## 后记：如何学习领域驱动设计

**《领域驱动设计实践》课程后记**

幸好，领域驱动设计（Domain Driven Design，DDD）不是一门容易衰亡的软件方法学。我从 2017 年 11 月写下本课程的第一个字到现在完成整个课程，已有两年多的时间了，好在 DDD 在这两年后依然算是一门“显学”，虽然它之耀眼更多地是在微服务与中台光芒的映衬下。

在这两年多备尝艰辛的写作过程中，我对于 DDD 的理解也在不断地蜕变与升华。当我敲完课程的最后一个字后，不由感叹自己终于可以浮出水面呼吸一口新鲜空气了；可是，隐隐又有一种意犹未尽的感觉。这或许囿于自己的学识有限，让课程内容留下不少遗憾的缘故吧！这些遗憾只有留待纸质书的写作去弥补和完善了。所以我还不能放松，在未来时间里，我将从头到尾再次审视课程的内容，以课程内容为蓝本去芜存菁，更有体系地梳理 DDD 的知识，争取打造一本 DDD 的原创精品。我给自己设置的时间期限是半年。如此算来，为了这一本 DDD 书籍，真可以说是“三年磨一剑”了！

从战略到战术，DDD 给出了诸多关于软件架构、设计、建模与编码的方法和模式，以用于应对业务复杂度。然而，许多开发人员对于 DDD 的价值仍然心存疑惑，相反，对于它的难以理解难以学习倒是确信不疑，甚至有人惊呼 DDD 是“反人类的难懂”。这正是现实给了 DDD 沉痛的当头一击啊！

从 2004 年 Eric Evans 出版《领域驱动设计》一书以来，已有十五余载。实事求是说，DDD 的推进与项目落地真的是举步维艰。个中原因，难以说清。DDD 是否真正反人类的难懂可以另说，但它是在反“早期的开发传统”，却是毋庸置疑。这一开发传统就是从实现技术出发，由数据驱动软件设计。软件开发人员往往擅长解决技术难题，却不善于（或者说不愿意）理清复杂的领域逻辑，对领域概念进行抽象。领域建模本身是一个主观思考的结果，这也带来优劣判定的不可衡量。

只要克服对 DDD 的畏难情绪（甚至是反感情绪），其实，DDD 的学习并没有想象的那么困难。最大的挑战在于如何落地？当一个企业或者一个团队希望选择 DDD 帮助他们提升软件设计与开发质量时，他们是否想过：

* 团队有没有专门的业务分析师，或者领域专家？
* 是否组建了特性团队，并以迭代的方式进行开发？
* 是否愿意以可视化的工作坊形式沟通需求，确定统一语言？
* 是否创造了足够的条件让特性团队的所有成员与角色能够面对面地高效沟通？
* 是否愿意为打造高质量的核心领域模型而为成本买单？

这些问题并非 DDD 能解决的，但却是成功实施 DDD 时需要确保的 **场外因素** ！因此，DDD 实施成败的关键，不仅在于 DDD 的本身，还在于企业或团队能力成熟度是否达到了实施 DDD 的要求！这也正是我为何在课程中提出“领域驱动设计能力评估模型（DDD Capability Assesment Model，DCAM）”的原因所在。

我眼中的 DDD 已经超越了软件设计技术的范畴，它更像是一门哲学！何谓“哲学”，可以理解为是对人生、世界乃至宇宙的智慧思考。而 DDD 就是对软件世界的一种思考形式，它提出以抽象的领域模型去反映混乱的现实需求世界，以有序、合规、演进的方式去打造满足业务需求的软件世界，并尽量将技术因素推出这个世界的大气层边界之外。简言之，DDD 是我们观察软件世界的态度！

因此，对于学习 DDD 的开发人员而言，第一重要的不是掌握 DDD 的模式，而是要改变分析思维与设计思维的方式。将这种思维方式运用到软件项目开发过程中，就是我在课程中提到的“领域模型驱动设计”，它的核心内容可以通过层层推进的形式汇集为如下三句话：

* 以 **领域** 为分析建模的驱动力
* 以 **场景** 为设计建模的驱动力
* 以 **任务** 为实现建模的驱动力

如何理解这三句话？

当你在开始领域模型驱动设计时，必须在分析建模阶段抛开实现技术对你的影响，与需求分析人员、测试人员一起单纯针对“领域”进行分析建模，即提炼与抽象领域概念，并以统一语言和模型的形式来表达。在设计建模阶段，围绕着一个完整的“场景”开展设计工作。需求分析人员为“场景”编写用户故事，测试人员为“场景”编写验收标准，开发人员则开始解剖“场景”，将其分解为组合任务与原子任务，然后各自分配给不同的角色构造型。到了实现建模，就针对这些任务定义测试用例，开始测试驱动开发，由内至外到达应用服务时，再将它们集成起来。显然，领域模型驱动设计就是针对领域开展的“合而分分而合”的解构过程。

同时，必须谨记：领域模型驱动设计的基础是限界上下文。在领域驱动设计的战略阶段，同样是一个“合而分分而合”的解构过程：将领域分解为限界上下文，再通过上下文映射联合限界上下文共同实现多个领域场景。

以上内容正是我言犹未尽想要表达的精髓。学习领域驱动设计，就需要抓住 DDD 的根本和精髓。你需要理解什么是限界上下文，它带来的价值是什么；你需要理解如何进行领域建模，统一语言在其中扮演了什么样的角色；你需要理解为何领域驱动设计提倡以领域为驱动力，为什么需要领域专家参与到项目开发中来。提升了对这些内容的认识后，再去学习 DDD 给出的设计模式，学习我在课程中给出的固化设计过程，如场景驱动设计，然后找三两个不曾实施 DDD 的项目，寻两三个实施了 DDD 的项目，相互对比其模型与代码，你绝对会有一种醍醐灌顶的感觉。当然，这些都需要你沉下心来细心体会，认真思考，还需要你广泛涉猎更多软件设计与开发的知识，如此方能打通 DDD 的任督二脉。

至于团队实施 DDD，则不仅在于你个人的 DDD 知识与能力，而在于我前面提及的“场外因素”。企业或团队若期望在项目中实施 DDD，首先需要利用 DCAM 评估一下团队的能力成熟度，再来决策做不做 DDD，怎么做 DDD，并着手培养团队成员的 DDD 能力。《领域驱动设计实践》这门课程可以在一定程度提高读者的 DDD 能力，却无法确保成功实施 DDD 的场外因素。

课程写作结束了。战略篇一共 34 章，15 万 5 千字；战术篇一共 71 章，35 万 1 千字；合计 105 章，共 50 万 6 千余字，加上两篇开篇词与这篇可以称为写后感的后记，共 108 章，算是凑齐了一百零单八将。如此成果也足可慰藉我为之付出的两年多艰辛时光！不过，我的 DDD 征程还未结束，接下来的半年时间，我将和人民邮电出版社异步图书的杨海玲女士合作，重新整理本课程内容，为出版 DDD 原创精品（希望是国内的第一本 DDD 原创图书）而奋斗！至于书名，就暂定为《解构领域驱动设计，Domain Driven Design Explained》。

还没有评论



评论

## 更多资源下载交流请加微信：Morstrong,加入永久会员,网盘更新更快捷！

# 本资源由微信公众号：光明顶一号，提供支持