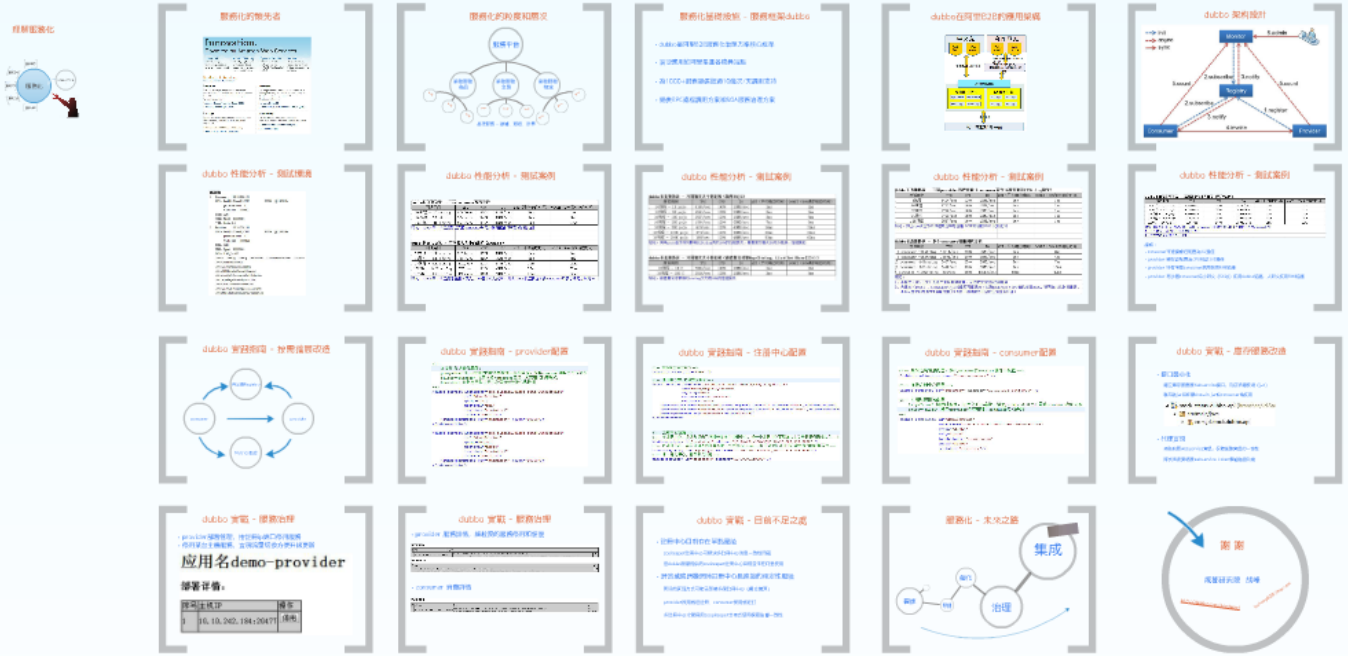
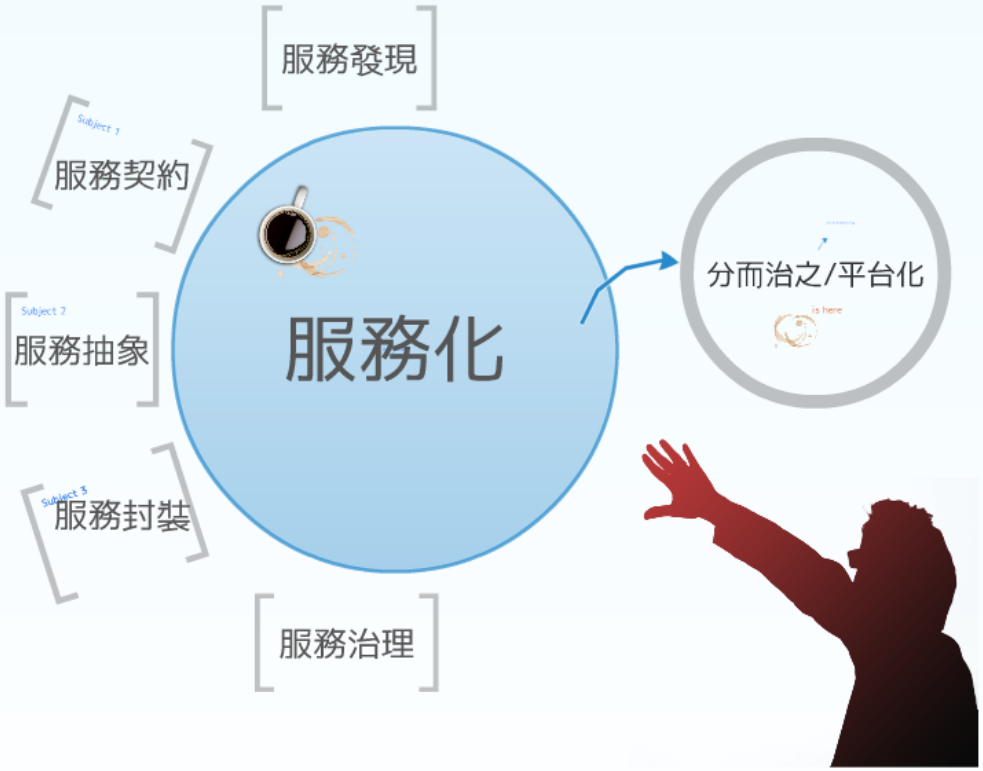
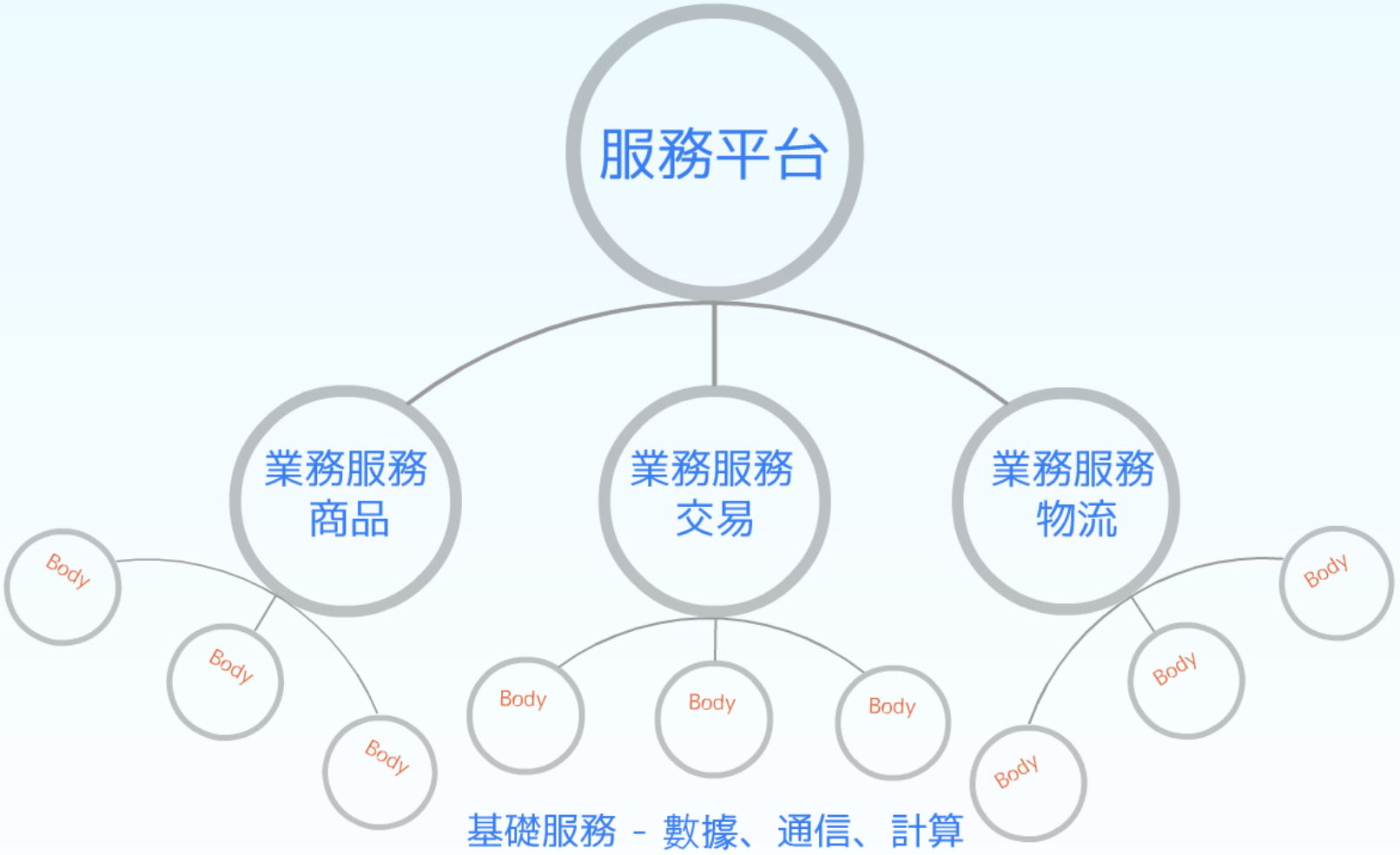

服務化的萌芽者
服務化的挑戰和需求
服務化基礎框架 - 服務框架dubbo
dubbo在阿里B2B的應用架構
dubbo 架構設計
dubbo 性能分析 - 測試結果
dubbo 性能分析 - 測試案例
dubbo 性能分析 - 測試案例
dubbo 性能分析 - 測試案例
dubbo 性能分析 - 測試案例
dubbo 實踐指南 - 按需指南改造
dubbo 實踐指南 - provider配置
dubbo 實踐指南 - 注冊中心配置
dubbo 實踐指南 - consumer配置
dubbo 實戰 - 應用優雅改造
dubbo 實戰 - 服務治理
应用名demo-provider
部署详情：
dubbo 實戰 - 服務治理
dubbo 實戰 - 目前不足之處
服務化 - 未來之路
集成
治理
測 測

理解服務化



服務化的領先者

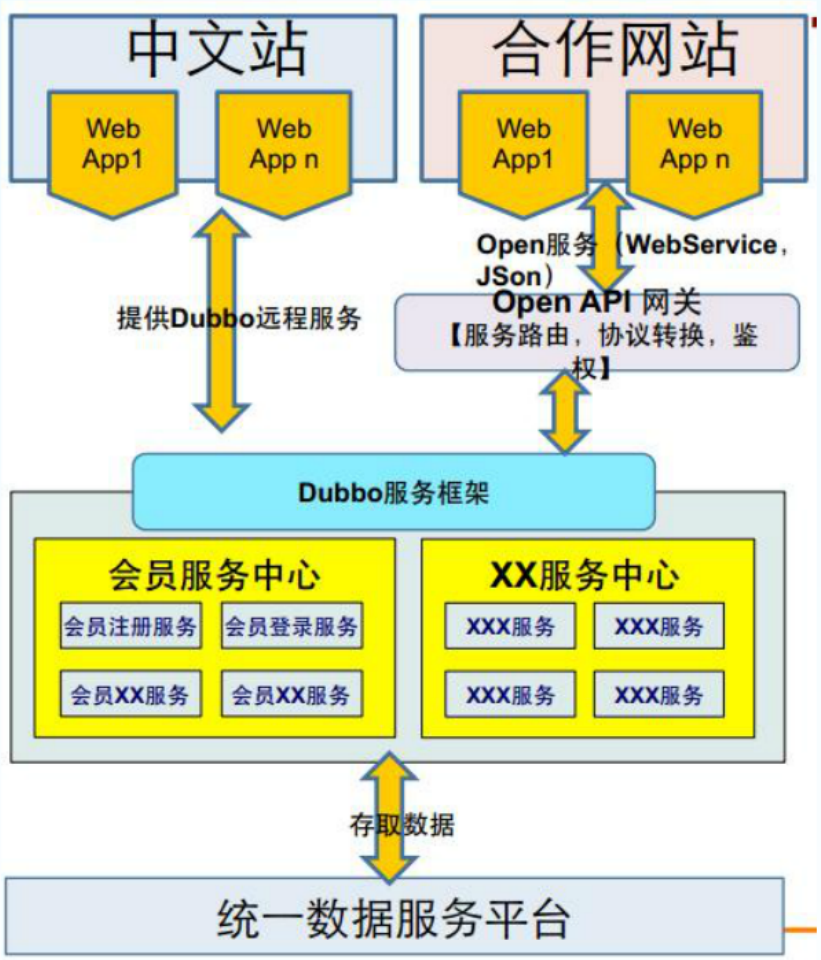
服務化的粒度和層次



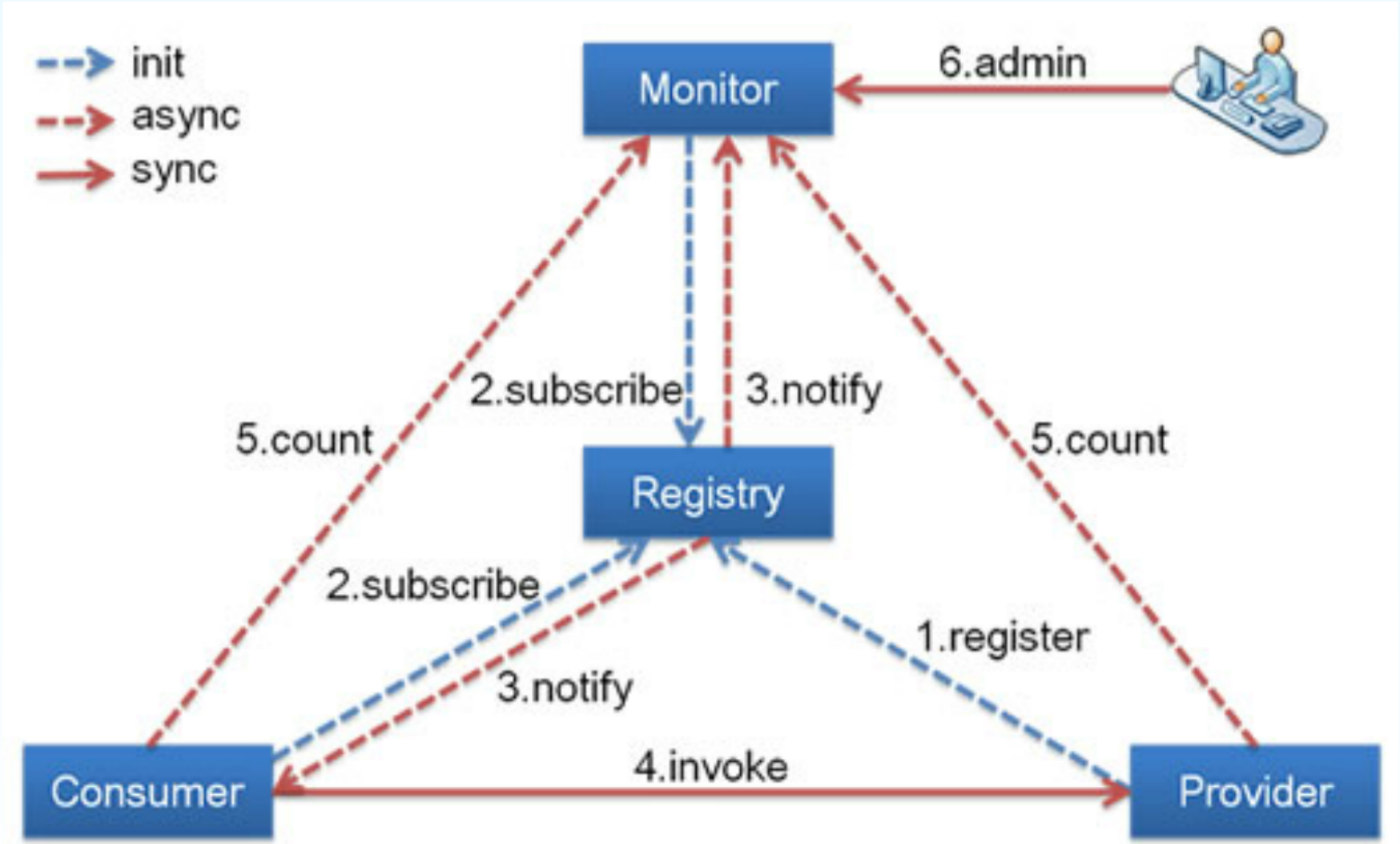
服務化基礎設施 - 服務框架dubbo

- dubbo是阿里B2B服務化治理方案核心框架
- 廣泛應用於阿里集團各成員站點
- 為1000+服務提供超過10億次/天調用支持
- 提供RPC遠程調用方案和SOA服務治理方案

dubbo在阿里B2B的應用架構



dubbo 架构設計



dubbo 性能分析 - 測試環境

测试环境

1. Consumer： 10.10.224.193
   CPU：Intel(R) Xeon(R) CPU E5506 @ 2.13GHz
   processor count 4
   Cache size 4096K
   内存：32G
   网络：Speed 1000Mb/s
   工具：Jmeter 2.4
2. Provider： 10.10.224.194
   CPU：Intel(R) Xeon(R) CPU E5506 @ 2.13GHz
   processor count 4
   Cache size 4096KB
   内存：32G
   网络：Speed 1000Mb/s
   JVM：1.6.0_24-b07

```
-server -Xmx2g -Xms2g -Xmn256m -XX:PermSize=128m -Xss256k
-XX:+DisableExplicitGC
-XX:+UseConcMarkSweepGC
-XX:+CMSParallelRemarkEnabled
-XX:+UseCMSCompactAtFullCollection
-XX:LargePageSizeInBytes=128m
-XX:+UseFastAccessorMethods
-XX:+UseCMSInitiatingOccupancyOnly
-XX:CMSInitiatingOccupancyFraction=70
```

dubbo 性能分析 - 測試案例

dubbo长连接协议 - 不同consumer线程比较

测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
10线程 - 1K string	9567/sec	150%	13MB/sec	1ms	1ms
20线程 - 1K string	9741/sec	140%	13MB/sec	1ms	3ms
100线程 - 1K string	9984/sec	140%	13MB/sec	9ms	11ms

结论：dubbo单一长连接最佳支持consumer采用10线程配置具有最好的性价比

dubbo长连接协议 - 不同報文大小的比較（String）

测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
10线程 - 10K string	2503/sec	120%	13MB/sec	3ms	5ms
10线程 - 20K string	1411/sec	120%	35MB/sec	7ms	8ms
10线程 - 50K string	625/sec	120%	38MB/sec	15ms	19ms
10线程 - 100K string	370/sec	120%	45MB/sec	26ms	32ms
10线程 - 200K string	190/sec	120%	48MB/sec	52ms	62ms

结论：dubbo单一长连接随着报文增大，I/O基本成为瓶颈，因此dubbo长连接适合多consumer场景才能充分利用IO

dubbo 性能分析 - 测試案例

dubbo长连接协议 - 不同報文大小的比較（简单POJO）

测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
10线程 - 1K pojo	8189/sec	145%	13MB/sec	1ms	2ms
10线程 - 5K pojo	4541/sec	135%	30MB/sec	2ms	3ms
10线程 - 10K pojo	2527/sec	120%	13MB/sec	3ms	5ms
10线程 - 20K pojo	1390/sec	120%	35MB/sec	7ms	8ms
10线程 - 50K pojo	619/sec	120%	40MB/sec	16ms	19ms
10线程 - 100K pojo	374/sec	120%	45MB/sec	26ms	31ms
10线程 - 200K pojo	188/sec	120%	48MB/sec	53ms	62ms

结论：简单pojo由于序列影响比String有约10%的性能损失，随着报文增大IO成为瓶颈，性能接近

dubbo长连接协议 - 不同報文大小的比較（嵌套集合对象Map<String, List<Set<User[]>>>）

测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
10线程 - 1K C	7884/sec	145%	13MB/sec	1ms	2ms
10线程 - 10K C	2516/sec	135%	13MB/sec	3ms	5ms

结论：嵌套集合对象相对string大约有18%的性能损失

dubbo 性能分析 - 測試案例

dubbo长连接协议 - 不同provider线程比较（consumer固定10线程发送1KString报文）

测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
8线程	8424/sec	120%	13MB/sec	1ms	1ms
16线程	9722/sec	140%	13MB/sec	1ms	1ms
32线程	9770/sec	140%	13MB/sec	1ms	1ms
64线程	9723/sec	140%	13MB/sec	1ms	1ms
128线程	9537/sec	140%	13MB/sec	1ms	1ms

结论：如上cpu设定为32线程最佳和经验值（CPU内核数×8）比较符合

dubbo长连接协议 - 多个consumer极限测试比较

测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
3 consumer -1Kstring	14314/sec	250%	25MB/sec	1ms	3ms
5 consumer -1Kstring	15901/sec	270%	25MB/sec	2ms	4ms
3 consumer -10Kstring	5007/sec	250%	60MB/sec	5ms	7ms
6 consumer -10Kstring	5418/sec	300%	80MB/sec	8ms	13ms
4 consumer -100Kstring	936/sec	350%	120MB/sec	48ms	42ms

结论：

1、小报文（1K、10K）无法产生足够的流量，无法使CPU或IO达到瓶颈

2、大报文（100K），4consumer×10线程即可压满IO（实测120M/sec）CPU消耗接近350%，说明在IO达到极限后，dubbo框架自身的CPU消耗占据了3.5核（测试服务不包含任何业务处理）

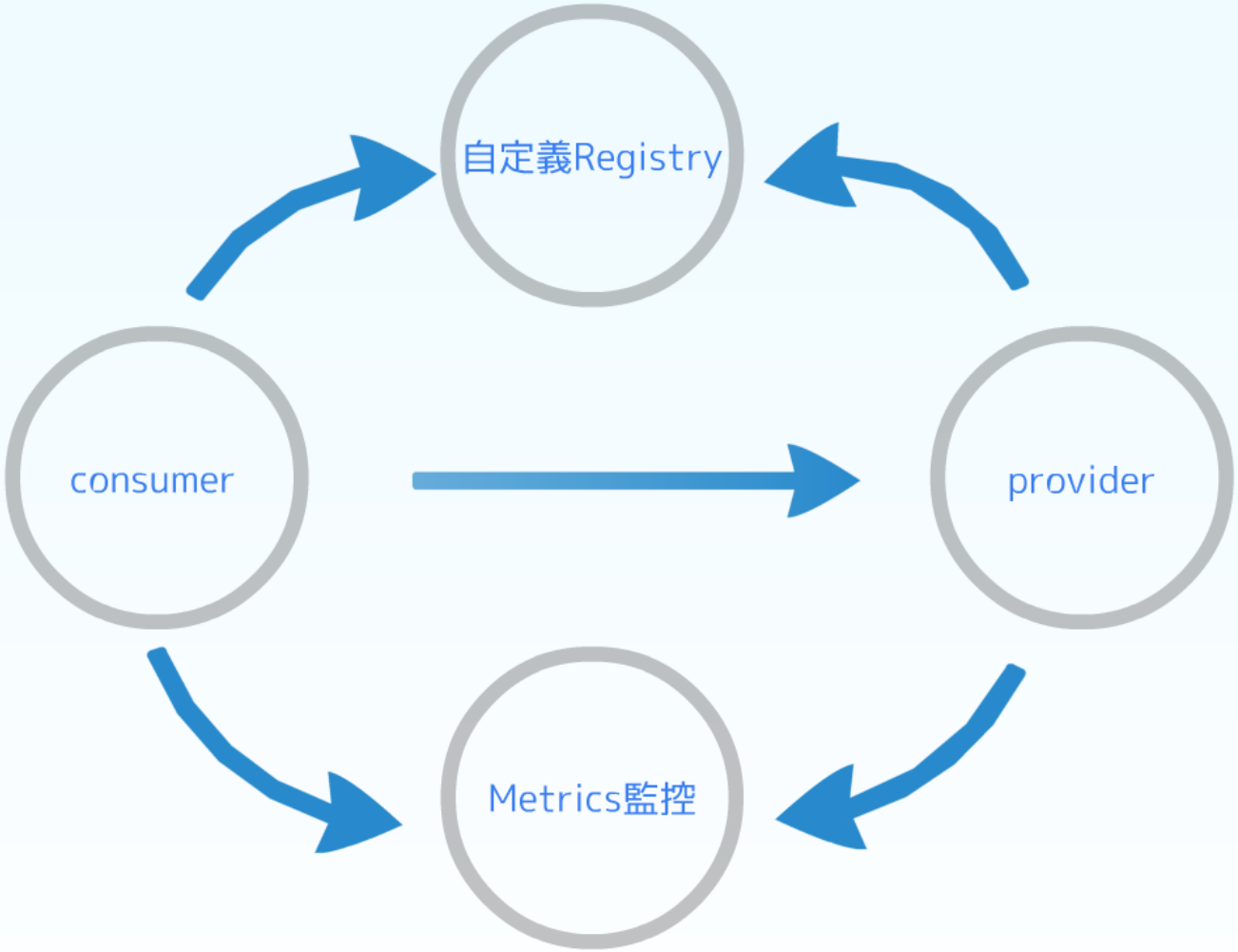
dubbo 性能分析 - 測試案例

dubbo RMI短连接协议 - 不同報文大小的比較（简单POJO）					
测试用例	TPS	CPU	IO	ART（平均响应时间）	90%RT（90%请求响应时间）
10线程 - 1K pojo	8402/sec	280%	80MB/sec	1ms	2ms
10线程 - 5K pojo	6806/sec	300%	105MB/sec	1ms	2ms
10线程 - 10K pojo	5527/sec	300%	120MB/sec	1ms	2ms
10线程 - 50K pojo	1650/sec	250%	120MB/sec	5ms	7ms
10线程 - 100K pojo	888/sec	250%	120MB/sec	11ms	12ms
结论:RMI协议由于采用短连接，在单一consumer同样线程数（10个）相比长连接的dubbo协议					
1、在小报文（5K以内）时没有明显优势					
2、大报文时优势很明显（>10K）					

總結：

- consumer單連接線程配置為10最佳
- provider 線程數配置為CPU核數×8最佳
- provider 只有單個consumer調用採用RMI協議
- provider 有多個consumer時小報文（5K以）使用dubbo協議，大報文使用RMI協議

dubbo 實踐指南 - 按需擴展改造



dubbo 實践指南 - provider配置

```
<!--
    多协议分组服务暴露：
    gourp="xx" 把不同协议的服务暴露成不同的分组，方便consumer端配置指定协议
    filter="metrics" 无侵入接入metrics监控，无需在代码中埋点
    heartbeat 参数表示和注册中心保持长连接心跳检测
-->
<dubbo:service interface="com.alibaba.dubbo.demo.DemoService"
               ref="demoService"
               group="rmi"
               protocol="rmi"
               registry="registry1"
               filter="metrics">
    <dubbo:parameter key="heartbeat" value="60000"/>
</dubbo:service>

<dubbo:service interface="com.alibaba.dubbo.demo.DemoService"
               ref="demoService"
               group="dubbo"
               protocol="dubbo"
               registry="registry1"
               filter="metrics">
    <dubbo:parameter key="heartbeat" value="60000"/>
</dubbo:service>
```

dubbo 實践指南 - 注册中心配置

```
<!-- 暴露服务协议配置 -->
<dubbo:protocol port="9090" />

<!-- 注册中心本社也暴露为服务 -->
<dubbo:service interface="com.alibaba.dubbo.registry.RegistryService"
               ref="memoryRegistryService"
               registry="N/A"
               ondisconnect="disconnect"
               callbacks="1000">
    <dubbo:method name="subscribe"><dubbo:argument index="1" callback="true" /></dubbo:method>
    <dubbo:method name="unsubscribe"><dubbo:argument index="1" callback="true" /></dubbo:method>
    <dubbo:parameter key="heartbeat" value="60000"/>
</dubbo:service>


<!-- 注册中心引用 -->
<!-- 多注册中心，竖线分隔表示同时连接多个注册中心，任一个注册中心不能连接上会导致服务无法启动 -->
<dubbo:registry id="registry1" address="10.10.224.193:9090|10.10.224.194:9090" />
<!-- 多注册中心，逗号分隔表示任意连接一个支持failover，但不能保证注册中心需自行保障数据同步 -->
<dubbo:registry id="registry2" address="10.10.224.193:9090,10.10.224.194:9090"/>
<!-- 单一注册中心，存在单点问题 -->
<dubbo:registry id="registry3" address="127.0.0.1:9090" />
```

dubbo 實践指南 - consumer配置

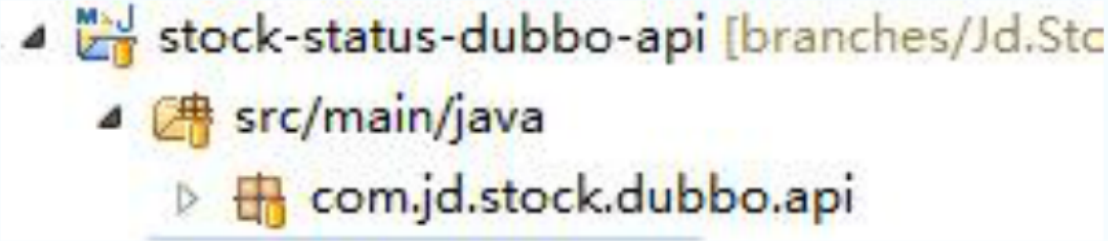
```
<!-- 当前应用信息配置，接入metrics后monitor属性不配置 -->
<dubbo:application name="demo-consumer" />

<!-- 连接注册中心配置 -->
<dubbo:registry id="registry" address="10.10.224.194:9090" />

<!-- 引用远程服务配置
    lazy="true" 调用时连接，初始化时不连接，避免provier不可用时导致consumer启动失败
    check="false" 避免provider不存在时，consumer启动失败
-->
<dubbo:reference id="demoService"
                 interface="com.alibaba.dubbo.demo.DemoService"
                 group="dubbo"
                 lazy="true"
                 loadbalance="roundrobin"
                 check="false"
                 registry="registry"/>
```

dubbo 實戰 - 庫存服務改造

- 接口最小化

  獨立庫存服務原Webservice接口，形成單獨模塊（jar）

  發布該jar和相關domain jar給consumer端使用



- 代理實現

  委託給原webservice實現，保障業務實現的一致性

  隱式傳參實現原webservice token攔截驗證功能

dubbo 實戰 - 服務治理

- provider部署管理，按註冊ip端口停用服務
- 停用某台主機服務，實現流量切換方便升級更新

dubbo 實戰 - 服務治理

- provider 服務詳情，細粒度的服務停用和恢復

提供服务详情：

序号	名称	协议	订阅者	URL	
1	rmi/com.alibaba.dubbo.demo.DemoService	rmi		rmi://10.10.242.184:1218/com.alibaba.dubbo.demo.DemoService?anyhost=true&application=demo-provider&dubbo=2.0.9&group=rmi&heartbeat=60000&interface=com.alibaba.dubbo.demo.DemoService&methods=sayHello&prompt=dubbo&revision=&service.filter=metrics	停用

已停用URL详情：

序号	URL	操作
1	dubbo://10.10.242.184:20881/com.alibaba.dubbo.demo.DemoService2?anyhost=true&application=demo-provider&dubbo=2.0.9&group=dubbo&heartbeat=60000&interface=com.alibaba.dubbo.demo.DemoService&methods=sayHello&prompt=dubbo&revision=&service.filter=metrics	恢复

- consumer 消費詳情

订阅服务详情：

序号	服务名称	参数	APP
1	dubbo/com.alibaba.dubbo.demo.DemoService	methods=sayHello dubbo=2.0.9 application=demo-consumer check=false singleton=true lazy=true interface=com.alibaba.dubbo.demo.DemoService group=dubbo loadbalance=roundrobin	demo-provider

dubbo 實戰 - 目前不足之處

- 註冊中心目前存在單點風險

  zookeeper註冊中心可解決多註冊中心消息一致性問題

  但dubbo開源提供的zookeeper註冊中心目前並未在阿里使用

- 跨地域或跨機房時註冊中心長連接的穩定性風險

  更好的實踐方式可能是部署多個註冊中心（獨立機房）

  provider採用就近註冊、consumer採用就近訂

  多註冊中心之間採用zoopkeeper分佈式協同保障數據一致性

服務化 - 未來之路

集成

優化

提煉

解耦

1st step

治理

謝 謝

成都研究院　胡峰

http://weibo.com/mindwind

hufeng@360buy.com