**Web开发技术报告**

**基于nodejs和mysql实现的电商产品管理和查询试验**

**提交日期： 2016年一月**

# 1新技术概述

随着互联网时代来临，web端的交互越来越多，而电商平台之类的网站往往伴随着高并发，传统的一些后端语言会在处理高并发的同时，会大量占用服务器内存从而影响性能，相对于single-process, multi-thread, 的传统语言例如JAVA, nodejs的single-process, single-thread机制让其在异步、非阻塞、高并发方面有极大作为。

Node.js是一个基于Chrome JavaScript运行时建立的平台， 用于方便地搭建响应速度快、易于扩展的网络应用。Node.js 使用事件驱动， 非阻塞I/O 模型而得以轻量和高效，非常适合在分布式设备上运行的数据密集型的实时应用。

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world. [1]

# 2新技术特点

分析该新技术的技术优势与劣势

nodejs的优势：

1. nodejs可以很快地在服务器端做原型，而且代码量相对JAVA/C会少很多。
2. nodejs使用事件驱动和非阻塞I/O，从而轻量而且高效，在处理高并发方面得心应手（最主要的优势）。
3. nodejs入门简单，开发工程易于上手，并且nodejs的npm包管理是世界上最大的开源库，这就意味着用户有很多资源选择的余地。
4. nodejs没有start-the-event-loop call，也就是时间循环调用，而是在某一个方法执行完毕之后使用回调，从而使代码更加清晰，而其promise机制也很好的避免了多层回调嵌套带来的代码混乱。 ［2］

nodejs的劣势：

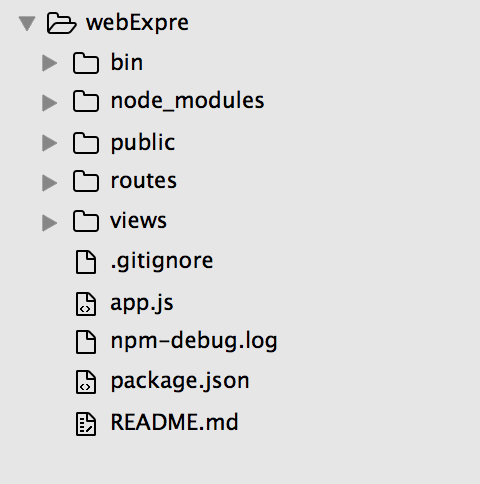
1. 只支持单核CPU，不能充分利用CPU
2. 由于single-process, single-thread，可靠性低，一个环节崩溃，整个系统崩溃。（解决方法：使用nginx反向代理，负载均衡，开多个进程，绑定多个端口）
3. 开源组件库质量参差不全，向下不兼容。Debug不方便。 ［3］

# 3新技术产品介绍与比较

分析该技术的架构,使用方法简介,同类技术产品间的比较

nodejs架构：

我们使用的是目前最流行的nodejs框架express，其自动生成的框架如下：



1. bin目录下是nodejs的底层执行文件，封装了最基础的方法，
2. node\_modules下是nodejs需要使用的相应包，里面是各种封装好的中间件，使用npm管理。（本身可以没有，可以根据package.json安装）
3. public下面是网站需要的各种静态资源，包含图片、css文件、以及充当controller作用的javascript文件等。
4. routes 下面是网站的路由层，即module，负责路由处理和各种逻辑、数据处理。
5. views 下面是试图文件。
6. app.js 中定义了各种控制器以及定义各类资源的使用。
7. npm-debug.log 纪录log日志
8. package.json 纪录版本信息，并且定义了该工程需要使用的各个依赖包及其版本。

使用方法：

1. 首先请确保安装node.js（详见官网）
2. 在shell中进入该工程主目录，执行npm install安装package.json中定义的依赖（需要sudo权限，如果npm install被墙，请百度cnpm并做相关操作）
3. 第二步完成之后，执行：npm start 即可访问本机3000端口查看网站效果

与其它语言和框架比较：

由于过于繁多，现贴上各类全文网站做的比较以供参考。

1）和php比较 <http://webapplog.com/php-vs-node-js/>

2）和python比较

<https://www.quora.com/What-are-the-benefits-of-developing-in-Node-js-versus-Python>

1. 和java比较

<http://www.infoworld.com/article/2883328/java/java-vs-nodejs-an-epic-battle-for-developer-mindshare.html>

毫无疑问的一点是： nodejs在处理高并发和多I/O方面更佳出色，这个只出现两年的新开发框架正在迅猛发展。

# 4新技术应用范围

适用场景：

1）. RESTful API

这是NodeJS最理想的应用场景，可以处理数万条连接，本身没有太多的逻辑，只需要请求API，组织数据进行返回即可。它本质上只是从某个数据库中查找一些值并将它们组成一个响应。由于响应是少量文本，入站请求也是少量的文本，因此流量不高，一台机器甚至也可以处理最繁忙的公司的API需求。

2）. 统一Web应用的UI层

目前MVC的架构，在某种意义上来说，Web开发有两个UI层，一个是在浏览器里面我们最终看到的，另一个在server端，负责生成和拼接页面。

不讨论这种架构是好是坏，但是有另外一种实践，面向服务的架构，更好的做前后端的依赖分离。如果所有的关键业务逻辑都封装成REST调用，就意味着在上层只需要考虑如何用这些REST接口构建具体的应用。那些后端程序员们根本不操心具体数据是如何从一个页面传递到另一个页面的，他们也不用管用户数据更新是通过Ajax异步获取的还是通过刷新页面。 [3]

3）. 大量Ajax请求的应用

例如个性化应用，每个用户看到的页面都不一样，缓存失效，需要在页面加载的时候发起Ajax请求，NodeJS能响应大量的并发请求。　　总而言之，NodeJS适合运用在高并发、I/O密集、少量业务逻辑的场景。

适用范围：

各种访问量很大但是逻辑并不复杂的互联网公司，或者内部管理系统。

已经使用的公司：

国内：BAT均在部分场景和业务使用nodejs，其他互联网公司也有使用。

国外：1. Netflix - [Building With Node.js At Netflix](https://www.talentbuddy.co/blog/building-with-node-js-at-netflix/)

1. New York Times - [Building With Node.js At The New York Times](https://www.talentbuddy.co/blog/building-with-node-js-at-the-new-york-times/)
2. PayPal - [Building With Node.js at PayPal](https://www.talentbuddy.co/blog/building-with-node-js/)
3. Medium - [On Building With Node.js At Medium](https://www.talentbuddy.co/blog/on-building-with-node-js-at-medium-and-growing-as-a-developer/)
4. LinkedIn - [Building With Node.js At LinkedIn](https://www.talentbuddy.co/blog/building-with-node-js-at-linkedin/)
5. Uber - [Node.js Meetup: Distributed Web Architectures - Curtis Chambers, Uber](https://www.joyent.com/developers/videos/node-js-office-hours-curtis-chambers-uber) [4]

# 5总结与体会

总结新技术学习的启发、对其应用前景进行展望

分析新技术对环境、社会可持续发展的影响，会产生什么样的社会经济价值

对于新技术的学习让我们体会到了技术的日新月异，就在几年前，我们完全无法想象脱离了后段工程师之后我们竟然还能进行整个网站的全部开发，前端工程师已经不再是后端工程师的附属，他们也能撑起一片天了。在这个发展迅速的互联网时代，我们只有不断学习和充实自己，才能适应这个飞速发展的环境。

而且nodejs的崛起势以及它的优缺点必然会导致人们对其工具和缺点的更严格的审查，网络安全也会更加受重视，这也恰好促进了互联网的发展。

Nodejs的应用，大大减少了很多公司的开发成本，加快了开发效率，从而会使中国的互联网行业发展进一步加速，未来中国互联网行业将更佳多彩缤纷。让我们拭目以待。 ［5］

# 参考文献

1. <https://nodejs.org/en/> nodejs官网概述
2. <https://nodejs.org/en/about/> nodejs官网about 第三段
3. <http://blog.sina.com.cn/s/blog_53786db50102e9m9.html>等其他博客等
4. <https://www.quora.com/What-companies-are-using-Node-js-in-production>
5. <http://www.infoworld.com/article/2882497/node-js/nodejs-goes-pro-new-opportunities-and-risks.html>