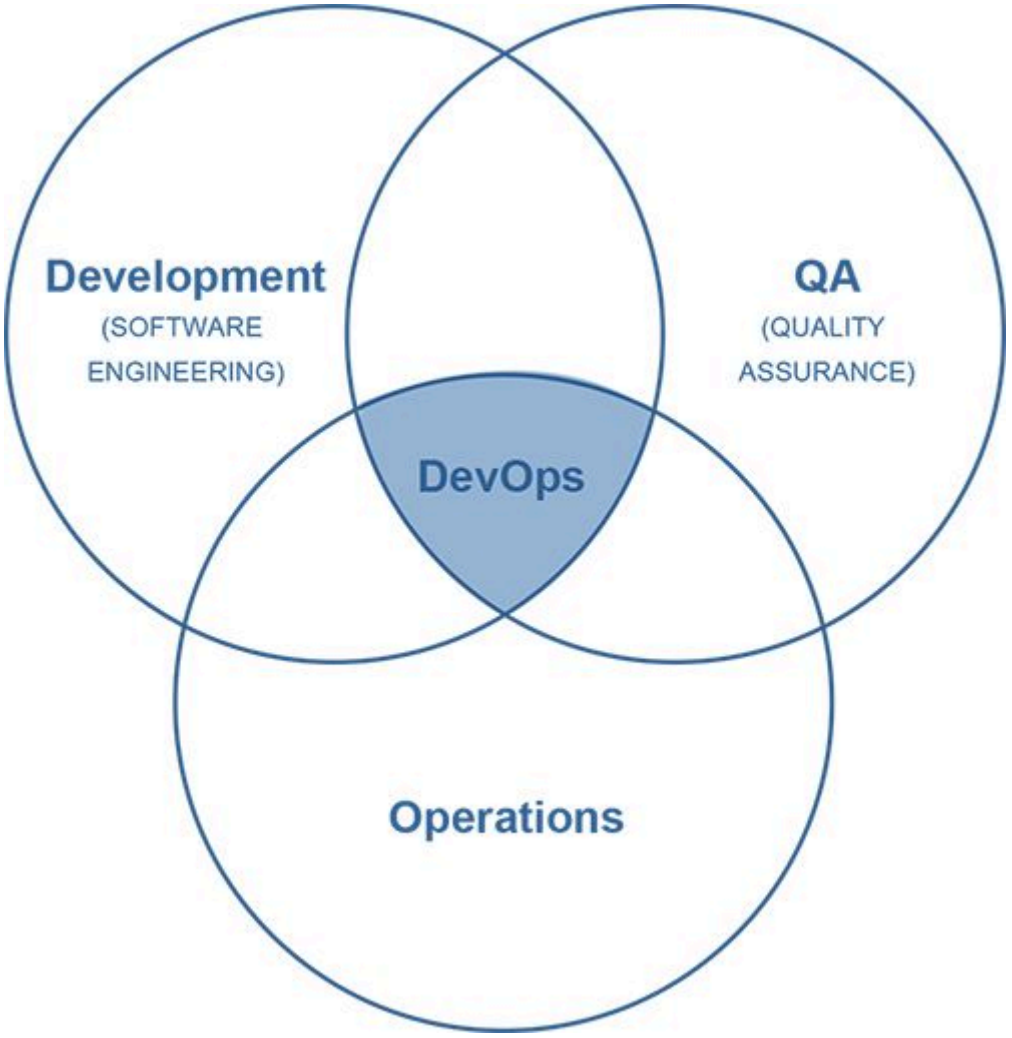
一、 DevOps

1. DevOps的定义

DevOps 是一个完整的面向IT运维的工作流，以 IT 自动化以及持续集成（CI）、持续部署（CD）为基础，来优化程式开发、测试、系统运维等所有环节。

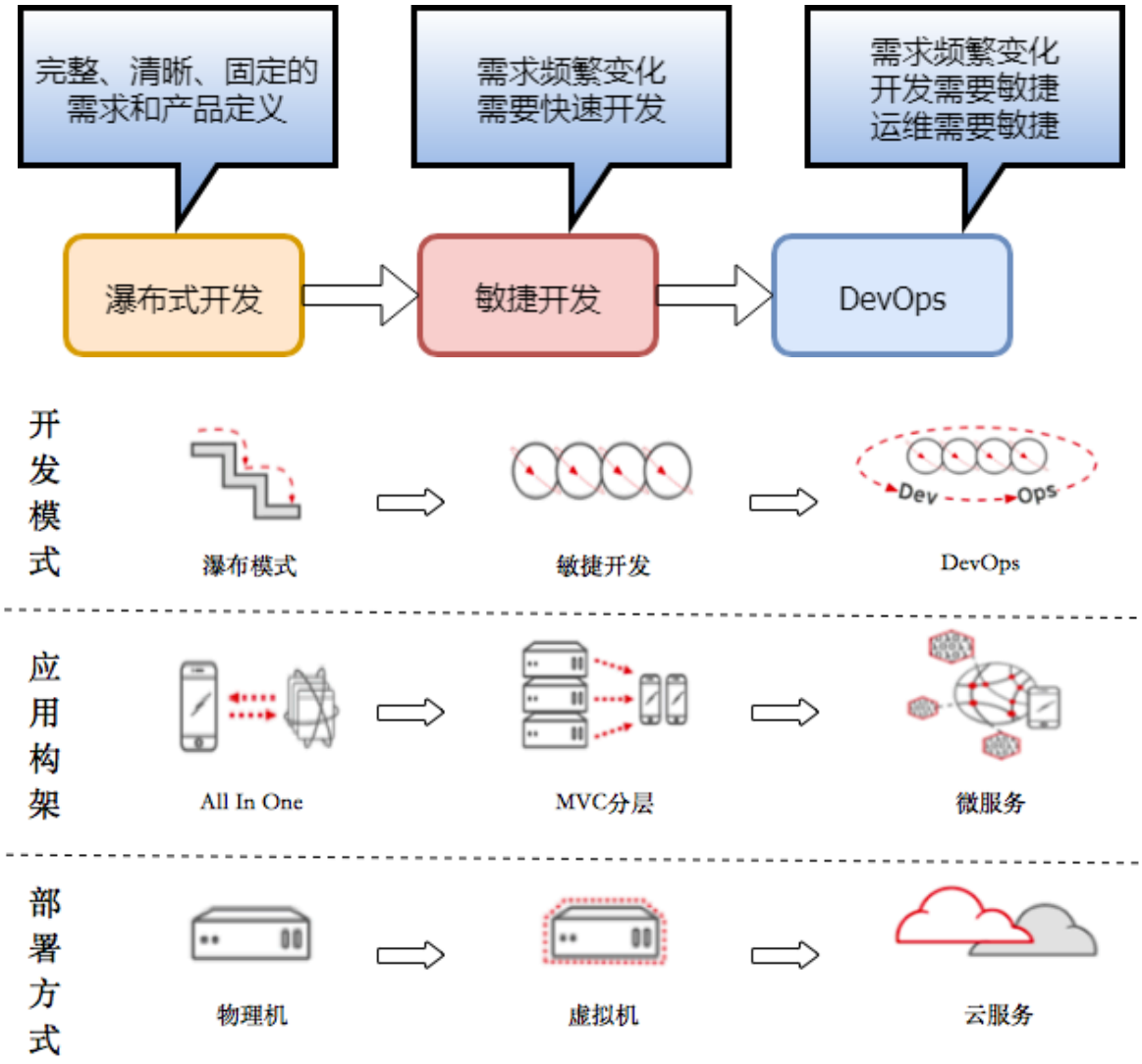
> DevOps **强调的是高效组织团队之间如何通过自动化的工具协作和沟通来完成软件的生命周期管理，从而更快、更频繁地交付更稳定的软件。**

DevOps是为了填补开发端和运维端之间的信息鸿沟，改善团队之间的协作关系。不过需要澄清的一点是，从开发到运维，中间还有测试环节。**DevOps其实包含了三个部分：开发、测试和运维。**



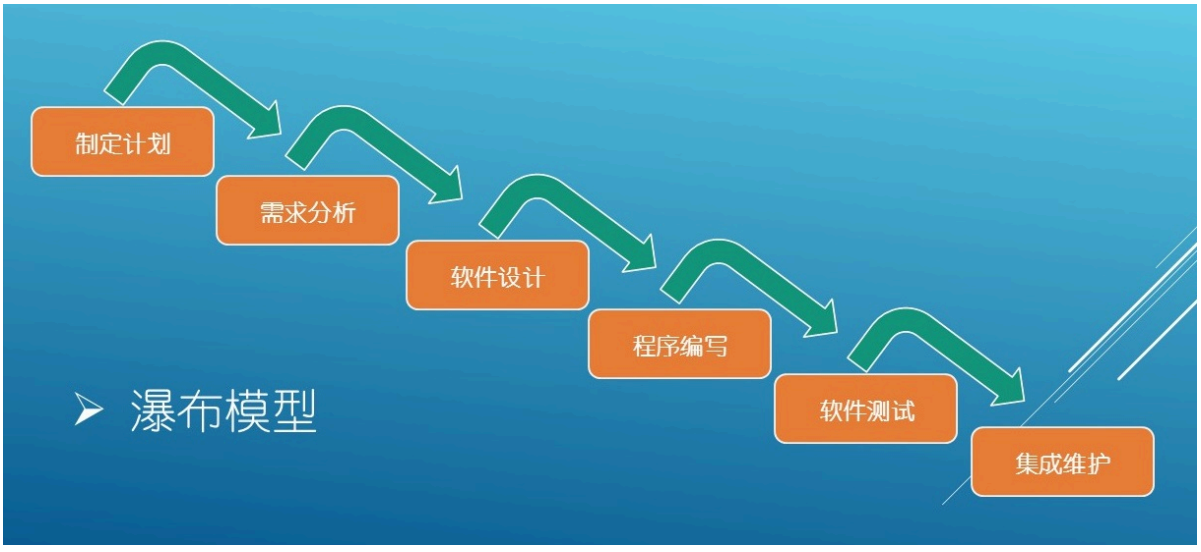
2. 历史变革

由上所述，相信大家对DevOps有了一定的了解。但是除了触及工具链之外，作为文化和技术的方法论，DevOps还需要公司在组织文化上的变革。回顾软件行业的研发模式，可以发现大致有三个阶段：`瀑布式开发`、`敏捷开发`、`DevOps`。



2.1. 瀑布式开发

瀑布开发模式大致的结构如下



核心的思想就是讲软件的生命周期分割为不同的阶段，每个阶段完成不同的任务，而且大多数情况下每一个阶段是由不同的团队完成的。

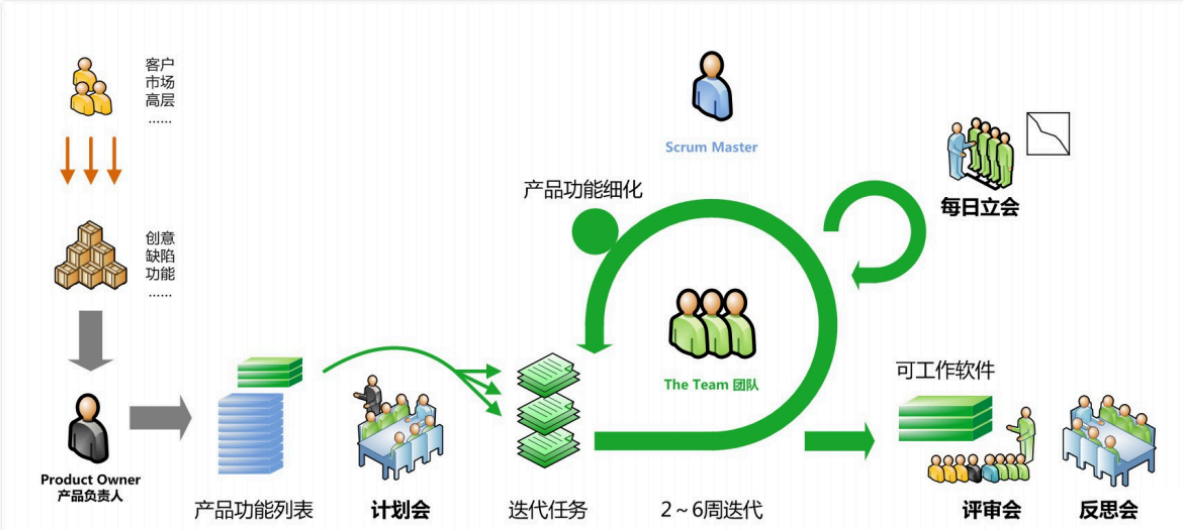
这种开发模式比较适合传统大型软件的开发流程，产品负责人从项目的开始阶段就便于估算，项目开发中的每一个阶段都被预先计划，每一个需求都得到确认，在代码编写之前项目的结束标准就能够确定项目是否成功。这样就保证了项目开发的目的明确性。

但是瀑布开发模式的缺点也是明显的，如果项目的任何一个阶段出现问题都可能导致整个项目的问题。从产品经理的角度来讲瀑布流可以提高整个产品的规划，但是对于开发人员来讲通常情况是噩梦一样的存在，特别是当开发人员在多个项目之间共享的情况。

2.2. 敏捷开发

敏捷开发是另一种在工作中场景的开发模式，敏捷开发的产生一定程度上解决了瀑布开发模式的弊端，将一个大型的瀑布开发流程切分成了非常多的小的子任务，通过连续迭代的方式一步一步的完成一个大型项目的开发。

产品负责人**建立条目化的产品待开发项**，并进行优先级排序。	在**迭代计划会**上，产品负责人讲解本迭代要开发的条目，团队进行估算并放入下一个迭代。	团队在迭代内完成所列需求，每天都开**每日”立“会**以沟通进度和问题。	在迭代终点的**迭代评审会**上，团队向产品负责人等展示开发成果。

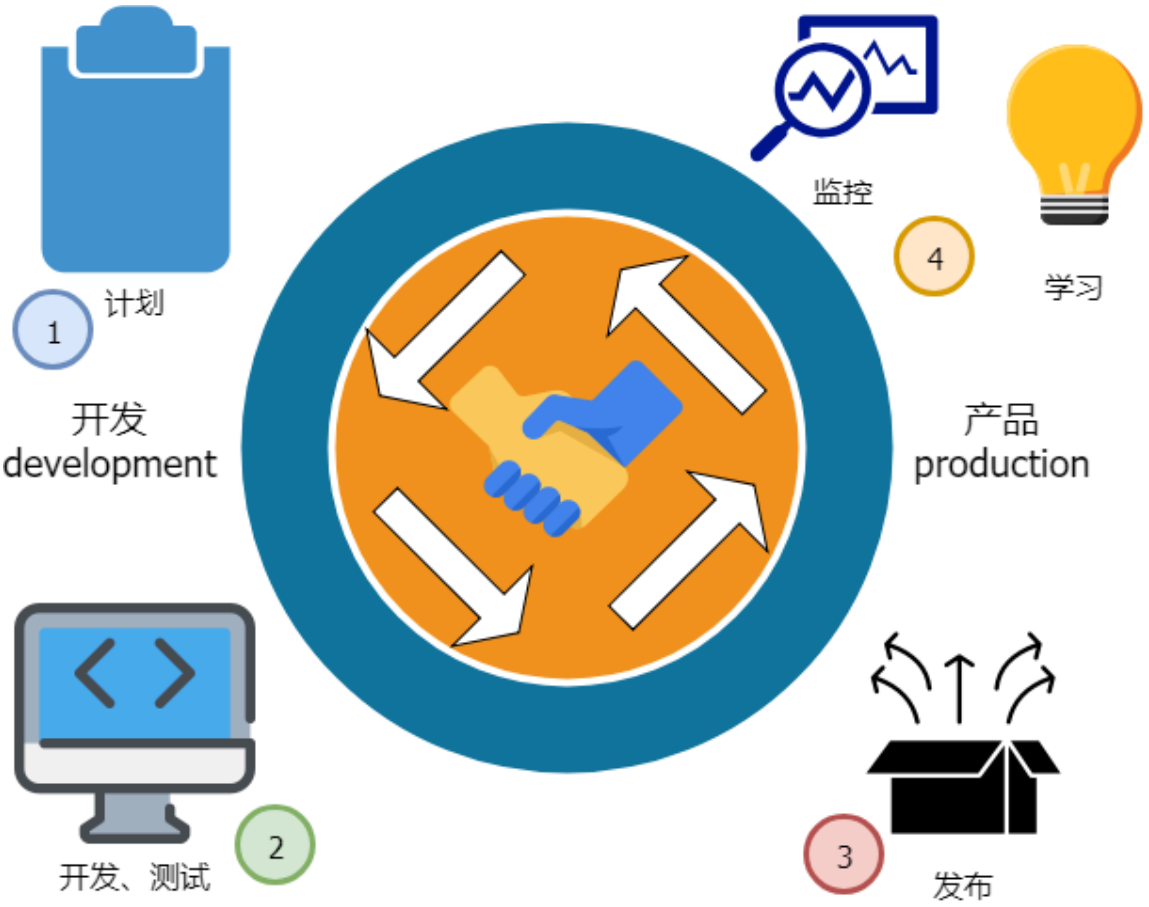


缓慢而繁琐的瀑布模型演变成敏捷，开发团队在短时间内完成软件开发，持续时间甚至不超过两周。如此短的发布周期帮助开发团队处理客户反馈，并将其与bug修复一起合并到下一个版本中。

虽然这种敏捷的SCRUM方法为开发带来了敏捷性，但它在运维方面却失去了敏捷实践的速度。开发人员和运维工程师之间缺乏协作仍然会减慢开发过程和发布。

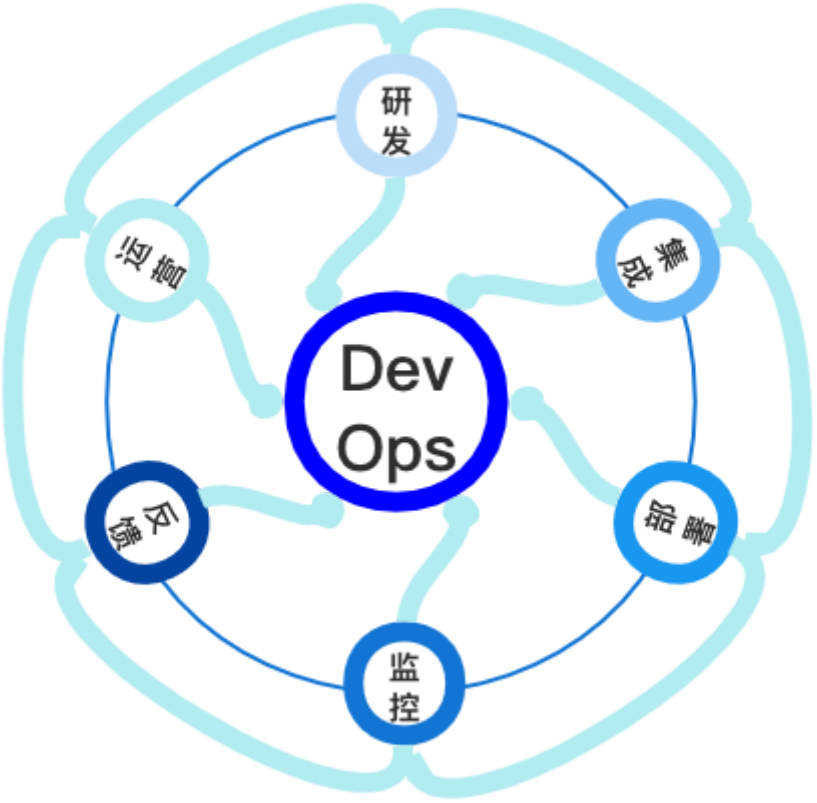
2.3. DevOps

DevOps方法就是基于对更好的协作和更快的交付的需求而产生的。DevOps允许用较少复杂问题的持续软件交付来修复和更快地解决问题。



> DevOps早在九年前就有人提出来，但是，为什么这两年才开始受到越来越多的企业重视和实践呢?
>
> 因为DevOps的发展是独木不成林的，现在有越来越多的技术支撑。微服务架构理念、容器技术使得DevOps的实施变得更加容易，计算能力提升和云环境的发展使得快速开发的产品可以立刻获得更广泛的使用 。

完整的 DevOps 生命周期一般包括以下六个阶段。



3. 好处是什么?

DevOps的一个巨大好处就是可以高效交付，这也正好是它的初衷。 Puppet和DevOps Research and Assessment (DORA) 主办了2016年DevOps调查报告，根据全球4600位各IT公司的技术工作者的提交数据统计，得出高效公司平均每年可以完成1460次部署。

与低效组织相比，高效组织的部署频繁200倍，产品投入使用速度快2555倍，服务恢复速度快24倍。在工作内容的时间分配上，低效者要多花22%的时间用在为规划好或者重复工作上，而高效者却可以多花29%的时间用在新的工作上。所以这里的高效不仅仅指公司产出的效率提高，还指员工的工作质量得到提升。

DevOps另外一个好处就是会改善公司组织文化、提高员工的参与感。 员工们变得更高效，也更有满足和成就感；调查显示高效员工的雇员净推荐值（eNPS:employee Net Promoter Score）更高，即对公司更加认同。

> ⚡ **快速部署同时提高IT稳定性。这难道不矛盾吗?**
>
> 快速的部署其实可以帮助更快地发现问题，产品被更快地交付到用户手中，团队可以更快地得到用户的反馈，从而进行更快地响应。而且，DevOps小步快跑的形式带来的变化是比较小的，出现问题的偏差每次都不会太大，修复起来也会相对容易一些。
>
> 🔥 因此，认为速度就意味着危险是一种偏见。此外，滞后软件服务的发布也并不一定会完全地避免问题，在竞争日益激烈的IT行业，这反而可能错失了软件的发布时机

4. 为什么DevOps会兴起?

条件成熟：技术配套发展

技术的发展使得DevOps有了更多的配合。早期时，大家虽然意识到了这个问题的，但是苦于当时没有完善丰富的技术工具，是一种“理想很丰满，但是现实很骨感”的情况。DevOps的实现可以基于新兴的容器技术；也可以在自动化运维工具Puppet、SaltStack、Ansible之后的延伸；还可以构建在传统的Cloud Foundry、OpenShift等PaaS厂商之上。

来自市场的外部需求：这世界变化太快

IT行业已经越来越与市场的经济发展紧密挂钩，专家们认为IT将会有支持中心变成利润驱动中心。事实上，这个变化已经开始了，这不仅体现在Google、苹果、Amazon这些大企业中，而且也发生在传统行业中，比如出租车业务中的Uber、酒店连锁行业中的Airbnb等等。能否让公司的IT配套方案及时跟上市场需求的步伐，在今天显得至关重要。

DevOps 2016年度报告给出了一个运维成本的计算公式：

```
停机费用成本 = 部署频率 * 版本迭代失败概率 * 平均修复时间 * 断电的金钱损失
```

来自团队的内在动力：工程师也需要

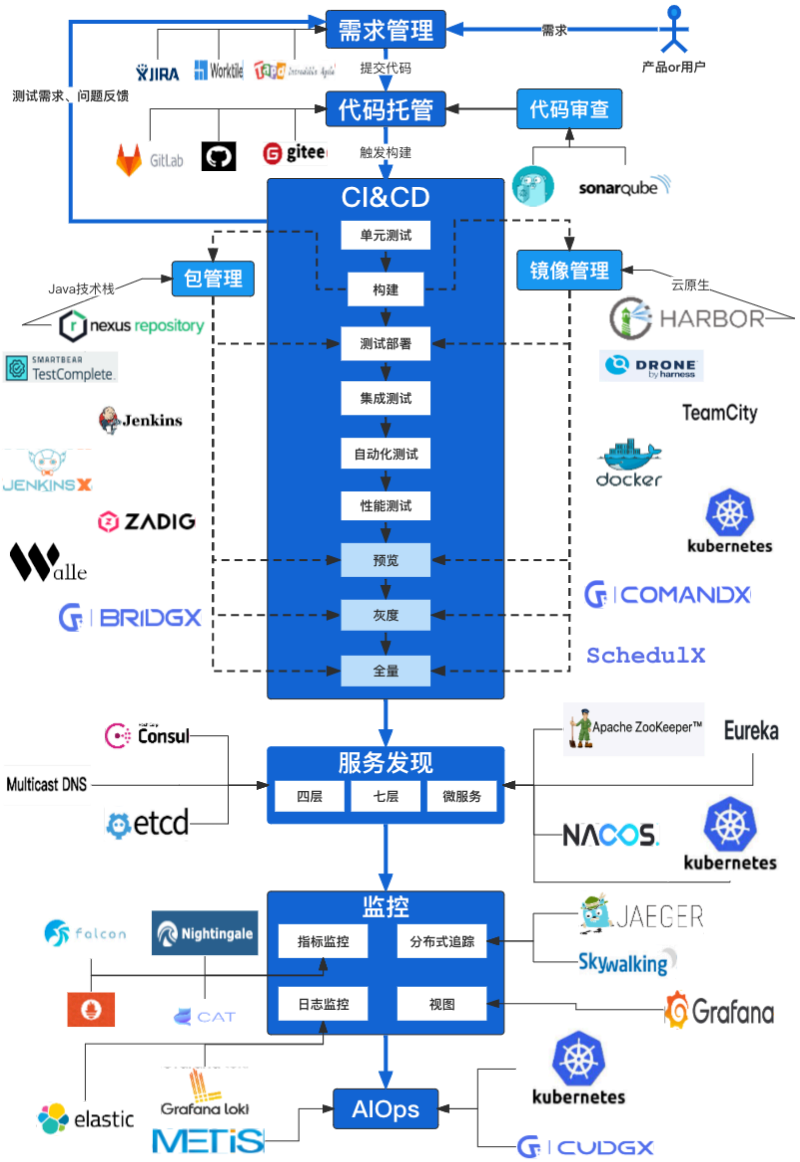
对于工程师而言，他们也是DevOps的受益者。微软资深工程师Scott Hanselman说过“对于开发者而言，最有力的工具就是自动化工具”（The most powerful tool we have as developers is automation）。

工具链的打通使得开发者们在交付软件时可以完成生产环境的构建、测试和运行；正如Amazon的VP兼CTO Werner Vogels那句让人印象深刻的话：“谁开发谁运行”。（You build it, you run it）

4.1. 实现DevOps需要什么?

4.1.1. 硬性要求：工具上的准备

上文提到了工具链的打通，那么工具自然就需要做好准备。现将工具类型及对应的不完全列举整理如下：



- 代码管理（SCM）：**GitHub**、GitLab、BitBucket、SubVersion
- 构建工具：**Ant**、Gradle、**maven**
- 自动部署：Capistrano、CodeDeploy
- 持续集成（CI）：Bamboo、Hudson、**Jenkins**
- 配置管理：Ansible、Chef、Puppet、SaltStack、ScriptRock GuardRail
- 容器：**Docker**、LXC、第三方厂商如AWS
- 编排：**Kubernetes**、Core、Apache Mesos、DC/OS
- 服务注册与发现：**Zookeeper**、etcd、Consul
- 脚本语言：python、ruby、shell
- 日志管理：ELK、Logentries
- 系统监控：Datadog、Graphite、Icinga、Nagios
- 性能监控：AppDynamics、New Relic、Splunk
- 压力测试：JMeter、Blaze Meter、loader.io
- 预警：PagerDuty、pingdom、厂商自带如AWS SNS
- HTTP加速器：Varnish
- 消息总线：ActiveMQ、SQS

- 应用服务器：Tomcat、JBoss
- Web服务器：Apache、Nginx、IIS
- 数据库：MySQL、Oracle、PostgreSQL等关系型数据库；cassandra、mongoDB、redis等NoSQL数据库
- 项目管理（PM）：Jira、Asana、Taiga、Trello、Basecamp、Pivotal Tracker

在工具的选择上，需要结合公司业务需求和技术团队情况而定。（注：更多关于工具的详细介绍可以参见此文：51 Best DevOps Tools for #DevOps Engineers）

4.1.2. 软性需求：文化和人

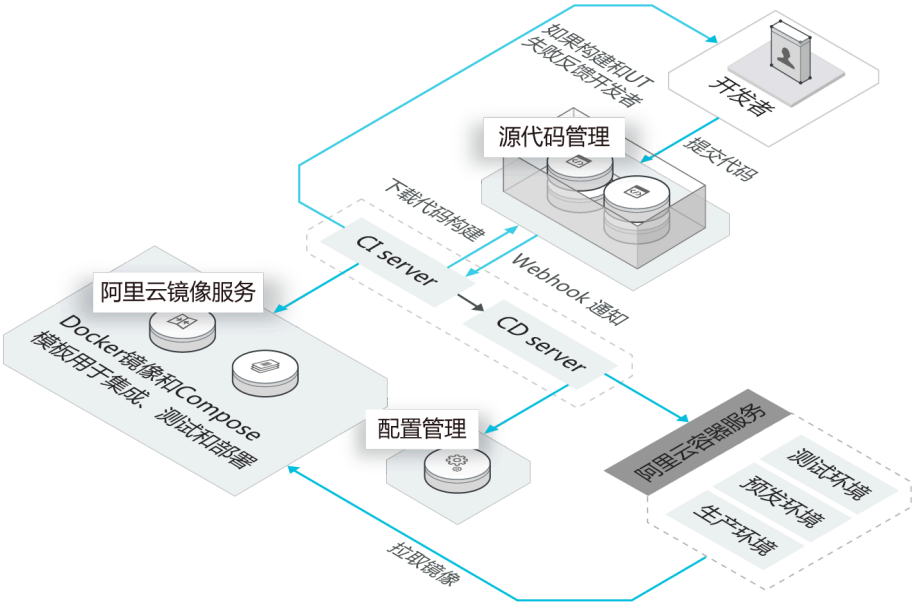
DevOps成功与否，公司组织是否利于协作是关键。开发人员和运维人员可以良好沟通互相学习，从而拥有高生产力。并且协作也存在于业务人员与开发人员之间。

出席了2016年伦敦企业级DevOps峰会的ITV公司在2012年就开始落地DevOps，其通用平台主管Clark在接受了InfoQ的采访，在谈及成功时表示，业务人员非常清楚他们希望在最小化可行产品中实现什么，工程师们就按需交付，不做多余工作。

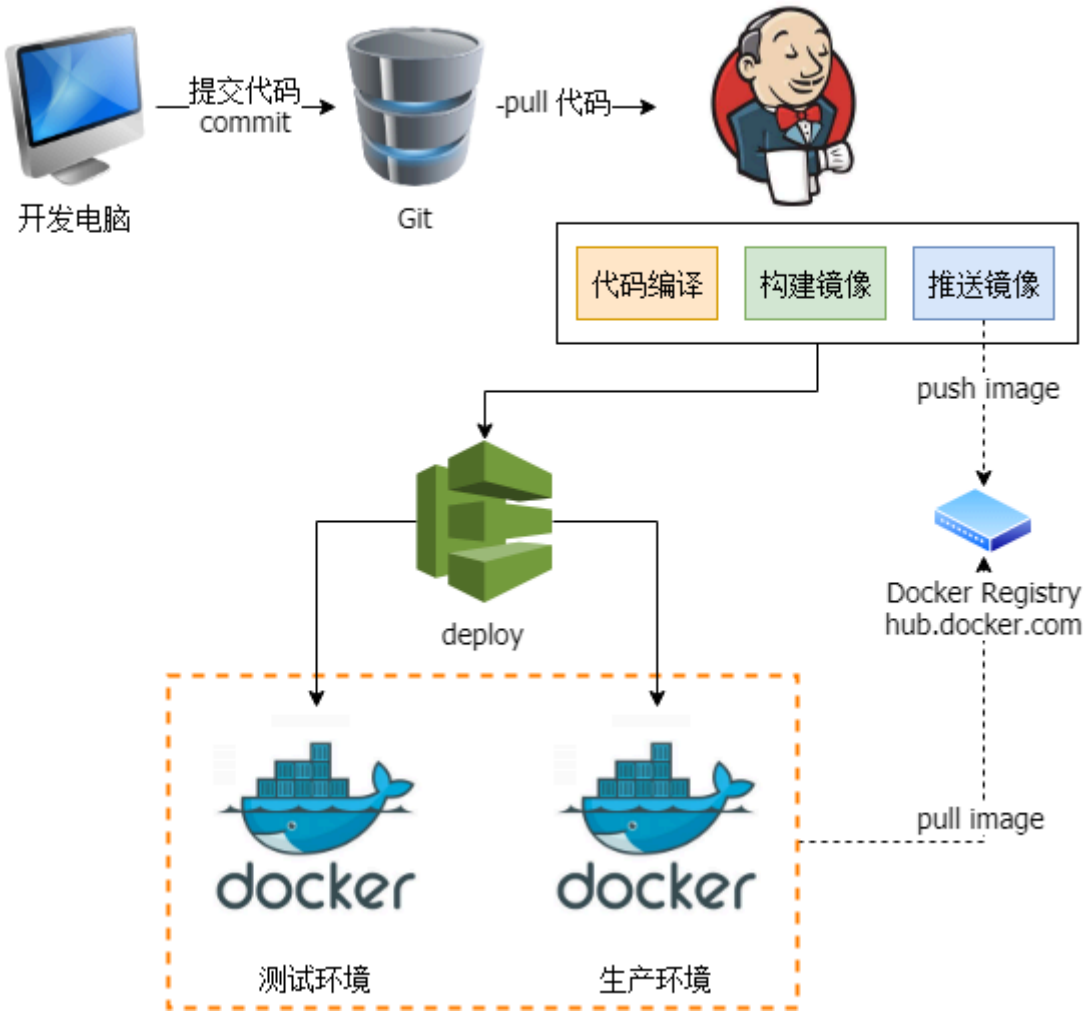
这样，工程师们使用通用的平台（即打通的工具链）得到更好的一致性和更高的质量。此外，DevOps对工程师个人的要求也提高了，很多专家也认为招募到优秀的人才也是一个挑战。

5. DevOps的实践

阿里云的DevOps解决方案：



持续交付是一种敏捷交付的方式，加速软件的开发、测试与交付。从整个产品生命周期的角度，通过自动化的方式减少从前由于流程或者人为因素干预而造成的开发周期冗长，人员效率低下，软件质量无法保障等问题。



5.1. GIT

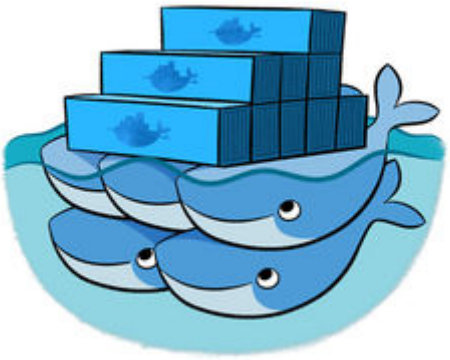
Git(读音为/gɪt/。)是一个开源的分布式版本控制系统，可以有效、高速地处理从很小到非常大的项目版本管理。[1] Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。

5.2. Jenkins

Jenkins是开源CI&CD软件领导者， 提供超过1000个插件来支持构建、部署、自动化， 满足任何项目的需要。

5.3. Docker

Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的镜像中，然后发布到任何流行的 Linux或Windows 机器上，也可以实现虚拟化。容器是完全使用沙箱机制，相互之间不会有任何接口。



6. 相关内容

- **敏捷开发实践**