto */
    struct mutex        unmap_mutex;    /* to protect unmapping */
} __attribute__((aligned(sizeof(long))));
```

低延迟分布式存储系统 – PageCache的毛刺内核源码分析

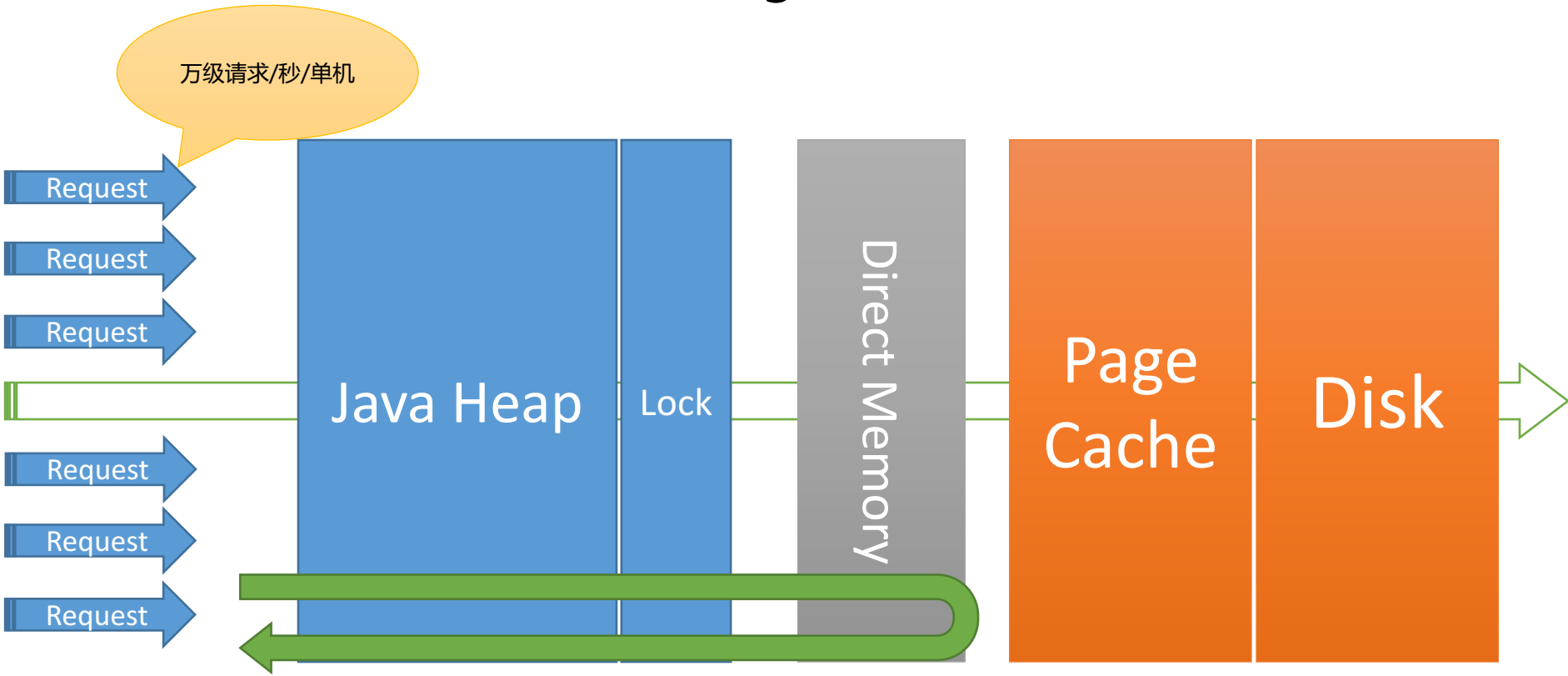
- Memory access latency issues:
  - Memory lock
  - Wake_up_page
    - Wait_on_page_locked()
    - Wait_on_page_writebacfk()

LOCKED	DIRTY	LRU
ACTIVE	WRITEBACK	RECLAIM
ANON	SWAPCACHE	SWAPBACKED
HUGE	RECLAIM	NOPAGE

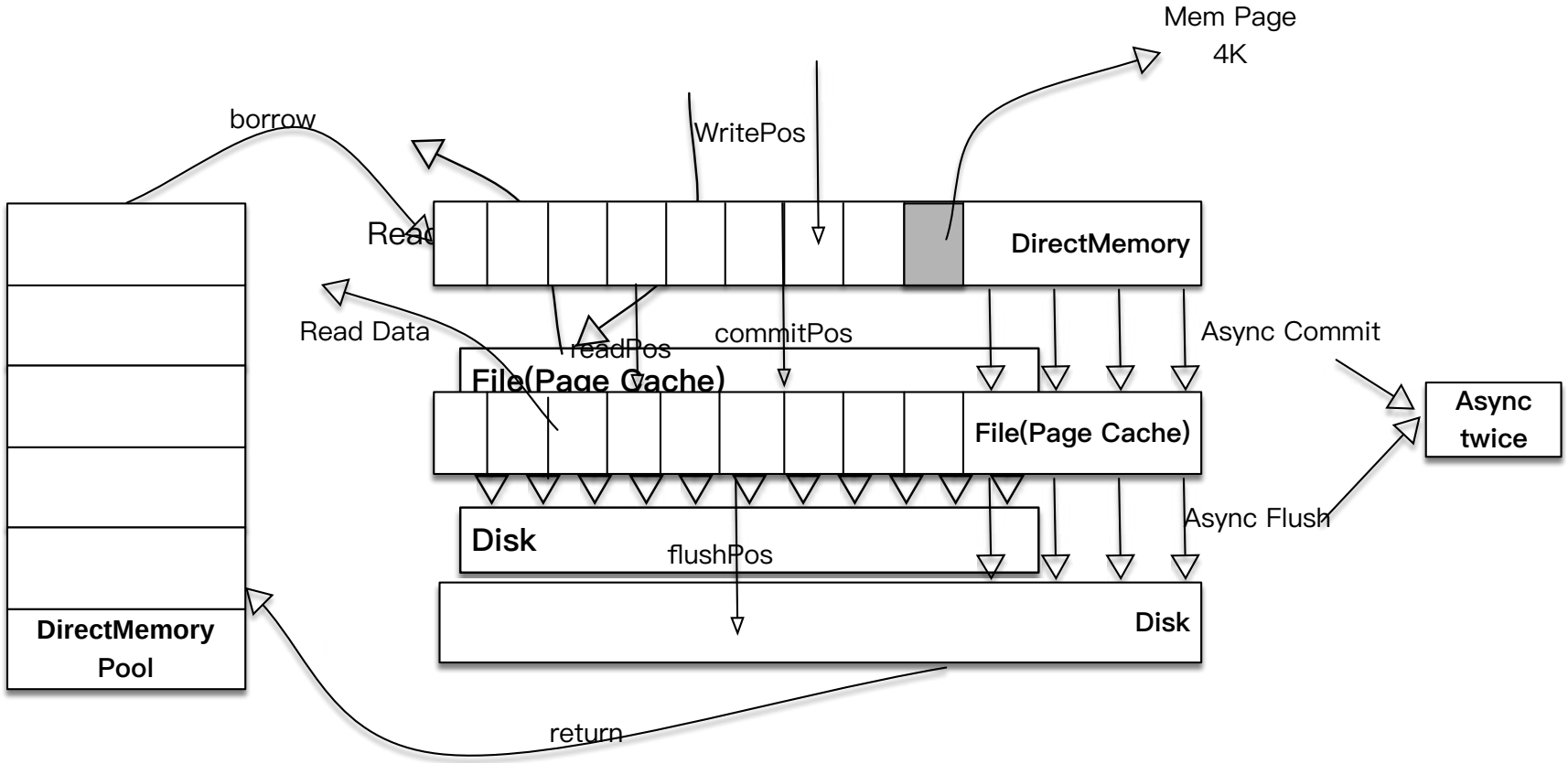
```
static inline int trylock_page(struct page *page)
{
    return (likely(!test_and_set_bit_lock(PG_locked, &page->flags)));
}

/*
 * lock_page may only be called if we have the page's inode pinned.
 */
static inline void lock_page(struct page *page)
{
    might_sleep();
    if (!trylock_page(page))
        __lock_page(page);
}
```

```
void unlock_page(struct page *page)
{
    VM_BUG_ON(!PageLocked(page));
    clear_bit_unlock(PG_locked, &page->flags);
    smp_mb__after_clear_bit();
    wake_up_page(page, PG_locked);
}
EXPORT_SYMBOL(unlock_page);
```

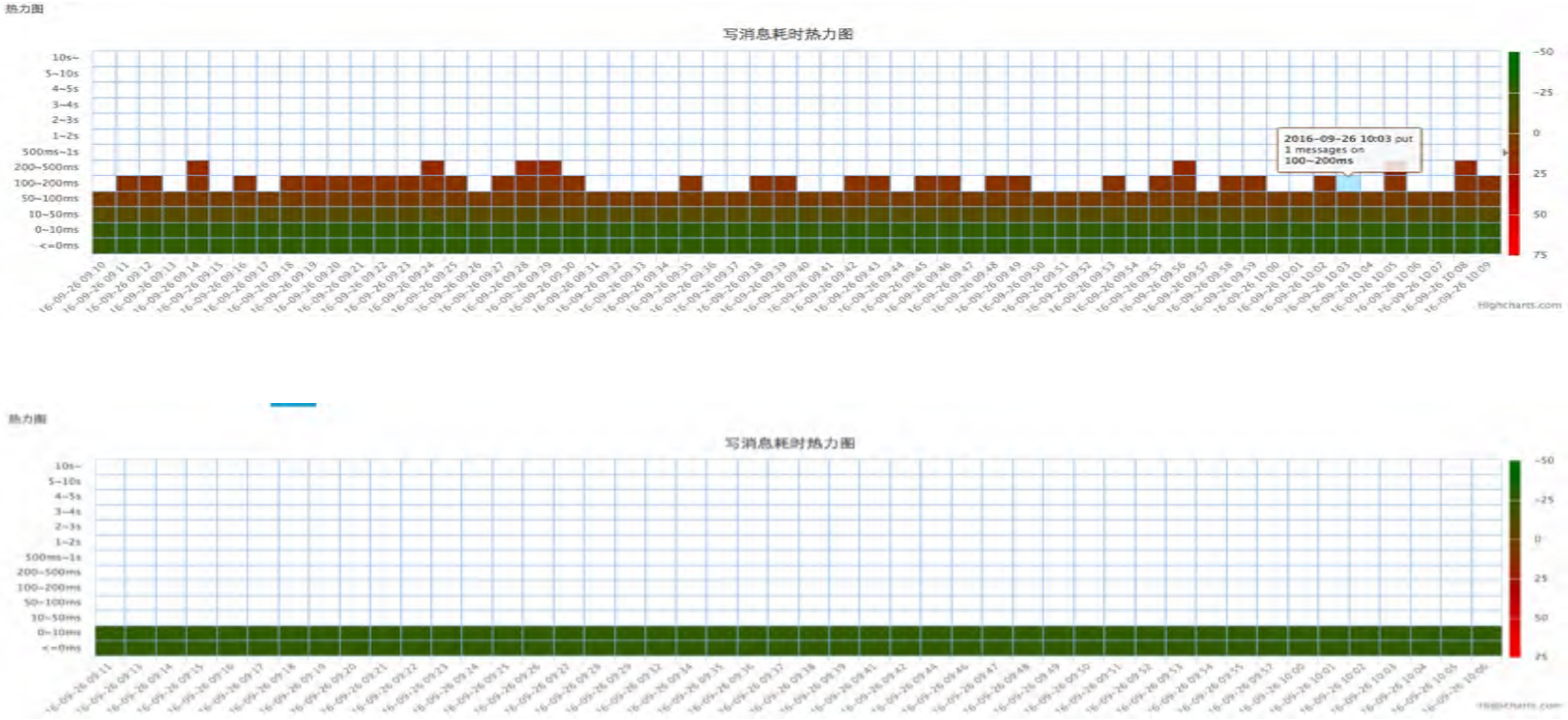
低延迟分布式存储系统 – PageCache的毛刺解决方案



低延迟分布式存储系统 – PageCache的毛刺解决方案



低延迟分布式存储系统 – PageCache的毛刺解决方案小结



低延迟分布式存储系统 – PageCache的毛刺解决方案小结

操作系统Page Cache Radix Tree 自旋锁，产生几秒的大毛刺
如果遇到坏盘，可能Block若干分钟，对系统产生致命影响。

操作系统Page Cache Radix Tree 每个Page的阻塞锁，产生几百毫秒小毛刺

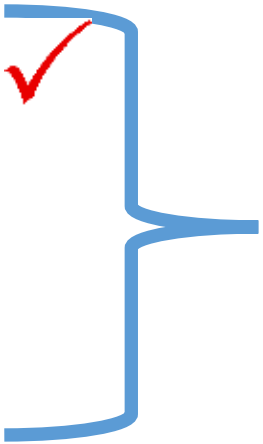
写入数据平均响应时间不超过1ms
写入数据最大响应时间不超过20ms（Java GC暂停线程引起）

双十一当天高可用要求 ~~ 100%

低延迟的分布式存储系统 ✓

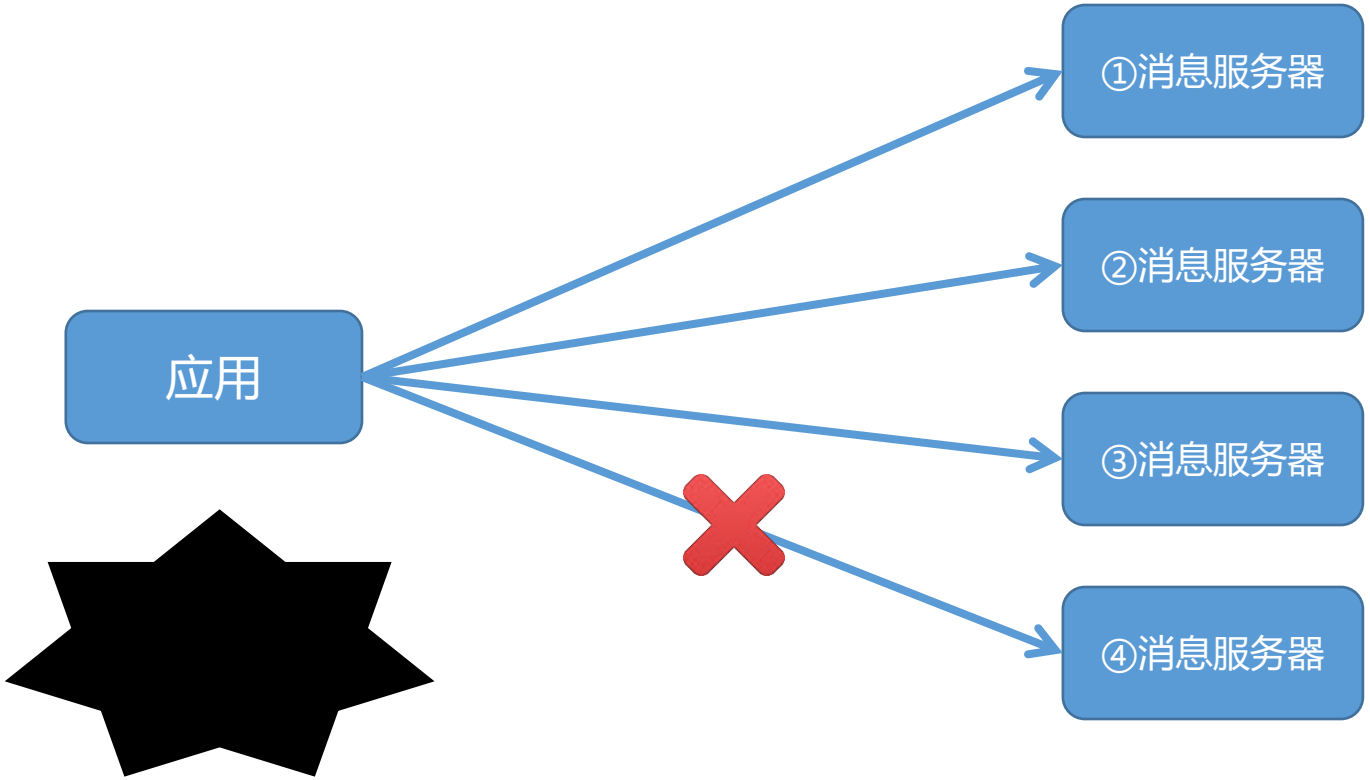
在线熔断机制

完善的容量评估



SLA=99.999%

在线熔断机制



规则

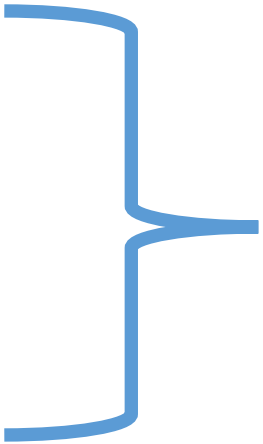
1. 最多只能隔离30%的机器。
2. 响应时间过长，开始隔离1分钟
3. 调用抛异常隔离1分钟
4. 如果隔离的服务器超过30%，则有部分调用会进入隔离列表中最早隔离的机器

双十一当天高可用要求 ~~ 100%

低延迟的分布式存储系统

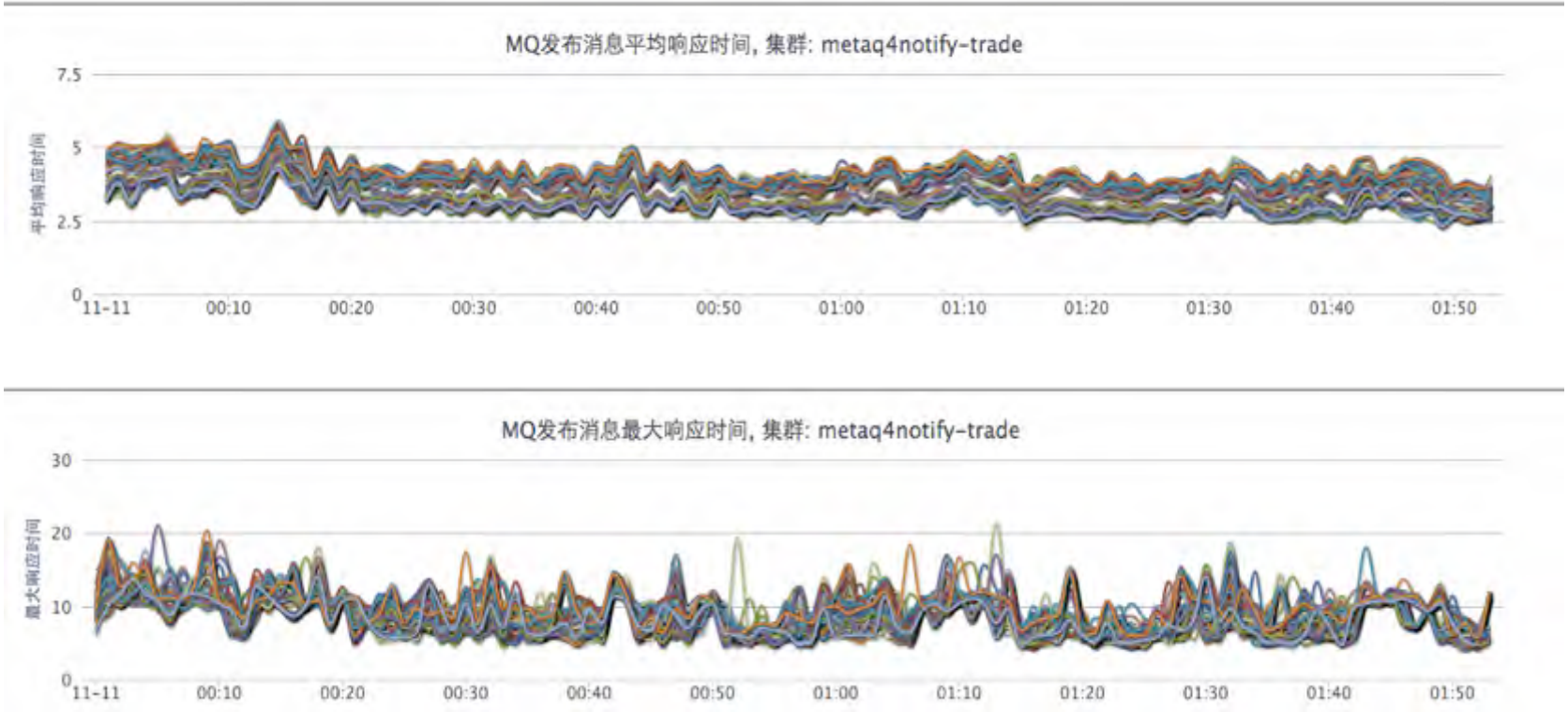
在线熔断机制 ✓

完善的容量评估



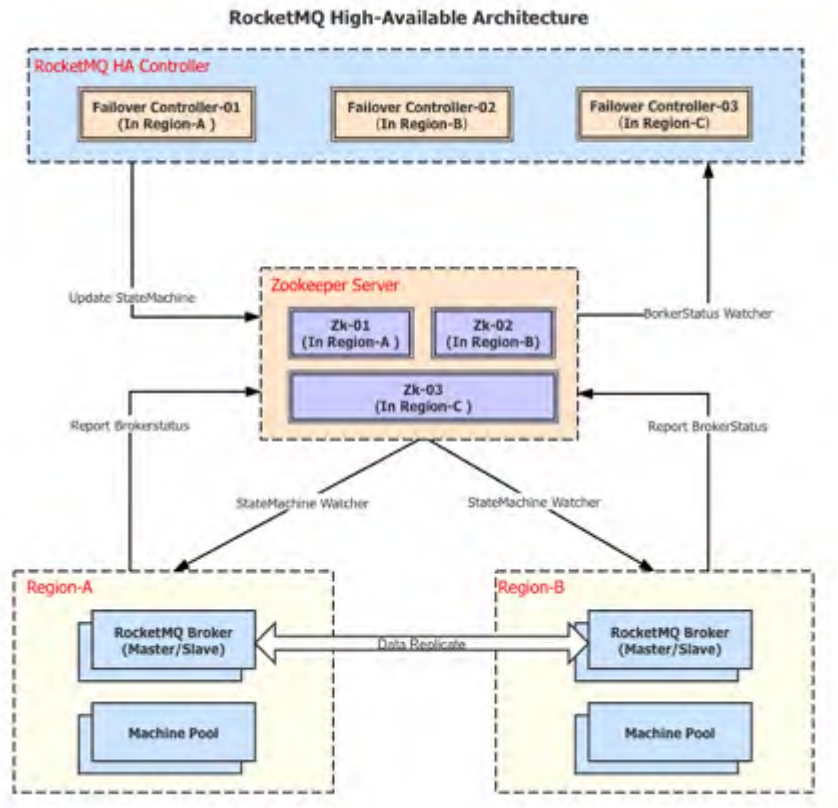
SLA=99.999%

双十一当天交易集群线上可用性



CONTENTS

RocketMQ 多副本高可靠



Controller provides:

- Watch broker status's change.
- Handle state machine transition and push new state to ZK.

ZK provides:

- Maintain persistent state Machine.
- Maintain ephemeral broker status.
- Notification mechanism when there is a change.

Report status ← Change status

Broker works as:

Watch state machine

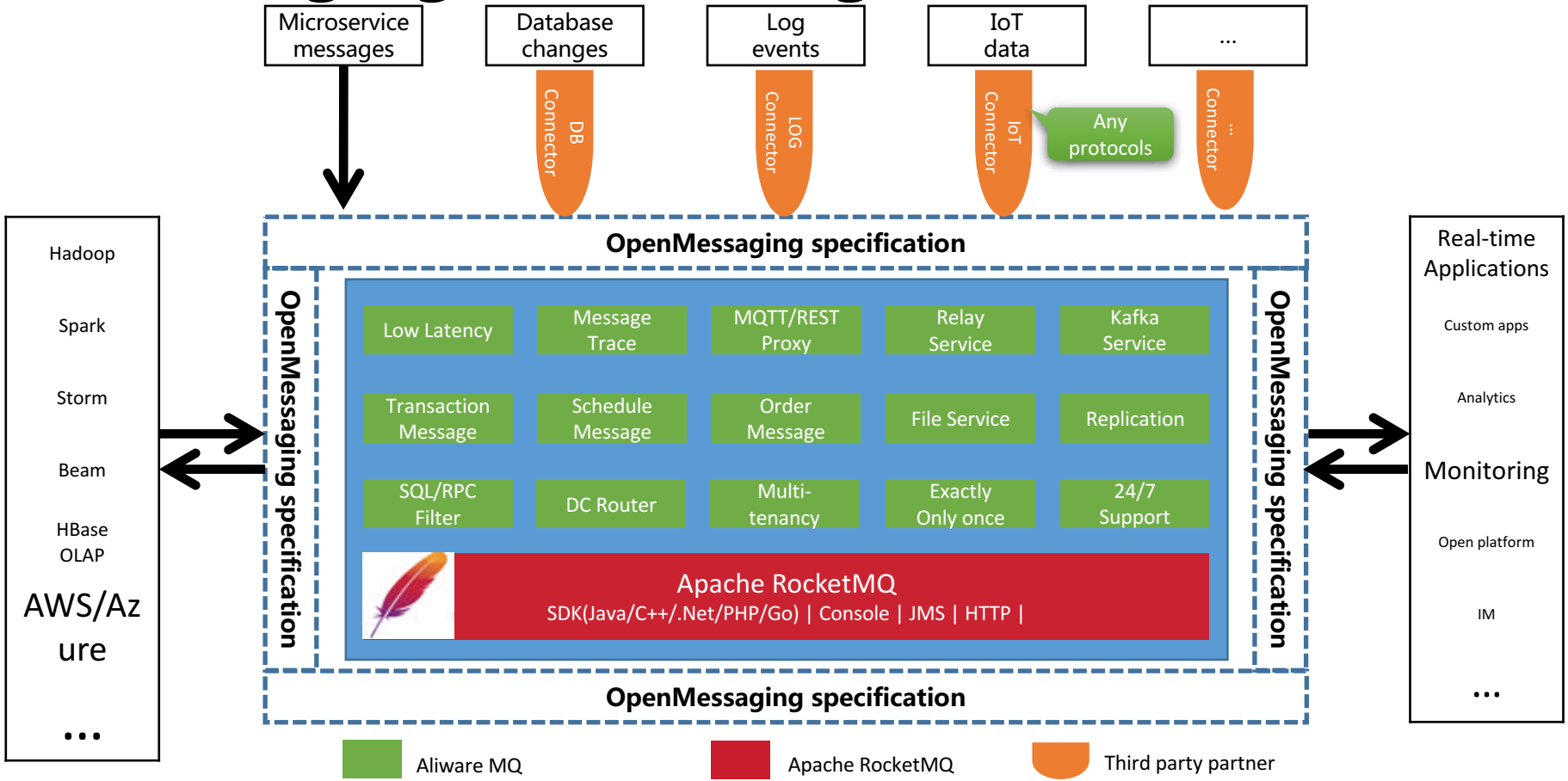
RocketMQ 多副本高可靠

Variables	Values & Description
MTBF of Disk(Hours)	1200000, from Seagate(希捷)
Time for recovery(Hours)	2, 1TB Disk, 75% capacity water level, 100MB/S

Cluster Scale	HA	Replica Nums	Reliability
1M	❌	1	99.27%
2M	❌	1	99.27%
1M1S	✅	2	99.9999976%
2M2S	✅	2	99.9999976%
2M2S	✅	3	99.99999999987%

CONTENTS

Messaging&Streaming Platform



Apache RocketMQ – 商业化版本

阿里中间件
Alibaba Middleware

王小瑞

浙江 杭州

感谢大家

THANKS!

Alibaba Middleware

- http://openmessaging.github.io/
- http://rocketmq.apache.org/
- http://www.aliyun.com/product/ons

扫一扫上面的二维码图案，加我微信