68 | 软件工程的宏观视角

2019-12-27 许式伟

许式伟的架构课 进入课程 >

讲述：姚迪迈
时长 07:40 大小 7.04M

你好，我是七牛云许式伟。

软件工程

今天开始，我们进入第六章，谈谈软件工程。

我理解的架构师的职责其实是从软件工程出发的。也许大家都学过软件工程，但如果我们把软件工程这门课重新看待，这门学科到底谈的是什么？是软件项目管理的方法论？

无论如何，软件工程是一门最年轻的学科，相比其他动辄跨世纪的自然科学而言，软件工程只有 50 年的历史。这门学科的实践太少了，任何一门学科的实践时间短的话，都很难沉淀

出真正高效的经验总结，因为这些总结通常都是需要很多代人共同推动来完成的。

为什么说它只有 50 年时间呢？

我们先来看看 C 语言，一般意义上来说，我们可能认为它是现代语言的开始。C 语言诞生于 1970 年，到现在是 49 年。再看 Fortran，它被认定为是第一个高级语言，诞生于 1954 年，那时候主要面向的领域是科学计算。Fortran 的程序代码量普遍都还不大，量不大的时候谈不上工程的概念。

这也是我为什么说软件工程这门学科很年轻，它只有 50 岁。对于这样一个年轻的学科，我们对它的认知肯定还是非常肤浅的。

我在这个架构课的序言 “开篇词 | 怎样成长为优秀的软件架构师？” 一上来就做了软件工程和建筑工程的对比。通过对比我们可以发现，二者有非常大的区别，具体在于两点：

其一，不确定性。为什么软件工程有很大的不确定性？大部分大型的软件系统都有几千甚至几万人的规模，而这几千几万人中，却没有两个人的工作是重复的。

虽然大家都在编程，但是编程的内容是不一样的。每个人昨天和今天的工作也是不一样的，没有人会写一模一样的代码，我们总是不停地写新的东西，做新的工作。这些东西是非常不同的，软件工程从事的是创造性的工作。

大家都知道创造是很难的，创造意味着会有大量的试错，因为我们没有做过。大部分软件的形成都是一项极其复杂的工程，它们远比传统的工程复杂得多，无论是涉及的人力、时间还是业务的变数都要多很多。这些都会导致软件工程有非常大的不确定性。

其二，快速变化。建筑工程在完工以后就结束了，基本上很少会进行变更。但在软件工程里，软件生产出来只是开始。只要软件还在服务客户中，程序员们的创造过程就不会停止，软件系统仍然持续迭代更新，以便形成更好的市场竞争力。

这些都与传统建筑工程的模式大相径庭。一幢建筑自它完成之后，所有的变化便主要集中在一些软装的细节上，很少会再发生剧烈的变动，更不会持续地发生变动。但软件却不是这样，它从诞生之初到其生命周期结束，自始至终都在迭代变化，从未停止。

以上这两点都会导致软件工程区别于传统意义上的所有工程，有非常强的管理难度。过去那么多年，工业界有非常多的工程实践，但是所有的工程实践对软件工程来说都是不适用的，因为二者有很大的不一样。

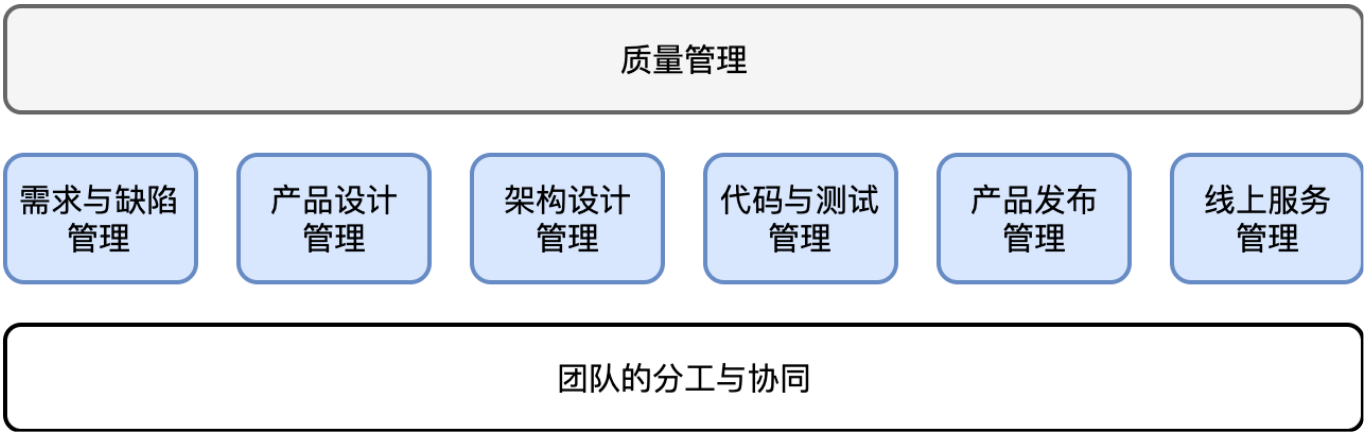
今天如果我们站在管理的视角再看软件工程的话，我们知道管理学谈的是确定性。管理学本身的目的之一就是要抑制不确定性，产生确定性。

比如，开发工期、时间成本是否能确定。比如，人力成本、研发成本以及后期运维的成本是否能确定。

所以，软件项目的管理又期望达到确定性。但软件工程本身是快速变化的，是不确定的。这就是软件工程本身的矛盾。我们的目标是在大量的不确定性中找到确定性，这其实就是软件工程最核心的点。

架构师的职责

如果用 “瀑布模型” 的方式来表达，现代软件工程的全过程大体如下：



从开始的需求与历史版本缺陷，到新版本的产品设计，到架构设计，到编码与测试，到最终的产品发布，到线上服务的持续维护。

贯穿整个工程始终的，还有不变的团队分工与协同，以及不变的质量管理。

更为重要的是，这个过程并不是只发生一遍，而是终其生命周期过程中，反复迭代演进。

它是一个生命周期往往以数年甚至数十年计的工程。对于传统工程，我们往往也把一个工程称为项目，项目工程。但软件工程不同，虽然我们平常也有项目的概念，但软件工程并不是

一个项目，而是无数个项目。每个项目只是软件工程的一个里程碑（Milestone）。

所以，光靠把控软件工程师的水平，依赖他们自觉保障工程质量，是远远不够的。软件工程是一项非常复杂的系统工程，**它需要依赖一个能够掌控整个工程全局的团队，来规划和引导整个系统的演变过程。这个团队就是架构师团队。**

软件架构师的职责，并不单单是我们通常理解的，对软件系统进行边界划分和模块规格的定义。从根本目标来说，软件架构师要对软件工程的执行结果负责，这包括：按时按质进行软件的迭代和发布、敏捷地响应需求变更、防范软件质量风险（避免发生软件质量事故）、降低迭代维护成本。

因此，虽然架构师的确是一个技术岗，但是架构师干的事情，并不是那么纯技术。

首先是用户需求的解读。怎么提升需求分析能力，尤其是需求演进的预判能力？它无关技术，关键是心态，心里得装着用户。除了需要 “在心里对需求反复推敲” 的严谨态度外，对用户反馈的尊重之心也至关重要。

其次是产品设计。产品边界的确立过程虽然是产品经理主导，但是架构师理应深度参与其中。原因在于，产品功能的开放性设计不是一个纯粹的用户需求问题，它通常涉及技术方案的探讨。因此，产品边界的确立不是一个纯需求，也不是一个纯技术，而是两者合而为一的过程。

以上两点，是架构本身的专业性带来的，在前面五章中已经谈过很多，我们这里不再展开。在本章中，我们更多是从工程本身出发。这些话题是因软件工程的工程性而来，属于工程管理的范畴，但它们却又通常和架构师的工作密不可分。

这里面最为突出但也非常基础的，是贯穿软件工程始终的 “团队分工与协同” 问题、“软件的质量管理” 问题。从 “团队分工与协同” 来说，话题可以是团队的目标共识，也可以是做事方式的默契，各类规范的制定。从 “软件的质量管理” 来说，话题可能涉及软件的版本发布，质量保障的过程体系等等。

从更宏观的视角看，我们还涉及人力资源规划的问题。什么东西应该外包出去，包给谁？软件版本的计划是什么样的，哪些功能先做，哪些功能后做？

看起来，这些似乎和架构师的 “本职工作” 不那么直接相关。但是如果你认同架构师的职责是 “对软件工程的执行结果负责”，那么就能够理解为什么你需要去关注这些内容。

结语

软件工程本身是一个非常新兴、非常复杂的话题。可能需要再花费 50 年这样漫长的时间才能形成更清晰的认知（例如，我们第四章 “服务治理篇” 专门探讨了现代软件工程全过程最后一个环节 “线上服务管理” 这个话题）。

作为架构课的一部分，这一章我们将主要精选部分与架构师的工作关系密切的话题来进行讨论，主要包括：

团队的共识管理；

如何阅读别人的代码；

怎么写设计文档；

发布单元与版本管理；

软件质量管理：单元测试、持续构建与发布；

开源、云服务与外包管理；

软件版本迭代的规划；

软件工程的未来。

如果你对今天的内容有什么思考与解读，欢迎给我留言，我们一起讨论。下一讲我们的话题是 “团队的共识管理”。

如果你觉得有所收获，也欢迎把文章分享给你的朋友。感谢你的收听，我们下期再见。

上一篇　67 | 架构思维篇：回顾与总结

精选留言 (7)

写留言

milley
2019-12-27

非常期待后面的课程，特别是如何阅读别人的代码

展开

1

Charles
2019-12-27

许老师，如你所说产品设计阶段架构师参与其中，虽然有比较明确的职责，产品负责需求边界、架构负责技术方案，决策的时候有冲突了，这个时候依靠什么去决策？七牛一般怎么做？盼复，谢谢

作者回复: 还是产品经理决策。如果还有疑义，可以找双方共同的上级。

py
2019-12-27

项目管理有去评判一个项目成不成功狭义来讲是项目有没有在限定的时间、资源把项目成功交付， 广义的来讲是项目客户满不满意、项目的成果有没有被应用并产生预期效果。 视

角不一样 境界完全不一样，架构师也一样

展开

作者回复: 👍

诗泽
2019-12-27

许老师后续是否可以讲一下ai的浪潮下对做工程的同学以及架构师们带来的改变有哪些

作者回复: ai 背后更大的浪潮，是马云说的 DT（数据科技）

leslie
2019-12-27

借用一个老师之前课程中的比喻“架构与建筑”：一路学习、一路强化、一路领悟，软件工程与建筑工程的特性不知不觉中发现确实可以相互解释。前几天一群人聊天问架构是干什么的有些事情为何那么做，边上一个资深架构师解释了一堆没解释清楚，我直接用建筑去解释，然后对方立刻明白了。

一个很生活化的例子和解释反而让我去明白老师开课时所说的“架构与建筑”：当我...

展开

Aaron Cheung
2019-12-27

期待软件工程的未来 在老师说的很多基础服务固化之后 后端工程师还能在何处进行发展

丁丁历险记
2019-12-27

其二，快速变化。建筑工程在完工以后就结束了，基本上很少会进行变更。但在软件工程里，软件生产出来只是开始。只要软件还在服务客户中，程序员们的创造过程就不会停止，软件系统仍然持续迭代更新，以便形成更好的市场竞争力。

这一段语音读错了，修改一下。

展开

作者回复: 收到