《软件项目管理》读书笔记

——读《软件架构实践》有感

姓名：陈科宇　　学号：2022141461064

一、起因

读这本书的起因原本是因为这学期正好学选了学院的“软件架构与设计”这门课程，而这本书就是这门课程的教材，所以我也同样将这本书选为本门课的课外阅读书籍。

二、主要内容回顾

首先我先回顾一下这本书的主要内容

最初，这本书先介绍了“什么是软件架构”。总的来说，软件架构就是将软件系统抽象化后用来对软件进行设计的一组软件结构。而结构是由关系维系再一起的一组元素。软件的结构大致分为三类。1.组件及连接器结构 2.模块结构 3.分配结构。分别用于关注软件的不同方面。

理解了什么是软件架构之后，这本书又继续讲到软件架构的重要性。也就是我们可以通过一个软件的架构对其进行质量评估、设计以及决策等操作，使得我们可以从一个更加全局的视角构建我们的软件，从而实现风险的提前规避和软件的高效构建。其中作者还提到了一个用于量化软件系统各方面质量的标准，也就是“软件的质量属性”。

然后是质量属性的定义。质量属性是系统的一种可测量或可测试属性，用于表明系统在基本功能之外满足利益相关者需求的程度。根据质量属性，设计者可以明确自己设计的软件系统需要实现的在基本功能之外的目标。接着作者说明了质量属性的表示方法，然后又提到我们用来实现某种质量属性的方法叫做战术，而针对一系列典型质量属性的、被反复验证过的、成熟的架构设计方案被称作架构模式。了解了质量属性和战术之后，可以说，软件架构的设计就是根据系统的需求，分析出系统所需要的质量属性，然后选择或设计相应战术和架构模式，构建出符合需求的软件结构的过程。

接着就是对各种常见软件质量属性的介绍。比较典型的有：

1.可用性：描述软件系统的可靠性以及面对故障时屏蔽或者修复故障的能力。

2.可修改性：指软件面对时刻发生的变更所导致的成本和风险是否足够地低（也就是软件是否容易被修改）。

3.性能：指软件对于可能发生地事件所作出的 相应是否足够地快。

学习完一些基本的质量属性之后，本书接着介绍了一些成熟的系统。比如空中交通管制系统、云和分布式计算，以及它们所体现的一些针对质量属性的架构设计。然后也介

绍了一些不同类型软件结构下的成熟的架构模式，比如 Broker、Pipe&Filter、P2P 模式等。还有一些常用的设计模式，比如工厂、单例、命令模式等。

最后本书讲到如何进行完整的一次软件架构设计。书中主要介绍了 ADD 方法，也就是属性（Attributes）驱动（Drives）设计（Design）。总的步骤是：

1.确认输入的属性是可用的。

2.聚焦于某个特定的属性设立设计目标。

3.对系统进行分解细化，筛选出其中与目标有关的部分。

4.选择合适的设计理念。

5.实例化架构元素，分配职责，定义接口。

6.制作相应的设计图和相关文档。

7.执行设计，并进行分析，检查目标实现情况。

8.必要的时候进行迭代（重复 2~7）。

总之。可以看到，本书内容详实，从软件架构的定义、意义、目标、方法、模式到步骤和评估，覆盖了软件架构设计的方方面面。下面我谈一下学完本书后的感受。

三、阅读后感受

首先，在学习软件架构设计之前，我也做过不少的项目，但是最初大多是没有设计就直接上手编码的课程作业，这就导致我不得不一边编写一边思考，然后一边修改，使得软件构件效率低下的同时还往往会不断妥协，导致软件并不能满足之前设想的种种需求，使得软件质量总是不尽如人意。

后来经验稍微多了一些后，开始有了做软件项目之前要进行设计的意识，但是也仅仅是局限于绘制相关的类图，对代码的组织结构、接口和算法进行预先的设计。这样虽然使得编码的效率提高了，在编码时的思考和修改也减少了，不过最终也只能实现基本功能的实现，并不能做到更高层面的质量提升。

因此，在读完这本书后，给我印象最深的就是书中提到的种种质量属性，尤其是可用性。因为在这之前，虽然我常常也会为软件因为种种原因异常崩溃或不能使用而感到苦恼，但是我并没有想到这些其实也是架构设计的范畴之内。可以说，读了这本书之后，我对软件架构设计的理解变得更深更宽了。

其次，这本书不仅告诉了我软件架构设计应该是什么样子的，更是告诉了我一套完整的进行架构设计的方法，那就是 ADD（质量驱动设计），在这之前，我全都是采用的“功能驱动设计”，也就是只为方便实现功能而设计，如今才发现，这就是我做的项目只能实现基本功能而不能有更高质量的原因。在了解了 ADD 这种架构设计方法之后，我确信

我会在更高层次的角度审视软件系统，从而做出更高质量的软件架构设计。

最后，在本书提到的针对各种质量属性的战术和各种架构模式、设计模式种，不仅有使我大开眼界的可用性战术、Pipe&Filter 架构模式和命令设计模式，还有我以前就见过的“高内聚、低耦合、延迟绑定”，“P2P 架构模式”，“单例模式和工厂模式”。不过特别的地方在于，通过系统性的学习，我更是对之前的“老知识”有了更加清晰的梳理，使得我学过的知识得到了更加细致的整合。

总之，我认为，在读这本书之前，我最多只能称得上是编写程序“程序员”，而读了这本书，有了对软件架构设计有了系统性的整理之后，才有能算的上“软件工程师”的资格。可见这本书背后体现的知识对于软件学院学生的重要性。

四、个人不足之处

读了这本书虽然确实让我受益匪浅，不过仍有不足的地方是，我还非常欠缺相应的实践，导致我对书上提到的概念和方法还处于一知半解的状态。

不过这是一个不错的起点，在对软件架构设计有了一个初步的了解之后，再通过我不断地在后续的项目种进行实践，最后就应该可以实现对知识的融会贯通。可以说，这本书给我后面的成长提供了一个目标。

以上就是我们目前在读完这本书之后的感受在梳理后所汇成的读书笔记。