Node.js开发数据统计系统

-- 基于Node.js、MongoDB的企业级实践

by 杨德模（dmyang）

About me

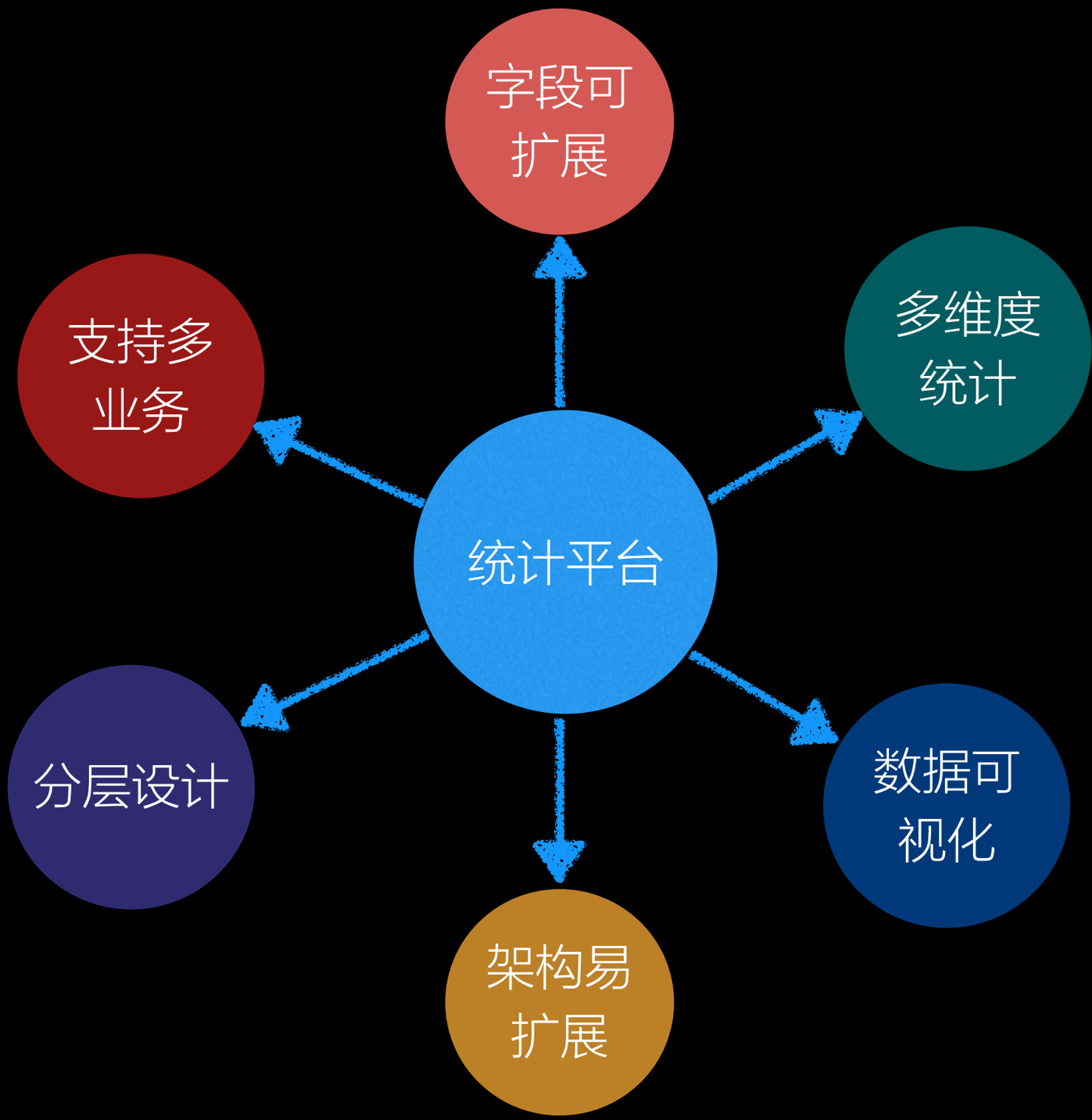
- 腾讯（2011.7-2014.6），唯品会移动事业部（2014.7-）
- 前端开发，Node.js开发（近2年），热爱Linux，热爱GitHub

- github: @chemdemo
- wechat: @chemdemo
- weibo: http://weibo.com/chemdemo

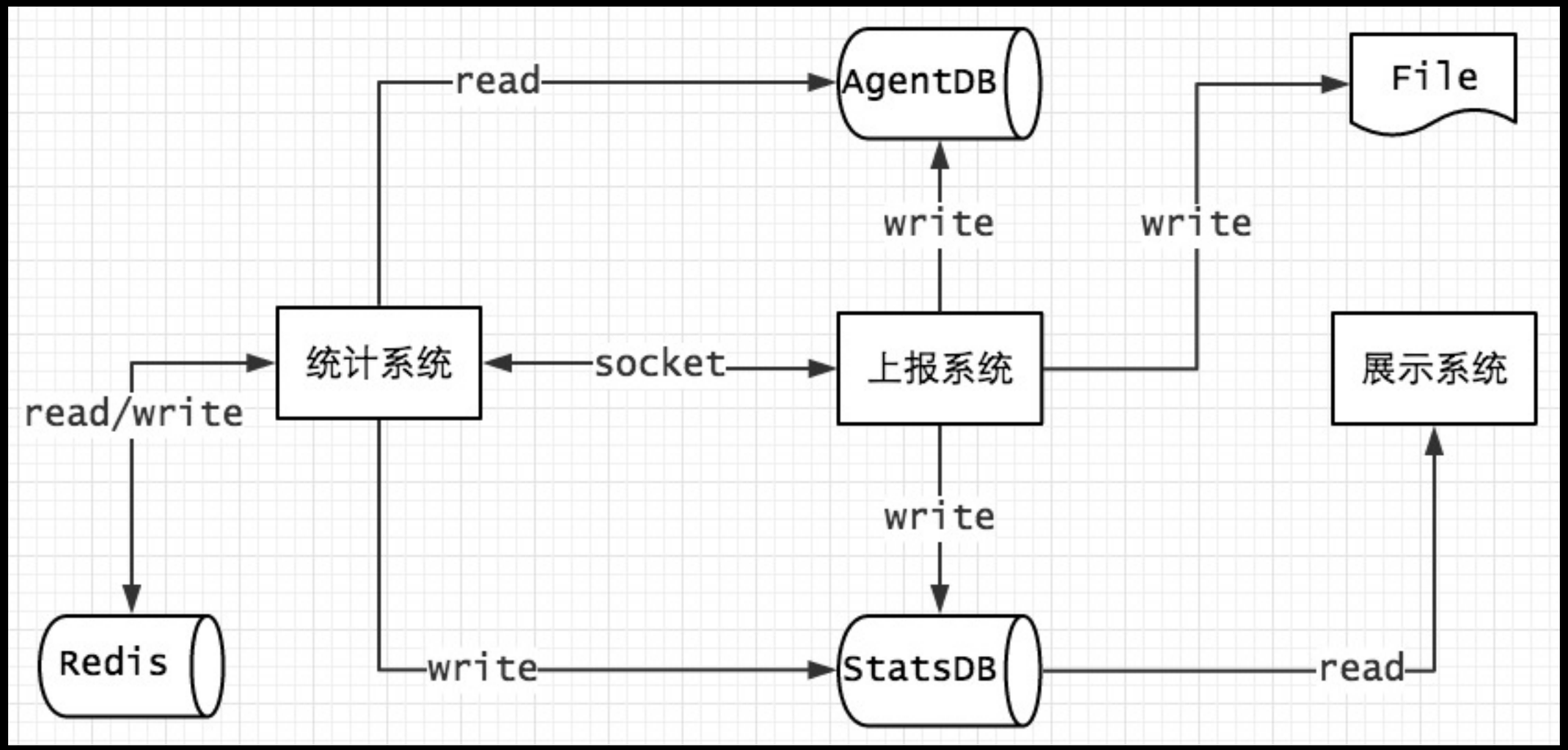
为什么自己做

- 数据团队提供的接口不能满足我们业务需求
- 选择Node：轻量、高效、有实践

功能特性



整体架构图


read
AgentDB
File
write
write
统计系统
socket
上报系统
展示系统
read/write
write
Redis
write
StatsDB
read

项目结构

master › mstats-report

Name	Last
api	a day
bin	5 day
conf	a day
doc	2 mo
lib	a day
test	22 da
.gitignore	abou
README.md	2 mo
index.js	2 mo
package.json	21 da
pm2_deploy.json	abou
sites.txt	8 day

master › mstats-monitor / -

Name	Last Up
api	a day ag
bin	a day ag
conf	a day ag
lib	a day ag
public	a day ag
test	15 days
views	a day ag
.gitignore	15 days
README.md	2 month
index.js	15 days
package.json	15 days

dev › mstats-stats / -

Name	Last U
bin	about a
conf	a day ag
lib	a day ag
tasks	a day ag
test	20 days
.gitignore	2 days a
index.js	a day ag
package.json	22 days
pm2_deploy.json	about a

输出

WAP数据统计平台

基本信息 | 性能统计 | 业务埋点 | js错误 | 日志跟踪

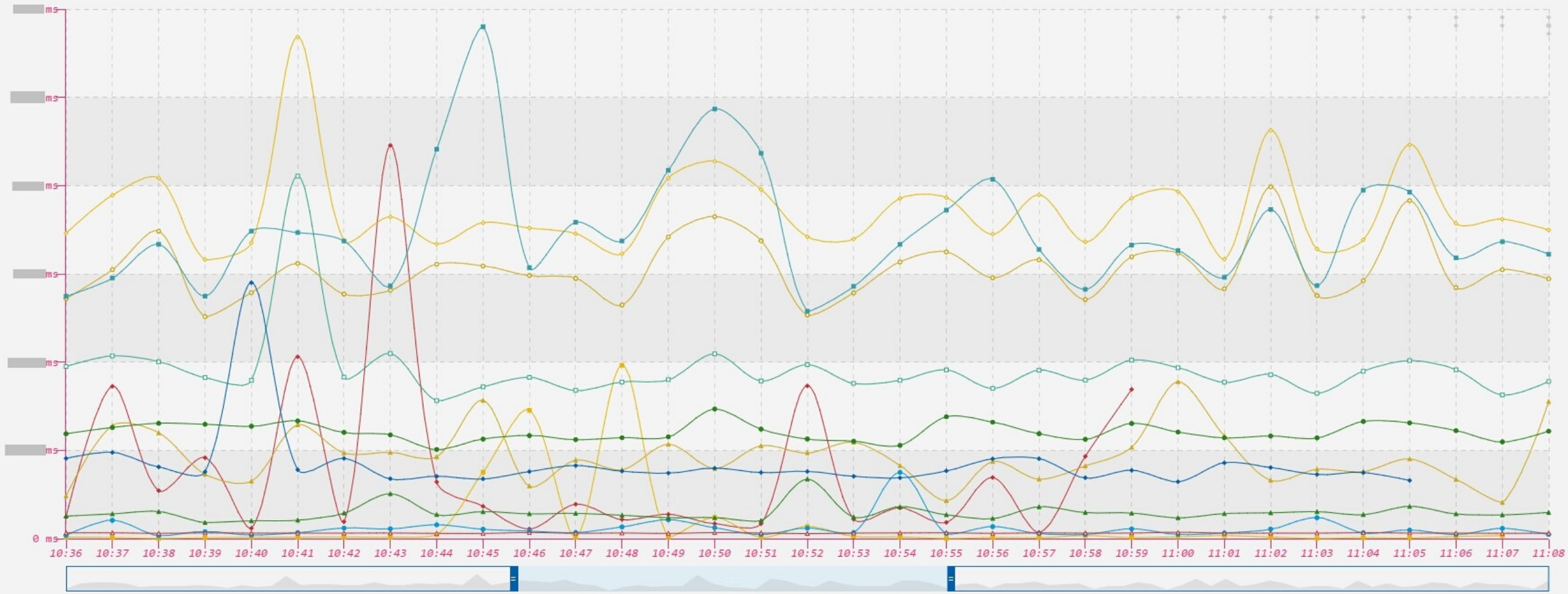
Url brand | 区域 All | 平台 All | 浏览器 All | 开始时间 2014-12-05 10:7 | 结束时间 2014-12-05 12:7 | 筛选

一共 102 条数据

WAP性能曲线

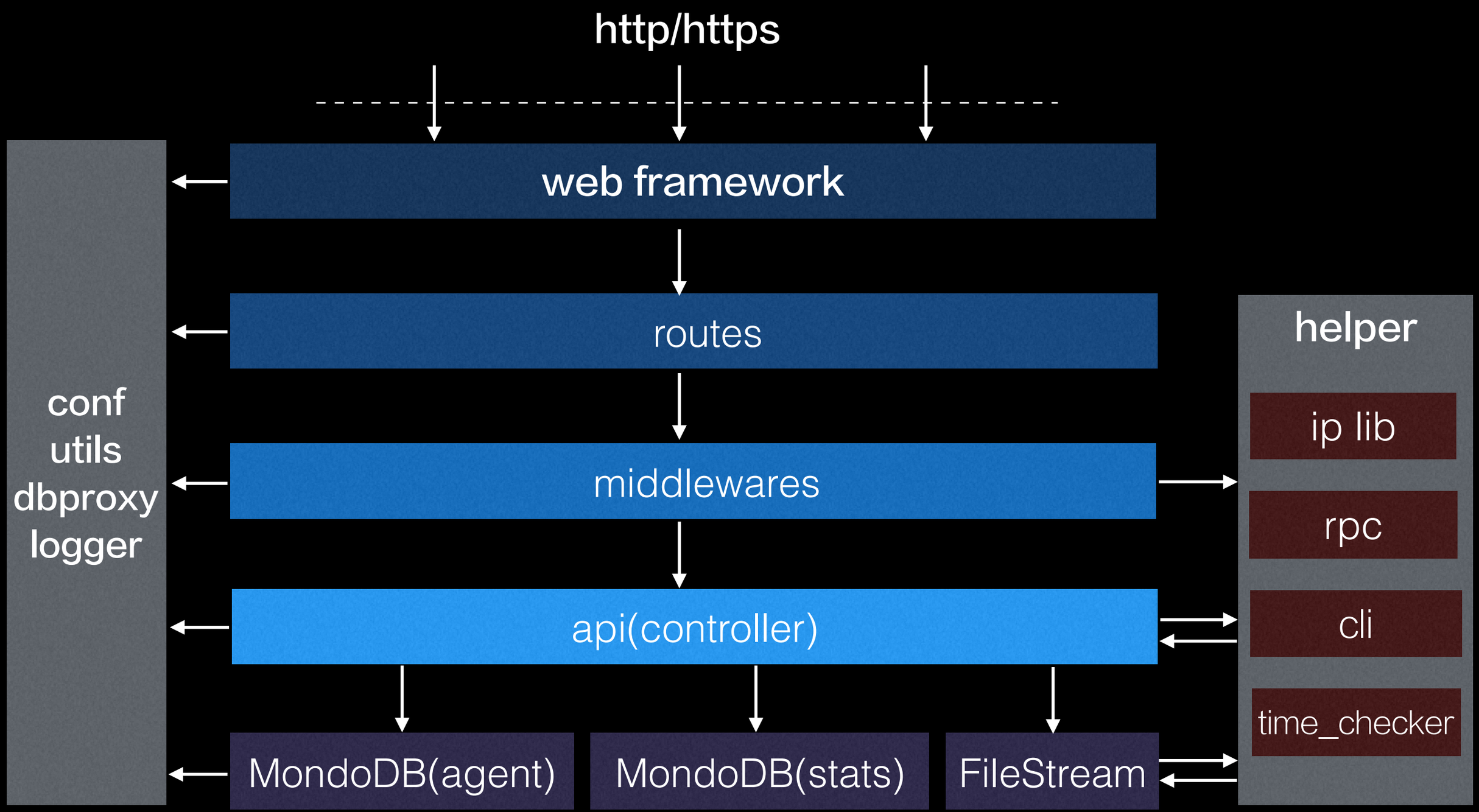
Alpha版

页面跳转 DNS TCP 首字节 接收响应 前页面卸载 初始化dom 可交互 domContentLoaded 页面加载 头部加载 domReady html渲染 onload 首屏



主题切换： shine

上报系统内部架构


http/https
web framework
routes
middlewares
api(controller)
MondoDB(agent)
MondoDB(stats)
FileStream
conf
utils
dbproxy
logger
helper
ip lib
rpc
cli
time_checker

简单数据类型统计

如：点击流统计

```
/api?name=lottery-first-failed&id=btn
```

↓

一些数据

- **3台**机器
- MongoDB单表每天写入量：**百万级**
- 上报接口PV：**千万级**

App name	id	mode	PID	status	restarted	uptime	memory
mstats-monitor	0	fork	14284	online	5	17h	31.816 MB
log.io-server	1	fork	11752	online	22	17h	122.367 MB
mstats-stats	2	fork	15448	online	0	16h	76.195 MB
log.io-harvester	5	fork	1913	online	0	3D	52.434 MB

实现细节

- 将多个查询条件组合成一个
- 每个请求都判断所处时间段

利用MongoDB的increase操作符（$inc）实现自更新!

即数据合并成一条

- 不同时间段，新插入一条数据

组合查询条件

不管多少个key，排列组合组合拼成一个id:

```
var query = {id: ‘click’};
```

```
/api?name=click&id=foo  ⟶  var query = {id: ‘foo’};
```

```
var query = {id: ‘click|foo’};
```

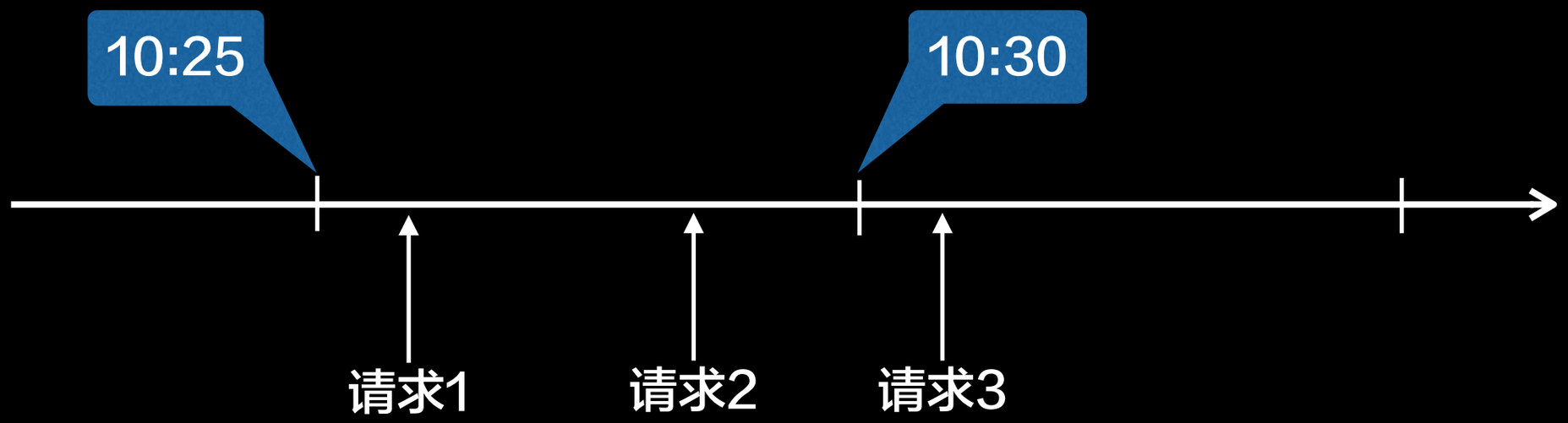
```
var query = {id: ‘view’};
```

```
/api?name=view&id=bar  ⟶  var query = {id: ‘bar’};
```

```
var query = {id: ‘view|bar’};
```

time字段

假设某业务，要求统计时间间隔是5分钟：



请求1：
```
query.time = '10:30';
```

请求2：
```
query.time = '10:30';
```

请求3：
```
query.time = '10:35';
```

插入数据

```
db.update(query, {‘$inc’: {count: 1}}, true, true);
```

事先对id和time字段建联合索引:

```
db.ensureIndex({time: 1, id: 1});
```

时间间隔可配置

- 不同的业务、不同的api，可以配置统计间隔
- 程序启动后不支持动态调整时间间隔

```
module.exports = {
    moduleA: {
        interval: {
            api_x: 5 // 分钟
        }
    }
};
```

非数值类型的统计

如：性能统计


```
{
    url: ‘/foo/bar’,
    browser: ‘Safari’,
    os: ‘iOS’,
    region: ‘广州’,
    data: {
        dns: 1,
        tcp: 2,
        ttl: 3,
        首屏: 4,
        domReady: 5,
        onload: 6
        …
    }
}
```

数值字段处理

```
data = {
    dns: 0,
    tcp: 1,
    ttl: 2
}
```

```
$inc = {
    dns: 0,
    tcp: 1,
    ttl: 2,
    count_dns: 0,
    count_tcp: 1,
    count_ttl: 1
}
```

其他字段同样的方法处理

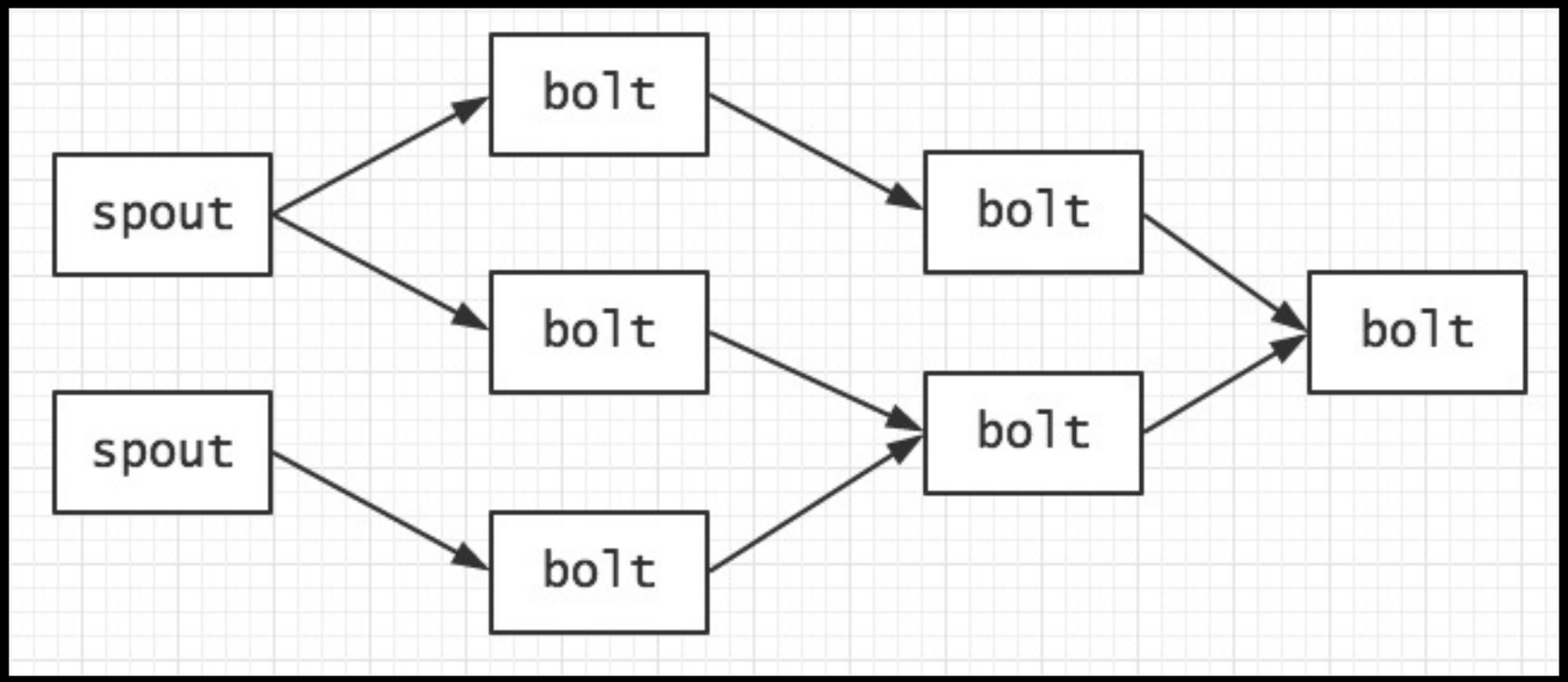
```
delete query.data;

db.update(query, $inc, true, true);
```

统计系统

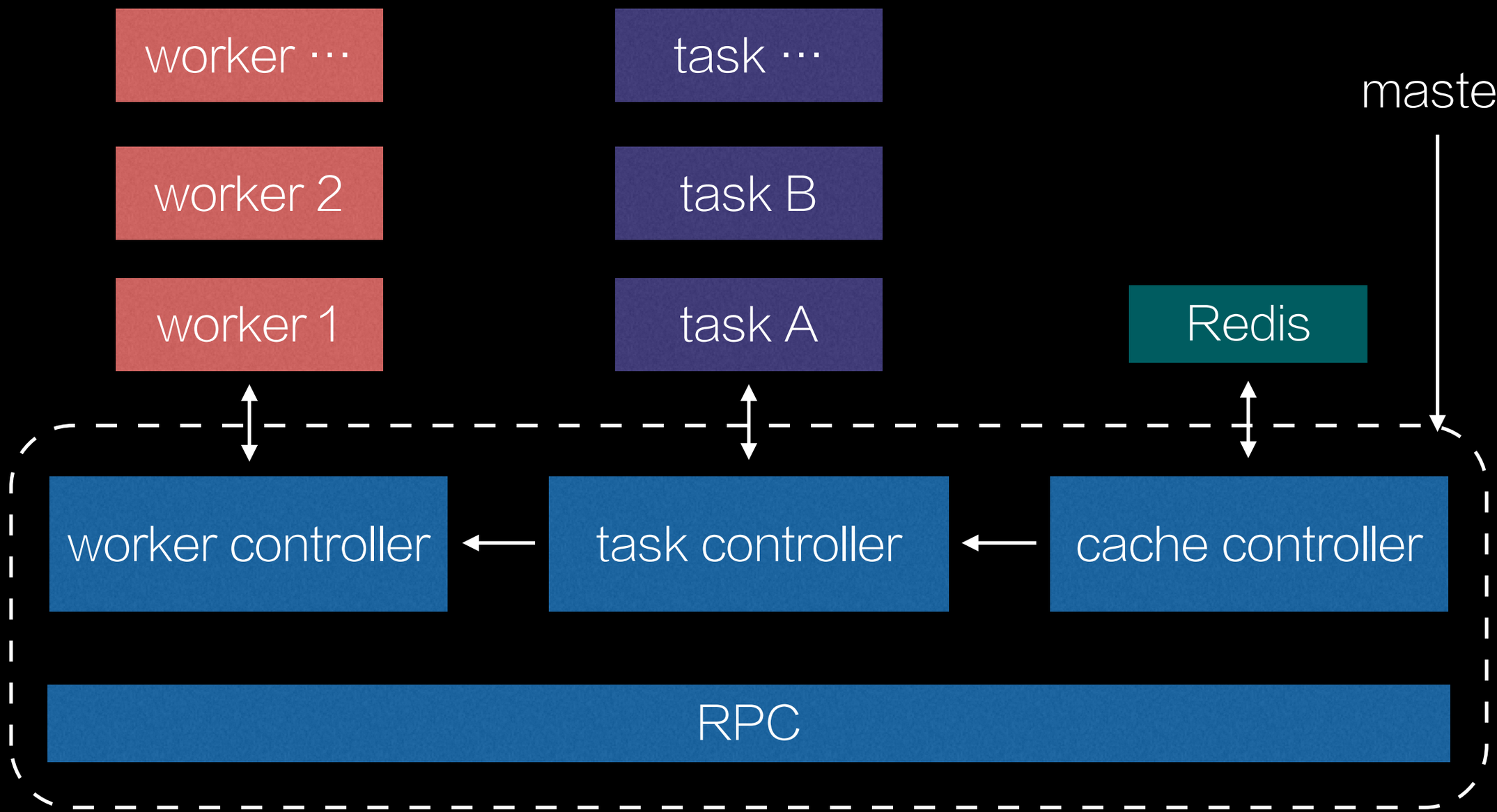
- 二次统计需要
- 实时统计需要

Twitter storm Topology



- 利用Thrift支持跨语言定义spout和bolt
- spout可以是rpc（DRPC）
- https://github.com/rkatti/node-drpc
- https://github.com/chemdemo/storm-drpc-node

统计系统内部架构


worker …
task …
master
worker 2
task B
worker 1
task A
Redis
worker controller
task controller
cache controller
RPC

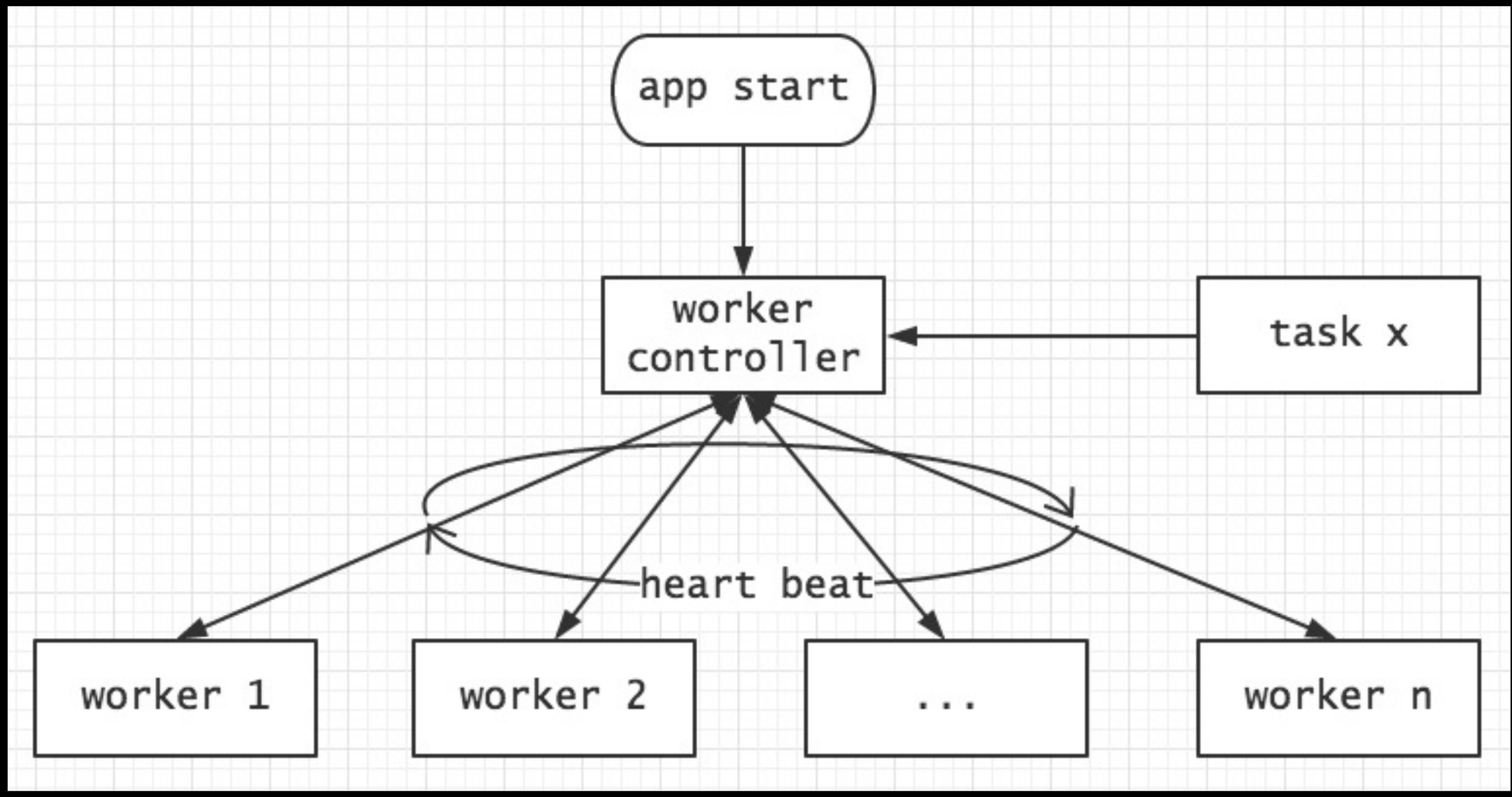
cache controller

- 消息队列（Redis list）
- 防止统计服务器重启丢信息
- 延迟统计

worker controller

- 进程池，负责进程管理子进程消息处理
- 按照cpu个数启动worker，绑定cpu（taskset）
- 心跳检测
- 还负责进程分配

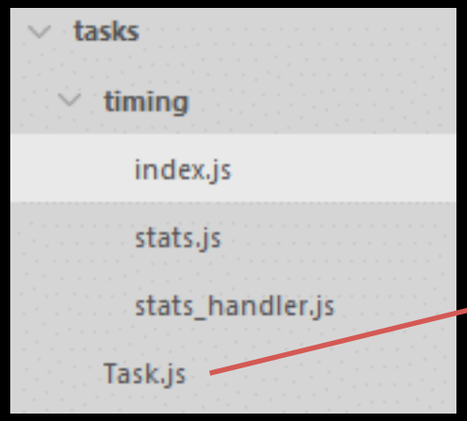
worker pool


app start
worker controller
task x
heart beat
worker 1
worker 2
...
worker n

task controller

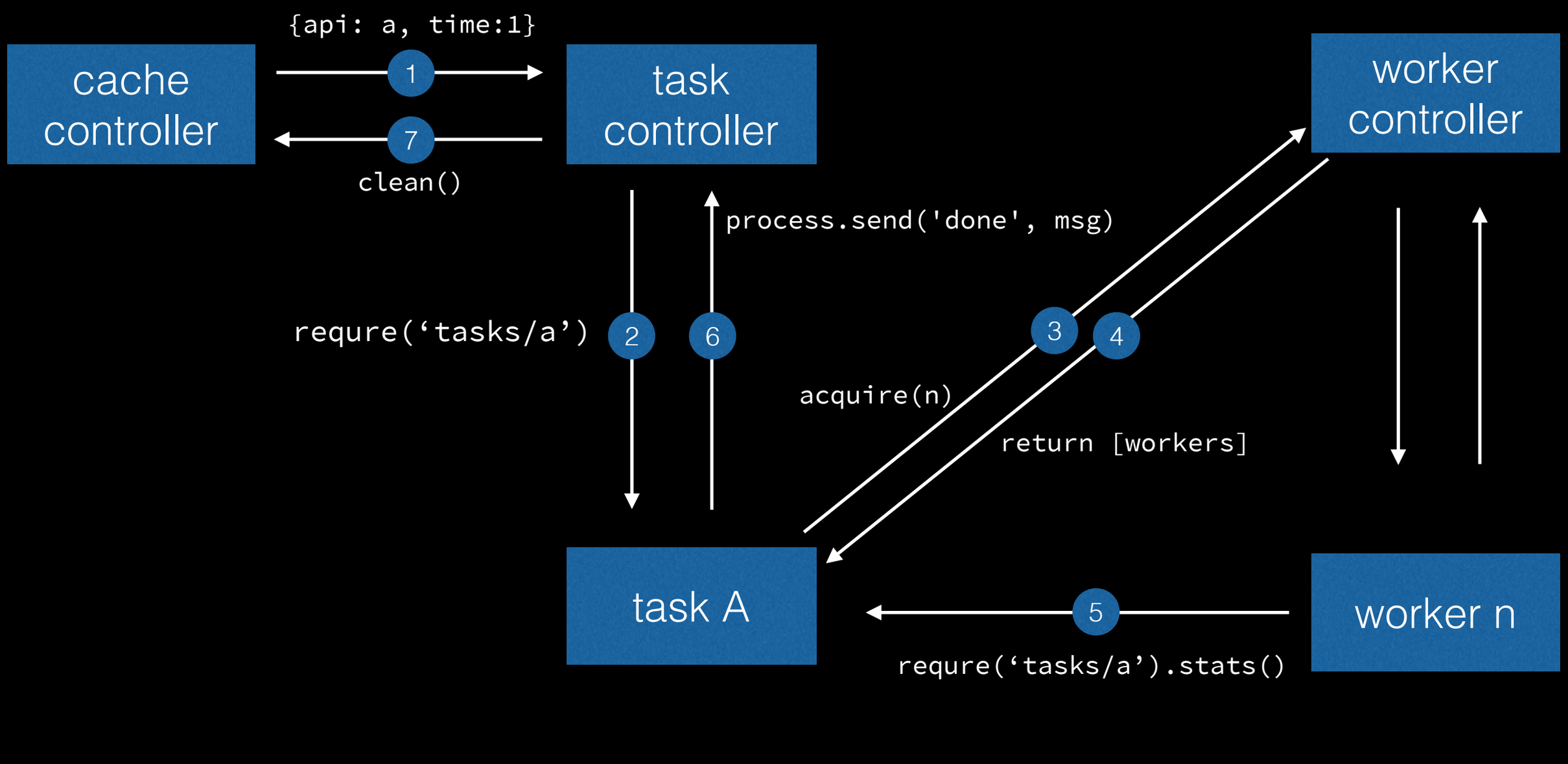
- 将具体执行统计运算的逻辑抽象为一个task，和底层服务解耦
- 统一处理子进程来的消息，进行转发和重试

定义统计任务




```
// runs in master
Task.prototype.statsDoneHandler = function(msg) {
    throw Error('');
};
// runs in master
Task.prototype.emitStats = function(module, time) {
    throw Error('');
};
// runs in worker
Task.prototype.stats = function(msg) {
    throw Error('');
};
```

实时统计流程


{api: a, time:1}
cache controller
1
7
clean()
task controller
worker controller
process.send('done', msg)
require(‘tasks/a’)
2
6
3
4
acquire(n)
return [workers]
task A
5
worker n
require(‘tasks/a’).stats()

遇到过的问题

- MongoDB单次update时间很长很长
- 程序启动后过几分钟请求全部502

连接耗尽

lsof显示结果：

```
tcp        0      0 192.168.42.14:28000     10.200.84.23:10281      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:11049      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:11561      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:11305      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:11817      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:8489       TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:9001       TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:14633      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:14889      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:15657      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:15401      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:16169      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:15913      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:12585      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:12329      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:13097      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:13609      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:13353      TIME_WAIT   -
tcp        0      0 192.168.42.14:28000     10.200.84.23:13865      TIME_WAIT   -
```

解决方案：

- **严格的读写分离**
- **分库/分表，同时自动建索引**

没有索引时：24ms

(1)	{ 15 fields }
cursor	BasicCursor
isMultiKey	false
n	1
nscannedObjects	7454
nscanned	7454
nscannedObjectsAllPlans	7454
nscannedAllPlans	7454
scanAndOrder	false
indexOnly	false
nYields	58
nChunkSkips	0
millis	28

对id和time字段建联合索引：1ms

(1)	{ 16 fields }
cursor	BtreeCursor time_1_id_1
isMultiKey	false
n	1
nscannedObjects	1
nscanned	1
nscannedObjectsAllPlans	1
nscannedAllPlans	1
scanAndOrder	false
indexOnly	false
nYields	0
nChunkSkips	0
millis	0

遇到过的问题

- 内存泄漏
- Redis一个list长度达到20万之多

解决方案：

- **对url进行转换处理，只处理关键路径**
- **谨慎使用常驻内存的js数组、对象**

遇到过的问题

- MongoDB连接数持续升高

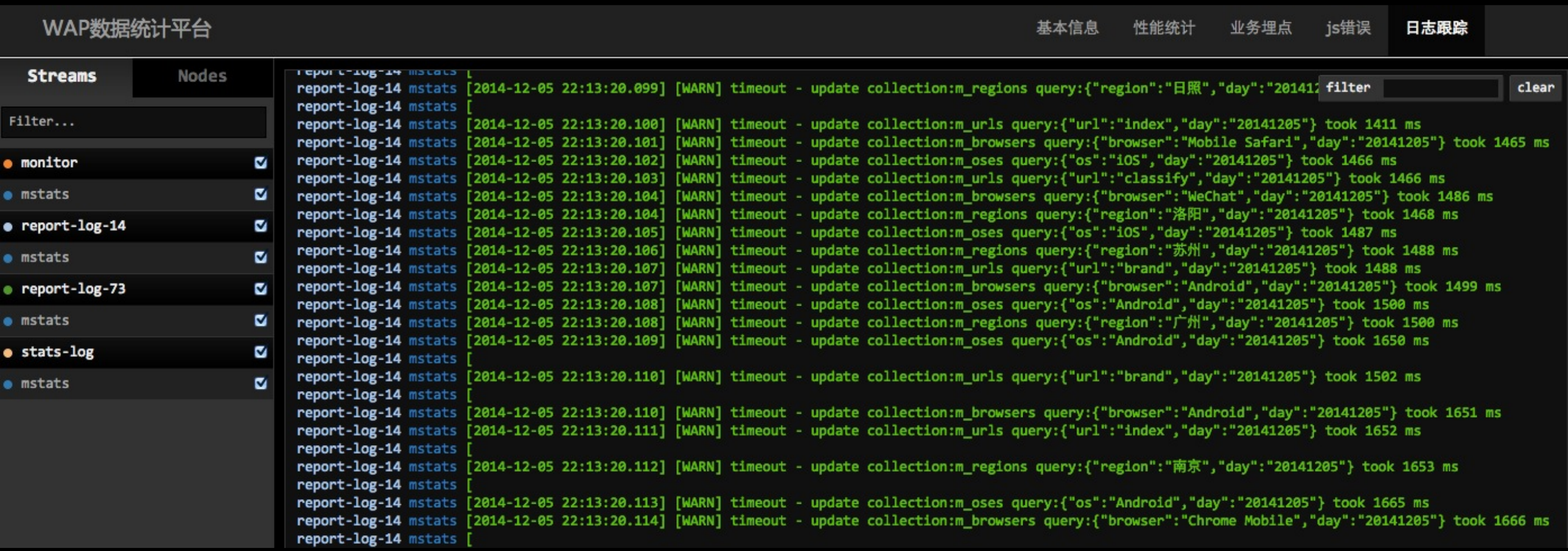
解决方案：

- **调整默认连接数**
- **使用连接池并调整并发连接数**

https://github.com/coopernurse/node-pool

➡ 连接数稳定了，但是很高

Tips: debug


WAP数据统计平台
基本信息
性能统计
业务埋点
js错误
日志跟踪
Streams
Nodes
Filter...
monitor
mstats
report-log-14
mstats
report-log-73
mstats
stats-log
mstats
filter
clear
report-log-14 mstats [2014-12-05 22:13:20.099] [WARN] timeout - update collection:m_regions query:{"region":"日照","day":"201412
report-log-14 mstats [2014-12-05 22:13:20.100] [WARN] timeout - update collection:m_urls query:{"url":"index","day":"20141205"} took 1411 ms
report-log-14 mstats [2014-12-05 22:13:20.101] [WARN] timeout - update collection:m_browsers query:{"browser":"Mobile Safari","day":"20141205"} took 1465 ms
report-log-14 mstats [2014-12-05 22:13:20.102] [WARN] timeout - update collection:m_oses query:{"os":"iOS","day":"20141205"} took 1466 ms
report-log-14 mstats [2014-12-05 22:13:20.103] [WARN] timeout - update collection:m_urls query:{"url":"classify","day":"20141205"} took 1466 ms
report-log-14 mstats [2014-12-05 22:13:20.104] [WARN] timeout - update collection:m_browsers query:{"browser":"WeChat","day":"20141205"} took 1486 ms
report-log-14 mstats [2014-12-05 22:13:20.104] [WARN] timeout - update collection:m_regions query:{"region":"洛阳","day":"20141205"} took 1468 ms
report-log-14 mstats [2014-12-05 22:13:20.105] [WARN] timeout - update collection:m_oses query:{"os":"iOS","day":"20141205"} took 1487 ms
report-log-14 mstats [2014-12-05 22:13:20.106] [WARN] timeout - update collection:m_regions query:{"region":"苏州","day":"20141205"} took 1488 ms
report-log-14 mstats [2014-12-05 22:13:20.107] [WARN] timeout - update collection:m_urls query:{"url":"brand","day":"20141205"} took 1488 ms
report-log-14 mstats [2014-12-05 22:13:20.107] [WARN] timeout - update collection:m_browsers query:{"browser":"Android","day":"20141205"} took 1499 ms
report-log-14 mstats [2014-12-05 22:13:20.108] [WARN] timeout - update collection:m_oses query:{"os":"Android","day":"20141205"} took 1500 ms
report-log-14 mstats [2014-12-05 22:13:20.108] [WARN] timeout - update collection:m_regions query:{"region":"广州","day":"20141205"} took 1500 ms
report-log-14 mstats [2014-12-05 22:13:20.109] [WARN] timeout - update collection:m_oses query:{"os":"Android","day":"20141205"} took 1650 ms
report-log-14 mstats [2014-12-05 22:13:20.110] [WARN] timeout - update collection:m_urls query:{"url":"brand","day":"20141205"} took 1502 ms
report-log-14 mstats [2014-12-05 22:13:20.110] [WARN] timeout - update collection:m_browsers query:{"browser":"Android","day":"20141205"} took 1651 ms
report-log-14 mstats [2014-12-05 22:13:20.111] [WARN] timeout - update collection:m_urls query:{"url":"index","day":"20141205"} took 1652 ms
report-log-14 mstats [2014-12-05 22:13:20.112] [WARN] timeout - update collection:m_regions query:{"region":"南京","day":"20141205"} took 1653 ms
report-log-14 mstats [2014-12-05 22:13:20.113] [WARN] timeout - update collection:m_oses query:{"os":"Android","day":"20141205"} took 1665 ms
report-log-14 mstats [2014-12-05 22:13:20.114] [WARN] timeout - update collection:m_browsers query:{"browser":"Chrome Mobile","day":"20141205"} took 1666 ms

Tips: memory leak

```
var memwatch = require(‘memwatch');

// online
memwatch.on('leak', function(info) {
    memwatchLogger.warn('leak', JSON.stringify(info));
});
```

- memwatch给出的信息很少
- 还得靠人、靠经验

Tips: live reload

- `fs.watchFile();`
  - `delete require.cache[filepath + ‘.js'];`
  - `require(filepath);`

Tips: 定期任务

- 程序内部起一个http server，外部通过curl和程序交互
- 通知report服务器更新写入的表
- 同时创建响应的索引

```
0 0 * * * curl "http://127.0.0.1:28010?cmd=up_time_daily"
```

Q & A

Join us：dm.yang@vipshop.com