

## 02 | 影响持续交付的因素有哪些？

2018-07-07 王潇俊

持续交付36讲

[进入课程 >](#)



讲述：王潇俊

时长 15:53 大小 7.28M



在上一篇文章中，我和你聊了聊“持续交付”的价值。现在，你是不是感觉热血沸腾，似乎找到了解决一些问题的良方？你是不是跃跃欲试，想在团队立刻实施看看效果如何？

但别急，就像我在开篇词里说的一样，“持续交付”可真不是一件简单的事情。你一定会在实施过程中碰到各种各样的问题和困难，但也不要气馁，我现在就和你说说：影响持续交付的各种因素。知己知彼，方可百战不殆。

与绝大多数理论分析一样，影响持续交付的因素也可归结为：人（组织和文化），事（流程），物（架构）。

### 组织和文化因素

谈到组织，你是不是一下就想到了部门划分，跨部门合作等？的确，这就是我要和你讲的第一个影响因素。因为“持续交付”一定是整个组织层面的事情，是跨部门合作的产物，所以组织和文化因素，是要首先考虑的问题。

什么样的组织文化，才是“持续交付”成长的沃土（当然这也是定义好的组织的标准），我把它分成了三个层次：

**第一个层次：紧密配合，这是组织发展，部门合作的基础。**

一般企业都会按照职能划分部门。不同的职能产生不同的角色；不同的角色拥有不同的资源；不同的资源又产生不同的工作方式。这些不同的部门紧密配合，协同工作于共同的目标，就能达到成效。

**第二个层次：集思广益，这就需要组织内各个不同部门，或不同职能的角色，跳出自身的“舒适区”。**

除思考 and 解决本身职能的问题外，各部门还要为达到组织的共同目标，通盘考虑和解决所遇到问题和困难。这个层次需要增加组织的透明度，需要接受互相批评和帮助。

**第三个层次：自我驱动，是理想中的完美组织形式。**

如果第二个层次能够持续地运转，就会形成自我学习、自我驱动의飞轮效应，并且越转越快，它甚至能自发式的预见困难，并自驱动解决问题。

这三个层次看起是不是有点眼熟，和我在上一篇文章中讲到的持续集成的三个层次：

1. 分模块编码；
2. 整体集成；
3. 实现以上两个过程的自动化，并形成闭环；

好像是一样的。真是有趣，持续交付其实也是帮企业建立更好的组织形式的一种方法。

那么，在形成理想组织的实际执行中会遇到哪些问题呢？

**一般软件企业与交付有关的研发部门包括四个：产品、开发、测试和运维。而这四个部门天然地形成了一个生产流水线，所以形成理想组织的第一层次紧密配合，基本没什么问题。**

但是，要达到第二层次集思广益的难度，往往就很大。因为，每个部门有自身的利益，以及自己的工作方式和目标。

比如，产品人员和测试人员就是一对矛盾体：产品人员希望产品尽快上线，而测试人员则希望多留时间进行更完整的测试。

又比如，开发人员和运维人员也经常矛盾：开发人员希望能有完全权限，而运维人员却控制着生产的 root。

从各自的小目标的角度看，这些矛盾是正常的。但是，产品、开发、测试和运维这些部门的小目标往往就是实施持续交付的阻碍，只有它们把眼光放到更高地持续交付可用的产品上，有了共同的目标，问题才会迎刃而解。

那么，靠各个部门自己能解决这个问题吗，其实很难。组织的问题，还是需要通过组织变革来解决。通常会采用以下三种方案：

成立项目管理办公室（Project Manage Office，简称 PMO）这样的监督型组织，帮助持续交付落地；

独立建立工程效能部门，全面负责包括持续交付在内的研发效率提升工作；

使用敏捷形式，如 Scrum，打破职能部门间的“隔离墙”，以产品的形式组织团队，各团队自行推进持续交付。

当然，这三种方案各有利弊。比如：

成立项目管理办公室，虽然会带来非常强大的项目推进力，但它往往需要通过流程把控进行监督，这样就很有可能把流程变得更加复杂；

而独立的工程效能部门，虽然能最大化地去做持续交付工作，但其研发成本的投入也是需要考虑的，小团队的话，就不太适用了；

敏捷形式是比较适合中小团队的一种组织变革方式，但对个人能力的要求也会比较高，而且往往需要一个很长时间的磨合才能见效。

所以，你需要根据当前组织的情况来选择。**总而言之，持续交付必须有与其相适应的组织和文化，否则将很难实施。**

## 流程因素

要说持续交付对企业和组织改变最多的是什么，那么一定是流程。

持续交付一定会打破的这三类流程是：

1. **耗时较长的流程。**比如，一个功能的研发迭代周期为 5 天，而其中有一个上线审核流程，需要花费 3 天时间，那这个流程就严重影响了持续交付，必须被打破。
2. **完全人工类的流程。**完全人工操作的流程，一般效率低下，且质量难以保证，持续交付的逐步深入会通过自动化替代这些人工流程的存在。
3. **信息报备类的流程。**持续交付过程中同样会产生各种信息流，这些信息有些需要广播，有些需要定点传递。实施持续交付后，这些信息报备类的流程一定会通过异步消息等方式进行改造。

其中，如何对待审批流程是重点。

在持续交付过程中，其实最让你头痛的应该是一些审批流程。这些流程既然叫做审批，那就代表着授权与责任，代表着严谨与严肃，因此也一定有其存在的价值和意义，不能轻易被去除或打破。

但是，你我都知道，审批往往指的是由人进行审核和批准，既是一个全人工流程，又是一个信息流转类流程。那么如何打破它呢？同样，也有几种思路：

1. 该审批流程是否确实需要，如果能够通过系统来保证，则可以去除；
2. 该审批流程是否可以从事前审批转化为事后审核；
3. 该审批流程是否可以被简化。

但是，每家公司的流程都不太一样，所以我的这几个思路并不一定是放诸四海而皆准，但我希望你可以借鉴，或者从中学习到一些新的思路，并结合你自己的情况进行合理调整。

相对于组织文化和流程因素，架构是真正和技术相关的因素，也是我要和你重点分享的内容。

## 架构因素

技术架构对于持续交付来说，是万分重要的。如果遇到混乱的架构，那持续交付会处处受制，痛苦不堪。但与之前讨论的组织、文化和流程因素相比，架构的问题解决起来也会相对容易，因为凡是技术上的东西，都比较愿意接受优化，并且可以随着持续交付一起慢慢重构。

影响持续交付的架构因素，主要有两大部分：系统架构和部署架构，接下来我会给你详细展开。

### 第一，系统架构

系统架构指系统的组成结构，它决定了系统的运行模式，层次结构，调用关系等。我们通常会遇到的系统架构包括：

1. 单体架构，一个部署包，包含了应用所有功能；
2. SOA 架构，面向服务，通过服务间的接口和契约联系；
3. 微服务架构，按业务领域划分为独立的服务单元，可独立部署，松耦合。

那么，这些架构对持续交付又有什么影响和挑战呢？

#### 对单体架构来说：

1. 整个应用使用一个代码仓库，在系统简单的情况下，因为管理简单，可以快速简单地做到持续集成；但是一旦系统复杂起来，仓库就会越变越大，开发团队也会越来越大，多团队维护一个代码仓库简直就是噩梦，会产生大量的冲突；而且持续集成的编译时间也会随着仓库变大而变长，团队再也承受不起一次编译几十分钟，结果最终失败的痛苦。
2. 应用变复杂后，测试需要全回归，因为不管多么小的功能变更，都会引起整个应用的全新编译和打包。即使在有高覆盖率的自动化测试的帮助下，测试所要花费的时间成本仍旧巨大，且错误成本昂贵。
3. 在应用比较小的情况下，可以做到单机部署，简单直接，这有利于持续交付；但是一旦应用复杂起来，每次部署的代价也变得越来越高的，这和之前说的构建越来越慢是一个道理。而且部署代价高会直接影响生产稳定性。这显然不是持续交付想要的结果。

总而言之，一个你可以完全驾驭的单体架构应用，是最有容易做到持续交付的，但一旦它变得复杂起来，一切就都会失控。

## 对 SOA 架构来说：

1. 由于服务的拆分，使得应用的代码管理、构建、测试都变得更轻量，这有利于持续集成的实施。
2. 因为分布式的部署，使得测试环境的治理，测试部署变得非常复杂，这里就需要持续交付过程中考虑服务与服务间的依赖，环境的隔离等等。
3. 一些新技术和组件的引入，比如服务发现、配置中心、路由、网关等，使得持续交付过程中不得不去考虑这些中间件的适配。

总体来说，SOA 架构要做到持续交付比单体架构要难得多。但也正因架构解耦造成的分散化开发问题，持续集成、持续交付能够在这样的架构下发挥更大的威力。

## 对微服务架构来说：

其实，微服务架构是一种 SOA 架构的演化，它给持续交付带来的影响和挑战也基本与 SOA 架构一致。

当然，如果你采用容器技术来承载你的微服务架构，就另当别论了，这完全是一个持续交付全新的领域，这部分内容我将在后续文章中跟你分享。

## 第二，部署架构

**部署架构指的是，系统在各种环境下的部署方法，验收标准，编排次序等的集合。它将直接影响你持续交付的“最后一公里”。**

**首先，你需要考虑，是否有统一的部署标准和方式。** 在各个环境，不同的设备上，应用的部署方式和标准应该都是一样的，可复用的；除了单个应用以外，最好能做到组织内所有应用的部署方式都是一样的。否则可以想象，每个应用在每个环境上都有不同的部署方式，都要进行持续交付的适配，成本是巨大的。

**其次，需要考虑发布的编排次序。** 特别是在大集群、多机房的情况下。我们通常会采用金丝雀发布（之后讲到灰度发布时，我会详解这部分内容），或者滚动发布等灰度发布策略。那么就需要持续交付系统或平台能够支持这样的功能了。

**再次，是 markdown 与 markup 机制。** 为了应用在部署时做到业务无损，我们需要有完善的服务拉入拉出机制来保证。否则每次持续交付都伴随着异常产生，肯定不是大家愿意见到的。

**最后，是预热与自检。** 持续交付的目的是交付有效的软件。而有些软件在启动后需要处理加载缓存等预热过程，这些也是持续交付所要考虑的关键点，并不能粗暴启动后就认为交付完成了。同理，如何为应用建立统一的自检体系，也就自然成为持续交付的一项内容了。

关于部署的问题，我也会在之后的篇章中和你详细的讨论。

## 总结

今天，我和你分享的主题是影响持续交付的因素，为了便于你理解，我将其划分为人（组织和文化），事（流程），物（架构）三个方面：

1. 组织和文化，是最重要的因素，是持续交付推进的基础；
2. 流程因素，实施持续交付也是一次流程改造之旅；
3. 系统架构，与持续交付相互影响，但技术可以解决一切问题；部署架构，千万不要失败在“最后一公里”，这部分你也需要重点关注。

最后，也请你思考一下，如果你的组织实施持续交付，最大的影响因素会是什么？如果是系统架构方面的因素，你将如何应对？

欢迎你给我留言。

---



# 持续交付36讲

量身定制你的持续交付体系

王潇俊 携程系统研发部总监



新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 01 | 持续交付到底有什么价值？

下一篇 03 | 持续交付和DevOps是一对好基友

## 精选留言 (8)

写留言



旭东

2018-07-14

7

大公司不关注技术，关注办公室政治，导致持续集成徒有虚名。一个CTO和中高层无此意识很难推动。

作者回复：相信现在绝大多数的领导能够看到持续交付的价值，否则就会像我老板说的那样，不能从上进行改革，就只能等着自下的革命了：)



JinSong

2018-07-08

4



人和事是最难搞定的

展开 ∨

作者回复: 他山之石 可以攻玉, 哈哈



宝爷

2018-07-14

👍 2

我觉得是自动化测试这个环节, 对于小团队而言, 自动化测试的成本太高

展开 ∨

作者回复: 测试工作其实很讲究分层思路, 也最讲究ROI, 自动化可以覆盖不同的层面, 代价也都不同, 并不是一定要覆盖最高层UI才算的, 而且也没要求过100%的自动化覆盖率。一个QA的好坏本质上看的就是工作投入和质量产出比



翔翔

2018-07-07

👍 2

我觉得系统架构的选择是不是应该首先服务于业务需求和应用实际, 然后才是看采用什么样的持续交付方式去适应这种系统架构, 而不能是为了更方便的去持续交付, 去选择一种系统架构吧?

展开 ∨

作者回复: 说得很对, 就是用户问题, 架构服务于业务, 持续交付服务于研发



frankie

2018-11-12

👍 1

老师能解释一下Markdown和markup吗?

展开 ∨

作者回复: 是否接入流量



大M

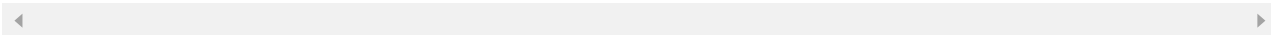
2018-07-10

👍 1

后台的代码都微服务化了，前端工程有什么拆分最佳实践吗？

展开 ▾

作者回复: 前端还是比较传统的部署方式，做好依赖管理，包管理比较重要



胖宝王

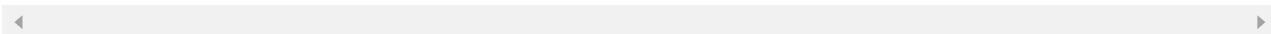
2018-07-07

👍 1

我觉得应该是混乱的部署环境，有的环境是编译，有的是包管理，开发人员本地测试也不可控。经常出现本地ok，到其他环境出现异常。先上 docker，统一所有环境！

展开 ▾

作者回复: 我后面会有一讲专门讲docker对环境的问题，这里卖个关子先，但是docker不是银弹，而且如果能力未到，它还会带来更多麻烦



志海...

2018-09-21

👍

现阶段想做，测试环境的持续交付，小步快跑快速迭代

展开 ▾