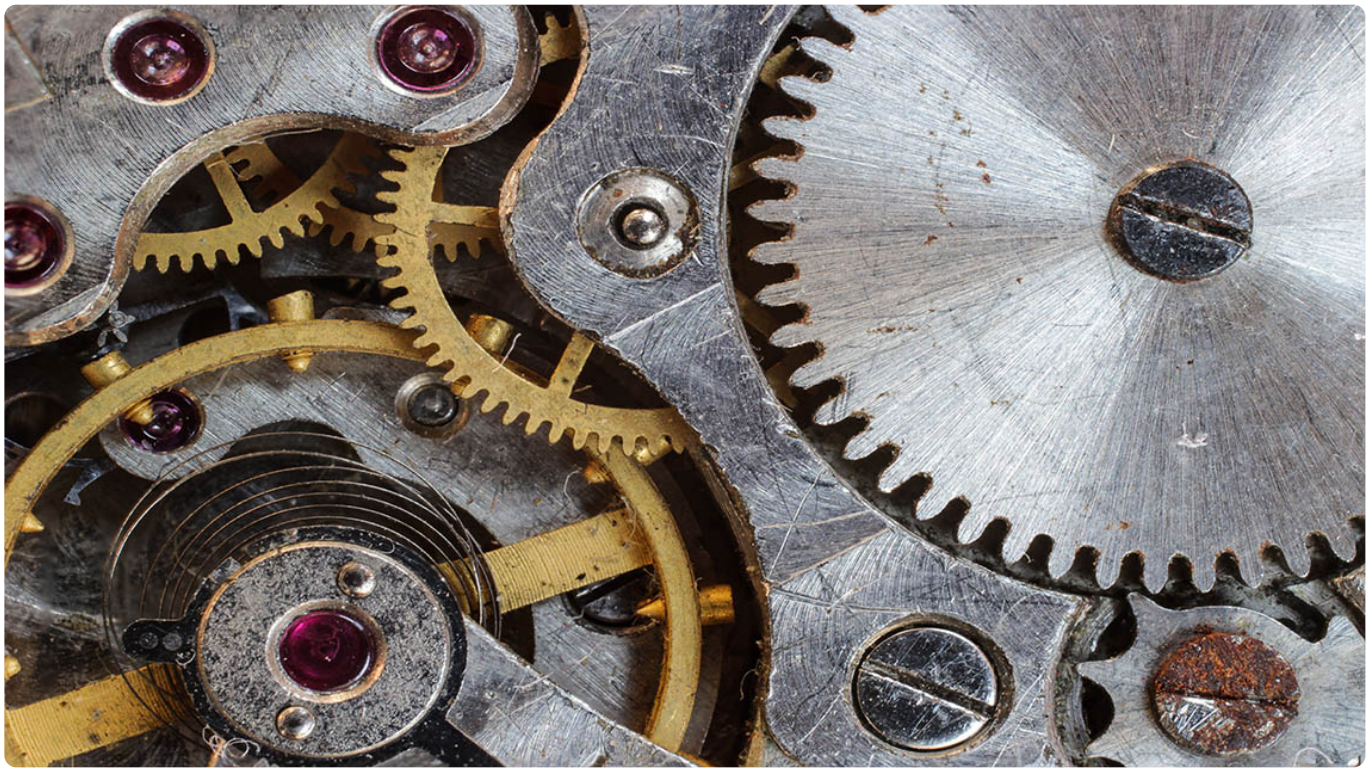
03 | 瀑布模型：像工厂流水线一样把软件开发分层化

2019-02-28 宝玉

软件工程之美 进入课程 >



讲述：宝玉
时长 20:08 大小 18.44M

你好，我是宝玉，我今天分享的主题是：瀑布模型，像工厂流水线一样把软件开发分层化。

可以这么说：**瀑布模型算是现代软件工程的起源，软件工程的发展，很大部分都是构建于瀑布模型的基础之上的。**我们后面所学的软件工程的很多内容，都是源自瀑布模型的衍生，或者其中某个阶段的细分。

我在上大学期间，还并不懂软件工程瀑布模型这些知识。当时我自学了点编程知识，然后开始在外面接点做网站的小活，开发模式非常简单，接到活直接写代码，有问题就改。这样下来居然也做了不少小网站，但是大一点的网站项目就搞不定了，甚至手头的小网站项目，找个同学帮忙都不知道大家该怎么分工。

所以当时我也很好奇，大的软件系统是如何开发出来的？那么多人一起开发一个软件，系统是如何分工协作的？

后来到大三的时候，开始系统学习软件工程课程，我才开始了解到一些理论知识，包括我做小网站的这种开发模式，都有一个专业术语，叫边写边改（Code And Fix）模型。

这不是我的发明。在 1960 年初，软件开发刚开始起步，这时的软件开发是混沌无序的，那时候编程语言还是汇编语言为主，开发模式就是边写边改模型。如果程序员水平高，功能简单，还是可行的。

后来软件开发需求越来越多，功能越来越复杂，从事软件开发的人员水平也参差不齐，这种落后的软件生产方式已经无法满足迅速增长的计算机软件需求，从而导致软件开发与维护过程中出现一系列严重问题，这个现象也被称之为“软件危机”。

像这种边写边改的开发模式，为什么说不能满足复杂软件项目的需要呢？主要是有几方面的原因：

- 整个开发过程不可控，想基于这种开发模式做项目计划太难；
- 项目的人数多了后，无法有效分工协作；
- 项目开始的时候对需求几乎没有进行有效分析，对需求的理解容易出现偏差，后期导致很多返工；
- 项目编码完成后，没有有效测试，运行时 Bug 非常多。

瀑布模型的诞生

为了解决软件危机中的这些问题，在 1970 年，Winston Royce 博士借鉴了其他工程领域的思想，比如建筑工程，提出了瀑布开发模型，指出软件开发应有完整之周期，并将软件开发过程分成了若干阶段。像瀑布一样，从上往下，完成一个阶段继续下一个阶段。

瀑布模型把整个项目过程分成了六个主要阶段：

一、问题的定义及规划

这个阶段是需求方和开发方共同确定软件开发目标，同时还要做可行性研究，以确定项目可行。这个阶段会产生需求文档和可行性研究报告。

二、需求分析

对需求方提出的所有需求，进行详细的分析。这个阶段一般需要和客户反复确认，以保证能充分理解客户需求。最终会形成需求分析文档。

三、软件设计

根据需求分析的结果，对整个软件系统进行抽象和设计，如系统框架设计，数据库设计等等。最后会形成架构设计文档。

四、程序编码

将架构设计和界面设计的结果转换成计算机能运行的程序代码。

五、软件测试
在编码完成后，对可运行的结果对照需求分析文档进行严密的测试。如果测试发现问题，需要修复。最终测试完成后，形成测试报告。

六、运行维护
在软件开发完成，正式运行投入使用。后续需要继续维护，修复错误和增加功能。交付时需要提供使用说明文档。

瀑布模型在提出后，因为其简单可行，切实有效，马上就在很多软件项目中应用起来，一直到 2000 年前后，都是最主流的软件开发模型，即使到现在，你也能在很多软件项目中看到它的影子。

也是从那时开始，有了“软件生命周期”(Software Life Cycle,SLC) 的概念。

> 软件生命周期是软件的产生直到报废或停止使用的生命周期。而像瀑布模型这样，通过把整个软件生命周期划分为若干阶段来管理软件开发过程的方法，叫软件生命周期模型。

虽然现在瀑布模型已经不是最主流的开发模式，那为什么我们现在还要学习瀑布模型呢？

因为不管什么软件项目，不管采用什么开发模式，有四种活动是必不可少的，那就是需求、设计、编码和测试。而这四项活动，都是起源自瀑布模型，也是瀑布模型中核心的部分。

学好瀑布模型，才可以帮助你更好的理解这些内容。

如何用瀑布模型开发项目？

如果单纯看这些阶段的概念介绍，还是有点难以直观地理解整个软件开发过程，在这里拿我经历过的一个网站开发项目作为案例，来看一下如何使用瀑布模型来开发一个软件项目。

问题的定义及规划的阶段

大概在 2009 年的时候，Web2.0 还正火，公司老板打算做一个游戏领域的社交网站。

问题很明确，就是要做一个社交网站，并且用户能按照游戏来交友。至于可行性分析嘛，按照当时 Web2.0 的热度，这个似乎是可行的。那么就立项了。

然后老板问项目经理，这么样一个网站，你大概得多久做出来？项目经理一看，这么复杂一个网站，怎么也得半年才能做出来一个版本，于是说半年。老板说半年太久了，给你三个月吧，项目经理心中叫苦，最后讨价还价，决定四个月上线。

于是，项目经理按照四个月开始倒推项目计划：

> 需求分析——2 周；
> 软件设计——4 周；
> 程序编码——6 周；
> 软件测试——4 周。

需求分析的阶段

在项目立项后，产品经理首先和老板充分的沟通，了解老板的想法是什么，要做一个什么样的网站。在了解老板的想法后，产品经理对市场上同类的社交网站进行了调研，然后用原型工具设计了网站的原型。原型虽然很简陋，但是从原型可以看出来，项目要做成什么样子，便于确认需求。

原型拿给老板看后，老板再根据自己的想法提一些反馈，这样反复沟通确认，在原型设计确认清楚后，产品经理开始撰写产品设计文档，将原型设计落实到文档，将整个网站划分成不同的功能模块，例如用户注册、登录、添加好友等，确定每个功能模块需要哪些功能。

这个阶段产品经理是最忙的，那这时候其他人在干嘛呢？其他人都还挺轻松的，架构师研究网上流行的社交网站都采用什么架构，程序员、测试看看技术文档。

虽然最终确定了产品设计文档，但是因为中间反复确认的时间过长，原定 2 周能完成的需求分析，最后拖到了 3 周。项目经理一看，最终上线时间点没法延，那就只好压缩编码时间了，不行加加班！

项目计划变成了：

> 需求分析——3 周；
> 软件设计——4 周；
> 程序编码——5 周；
> 软件测试——4 周；

软件设计

产品经理的产品设计文档确定后，架构师开始做架构设计，UI 设计师开始设计 UI，测试经理开始针对产品设计文档写测试用例，产品经理还要进一步设计交互。

由于前期原型设计工作做的好，所以 UI 设计还是很顺利的，主风格定下来以后，各个界面就是细节的确认了。

因为产品设计文档写的详细，输入输出很清楚，测试用例也进展顺利。

至于架构设计这边，架构师很有经验，先把整体架构确定，写了个技术方案文档，和大家一起开会讨论，几次后确认了整体技术方案。按照功能模块一拆分，把其中一个功能模块做了一个样板，然后把各个子模块分给开发人员，大家一起协助做详细设计，然后再分别确认。

大家都如火如荼地忙起来了。如果一切顺利的话，软件设计 4 周应该能完成，可以进入编码阶段了。但是软件设计进行到第 3 周的时候，老板的想法发生了一些变化。

因为市场上已经有了游戏社交的网站，而且运营结果不算太好，而网页游戏正流行，如果我们的平台能接入网页游戏，这会是个不错的机会。

于是需求变更了，我们要能和其他网页游戏的用户系统对接，这个需求最开始是没有提出来，也没有考虑的。

项目经理考虑再三，决定还是接受这个需求变更，但是希望能多一些时间，老板没同意，认为时间点很重要，哪怕砍一点功能，牺牲一点质量也要如期上线。但就算这时候砍功能，设计工作还是少不了多少。

于是产品经理重新修改相应原型，再确认，再重新修改产品设计文档。变更完后，UI 设计的相关页面重新修改设计、测试人员修改测试用例，最苦的是架构师，当初没有考虑到要和其他用户系统对接，现在用户系统的设计都要重新考虑了。

于是为了赶进度，项目组开始加班，即使如此，软件设计阶段也推迟到了第 5 周才勉强完成。

项目计划又变了：

> 需求分析——3 周；
> 软件设计——5 周；
> 程序编码——5 周；
> 软件测试——3 周。

程序编码

终于进入编码阶段了，为了保证进度，加班还在继续，哪怕前期做了大量的设计，真到编码的时候还是有好多没有考虑到的，同时各个模块之间还存在相互依赖，有时候虽然自己功能开发完成，还需要等待其他人的功能完成才能调试，所以 5 周时间很快就过去了，而程序还不能完整地跑起来。

其实中间还有个小插曲，老板觉得还要加上支付的功能，但是项目经理觉得这个阶段改需求已经不可能了，以辞职为威胁总算顶回去了，打算放在下个版本加上。

终于到第 6 周的时候，有了一个勉强可以测试的版本。

项目计划现在变成了：

> 需求分析——3 周
> 软件设计——5 周
> 程序编码——6 周
> 软件测试——2 周

软件测试

留给测试的时间只有两周了，但是前期实在 bug 太多，两周测试时间过去，软件质量还是很糟糕，完全无法正常使用，于是项目不得不延期，一直延期了 4 周后，才算具备上线条件。

所以最终的项目计划差不多是：

> 需求分析——3 周
> 软件设计——5 周
> 程序编码——6 周
> 软件测试——6 周
> 和原定计划已经延迟了 4 周。

运行维护

网站上线后，好在前期并没有多少用户，但是线上 Bug 还是不少，需要继续修复线上发现的 Bug。

瀑布模型的优缺点

以上案例是我参与过的、用瀑布模型开发的软件项目的一个缩影，你会发现瀑布模型其实跟我们传统的建筑建造方式非常类似。我们拿盖房子的过程来看看瀑布模型。

客户想要盖一栋房子（**初步的想法**）。

客户一开始可能没想清楚想要什么样子的房子。（**客户对需求还不清楚**）

施工方开始找客户确认：用途是什么，要个几层的房子，什么建筑风格，希望什么时间完工，预算多少。（**问题定义**）

施工方根据客户提的需求，对比工期和预算，评估是不是值得做。（**可行性研究**）

施工方评估后觉得可行，于是和客户签订合同，约定价钱和工期。（**立项，制定项目计划**）

施工方开始跟客户沟通确认需求，例如每层户型如何，将来的装修风格等。（**需求分析**）

确认完需求后，施工方开始出建筑施工图，还画了漂亮的建筑效果图。（**系统设计和 UI 设计**）

施工方按照设计图开始施工。（**程序编码**）

这期间如果客户去参观施工情况，客户只能看到毛胚，只有最后施工完成才能看到最终样子。（**在中间客户看不到结果，只有最后能看到结果**）

原定二层是两个卧室，在房子施工过程中，突然客户说两个卧室不够，要改成三个卧室。这意味着施工方要对施工图重新设计，很多已经建好的房间要拆掉重建。（**瀑布模型是很**

难响应需求变更的，而且越到后期代价越大）

工程质量检查人员对施工结果进行质量检测，如果不满足质量要求，需要修改。（**测试**）

最后验收通过后，客户入住。（**上线**）

所以你看，用瀑布模型开发软件，就像建筑工程里，盖房子一样简单和自然。每个阶段都有侧重的事情，就像需求阶段专注于搞清楚需求，编码阶段专注于实现。

最重要的是，这种编码前先设计、编码后测试、整个过程重视文档的方式，开发出来的产品，质量相对是有保障的。

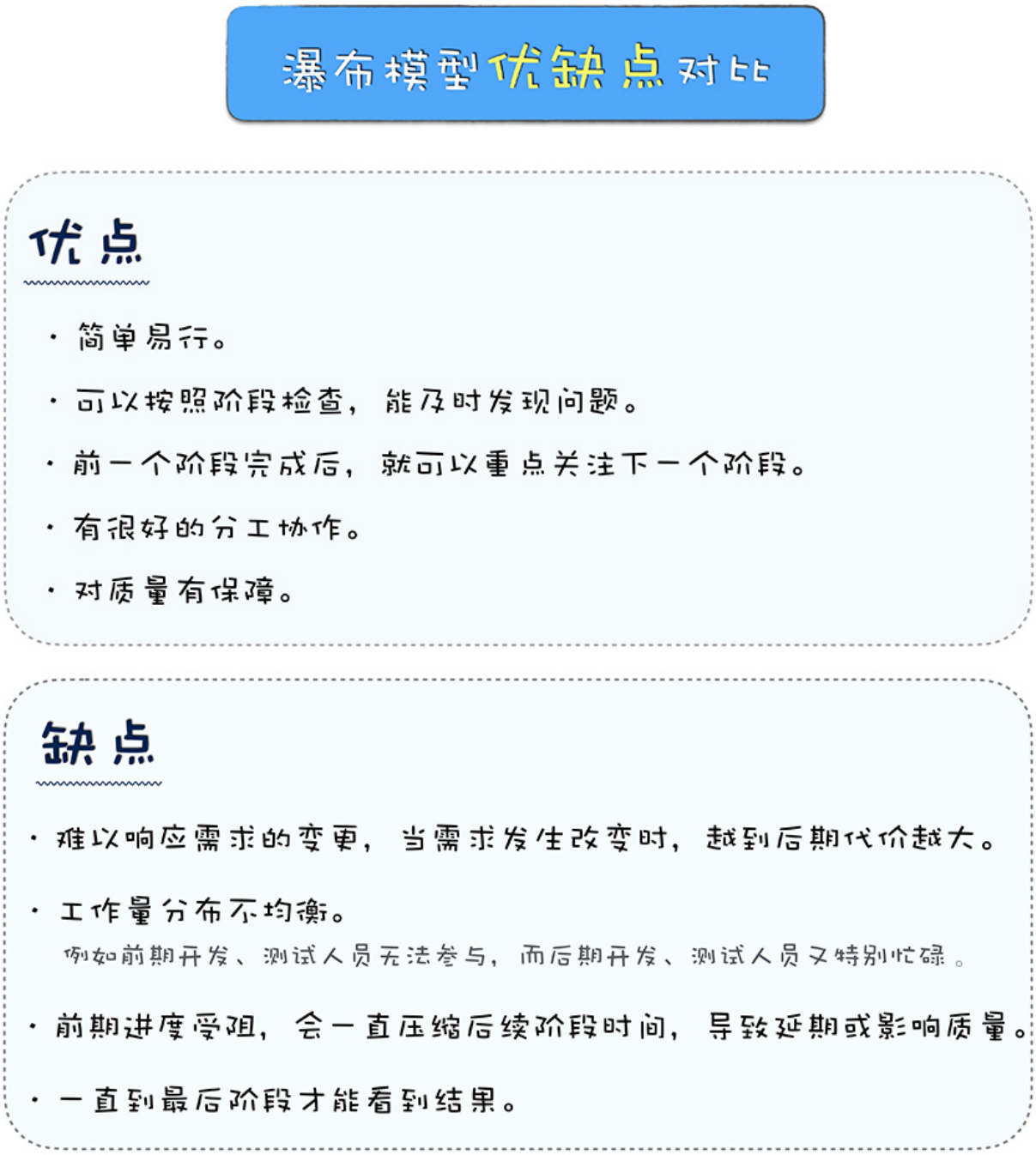
但用瀑布模式开发，也存在一些问题。

最大的问题就是不能及时响应需求变更，越到后期变更代价越大。另外，通常要到最后阶段才能看到结果是什么样子。

我以前参与过的用瀑布模型方式开发的项目中，在开发和测试阶段加班是常态，原因就在于需求分析和系统设计可能会有延误，从而延迟了编码阶段的开始时间，压缩了编码实现的时间。

而在编码阶段，通常会发现很多设计时没有考虑清楚的问题，或者遇到需求变更，导致编码阶段即使加班加点也会大大延期，最后留给测试阶段的时间就不够多了。

鉴于瀑布模型存在的这些问题，后来又有很多人提出了其他的软件生命周期模型，比如快速原型开发模型、增量模型、迭代模型，以期保留瀑布模型的这些优点，克服瀑布模型中存在的问题。我们将会在后面的章节中，详细介绍瀑布模型衍生出的其他开发模型。



总结

从瀑布模型提出将近 50 年过去了，虽然现在大家一提起瀑布模型，似乎已经成了落后的代名词，但在当时是有划时代意义的。如果类比一下，我觉得瀑布模型的价值相当于工业界第一次提出流水线作业。

> 1769 年，英国人乔赛亚·韦奇伍德开办埃特鲁利亚陶瓷工厂。以前制作陶瓷只有“制陶工”一个工种，一个人从挖泥、制胚到最后烧制，要求很高。但是乔赛亚把原本的制陶流程从开始到结束分成了若干阶段，每个阶段可以由不同的人完成，从单一的制陶工分成了挖泥工、运泥工、扮土工、制坯工等，这样就大大提高了生产效率，也降低对工人的要求。

同理，瀑布模型的出现，也解决了软件项目开发中的几个重要问题。

让软件开发过程有序可控。瀑布模型的每个阶段都有明确的任务，每个阶段都有明确的交付产物，都有相应的里程碑。这些让整个过程更可控，而且能及早发现问题。

让分工协作变成可能。瀑布模型的六个阶段，也让软件开发产生相应的基础分工：项目经理、产品经理、架构师、软件工程师、测试工程师、运维工程师。

质量有保障。瀑布模型每个阶段都需要交付相应的文档，而文档的撰写和评审，可以帮助在动手之前把问题沟通清楚，想清楚。瀑布模型在编码结束后，会有严密的测试，只有测试验收通过后，才能上线发布。这些措施都让软件的质量更有保障。

课后思考

通过今天的学习，你可以对照一下你工作中的软件项目开发是不是采用的瀑布模型？和瀑布模型有哪些不同？你认为瀑布模型有哪些好或不好的地方？欢迎你在留言区留言，和我分享讨论。

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。

上一篇　02 | 工程思维：把每件事都当作一个项目来推进

下一篇　04 | 瀑布模型之外，还有哪些开发模型？

精选留言 (44)

写留言

纯洁的憎恶
2019-02-28

23

我对瀑布模型感触颇多啊！首先瀑布模型把复杂的软件生产过程，按照时间线索，切分为若干较为独立和专业的部分，条理清晰。在每个阶段内只需要集中精力与阶段任务即可，不用胡子眉毛一把抓。每个节点有交付件，过程可控、权责清楚明白。破布模型特别符合我所在的大型央企的性格。但是我经手好几个项目，也被瀑布模型折腾的死去活来。比如我现在正在处理的项目。...

展开

作者回复: 很感动，写了这么长的留言。这是个很生动的案例。像这样的环境，虽然说根源不是瀑布模型造成的，但是瀑布模型确实加剧了问题。

对于这种开发模式，邹老师在《构建之法》中有精彩的论述：“老板驱动的流程：笔者在和中国一些企业的软件开发者交流的时候，听闻不少人提到开发流程事实上是由行政领导主导，或者由公司的老板驱动，我们姑且把它命名为老板驱动的流程。” 领导“有极大权力， 极小责任”的驱动方式。

这种项目可以考虑调整为快速迭代的模式（下一篇会提到，核心就是控制每次的需求），尽早上线核心功能，上了慢慢改，下个迭代改。就像老话说的：“生米先煮成熟饭”。

还有PM应该要能顶得住压力，在一个迭代中应该不改或者少改，才能继续推进。

另外，也能理解这种项目不是靠个人就能改变太多的，适当的时候提一些建议，改变能改变的，接受不能改变的。

最后，有具体问题也欢迎继续留言讨论，很乐意提供建议：）

西西弗与卡...
2019-02-28

9

瀑布模型分解和识别出了软件开发过程中的几种主要活动，以及每种活动所关注的价值点，并从时间尺度上划分了对应的阶段。这几个阶段形成了一个环。现代的其他软件工程

方法，举个不太恰当的比喻，就像是滚动更快的轮子。不管什么样的轮子，这几个阶段的功夫，我们都得练好

作者回复: 你这个比喻很有意思的👍，下一篇关于衍生模型的很多例子完全适应你这个比喻。

敏捷开发的Scrum可能还是有点不一样，每个Sprint并不完全按照这个周期，但是在一个Sprint的某个用户故事的开发，其实也适用于这个环。

所以你说的关于这几个阶段的功夫，确实都得练好！

一路向北
2019-02-28

5

目前我们的开发模式还是以瀑布型的为主。之前也引进过敏捷开发模式，但是因为团队不大，每个人的分工比较清楚，甚至经常是每个人负责一个项目，所以也就没有按照敏捷的那一套流程走。
我的理解是，如果产品的需求变化不大，产品设计者的前瞻性和设计能力比较强，设计的框架的伸缩性强，是不是瀑布式的开发会优于敏捷开发呢？也就是说，产品经理的需求...
展开

作者回复: > “如果产品的需求变化不大，产品设计者的前瞻性和设计能力比较强，设计的框架的伸缩性强，是不是瀑布式的开发会优于敏捷开发呢？”

如果满足你说的前提，再加上没什么进度压力的话，瀑布模型像计划性、质量、过程控制都是很好的，是不是比敏捷开发好这个我不好随便下结论，因为敏捷开发理想情况下一样可以质量高。问题是绝大部分时候真的不是理想情况。

> 对于较大的项目，如果能做到合理的项目分割，划分，那么在分项目中再实行瀑布式开发，是不是效率也会挺高？

是的，你说的这种大的拆小的是下一篇会讲的迭代模型，基于瀑布模型，但是更小，也更高效。在再下一篇我还会对比迭代模型和敏捷开发，希望能帮助你更好的理解其中的差别。

很多人不愿意尝试其他瀑布模型之外开发模式，可能是因为已经对瀑布模型太熟悉了，多看看多学习一下其他模型好的地方，再尝试实践也是很好的。

王二宝

2019-03-01

4

我是非科班出身的程序员，工作第三年，真的觉得《软件工程》是门必须要掌握的学科。我们公司是几人小团队，很多流程都是自己摸索的，我今天才知道，原来我们平时的工作流程就是瀑布模型啊。让我想到了吴军的专栏中说，很多野路子探索了很久才恍然大悟，并以此骄傲的东西，对于有理论基础的人来说，这就是很平常的理论啊，甚至是觉得这东西，是与生俱来的。

展开

作者回复: 你这个总结的特别好👍

我以前不是科班的时候，就是怎么都想不明白大项目怎么开展，后来改专业到软件工程后，就觉得这是很自然而然的事情了。

而且虽然我们大学只讲了瀑布模型，但是后来再看迭代模型，马上就明白怎么回事了。当然也有科班也有问题，例如我刚开始对敏捷就有点不屑一顾，觉得瀑布模型就很好呀：）

tongmin_ts...

2019-03-01

3

老师，我经历不同的公司，因为在较传统的行业，基本都采用瀑布模型，但是不同的公司，在瀑布模型的流程上大体差不多，就是每个流程，像您里面说到的 问题的定义及规划 -》需求分析-》软件设计-》程序编码-》软件测试-》运行维护等阶段，输出产出不同的成果，大体就是不同的公司，每个阶段产出不一致，特别是问题的定义及规划 -》需求分析-》软件设计这些阶段，比如基本都是产出相应的文档，有些还产出如原型图，流程图...

展开

作者回复: 文档这部分，没有什么统一的模板。我的观点是，文档最关键的地方在于达到其目的，格式是其次的。
文档的目的我觉得主要是两点：
1. 写文档的过程中帮助你梳理清楚逻辑
2. 写出来的文档是要用来沟通的，让其他人通过文档明白你的思路

所以建议不用太在意格式，重点放在内容上就好了。

纯洁的憎恶

2019-02-28

3

老师的留言区真是异彩纷呈啊😁

展开

作者回复: 感觉大家项目经历都很丰富，一些内容触动了大家留言的想法💡

很喜欢看大家的留言，可以了解大家想要什么有什么问题，也可以了解到很多有意思的项目，根据留言还可以调整后面的内容呢。

谢谢

javaadu
2019-02-28

3

我们是包在敏捷开发下的瀑布模型，实际实施的时候还有点边写边改模型的成分。

瀑布模型之于软件开发，就像流水线作业之于小作坊，有了这套理论指导下的软件小作坊终于有机会稍微正规一点。
...
展开

作者回复: 04就会讲迭代模型，是最容易从瀑布模型演进过去的，可以很有效的应对需求不清楚的情况。其实你这种情况最好是05敏捷开发，但是实施难度要大一些，不够了解的话还不如迭代模型。

具体到实施上来说呢，就是你考虑把大瀑布拆成小瀑布，固定迭代的周期（例如2-4周），每个迭代都发布一个可以运行的版本（非常重要），本质上还是瀑布，但是这种模式，可以让你先选择优先级高的需求，当前迭代如果没搞清楚需求也没关系，下个迭代改进完善。

可行性分析和需求分析在后面章节会涉及。

Tomcat
2019-02-28

3

瀑布模型原来才是最经典的软件开发模型！这一点大大出乎我的意料。也就是说，后续那些敏捷开发等模型都是在解决瀑布模型的缺点而努力的！

作者回复: 👍
是的，瀑布模型虽然问题比较多一点，但是意义重大。后续的模型都是为了解决瀑布模型而努力

的。

cljloves
2019-03-01

2

几年前用瀑布模型开发过一些手机APP项目，可能是APP比较量轻，除了第一版耗时4个月左右，之后就进入快速迭代（迭代周期在一个月内）。产品部会在当前版本的开发和测试的环节进入新版本迭代进程，等到测试完毕，差不多设计也快递交结果给开发了，工种间的空闲间隔并不长，工作量还是蛮饱和的。不过进入大功能迭代还是会“卡壳”，而且无论何时，开发和测试总在加班，哈哈哈。期待后续的专栏。

展开

作者回复: 大功能确实会超过一个迭代周期，这种情况下，还是应该坚持按时发布更新，先把完成的小功能和bug修复发布，没完成的放在后面的迭代中继续完成。不然就又回去大瀑布了。

Felix
2019-02-28

2

实际工作中我举一个例子，倒逼项目往往就自然而然地使用了瀑布模型，而中间一些领导的"创意"或竞品的对齐导致需求变更，最惨的是瀑布的下游——开发和测试 的疯狂加班，请问宝玉老师对于倒逼项目有没有自己的处理方式和建议

作者回复: 通常两种方案：
1. 砍需求，以保证可以按时完成
2. 快速迭代，可以及时响应变更

中年男子
2019-03-06

1

花了一个小时把这一篇的正文及评论看完，真是精彩，受益匪浅，评论区的案例和宝玉老师的回复看一遍都不过瘾。

我现在公司最大的问题就是使用瀑布模型，需求还频繁改动，程序员苦不堪言。老板驱动极大权利极小责任简直是说到心坎里了...

展开

作者回复: 是的，其实我这也算是抛砖引玉，提出一个观点，然后引发更多讨论，有时候甚至讨论的观点更精彩！👍

那你可以尝试快速迭代试试，如果周期能短一点，也许对于需求的响应能迅速一点，能改善一点。

卡皮 1
2019-03-05

定了很多课程，宝玉老师是每条留言都回复的，赞一个！

展开

作者回复: 谢谢夸奖。

其实这就像上课，课后的答疑。毕竟一篇文章不可能面面俱到，难免有没讲到和没讲清楚的地方，通过留言和回复，可以把这些问题补充说明清楚。

alva_xu 1
2019-03-02

另外，老师提到瀑布里夹敏捷，我觉得这是大型项目常用的方法。比如对于大型系统的架构设计和落地，我觉得采用瀑布方法比较好，架构就如地基，如果需求不明，设计上经常要修改，自然会影响项目质量和进度。我觉得还是以造房子为例，越是地基和框架，越要少改，尽量一次成型。

作者回复: 还是看项目类型吧，因为架构一定是构建在需求之上的，如果需求不明确，很难设计出来合适的架构，所以难免会改，反倒不如先设计合适的架构，后面慢慢修改，一定阶段后重构。

alva_xu 1
2019-03-02

在我们企业，由于软件开发以外包为主，由于合同的原因，往往在项目外包前就已经基于需求做出了工作量的预算，以便有明确的合同价格，所以，难以推行敏捷或者迭代的工作

方法，往往采用瀑布模式进行开发。但同时，又存在“老板驱动的流程” 也就是 领导“有极大权力， 极小责任”，需求变更随时随地，用户验收又不好好进行，导致项目质量和工期都难以把握。我觉得比较好的解决方法，有两个，1，是建立自开发团队，2是和外包...

展开

作者回复: 你说的第二个方案应该还是可行的，因为甲方关心的是产品质量和价钱，如果你的方案价钱差不多，质量可以更好，他们应该会支持的。

当然这个很多时候不是一个人就能解决的了的。及时是瀑布模型，也可以借鉴一些其他模型好的实践来优化，例如在开发阶段采用迭代模型或者敏捷开发，采用持续集成提升效率。

Charles

2019-03-02

1

我们现在用的也基本是瀑布模型+小版本瀑布迭代，碰到需求变更也尽量放到下个版本拥抱，你文章的案例和留言中的案例看完，感触颇深似曾相识，精彩😁

几个点请教下老师：

...

展开

作者回复: 《10 | 项目计划：代码未动，计划先行》会讲时间估算。

通常时间估算并不需要特别精确，初步的需求分析就可以大致估算出每个阶段需要的时间。大致时间确定后，就可以设置里程碑。里程碑定了后再确定细的计划。

虽然我觉得甩锅不是什么好事，但是如果你真要甩锅，最简单有效就是设置流程去划分责任：）

上线后有问题其实很正常的，重要的是要有合理的机制：
1. 及时发现问题，监控报警、用户投诉反馈等
2. 马上解决问题，对线上版本有专门的代码分支，可以随时打补丁修复，测试上线
3. 避免后续再犯同样的错误。要分析原因，看什么导致问题，然后改进流程。

希望以上回答能解答你的问题，如果还有不清楚的欢迎继续留言。

Linuxer

2019-03-01

1

老师您好，小公司有很多项目就是一两个人，没有那么多角色，怎么做到按这种流程去开发项目呢?比如常常在代码编写过程中发现很多问题都没考虑全面又感觉在交流需求的时候根本就没想到，要怎么在之后的项目中不再犯这种问题呢?

作者回复: 即使只有一个人，建议也要做简单的需求分析和设计，做完后，形成简单的文档，找人评审一下，提一些意见。

因为你写文档的过程，给别人讲的过程，其实是在帮助你思考，帮助你梳理清楚逻辑，避免在实现的时候发现好多问题没想清楚。

还有一个思路就是快一点迭代，每一个迭代解决优先级最高的问题，然后下一个迭代中改进上一个迭代的问题。

项目中犯错误其实很正常的，重要的时候要总结，看看通过什么方式能改进，避免犯类似的错误。

One day
2019-03-01

1

之前用瀑布模型，现在项目是敏捷开发，各有优劣，都是为完成项目，只不过实践方式有差别，需求变更基本是常态，和项目大小有很大关系，处理方式也不同

展开

作者回复: 是的，瀑布模型和敏捷开发各有优劣，还是得看项目情况，团队情况，灵活选择，合理应用，甚至于相结合或者借鉴好的实践。

aide
2019-03-01

1

感觉上课不能只看老师发的文章啊，看完文章之后还应该再认真看看留言，一些有工作经验学员就会对老师的文章很有感触，然后把自己的经验和总结当做实例来相结合老师的文章，能够更快更好的吸收知识。

展开

作者回复: 是的，很多留言很精彩的👍

beiler
2019-03-01

1

我觉得可以把需求细分，版本进行快速的迭代，比如列清1.1版本的需求，1.2版本的需求，每个月发布一个版本，这种快速迭代。但是这样就又有问题了，如果1.1版本我发现有bug，1.3版本都开发完了，那我bug怎么merge进去呢？可能每个版本的代码都出现了改动，这样合并会出现冲突。第二个问题，架构师如何把数据库设计的更合理能满足后续不断变更的需求呢？请老师指点下，有没有什么好的文章或书也可以推荐下

展开

作者回复: 如果你在1.1发现bug，需要甄别一下紧急程度，紧急的话，就1.1分支修复，然后hotfix，不紧急的，就1.2修复好了。

分支合并有两种策略：1. 不合并，如果有bug fix，各个分支同步修改；2. 分支合并回主分支，例如1.1上线时创建分支，1.2开始在主分支开发，开发时发现1.1的bug，只更新1.1分支，再定期把1.1的分支合并回主分支。

敏捷开发中，通常建议只做刚刚好的设计，不必要考虑太多后续的需求变更，定期再重构（例如你可以重新设计表，然后把旧数据导入）。

极客时间有个《从0开始学架构》的专栏（也出书了）不错，你可以看看。

Aaron
2019-02-28

1

现在还是用瀑布模型，期待尽快看到老师的文章

展开

作者回复: 下一篇就是介绍瀑布模型的衍生模型啦