30 | 持续交付中有哪些宝贵数据?

2018-09-11 王潇俊

持续交付36讲 进入课程 >

讲述：王潇俊
时长 13:12 大小 6.05M

你好，我是王潇俊。今天我和你分享的主题是：持续交付中有哪些宝贵数据。

通过我前面和你分享的内容，相信你已经掌握了持续交付流水线所包含的五个主要动作：代码管理、环境管理、集成和编译管理、发布管理，以及测试管理。而且，你也应该已经初步掌握了建设持续交付体系的基本方法。

那么，如何使这个初步建立的持续交付体系更上一层楼呢？现在我们都选择用数据说话，所以优化一套系统的最好办法，就是从数据角度进行分析，然后找出优化方向，再进行具体的改进。

所以，我今天就分享一下，我在携程建设持续交付系统时，遇到的几个与数据密切相关的案例。通过对这些数据的分析，我们可以明确优化系统本身处理能力的方向，也可以快速发现

日常工作中与持续交付相违背的行为，从而再次展现我们搭建的持续交付系统的作用。

案例 1：要用好的数据来衡量系统

让我记忆犹新的第一个案例，是我们持续交付平台刚上线时，就遇到了一个很大的问题。这个问题就是，这套系统的稳定性怎么样？

这个问题不仅老板会问你，用户会问你，其实你自己都会问自己。如果没有相关的数字指标，那我怎么证明这套系统的稳定性好呢？如果我无法证明这套系统的稳定性，又怎么说服整个公司，把它当做研发的核心流水线呢？

期间，我想过很多方案，比如用宕机时间来计算稳定性。但是实际使用起来，你就会发现，这个衡量指标很不靠谱。

第一，对于平台系统来说，有很多相关联的子系统，有些子系统属于旁支系统，对实际业务影响不大，而有些则影响非常大。那么，我怎么合理计算这些系统间的权重呢？

第二，毕竟是一套对内的系统，有的时候即使宕机了，特别是在夜间，因为没有用户使用，其实际影响几乎为 0；而反之有的时候，比如处在发布高峰的下午，系统宕机则会产生较大的影响。所以，用宕机时间这个指标衡量的话，就会把这些影响摊平，不能正确地反映真实的问题。

第三，宕机时间这个单一指标，不能全面地评价系统的稳定性。比如，有些子系统的运维会在宕机时进行降级（比如，增加排队时间等手段等等），使系统处于将死而不死状态。这种处理，虽然从业务的角度可以理解为降级，但却对系统的真实评价却起到了反作用。

其实上面的这三个问题，也会在真实的业务系统上碰到。所以，我们借鉴了携程业务系统的稳定性评估方案，最后决定采用如下的实施方案：

首先，我们通过监控、保障、人为记录等手段，统计所有的故障时间。需要统计的指标包括：开始时间、结束时间和故障时长。

然后，计算过去三个月内这个时间段产生的持续交付平均业务量。所谓业务量，就是这个时间段内，处理的代码提交、code review；进行的编译、代码扫描、打包；测试部署；环境处理；测试执行和生产发布的数量。

最后，计算这个时间段内的业务量与月平均量相比的损失率。这个损失率，就代表了系统的不稳定性率，反之就是稳定性率了。

这样计算得到的不稳定性率指标，就要比简单粗暴地用宕机时间要精确得多，也不再会遇到前面提到的三种问题。

由此得到的计算模型一旦固定下来，你只要做好业务量的数据统计，其计算难度也会不大。

因此，我推荐你也可以使用这个数据指标来衡量自己系统的稳定性。

案例 2：数据既要抓大势，也要看细节

第二个让我映像深刻的案例，是一个与长尾数据有关的案例。

自从我们的持续交付平台在携程上线以后，一直颇受好评。当然，在系统上线后，我们也进行过几次优化，编译和打包速度被提升了非常多。而这些优化的方法，我也已经在专栏前面的文中进行了分享，如果有哪些不太清楚的地方，你可以去回顾一下这些文章，也可以给我留言我们一起讨论我在搭建系统时遇到的具体问题。

而且，我们运维团队也一直谨记，要通过数据分析不断优化系统。所以，我们一直非常关注总体的集成编译速度，因此除了追踪平均速度外，还会定期进行全量个例分析。

从整体的平均速度这个指标来看，系统一直表现良好。但，在 99 线以上，我们却发现存在一些长尾特例。正常的 Java 代码平均编译时间大概在 1 分钟之内，而这些特例却在 7 分钟以上。在几次的全量个例分析中，我们虽然发现了这个问题，但并没有特别关注。而且，在查看了编译过程的几个主要计时点后，我们也确实没有发现任何问题。

所以，这时我们都认为，这些长尾数据可能真的只是一些特殊个案。毕竟相对于每天几万次的集成编译来说，这个数量实在太小。

但就是这个小小的数据疏忽，我们差点忽略一个非常重要的故障点。随着时间的推移，我们发现这个长尾在慢慢变大。最终，我们还是尝试去一探究竟，发现原因其实是：

持续交付系统在打包之后，会通过网络专线向另外一个 IDC 分发部署包和容器镜像。但由于历史原因，两个 IDC 之间的防火墙对流量进行了不恰当的限制。随着持续交付在全公司的开展，这个分发量也越来越大，使网络流量达到了瓶颈，从而形成了之前的长尾现象。

当然，这个问题的解决方案十分简单。但是，从中我们看到：**大的故障和影响，往往都是出于一些非常愚蠢的失误。**

所以，这个案例也一直在提醒着我，看数据不仅要抓大势，也要关注细节，特别是异常细节。

案例 3：数据可以推动持续交付

第三个案例是，关于怎么利用数据来改善业务开发团队的持续交付过程。

任何一个团队，都会有它自己的研发习惯、迭代速度以及交付频率。自从携程上线了持续交付平台之后，我们从数据上就能很明显地发现，每个团队乃至整个公司，每周的发布数据是一个固定的趋势。

从周跨度上看，周三、周四为发布高峰；从日的维度看，中午 12 点开始发布数量逐步增加，下午 4 点达到发布数据的高峰，晚上 7 点之后发布数据逐步回落。

几乎每个团队的发布数据趋势都是这样，只有少数几个团队呈现了不同的趋势：它们的周趋势与其他团队基本一致，但日趋势则非常不同，高峰出现在下午 5 点，然后立即回落，且高峰值远远高于其他团队的平均值。

这是怎么回事呢？很明显，这是一种集中式发布的形态。但是，我在携程也没听说过有这样的流程。之后，我们经过了解，发现这种集中发布的情况是：这几个团队一直在沿用以前的发布模式，即将所有发布会汇总到一个发布负责人处，由他专门负责发布。所以，为了方便工作，这些发布负责人员选择在下午 5 点集中开始发布。

虽然这种做法和流程没什么问题，但却有违于我们推崇的“谁开发，谁运行”理念，并且也因此增加了一个实际不是必须的工作角色。在这之后，我们改造了这几个团队的流程，相当于是推动了整个公司的持续交付。

这个案例第一次让我们认识到，**我们可以用手上的数据去推动、去优化持续交付体系。**

这三个案例，都充分说明了数据对持续交付、持续交付平台的重要性，所以我们也要善用这些宝贵的数据。接下来，我再和你分享一下，持续交付体系中还有哪些数据值得我们关注。

常规系统指标数据列举

在日常工作中，我把需要关注的系统指标数据做了分类处理。这样，我可以通过这些数据指标去了解每一个持续交付子系统的当前状况，并确定需要优化的指标。

第一类指标，稳定性相关指标

作为基础服务，稳定性是我们的生命线。所以，对于所有的子系统，包括：代码管理平台、集成编译系统、环境管理系统、测试管理系统和发布系统，我们都会设立必要的稳定性指标，并进行数据监控。这些稳定性相关的数据指标，代表整个系统的可用度。

各系统的稳定性计算则可以参考第一个例子中的算法。

第二类指标，性能相关指标

与系统性能相关的指标，通常可以直接反应系统的处理能力，以及计算资源的使用情况。更重要的是，速度是我们对用户服务能力的直观体现。很多时候，系统的处理速度上去了，一些问题也就不再是问题了。比如，如果回滚速度这个指标非常优秀，那么业务发布时就会更有信心。

与性能相关的指标，我比较关注的有：

push 和 fetch 代码的速度；

环境创建和销毁的速度；

产生仿真数据的速度；

平均编译速度及排队时长；

静态检查的速度；

自动化测试的耗时；

发布和回滚的速度。

第三类指标，持续交付能力成熟度指标

与持续交付能力成熟度相关的指标，可以帮助我们度量组织在持续交付能力上的缺陷，并加以改善。

不同的子系统，我关注的指标也不同。

与代码管理子系统相关的指标包括：commit 的数量，code review 的拒绝率，并行开发的分支数量。
这里需要注意的是，并行开发的分支数量并不是越多越好，而是要以每个团队都保持一个稳定状态为优。

与环境管理子系统相关的指标包括：计算资源的使用率，环境的平均大小。
这里需要注意的是，我一直都很关注环境的平均大小这个数据。因为我们鼓励团队使用技术手段来避免产生巨型测试环境，从而达到提高利用率、降低成本的目的。而且，这个指标也可以从侧面反映一个团队利用技术解决问题的能力。

与集成编译子系统相关的指标包括：每日编译数量，编译检查的数据。
我们并不会强制要求编译检查出的不良数据要下降，因为它会受各类外部因素的影响，比如历史代码问题等等。但，我们必须保证它不会增长。这也是我们的团队在坚守质量关的体现。

与测试管理子系统相关的指标包括：单元测试的覆盖率，自动化测试的覆盖率。
这两个覆盖率代表了组织通过技术手段保证质量的能力，也是测试团队最常采用的数据指标。

与发布管理子系统相关的指标包括：周发布数量，回滚比率。

发布数量的增加，可以最直观地表现交付能力的提升；回滚比率，则代表了发布的质量。综合使用周发布数量和回滚比例这两个指标，就可以衡量整个团队的研发能力是否得到了提升。

以上这些数据指标，就是我们在携程要关注的了。希望通过我对这些数据指标的介绍，可以帮助你了解如何衡量自己的持续交付体系，并通过分析这些数据找到优化当前体系的方向。

总结

今天我通过三个实际工作中的例子，和你分享了应该如何利用持续交付中产生的数据。

首先，你可以利用与业务量相关的数据模型来衡量持续交付系统的稳定性；

然后，在日常的数据分析中，除了要抓住主要数据的大趋势外，你还要关注那些异常的个性数据，它能帮你及早地发现问题；

最后，通过日常的数据分析，你也能发现持续交付流程上的一些问题，并协助团队一起改进。

当然，这只是三个比较突出的例子而已。在实践中，实施持续交付的过程中还有很多数据需要我们关注。我也一并把这些数据分成了三类，包括：

1. 稳定性相关指标；
2. 性能相关指标；
3. 持续交付能力成熟度指标。

希望这些案例，以及这些数据指标可以对你日常的分析工作有所帮助。

思考题

你有没有在推进持续交付过程中遇到一些阻力呢？你有没有尝试通过数据分析去解释和解决这些问题呢？你又有哪些案例，想要和我分享呢？

感谢你的收听，欢迎你给我留言。

© 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　29 | 计算资源也是交付的内容

下一篇　31 | 了解移动App的持续交付生命周期

精选留言 (1)

写留言

九脉一谷
2018-09-11

产品质量问题是现在最头疼的问题，开发到现在一直都没有一个稳定版本。在devops推进过程中不知道用哪些指标来考核产品的质量。

> 作者回复: 分级的bug数量统计即可，bug/commit数量也可以。
>
> 你说的问题看起来还是功能切分没做好，从分支开始就要考虑功能开发的粒度和解耦。一般一个功能分支，2周都合不进主干的，要么就是太复杂了，要么就是初期就买划分好。要重新规划。切分分支要考虑很多，难度、资源、外部依赖等等