

JOBOLE

伯乐在线

马哥教育

www.magedu.com



2016全新Linux运维实战+python实战班震撼上线!  
先就业,后付款!年薪24W!问能班还有少量名额!

首页

最新文章

开发

IT技术

职场

业界

极客

创业

访谈

在国外

伯乐在线 > [首页](#) > [所有文章](#) > [Web前端](#) > 前后端分离的思考与实践（六）

## 前后端分离的思考与实践（六）

2014/06/20 · [Web前端](#), [开发](#) · [java](#), [Nginx](#), [Node.js](#)

分享到:

4

进击Node.js基础（二）  
玩儿转Swift 2.0（第四季）

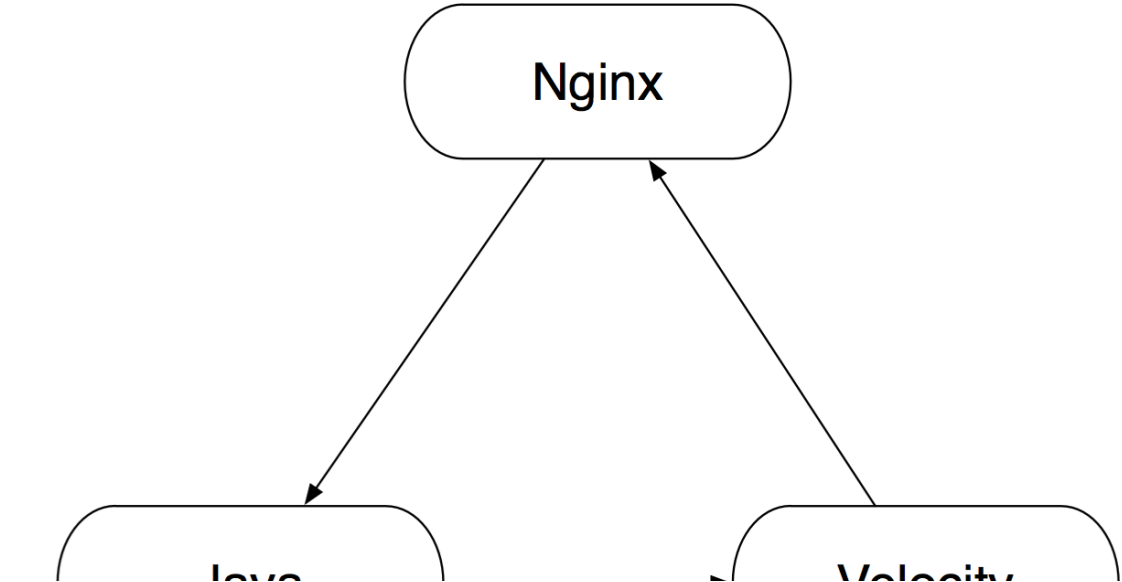
node+mongodb 建站攻略（一期）  
jQuery源码解析（架构与依赖模块）

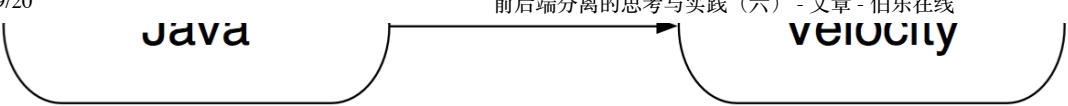
原文出处：[淘宝UED - 筱谷](#)

### Nginx + Node.js + Java 的软件栈部署实践

起

关于前后端分享的思考，我们已经有五篇文章阐述思路与设计。本文介绍淘宝网[收藏夹](#)将 Node.js 引入传统技术栈的具体实践。淘宝网线上应用的传统软件栈结构为 Nginx + Velocity + Java，即：





在这个体系中，Nginx 将请求转发给 Java 应用，后者处理完事务，再将数据用 Velocity 模板渲染成最终的页面。

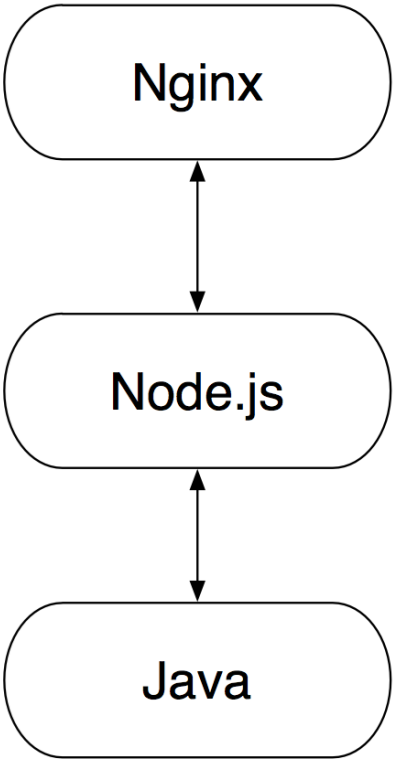
引入 Node.js 之后，我们势必要面临以下几个问题：

- 1. 技术栈的拓扑结构该如何设计，部署方式该如何选择，才算是科学合理？
- 2. 项目完成后，该如何切分流量，对运维来说才算是方便快捷？
- 3. 遇到线上的问题，如何最快地解除险情，避免更大的损失？
- 4. 如何确保应用的健康情况，在负载均衡调度的层面加以管理？

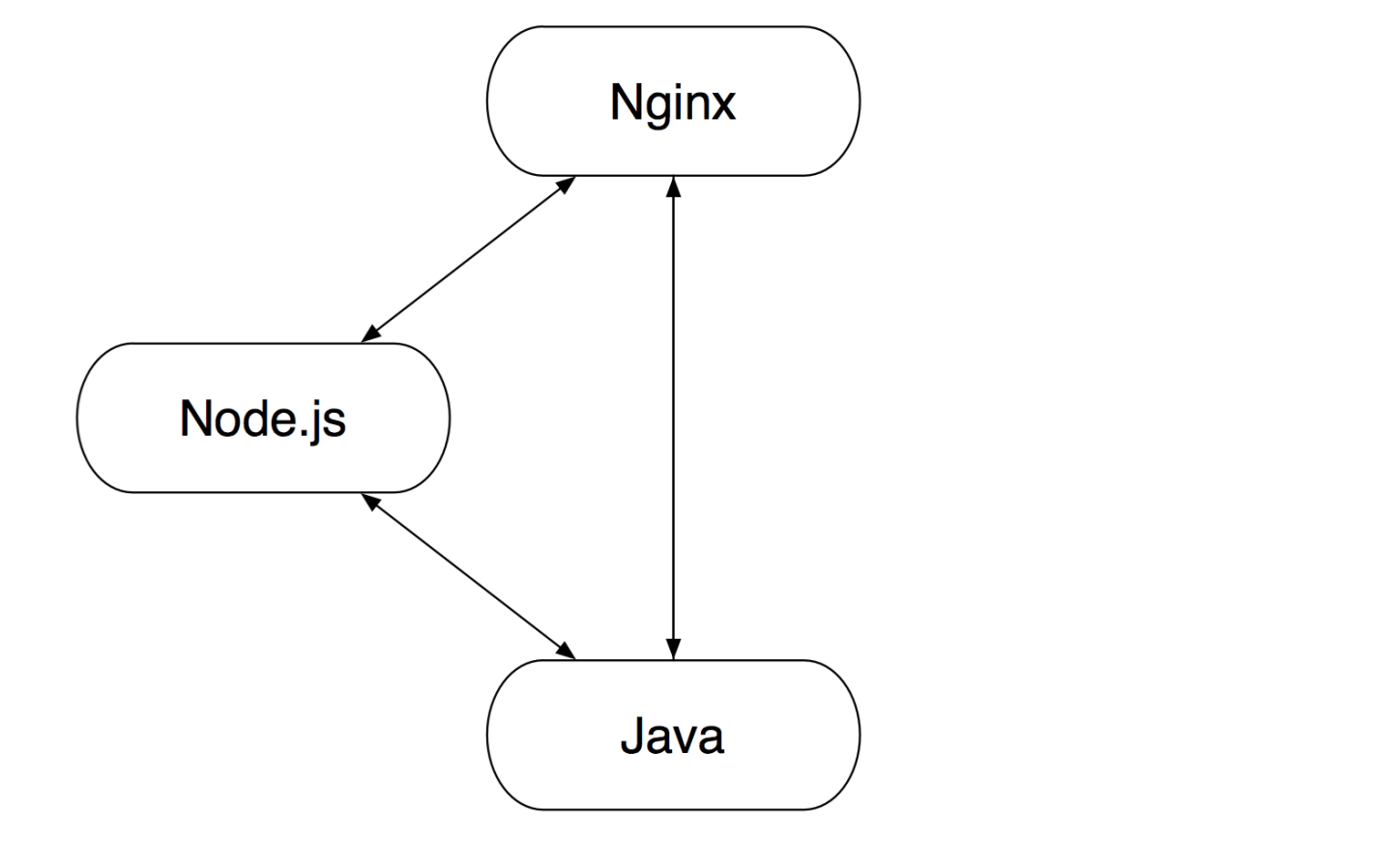
承

系统拓扑

按照我们在[前后端分离的思考与实践（二） - 基于前后端分离的根版探索](#)一文中的思路，Velocity 需要被 Node.js 取代，从而让这个结构变成：



这当然是最理想的目标。然而，在传统栈中首次引入 Node.js 这一层毕竟是个新尝试。为了稳妥起见，我们决定只在收藏夹的宝贝收藏页面 ([shoucang.taobao.com/item\\_collect.htm](#)) 启用新的技术，其它页面沿用传统方案。即，由 Nginx 判断请求的页面类型，决定这个请求究竟是要转发给 Node.js 还是 Java。于是，最后的结构成了：



**部署方案**

上面的结构看起来没什么问题了，但其实新问题还等在前面。在传统结构中，Nginx 与 Java 是部署在同一台服务器上的，Nginx 监听 80 端口，与监听高位 7001 端口的 Java 通信。现在引入了 Node.js，需要新跑一个监听端口的进程，到底是将 Node.js 与 Nginx + Java 部署在同一台机器，还是将 Node.js 部署在单独的集群呢？我们来比较一下两种方式各自特点：

同集群部署（Nginx、Java、Node.js 的进程都跑在一台机器上）	分集群部署（Nginx + Java 一个集群，Node.js 单独一个集群）
发布时打成一个包，形成强约束，方便一起发布，有问题时一并回滚	发布与回滚时需要负载均衡入口来精确控制机器 up 状态，分别操作两个集群
通讯在本地进行，网络性能优于经过集群之间的内网交换机	通讯在集群之间发生，有轻微的网络通讯成本
Node.js 与 Java 的配比不能做到一对多或者多对一	Node.js 与 Java 的配比可以做到多对一或者一对多

淘宝网收藏夹是一个拥有千万级日均 PV 的应用，对稳定性的要求性极高（事实上任何产品的线上不稳定都是不能接受的）。如果采用同集群部署方案，只需要一次文件分发，两次应用重启即可完成发布，万一需要回滚，也只需要操作一次基线包。性能上来说，同集群部署也有一些理论优势（虽然内网的交换机带宽与延时都是非常乐观的）。至于一对多或者多对一的关系，理论上可能做到服务器更加充分的利用，但相比稳定性上的要求，这一点并不那么急迫需要去解决。所以在收藏夹的改造中，我们选择了同集群部署方案。

灰度方式

为了保证最大程度的稳定，这次改造并没有直接将 Velocity 代码完全去掉。应用集群中有将近 100 台服务器，我们以服务器为粒度，逐渐引入流量。

也就是说，虽然所有的服务器上都在跑着 Java + Node.js 的进程，但 Nginx 上有没有相应的转发规则，决定了获取这台服务器上请求宝贝收藏的请求是否会经过 Node.js 来处理。其中 Nginx 的配置为：

XHTML

```
1 location = "/item_collect.htm" {
2     proxy_pass http://127.0.0.1:6001; # Node.js 进程监听的端口
3 }
```

只有添加了这条 Nginx 规则的服务器，才会让 Node.js 来处理相应请求。通过 Nginx 配置，可以非常方便快捷地进行灰度流量的增加与减少，成本很低。如果遇到问题，可以直接将 Nginx 配置进行回滚，瞬间回到传统技术栈结构，解除险情。

第一次发布时，我们只有两台服务器上启用了这条规则，也就是说大致有不到 2% 的线上流量是走 Node.js 处理的，其余的流量的请求仍然由 Velocity 渲染。以后视情况逐步增加流量，最后在第三周，全部服务器都启用了。至此，生产环境 100% 流量的商品收藏页面都是经 Node.js 渲染出来的（可以查看源代码搜索 Node.js 关键字）。

## 转

灰度过程并不是一帆风顺的。在全量切流量之前，遇到了一些或大或小的问题。大部分与具体业务有关，值得借鉴的是一个技术细节相关的陷阱。

### 健康检查

在传统的架构中，负载均衡调度系统每隔一秒钟会对每台服务器 80 端口的特定 URL 发起一次 get 请求，根据返回的 HTTP Status Code 是否为 200 来判断该服务器是否正常工作。如果请求 1s 后超时或者 HTTP Status Code 不为 200，则不将任何流量引入该服务器，避免线上问题。

这个请求的路径是 Nginx -> Java -> Nginx，这意味着，只要返回了 200，那这台服务器的 Nginx 与 Java 都处于健康状态。引入 Node.js 后，这个路径变成了 Nginx -> Node.js -> Java -> Node.js -> Nginx。相应的代码为：

XHTML

```
1 var http = require('http');
2 app.get('/status.taobao', function(req, res) {
3     http.get({
4         host: '127.1',
5         port: 7001,
6         path: '/status.taobao'
7     }, function(res) {
8         res.send(res.statusCode);
9     }).on('error', function(err) {
10        logger.error(err);
11        res.send(404);
12    });
13 });
```

但是在测试过程中，发现 Node.js 在转发这类请求的时候，每六七次就有一次会耗时几秒甚至十几秒才能得到 Java 端的返回。这样会导致负载均衡调度系统认为该服务器发生异常，随即切断流量，但实际上这台服务器是能够正常工作的。这显然是一个不小的问题。

排查一番发现，默认情况下，Node.js 会使用 HTTP Agent 这个类来创建 HTTP 连接。这个类实现了 socket 连接池，每个主机+端口对的连接数默认上限是 5。同时 HTTP Agent 类发起的请求中默认带上了 Connection: Keep-Alive，导致已返回的连接没有及时释放，后面发起的请求只能排队。

最后的解决办法有三种：

- 禁用 HTTP Agent，即在在调用 get 方法时额外添加参数 agent: false，最后的代码为：

XHTML

```
1 var http = require('http');
2 app.get('/status.taobao', function(req, res) {
3     http.get({
4         host: '127.1',
5         port: 7001,
6         agent: false,
7         path: '/status.taobao'
8     }, function(res) {
9         res.send(res.statusCode);
10    }).on('error', function(err) {
11        logger.error(err);
12        res.send(404);
13    });
14 });
```

设置 http 对象的全局 socket 数量上限：

XHTML

```
1 http.globalAgent.maxSockets = 1000;
```

在请求返回的时候及时主动断开连接：

XHTML

```
1 http.get(options, function(res) {
2     }).on("socket", function (socket) {
3         socket.emit("agentRemove"); // 监听 socket 事件，在回调中派发 agentRemove 事件
4     });
```

实践上我们选择第一种方法。这么调整之后，健康检查就没有再发现其它问题了。

## 合

Node.js 与传统业务场景结合的实践才刚刚起步，仍然有大量值得深入挖掘的优化点。比如，让 Java 应用彻底中心化后，是否可以考分集群部署，以提高服务器利用率。或者，发布与回滚的方式是否能更加灵活可控。等等细节，都值得再进一步研究。

【附】相关文章列表

1. [《前后端分离的思考与实践（一）》](#)
2. [《前后端分离的思考与实践（二）》](#)
3. [《前后端分离的思考与实践（三）》](#)
4. [《前后端分离的思考与实践（四）》](#)
5. [《前后端分离的思考与实践（五）》](#)

点赞

只5收藏

评论



### 相关文章

- [如何使用 Datalag 监控 NGINX（第三篇）](#)
- [NGINX 1.9.1 中的 Socket 切分](#)
- [短网址开发运维经验总结分享](#)
- [基于 JavaScript 的操作系统你听说过吗？](#)
- [前后端分离的思考与实践（二）](#)
- [Node.js初体验](#)

### 可能感兴趣的话题

- [\[失业就失业\]Finally it comes to me](#)
- [女程序员发奋于一夜之间？有没有发展前景啊？ · · · \[Q 44\]\(#\)](#)
- [中研网公布薪酬数据：晚上加班津贴，中秋节当天也要来加班，我也是醉了... · · · \[Q 56\]\(#\)](#)
- [预算 2K 以内，求推荐开发用笔记本电脑 · · · \[Q 73\]\(#\)](#)
- [昨晚加班到4点多，4:30左右才到家，好久没发帖了，来发个帖聊聊 · · · \[Q 51\]\(#\)](#)
- [后端转前端，转职萌新求指点 · · · \[Q 12\]\(#\)](#)

登录后评论

新用户注册

直接登录

文章 3

输入搜索关键字

搜索



- [本周热门文章](#)
- [本月热门文章](#)
- [热门标签](#)

- 0 [无休止加班的成因](#)
- 1 [我见过最有趣的代码注释，都在这里了...](#)
- 2 [HTTPS 科普扫盲帖](#)
- 3 [PM 叫你去改一个 Bug，后来.....](#)
- 4 [在谷歌招工程师，我看重的是这些](#)
- 5 [HTTPS 工作原理和 TCP 握...](#)
- 6 [TCP/IP之TCP协议：流量控制...](#)
- 7 [LOL设计模式之「策略模式」](#)
- 8 [.Net 高效开发之不可错过的实用工具](#)
- 9 [跟我一起写shell补全脚本（Zsh篇）](#)

加入伯乐在线专栏作者

扩大知名度

还能得 赞赏

业界热点资讯

更多»



[甲骨文准备将NetBeans交给Apache管理](#)  
11 小时前 · 2



[让计算机崩溃的简短代码](#)  
3 天前 · 20



[PHP 7.1 新特性一览](#)  
14 小时前 · 3



[redis3.2新功能 - GEO地理位置命令介绍](#)  
1 天前 · 4



[JS 又是第一编程语言：GitHub 2016 年度报告](#)  
3 天前 · 33 · 3

精选工具资源

更多资源»



[android-remote-notifications: 从远程JSON文件拉取...](#)  
[Android](#), [通知](#)



[PHP CPP: 一个开发PHP扩展的C++库](#)  
[PHP](#), [扩展](#)



[Apache Mahout: 经典机器学习算法库](#)  
[Java](#), [机器学习](#)





[WilliamChart: 优美和直观的图表库](#)  
[Android](#), [图表](#)




[Singularity: 方便部署和操作的Mesos框架](#)  
[Java](#), [集群管理](#)


最新评论

-  [Re: 我见过最有趣的代码注释，都在这里...](#)  
12不是很信
- 


- Re: [PM 叫你去改一个 Bug，后来...](#)

现在初三，因为这个原因，理想就是不受雇于人，有一家自己的（小）公司。这种活法太恶心
- 


Re: [你应该知道的计算机网络知识](#)

同意，因为一般一台机器只配有一个网络适配器(网卡)，所以也就默认一个MAC地址对应一台唯一的电脑...
- 


Re: [133 行代码实现质感地形](#)

应该感谢博主搜集了这么好的资源！！辛苦啦--
- 


Re: [133 行代码实现质感地形](#)

谢谢提醒。已经在正文更新了：)
- 

Re: [PM 叫你去改一个 Bug，后来...](#)

很赞的文章，翻译得也非常赞。解决客户的问题，这个是最大的价值，当然也不必要放弃原则：P
- 

Re: [133 行代码实现质感地形](#)

你的连接错啦！！现在修改了地址的 <http://demos.playfuljs.com/terrai...>
- 

Re: [加入伯乐在线专栏作者，扩大知名...](#)

MD 的教程，网络上有的，你自己搜搜。

关于伯乐在线博客

在这个信息爆炸的时代，人们已然被大量、快速并且简短的信息所包围。然而，我们相信：过多“快餐”式的阅读只会令人“虚胖”，缺乏实质的内涵。伯乐在线内容团队正试图以我们微薄的力量，把优秀的原创文章和译文分享给读者，为“快餐”添加一些“营养”元素。

- 快速链接
- [网站使用指南 »](#)
- [问题反馈与求助 »](#)
- [加入我们 »](#)
- [网站积分规则 »](#)
- [网站声援规则 »](#)

关注我们

新浪微博：@伯乐在线官方微博  
RSS： [订阅地址](#)



[程序员的那些事](#)   [UI设计达人](#)   [极客范](#)

合作联系  
Email: [bd@jobbole.com](mailto:bd@jobbole.com)  
QQ：2302462408（加好友请注明来意）

更多频道

- 小组 - 好的话题、有自发的回复、值得信赖的圈子
- 头条 - 分享和发现有价值的内容与观点
- 相亲 - 为IT单身男女服务的征婚传播平台
- 资源 - 优秀的工具资源导航
- 翻译 - 翻译传播优秀的外文文章
- 文章 - 国内外的精选文章
- 设计 - UI、网页，交互和用户体验
- iOS - 专注iOS技术分享
- 安卓 - 专注Android技术分享
- 前端 - JavaScript、HTML5、CSS
- Java - 专注Java技术分享
- Python - 专注Python技术分享

