第 22 章 结论

这本书只针对一件事：复杂性。处理复杂性是软件设计中最重要的挑战。这也是为什么系统难以构建和维护的原因，而且复杂的系统通常运行也很缓慢。在本书中，我试图描述导致复杂性的根本原因，例如依赖性和模糊性。我已经讨论了可以帮助您识别不必要的复杂性的危险信号，例如信息泄露、不必要的错误情况或名称过于笼统。我提出了一些通用的想法，可以用来创建更简单的软件系统，例如，努力创建更深和更通用的类、通过定义来规避错误以及将接口文档与实现文档分离。最后，我讨论了产生简单设计所需的投资思维。

所有这些建议的缺点是它们会在项目的早期阶段需要额外的工作量。此外，如果您不习惯于思考设计问题，那么当您在学习良好的设计技巧时，您的速度还会更慢一些。如果对您而言唯一重要的事情让您当前的代码尽快运行，那么思考设计就好像是件苦差事，而这实际上妨碍了您实现真正的目标。

另一方面，如果良好的设计对您来说是重要的目标，那么本书中的思想会让编程变得更有趣。设计是一个令人着迷的谜题：如何用最简单的结构解决特定问题？探索不同的方法很有趣，找到一种既简单又强大的解决方案是一种很棒的感觉。整洁、简单和易理解的设计是一件美事。

此外，您对良好设计的投资将很快获得回报。在项目开始时仔细定义的模块，在您一遍又一遍地重复使用它们时，会节省您的时间。您六个月前编写的清晰文档将为您节省修改代码以添加新功能的时间。花在磨练设计技能上的时间也将有所回报：随着技能和经验的增长，您会发现您可以越来越快地做出好的设计。一旦您掌握了方法，好的设计实际上并不会比草率的设计花费更多的时间。

成为一名优秀设计师的回报是，您可以将大部分时间花在设计阶段，这很有趣。而糟糕的设计师花费大量时间在复杂而脆弱的代码中寻找缺陷。如果您提高了设计技能，不仅可以更快地生产出更高质量的软件，而且软件开发过程也将也会更加愉快。