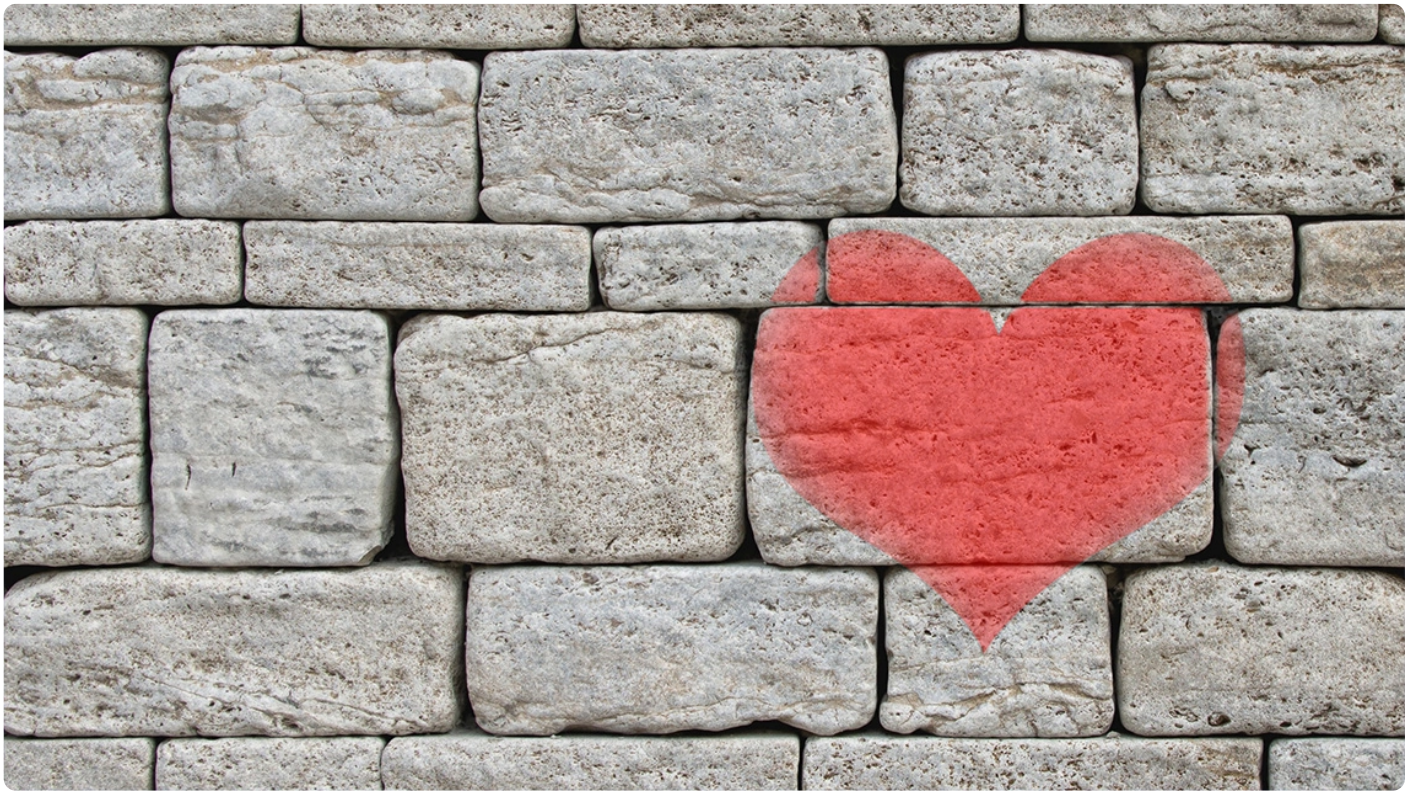
35 | 版本发布：软件上线只是新的开始

2019-05-21 宝玉 来自北京

《软件工程之美》



你好，我是宝玉。上一章我们学习了软件测试篇，今天，我们将从版本发布这个话题开始，进入到运行维护篇的学习。

说到版本发布，对于很多开发人员来说，觉得是一件很简单的事情，就是将程序编译打包部署，但实际发布的时候，却经常出现发布错版本的问题，或者是发布前修改了一点代码导致上线出现 Bug 的情况发生。

而版本发布对于很多项目管理者来说，又是一个很纠结的事情，觉得还有很多功能没完成，很多 Bug 还没改完，害怕用户负面评价，结果时间一拖再拖，迟迟无法上线。

所以今天我将带你一起学习一下如何做好版本发布，保障好发布产品的质量。

关于软件版本

在讨论这个话题之前，我们需要了解“版本”的含义。也许你已经知道版本的含义，但是这里还是有必要明确一下，因为在不同的语境下，版本的含义也有所不同。比如产品经理会对开发人员说：“这个功能我们会放到下个版本中实现”，开发人员会对测试人员说：“这个 Bug 在昨天的版本中已经修复了”。

这里产品经理说的“版本”是指特定功能集合，开发人员说的“版本”是指某一次程序的构建结果。也就是说对软件版本来说，包含两部分含义，一部分代表特定功能集合，一部分代表某一次特定的代码构建结果。

为了明确标识软件版本，需要对版本进行编号。目前业界在软件版本的命名上，通常会采用以下方式：

> 主版本号 . 子版本号.[. 修正版本号.[构建版本号]]

比如说：1.2.1、2.0、3.0.1 build-123。

其中主版本号和子版本号用来标识功能变化，小的功能变化增加子版本号，大的功能变化增加主版本号。修正版本号则表示功能不变化的情况下修复 Bug，而构建版本号表示一次新的构建，这个通常由编译程序自动生成。

团队中对版本有了清晰的定义和良好的版本编号，在讨论版本时，你就可以根据版本号清楚地知道应该有哪些功能，属于哪一次的构建结果。在修复 Bug 或者增加功能时，开发人员也能清楚地知道代码应该加入哪个版本。在验证 Bug 时，测试人员就可以知道应该在哪个版本中验证 Bug 有没有被修复。

版本发布前，做好版本发布的规划

回到前面的问题，为什么有的项目管理者会在发布前感觉没准备好，害怕上线发布呢？根源上，他们还是对于功能和质量没有信心，担心发布后获得负面的评价。

而实际上，并不代表你需要完成所有的功能，或者没有任何 Bug，有一个完美的版本才能上线。毕竟追求完美是没有止境的，这世界上也不存在完美的软件，很多著名的软件，比如 Windows、Office、iOS 都免不了在发布后还要打补丁。

这里的关键在于，**要在用户（或客户）的心理预期和你软件的实际情况之间，达到一种平衡，让软件的功能和质量，满足好用户的预期。**

要合理管理好用户的预期，达到好的发布效果，就需要在版本发布前先做好版本发布的规划。

那么，版本的发布规划，是指规划哪些内容呢？

首先是规划好要发布的功能。在发布前，搞清楚哪些是用户必须要有的功能，哪些是用户可以没有的功能。对于必须要有的功能，那么要保证软件中有这个功能才能发布，对于不是必需的功能，可以以后再逐步完善。

有一个经典的案例就是第一代的 iPhone，复制粘贴的功能都没有，但是在用户界面和触屏设计上做的非常好，一样取得了极大的成功。

然后是定义好发布的质量标准。在发布前，搞清楚你的用户对质量的容忍度如何，对哪些功能的质量要求高，对哪些功能的质量要求没那么高。对于那些用户在意的功能，要具有较高的发布标准，反之，则可以有较低的质量标准。

举例来说，像微博这种社交类应用软件，界面布局存在一点问题，用户是可以接受的，但是如果撰写内容的时候，因程序异常导致辛辛苦苦写的内容丢失，用户是比较难以忍受的。

再有就是要设计好发布的策略。考虑好是直接发布给所有用户？还是先让一部分用户试用？比如说可以先让内部用户使用，内部用户对软件质量问题容忍度是很高的，还可以帮助发现很多问题。

让一部分用户使用 Beta 版也是一个好的发布策略，当用户知道你的软件还是 Beta 版的时候，要求会比较低一点，可以接受一些不那么严重的 Bug。 还有就是采用灰度测试的发布策略，让一小部分用户先用新功能，如果没发现什么问题，再继续扩大使用的用户规模，如果有问题，也只是影响少量用户。

像 Gmail 在最初的几年，一直是 Beta 版本测试。还有像苹果的 iOS，用户也可以选择安装最新的 Beta 版本，可以先体验新功能，但是必须忍受系统的不稳定。

最后，就是有一个综合性的版本发布计划。在确定了要发布的功能、定义好了质量标准、设计好了发布策略，就可以制定一个综合性的版本发布计划了，确定好发布的时间点。

这个发布计划，不只是项目内部成员，还需要和项目之外利益相关方，比如客户、市场运营人员，大家一起确定最终的发布计划。

在对版本的发布做好规划后，就不用再纠结于该不该发布，该什么时候发布的问题。

有功能没完成没关系，关键要看这个功能是不是必须要有；有 Bug 没有修复完成，是不是影响发布，要看这些 Bug 是不是影响发布的质量标准；还可以采用一些像 Beta 版、小规模用户试用的发布策略，降低用户对功能和质量的预期。

规范好发布流程，保障发布质量

在规划好发布的版本后，要发布版本似乎是一件很简单的事，就是将源代码编码、部署。

但发布版本，可能并不是像你想的那么容易，这其中有几个需要注意的问题。

首先是必须保证要编译部署的是正确的版本。虽然一般来说，开发人员不会犯这样的错误，但是如果发布了错误的版本，后果可能很严重，所以要引起足够重视。比如说当年拼多多造成了几千万损失的薅羊毛事件，就是错误发布了一个测试用的无门槛券，导致被大量滥用。

然后要保证版本稳定可靠。如果你有开发经验的话，应该知道开发软件，一个常识就是每一次对代码的修改，都可能导致新的 Bug 产生。如果你的代码库在发布之前还一直在增加新的功能或者是不停地修复 Bug，那么质量是难以稳定下来的。

再就是要在发布失败后能回滚。没有谁能保证程序发布后没有严重问题，所以最保险的办法就是要在部署后，如果发现发布的版本出现严重问题，就应该对程序进行回滚操作，恢复到部署之前的状态。即使有些不可逆的升级，也需要事先做好应对措施，比如发布公告，停止服务，尽快修复。

针对这些问题，已经有些好的实践，比如说代码冻结、Bug 分级、回归测试等可以降低发布风险，保障发布产品的质量。在《12 | 流程和规范：红绿灯不是约束，而是用来提高效率》

这篇文章中，我也提到了：**流程和规范能将好的实践标准化流程化，让大家可以共享经验。**所以在版本发布上，我们也可以制定合理的流程，来应用这些好的实践，保证发布的质量。

那么一般大厂都是什么样的发布流程呢？下面这个流程可以作为一个参考。

在发布之前要做代码冻结。

什么是代码冻结呢？就是在发布之前，对于要发布的版本，在源代码管理工具中，专门创建一个 release 分支，然后对于这个分支的代码，冻结功能的修改，不接受新功能的增加，甚至重要性不高的 Bug 都不修改，只修复重要的 Bug。

由于严格的控制代码的修改，这样可以让版本的质量逐步趋于稳定。

对代码冻结后发现的 Bug 要分级

在代码冻结后，可能还存在一些 Bug，测试的过程中也会新增一些 Bug。代码冻结的原则就是尽可能减少代码的修改，避免引起不稳定。所以对于这些 Bug，要有一个简单的分级：是否在发布前修改，还是留在发布后再修改。

至于如何对一个 Bug 分级，这需要项目负责人和产品负责人一起确认。

每次修复 Bug 后，发布新的候选版本

进入代码冻结后，开发人员还需要对一些 Bug 进行修复，每一次修复完 Bug 后，就要生成一个新的候选发布版本，比如说 1.1 RC1、1.1 RC2。

关于生成发布版本，现在比较流行的做法是和持续集成系统整合，完全自动化。也就是在自动化测试通过之后，会自动构建，生成各个环境的发布版本。这样好处是，可以避免人为失误导致的错误，另外程序的配置管理做好了的话，只要测试环境的版本在测试环境测试没问题，那么就可以认为在生产环境的版本也是正常的。

自动化构建，生成发布版本并不复杂，各个语言都有成熟的方案，如果你还不了解的话，可以通过搜索引擎搜索关键字：“[对应平台] 自动打包”，例如搜索“iOS 自动打包”、“iOS build automation”这样的关键字。

其中稍微有点麻烦的就是如何应用不同环境下的不同配置，比如说测试环境连测试环境服务器，生产环境连生产环境服务器。有关程序配置管理部分，可以参考这篇文章：《大型项目程序配置管理演化之路》

每次部署新的候选发布版本后，要做回归测试

在每次开发人员部署新的候选发布版本到测试环境后，还需要做一次回归测试。也就是说在 Bug 修复完，对主要流程要重新测试一遍，同时还要对之前确认过的 Bug 再确认一遍，以确保 Bug 确实修复了，并且没有引入新的 Bug。

如果当前候选发布版本达到版本发布的质量标准后，就可以准备发布了。

申请上线发布

上线发布是一件很严谨的事，所以在正式上线发布前，通常还需要有一个申请和审批的流程。审批的主要目的是要有人或者有部门统筹对所有的上线发布有一个全面的了解和控制，避免上线过于随意导致问题，避免和其他部门的上线冲突。

部署发布

如果已经实现了自动化，部署发布应该是非常简单的一步。如果还没有自动化部署发布，也需要事先将详细的操作步骤写下来，避免部署发布时发生纰漏，这样在实际部署发布时，按照事先写好的步骤操作就不容易出现错误。

上线后的测试

项目上线后，测试人员需要马上对已经上线的版本做一个主要功能的测试，以确保线上运行正常。如果做好了数据监控，还同时要对一些关键数据进行监控，例如服务器 CPU 利用率、内

存占用率、服务出错率等数据。

如果万一发现版本上线后出现问题，需要考虑按照事先准备好的回滚方案进行回滚操作，尽量将损失降到最低。通常不到万不得已，不建议马上对问题打补丁进行修复。因为哪怕很小的代码修改，都可能会引入新的 Bug。而重新做一遍回归测试，耗时会比较长。

以上就是版本发布的一个常见流程，你也可以基于这个流程制定适合你项目的流程，让你的版本发布更加稳定可靠。

软件上线只是新的开始

当你的软件上线后，这不代表你的项目就结束了，可能这才只是新的开始。

用户在使用你的产品的时候，可能会遇到一些 Bug 或者是有一些建议，所以需要给用户反馈的渠道，让用户可以有途径对于 Bug 或者功能去反馈。通过收集用户的反馈，可以进一步完善你的软件产品。

只是靠用户主动反馈问题还是不够的，需要主动的对发布的版本进行监控，比如说要收集 App Crash 的 Log、监控服务器资源占用情况、监控 API 出错的比例、监控网页响应的速度等数据。当发现数据异常时，很可能说明发布的版本是有问题的，需要及时的应对，回滚版本或者发布新的更新补丁。

有关线上监控和报警的内容，将会在后续我们课程《38 | 日志管理：如何借助工具快速发现和定位产品问题？》中带你继续学习。

不管怎么样，软件成功上线了都是一件值得祝贺的事情，同时也是时候回顾总结一下整个项目过程了，关于这一点，我也会在后续专栏文章《39 | 项目总结：做好项目复盘，把经验变成能力》中跟你一起探讨如何做好项目总结复盘，把经验变成能力。

总结

今天带你一起学习了版本发布的相关知识。做好版本发布，关键在于版本发布前做好版本发布的规划，以及采用一个科学的发布流程。

版本规划，其实就是通过合理的规划，尽可能的让软件的功能和质量，满足好用户的预期。所以一方面要尽可能提供应有的功能和保证质量，另一方面也可以通过合理的发布策略，例如 beta 测试，降低用户预期。

通过规范的发布流程，可以确保要发布的版本正确以及发布质量的稳定。流程的关键在于发布前要对代码冻结，避免发布前频繁修改代码引入新的 Bug，同时在每次修复 Bug 后，要做回归测试保证 Bug 被修复以及没有引入新的 Bug，上线后还要对线上版本再一次测试，确保没有问题。整个流程中一些手工部署发布的操作应该尽可能自动化。

最后，软件上线只是新的开始，还需要收集用户的反馈，对线上服务进行监控和预警，对整个版本的开发过程进行总结回顾。

课后思考

你在发布版本前，会做哪些准备工作？你的发布流程是什么样的？通过对这篇文章的学习，你觉得有哪些可以改进的地方？或者你觉得有什么更好的方案可以分享的？欢迎在留言区与我分享讨论。

感谢阅读，如果你觉得这篇文章对你有一些启发，也欢迎把它分享给你的朋友。

精选留言 (10)

beiler
2019-06-02

老师，我想问个问题啊，比如我冻结了代码，但是测试测试出来了二十个bug，那回归测试是二十个bug修完了一起测试还是修一个测一个？最后再整体测试？

作者回复: 好问题！

冻结了代码后测试出来的Bug，首先要分级，不会20个都需要在当前版本修复的，有一部分影响不大的可以放到下一个版本修复。

然后剩下的这些Bug，什么时候测试取决于两个条件：1. 你什么时候部署测试环境，只有部署了才能测试；2. 测试自己的测试节奏，比如他们每天测试上一天的bug。

回归测试一般不会测试完一个就做一遍，但也不需要全部修复了才做，都是这一批测试完了再做一次，比如说今天测试完昨天所有修复的Bug后，会再做一遍回归测试，看有没有造成新的Bug。

8

熊猫

2019-05-21

多个项目发版时，要注意依赖，莫慌，匆匆忙忙很容易出现问题

作者回复: 👍依赖关系这个也很重要

如果自动化更新，应该能很好的改善这个问题，可以按照设定好的顺序去部署更新。

7

yellowcloud

2019-05-21

宝玉老师，您好，我们目前有一个项目是做实时数据采集的，对方将实时数据推送给我们，基本上每天每个时刻都可能有数据推送过来。这样就导致一个问题，我们部署新的版本时，他们的数据还在推送，这样就不可避免地丢失了部署过程中的数据，对方也没有重新推送的机制。请问下宝玉老师，这种问题有没有比较好的解决方案，以解决更新版本时数据丢失的问题。

作者回复: 这个问题其实不复杂，你可以将服务分拆，独立出来一个专门接受数据的服务，这个服务极其简单，只做一件事：接收数据，并存储到数据库或消息队列。

你原有的服务，改从数据库或者消息队列读取即可。

更新部署的时候，接受数据的服务就不要轻易更新了，这样就不担心会丢数据了。真要更新，只要和对方协商一下，暂停推送就好了。

--

@熊欲轻飞: 这种建议还是做个返回确认的机制，推送方对得到确认的数据做标记，没有得到确认的加入队列后续再推。

5

Charles
2019-05-22

我们目前人工管理版本很容易乱，看了文章才知道最后一位是构建版本并且是自动生成的，之前也没深究，豁然开朗，看来很多表面上看起来很简单的东西深究都是学问，谢谢！

另外问宝玉老师一个团队管理问题，如果是瀑布流带点敏捷部分实践的开发方式，总觉得UI这个岗位工作量不怎么饱和，一个版本过来特别是小版本迭代，周期可能是两周，UI可能1天就搞定了，其它岗位都差不多都要全程跟下来，这个问题出现在哪里？

作者回复: 这个问题有点不好回答，毕竟对你项目情况不够了解。

我觉得呢，如果UI这个岗位对你的团队来说是必须的，并且UI设计师很好的完成了他/她的工作，那么就很好，没有任何问题。毕竟有的人效率就是比较高，好过故意磨洋工看起来很忙。

如果UI这个岗位是可有可无，那么就可以考虑不设置这岗位，将工作外包出去，或者尽可能用一些标准的UI，或者让前端工程师兼职UI设计工作。

小老鼠
2019-10-07

一个产品的客户有许多，客户的环境各不相同，如何保证产品质量？(比如A电信企业生产的DNS服务器，在X产商集成的是B电信企业的DHCP服务器，在Y产商集成的是C电信企业的DHCP服务器…。)

作者回复: 我觉得对于你说的这种情况要保证产品质量，需要抓两头：

一头是在做需求分析和设计的时候，充分考虑到各种环境的差异；

另一头是在测试和验收的时候，要充分对哥哥环境进行测试。

纯洁的憎恶
2019-05-25

版本规划：
1.规划好要发布功能。必要的先发，锦上添花的后发。根据用户的质量容忍度，划分质量标

准，明确哪些功能质量必须一步到位，哪些可以慢慢来。
2.设计好发布策略。对内发布还是对外发布，beta版还是正式版，是否用灰度测试，管理好用户预期。
3.制定综合性发布计划。协调项目团队、客户、运营等多方利益确定发布时间进度。

科学规范的发布流程：
1.代码冻结。禁止修改即将发布到主干的程序，以避免出现不可控的新bug。
2.bug分级。冻结程序中的bug做到心中有数，但对于极其严重的bug仍要在发布前修正，修复后要生成新的候选版本号以示区分。
3.回归测试。最终候选版本发布生产环境前，对主流程和已知bug测试，确保无误。
4.上线申请、审批与部署。
5.上线测试。第一时间发现问题，如有必要立刻回滚，切忌急于打补丁。

作者回复: 👍感谢分享补充!

3

calvins
2020-04-08

2 b和2 c在版本上线差距真大，从文章来看，我们大部分的发版还是不够严谨。没有beta版，没有灰度，经常紧急修复，版本管理工具缺失，好多问题。不知道这么方面老师有没有好的方法或建议？

作者回复: 版本发布要做到稳定可靠，需要注意做到几点：
1. 自动化测试不可少，只依赖人工测试是不够和不可靠的，要想系统稳定，一定要写足够的自动化测试代码，覆盖主要的用户使用场景。借助持续集成CI/CD工具每次代码合并分支，每次部署之前都跑一次自动化测试，跑不过就要分析原因，修复为止。

2. 测试要充分，在正式部署到生产环境之前，在测试环境要运行一段时间，发现问题提交到Bug跟踪系统跟踪起来，根据优先级修复，通过测试后再部署生产环境。

3. 版本管理工具要用好，把测试版本和开发版本要分开，开发中的版本是不稳定的，如果一边开发新功能还一边修复bug，很难稳定下来。所以最好是要有两个分支，一个是开发新功能的，一个是已经开发好测试中的，部署后再合并一起，新版本完成了再创建新的测试分支。

4. 发布和回滚都实现自动化，一方面提升部署和回滚效率，另一方面可以减少人工导致的错误。

5. 一些新功能可能会导致系统不稳定，最好让模块尽可能独立，加上开关，让一小部分用户先使用，

逐步放开，如果出现故障，通过开关控制关闭，这样不必回滚。

6. 用好ELK、Splunk这样的日志收集分析系统，把关键信息记录起来，当发现问题的时候，可以借助日志快速定位到问题，即使问题还没发生，定期对日志信息进行分析，也可以对潜在问题提前发现。

1

E
2019-08-28

老师，我看您讲的大多数是针对2C，针对2B的有什么好的提议吗?

作者回复: 我个人对于2B确实没啥经验，所以讲得少。

2B的项目相对来说，迭代周期要长一些，对界面和用户体验要求低一些，更多是来源于客户需求，更难控制需求。

针对这些特点，要多花功夫在需求的管理和控制上，在开发上可以选择迭代模型或敏捷开发，优先实现已经确定的和核心的需求，勤和客户沟通。

1

javaadu
2019-05-24

这篇文章来得正好，很有收获，今天要组织项目的发布评审，昨天我在文档里主要梳理了几个点：本次发布的变更点；与前端的合作；自己内部应用的依赖关系；db变更；配置变更等等。

读了这篇文章，感觉自己需要改进几个点
第一，跟产品确认本次的功能点和标准
第二，不同应用的发布记录和灰度计划
第三，发布上线后健康状态的监控（这块我已经做了个降级开关，不过需要提出来，把信息同步出去）

另外，今年跟着老师的专栏学习下来，感觉自己的项目管理能力有不小的提升，表现在工作中就是试用期就开始做技术pm了，感谢老师🙏

作者回复: 👍赞，能学以致用最好!

也谢谢你的支持

1

ifelse

2022-07-05

软件上线只是新的开始，还需要收集用户的反馈，对线上服务进行监控和预警，对整个版本的开发过程进行总结回顾。--记下来