目 录

[1. 运维 1](#_Toc32415153)

[2. 开发 2](#_Toc32415154)

[2.1 编程语言 2](#_Toc32415155)

[2.1.1 C++ 2](#_Toc32415156)

[2.1.2 Python 2](#_Toc32415157)

[2.1.3 Go 2](#_Toc32415158)

[2.1.4 Java 2](#_Toc32415159)

[2.2 算法 2](#_Toc32415160)

[2.2.1 对称加密算法 2](#_Toc32415161)

[2.2.2 非对称加密算法 2](#_Toc32415162)

[2.3 设计模式 2](#_Toc32415163)

[3. 运营 3](#_Toc32415164)

[4. 其他 4](#_Toc32415165)

[4.1 知识共享 4](#_Toc32415166)

[4.2 4](#_Toc32415167)

[5. Python 5](#_Toc32415168)

# 运维

200台服务器以下：

使用开源的监控软件。

Cacti是因为对交换机的监控特别方便，几乎是傻瓜式的。

200——1000台服务器：

深入使用各种开源监控软件，开发自定义插件

1000台服务器以上：

更加深入使用开源软件，解决监控性能等问题，或者自己开发监控软件。

运维思路上的转变，远比单纯提升运维技术更有价值，

而运维真正的价值应该跟研发团队保持一致，

聚焦到效率、稳定和成本上来。

# 开发

## 编程语言

### C++

### Python

### Go

### Java

## 算法

### 对称加密算法

### 非对称加密算法

#### Https

## 设计模式

## 代码测试

充分的测试

使用安全性检查工具，通过软件分析的方法来辅助开发者发现潜在漏洞。

Oyente：一个python语言编写的工具，判断代码中有没有常见的安全漏洞，也会提示出可能有安全隐患的地方。

solidity-coverage: 一个nodejs编写的silidity代码覆盖率测试工具，需要结合测试网络一起使用。

solgraph: 一个nodejs工具，可以将一个智能合约作为输入，输出一个dot图文件，能将智能合约的功能控制流程花城一个流程图，也可以标注出潜在的安全漏洞。

## 项目需求评审

总结一下，其实就是一个不断拆解的过程

关于什么是需求评审，不在这里做解释与定义了，直接进入主题。

关于需求评审的几个原则：

1、需求评审不是谁要说服谁，而是我们要就某一个具体问题寻找到最优的解决方案。

我们有很多PM总是会抱着「我要说服研发」的态度来做需求评审，所以整个需求评审的基本氛围是辩论，是相互的寻找极端情况支撑自己的观点，试图让对方让步，这是非常错误的开局。

2、所有问题的讨论，都必须要先建立基本共识，然后基于这个共识再细化。

什么是基本共识呢？就是我们首先要把双方调整到一个频道上来。不要在不同的频道上相互努力，那没意义。比如，我们先要定义清楚什么是激活用户、什么是注册用户；比如我用到的这个名词指代的是什么这样的基础的小问题。

3、先不要在僵持不下的观点上浪费时间，先求同，后攻异。

在已经形成基本共识的基础上，如果一个小问题有分歧，且一时半会讨论不清楚，那就先放下，继续讨论其他的，等其他的解决了，再回来解决这个差异点上的问题。

关于需求评审的推进步骤：

1、先说我们这次需求的目标与目的。

简单说就是先讲明白我们要做什么，我们为什么要做这个。

就像是演讲的时候，先讲一个能够引起观众兴趣的故事或者观点，然后再提出一个在这个美好的故事中需要解决的难题。

这个问题是经常会产品经理忽略的，但也恰恰是需求评审中最最重要的，它的重要性甚至超过了后面所有的步骤！

我看过太多的产品经理在需求评审的时候，上来就打开画好的原型图，开始讲功能，讲流程。这是完全错误的。一定要花时间先讲为什么要做这个，要达到什么目标，并且一定要在这个目标上达成一致意见。

就像我之前说的，「[设计师，别急着打开设计软件](http://mp.weixin.qq.com/mp/appmsg/show?__biz=MjM5NTI5MzM2MA==&appmsgid=10000241&itemidx=1&sign=eb442507eb5e4db64139cba14be37e63#wechat_redirect)」，产品经理在需求评审之前，请别着急讲功能点与流程，这非常重要。

2、在目标与目的达成一致的基础上，再说你准备怎么做

在谈怎么做的时候，我们是默认为什么做大家达成了一致的。这个时候，不要再返回去讨论第一个问题了，聚焦在怎么做上。

在我们讨论问题的时候，在我们评审一个需求的时候，很多人会很着急，当你说要做这个的时候，他的思路会跑的很快，立刻就先跑到了想要是这么干，我们目前的架构会有什么问题，会存在什么潜在风险，已经跑到细节上了。

这样很不科学，也很不高效，很容易把问题混到一起去了，然后就会陷入到细节的争吵与辩论，然后就走偏了。这样的讨论也是低效的。

谈第一个层次问题的时候，就不要想第二个层次的事情，谈第二个层次的事情的时候，就不要再回去质疑第一个层次的结论了。

讲述你准备怎么做的过程，实际上就是要描述你的解答对解决问题而言的好处。

关于「准备怎么做」的讨论，会有2个结果：

2.1、你认同或者大部分认同我的方案

那么，就按照这个方案来执行，其中有一些不太认同的，我们坐下来讨论，看是否有更好的方案来做。

2.2、你不认同我的方案那么，我们先回到第一个问题「关于我们要做这个事情的目标与目的」是否认同。如果认同目标了，那就是实现方案的问题了，我们可以坐下来研究。如果不认同这个目标，那这评审也就没得玩了，这个状态一般极少出现。

好，现在问题出在不认同方案上了，又有2个结果：

2.2.1、这个方案可以做基本的修正吗？

可以，那我们探讨如何修正

2.2.2、你是否有更好的解决方案呢，这个方案是否可以勉强执行？

可以，那好，先按照这个方案干。如果你有更好的方案，我们按照你的方案干。

2.2.3、我没有更好的方案，我也不认同你的方案

靠，你这完全不是一个团队合作的状态，赶紧换人。

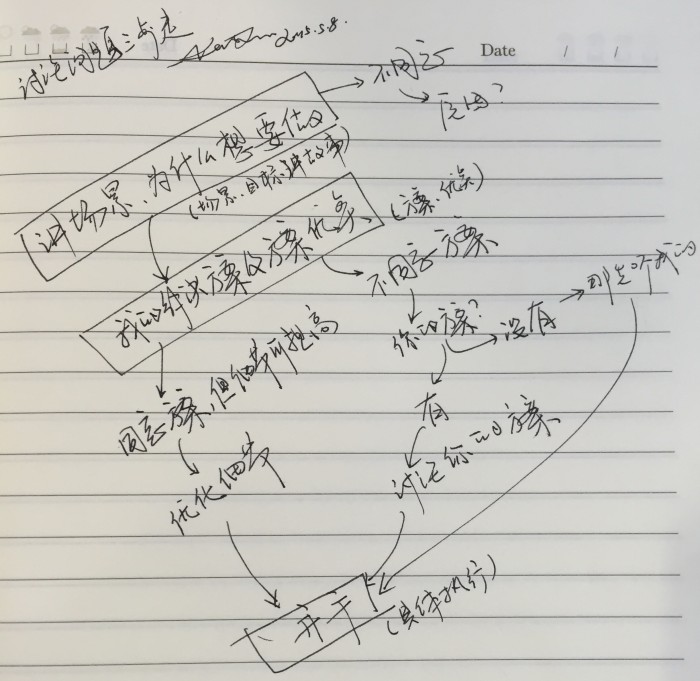
3、方案达成一致了，开始做排期与风险及收益预估

为什么做这个事情，都清楚了，也认同了；

是不是可以这么干，也基本达成一致了；

剩下的就是排期，安排人员，具体去执行了，这是一个新的话题，不展开了。

总结一下，其实就是一个不断拆解的过程，



1. 先说我们的目标与目的，在为什么要做这个事情上达成一致；
2. 再说我是这么想的，你觉得方案是否可行，我们一起看看有没有更好的方案，在怎么做上达成一致；
3. 按照我们讨论的这个做法，看看我们现有的人手，具体怎么一步一步执行。

讨论问题与需求评审一样，是一个解决问题的过程，解决问题的最合理方式就是不断拆解问题，拆解到不能再拆解了，然后分别的寻找解决方式。

另外，相比情商，智商反而有时候显得不是最重要的。

## 代码评审

# 运营

## 商业模式

### 网络效应和规模效应

创业艰难百战多，拼各种资源、执行力。但最终拼的还是认知——第一个就是你对自己的商业模式有多深刻的理解。这几年见过很多O2O、互联网+的项目要做”XX平台“，希望插上”互联网思维“的翅膀，一飞冲天。然而，绝大多数事与愿违，很多做的很累，却没有起色，这里的根源很可能是创业者没有深入理解自己的商业模式。

通常我们讲互联网思维，内涵是非常丰富的——网络效应、体验、免费、口碑、快等等——不同的成功人士都有不同的总结。但我认为其中最重要的还是网络效应，你想做一个商业模式、一个平台，最重要的是要理解自己的商业模式是否具备网络效应，这个决定了事情应该怎么做，各个阶段应该做哪些事情。往往，我们讲平台大部分都是指的网络效应。

网络效应关注的是用户之间的关系，新增用户的关系可以给网络（平台）带来无限的增值可能。至少对平台带来的增长不是线性的。一个有效的网络效应，随着用户的增加，平台的价值在指数级的增长，随着用户的增加，产品会越来越好用。举几个典型的例子，滴滴是一个双向的网络，链接了用户和司机，如果司机增加了，用户体验就会好，用户体验好了，口碑就会起来，用户就会增加，然后再带动司机数量的增加。这样滴滴就形成了一个正向的循环。所以，打车出行这个行业，是典型的网络效应，网络效应的结果也是容易出现头部效应，一家独大。所以，行业内能够留下一两家大的公司是比较正常的。同样的场景，美团外卖，商家和用户构成的网络也具备很强的网络效应。另外还有一些UGC的社区（User Generated Content），如抖音、快手，主播和用户之间的关系也形成了巨强的网络效应，有更多的提供优质内容的主播，用户就会大幅增加，用户大幅增加也会吸引更多有优质内容的主播。网络在不断壮大，网络的边更是在指数级的成长。能够带来的成交可能也就越来越大，整个平台的估值也就越来越大。

但有一些公司看似具有网络效应，但实际上不是，例如：肯德基、摩拜。肯德基和摩拜并没有形成一个良性的网络，随着肯德基开越来越多的门店，用户在稳步的线性增长。同样，摩拜也没有形成网络效应，投放更多的单车和更多的用户增长是线性关系。这类商业模式，对应的是规模效应，和网络效应有这本质的不同。体现在公司发展的过程中，就是该什么时候花钱，该花多少钱，该做什么样的增长。

另外，也有一些模式形成了网络，但很难享受到网络效应带来的快速发展。例如买房子、找工作形成的网络。我本人在做互联网招聘行业，在用人工智能解决一些招聘流程中的问题。这个过程中，我发现招聘的平台形成的网络没有带来网络效应。例如一个猎头平台，平台上形成的双向关系是猎头和企业。如果有更多的企业，会吸引更多优质的猎头；猎头多了起来，同样可以吸引更多的企业。一切看起来很美好，但是网络的效应却很难发挥出来，因为这里有一个很大的瓶颈。我们回顾滴滴，抖音的例子，双边网络中的用户是通过什么联系起来的呢？滴滴的算法调度，抖音的推荐算法，所以可以总结为机器算法。机器算法的好处是，当网络急剧扩张的时候，尤其是产生了更多的”边“，成本不会急剧的增加。所以网络效应的指数型增长就能够体现出来。但对于买房子，或者猎头招聘平台，双边网络建立关系的依据是中介，是人。不是机器。人的精力是有限的，平均一个人能处理的”边“是有限的，如果网络快速成长，边指数级的成长，那么对人的需求也是指数级的增长。成本的巨大增加，使得平台的增长被抵消掉了。所以，如果中间的人不能够被算法替代，那么商业模式就无法按照网络效应的做法进行。更多是一个规模效应的做法。

为什么同样是一个双边的网络，但是网络效应的强度有强弱呢？我觉得原因主要有两点：1）成本；2）需求的个性化程度。

成本主要是指用户使用一个产品过程中，做决策的成本，滴滴，抖音用户的决策成本很低，甚至通过调度算法、推荐算法，用户被动接受就可以了。建立有效连接的成本很低。但是对于买房子、找工作，决策的成本高，决策周期长，非常需要“中介”居中协调。所以，这样的网络里建立连接的成本高，网络效应就比较弱。机器算法在这里很难发挥更大的作用。

从需求个性化的角度，打个车和看一个短视频，个性化的要求不高，仅有的一些喜好，也在算法的帮助下很好的满足了。但买房子、找工作都是人生大事，都是要做深入的研究，谨慎的做出决定。尤其是找工作，我是一个Java工程师，并不是所有的Java的职位都适合我，我有自己想去的地方，喜欢的公司，期望的薪水，理想的办公环境等等。更个性的是，这些条件没有一个是硬性的。所以给建立连接，带来了巨大的成本。个性化的需求是成本的另一面。

网络效应强和规模效应强是不同的商业模式，对应不同的做法。网络效应强的模式，适宜在早期花大钱去推广，做用户的增长，细节效率的提升，产品细节的打磨可以稍稍延后，毕竟，这种模式下，第一要素就是形成强大的网络。规模效应强的模式，首先要做效率，要做体验，这个最小的闭环画圆了之后，才能高效的扩张，才敢开始花钱。如果对自己的商业模式有误解，一厢情愿的认为可以借助网络效应的东风，那么结果是钱花了，事情没有做成。

## 公司管理

### 反腐

【互联网公司反腐，搞不好就死了｜深氪】<https://36kr.com/p/5277885>

【阳光诚信联盟，互联网首个反腐败行业自治组织】<https://www.ctiea.com/>

### 知识共享

既然知识这么重要，那企业究竟应该如何管理知识呢？很多人听到这里，第一个反应就是写下来，委派一位负责人，将大家在实际工作中的好经验搜集上来，进行梳理汇总，形成电子文档，如有需要，可以随时更新补充，知识管理就建设的八九不离十了。这样做听上去十分合情合理，但也许是无用功。用强制的方式来书写经验，大家看不到做这件事的“利”，那他为什么不选择懒惰呢？而且把自己的核心竞争力都说出去，别人取代了自己怎么办？所以最后就会将知识管理演化为面子工程。

还有很多经验，都会和细节相关，这些细节往往在实际的工作中才会显现。比如A客户的痛点是解决传播的问题，B客户的痛点是哪个公司的返点多就选谁，C客户的痛点是要有效地解决战略规划的问题，作为销售人员针对不同的客户要有不同的套路，而且还需要使用到很多商业方面的知识套路，要做到活学活用。这些知识都是隐性知识，怎么用文字记载呢？即便记载了，也不会有人看，因为大家会认为这就是照本宣科，费了半天力气还不出成绩，何必呢？

如果再深想一层，平时我们遇到问题，最先做的并不是去翻书，而是会问问身边的资深专家该怎样做，归根到底是为了更好地解决问题。所以知识共享，共享的是对的人，这个人是处理这方面问题的高手。生病了我们去医院找医生看病，企业出问题了我们会找咨询师来帮忙，利益受到侵害了我们会找律师进行求助。等等类似的场景还有很多，这不就是找正确的人做正确的事吗？严谨的说共享的并不是人，而是不同领域人们头脑中的知识。

将上面的这套思路运用到企业中，我们的思路就会越来越明晰，如果建立共享知识体系，最重要的并不是形成文字，而是通过会议或者培训等具有仪式感的形式，让企业中的每个成员知道，遇到什么样的问题要找什么样的人来求助，这样就能高效能的解决知识共享的问题。所以啊，如果我们要建立属于自己的共享心智模式，就需要从这几个点进行考虑：组织高频率、仪式感强的知识分享活动；保证相关人员都参与，建立对他人知识结构的认知；帮助员工进行换位思考，理解他人行动背后的思维逻辑；格外重视关于失败、妥协、变通的经验分享。通过上面这几点的营建，就可以很有效的将知识分享给企业中的成员，这就是商业中“找专业的人，做专业的事”。

## 开源技术

很多大公司也开源自己技术，但是很多都是他们自己自定义的技术，应用的也只是暂时在他们公司内部。这种技术你敢用吗？他们大公司自己的技术当然有自己的技术人员可以及时帮忙查找修复问题。那其他用他们开源技术的公司呢？出了问题找谁呢？所以小公司都通常不建议用一些不常见的技术，因为怕出了问题，要解决很难。那种新技术风险大，要做技术支持也非常困难。而大公司自己因为有自己庞大的业务量，有实力雄厚的技术团队，使用新技术能改善一台服务器10%的性能就非常可观了，因为他们有上万台，甚至十万，几十万台服务器要管理，10%的性能提升能为他们降低很大一笔运营成本，他们当然有那个实力和动力去开发研究新的技术。而小公司就几十上百台服务器，用新技术风险大，运维难度大，而且实际能节省的运维成本并不高。所以大公司可以使劲折腾新技术，小公司就尽量采用稳定通用的技术方案好了。

# 公司管理章程

## 邮件

### 邮件自动回复



就像即时消息自动回复一样，非常有效。当人要暂时离开，比如关机开会等。就最好设置自动回复，让对方知道你现在在做什么，暂时不方便沟通。