# PRD2018-陈维-第二次作业

“杀死一个程序员只要改三次需求”虽然只是一句调侃，从其他的角度来看我认为能引人深思。作为刚入门的我，显然是不能理解这句玩笑话背后的沉重。没有深入的实战参与其实什么看法都是空中楼阁。尽管煞有介事，但我还是希望能够做一些粗浅的分析。

我了解到，从“夕阳”的瀑布模型到现在推崇的”敏捷开发”实际上就是人们对需求不明确或需求改动频率较大的一种担忧所促发而成的一类软件开发模式。

为什么？

不妨就以传统的瀑布模型为切入点作为我对”杀死程序员“的思考：

瀑布模型的优点非常具有逻辑性——进度可视。在开工之前就把全程可能的所有事务都设计好分析好。但是人是有局限性的，无论是用户还是开发者。当用户或者开发者的一些好的创意在开发过程或收尾阶段产生而不是在初始就已经备案，事实上是相当危险的“灵感”。没有什么能比在装修房子时突然要求再加装一个“伊甸园”更让设计师头疼的事情了。这是就是人的局限——我们无法预知未来。

“谁知道这个功能真他妈的难用?”。当客户的行为“朝秦暮楚“(我认为:好的想法和好的改进方案是贯穿整个开发过程的。)让人头疼，譬如加装一个”伊甸园”(加个功能所造成的后果还是可以接受的)。但我们都可以预见到，实际情况是：用户很大几率是想把已经打好地基的楼中楼做成别墅。虽然我不太懂建筑土木，但比喻是恰当的。无可奈何的设计师建筑师，只能硬着头皮再加班加点”迎难而上”。 当你费劲地脱下紧致的西装（也就是旧需求）送去洗衣店，很可能你的打火机或者定情戒指也在里边。牵一发而动全身，后果我想是不可估量的。

当改一次需求，造成开发人员再次通宵达旦为项目疲于奔命之时，他们很可能回不了家，巨大压力下开始掉头发，夫妻关系也出现裂痕。当试图改第二次时，可能发生的“家破人亡”会不会逼迫程序员产生反社会或者自残行为呢？这很难说，但是我觉得还是事不过三，不要有第三次了。