对于模式的“十大误解”

译者语：现在“模式”这个词真是非常流行。就象任何流行的东西一样，对它的误解也真是不少。甚至在一些发表出来的文章中，也存在着各种各样的误解，我想这会对读者造成非常糟糕的引导作用。早已想写一篇文章来澄清一些对模式的误解，却又因为水平所限难以成文。恰在此时，我看到 John Vlissides 先生的《十大误解》，于是我便乐得当文抄公了。

关于设计模式，下面有十种错误的观点——很多都是很流行的观点。且看 Vlissides 先生如何拨开这些迷雾。

——

最近，围绕着模式的讨论日嚣尘上，人们对模式的混淆、惊惶和以讹传讹已经不是一点半点。这也从一个侧面反映出对于主流软件开发者来说，模式是一个多么新鲜的领域——尽管严格说来，它并不是一个新的领域。同时，模式也是一个飞速发展的领域，留下了大片的空白。而且，没错，我们这些模式的支持者也应该受批评，因为我们没有把自己的知识完完全全的教给别人。尽管我们这样想过，但是却没有这样做[BMR+96, Coplien96, CS95, GoF95, MRB98, VCK96]。

因此，我觉得自己有责任来纠正一些模式方面越来越夸张的误解——我常常听到的那些误解都足以形成自己的模式了。甚至连我也曾经想以模式来组成自己的软件结构……后来我才明白：“把所有的东西都变成模式”，这种想法本身就是错误的！总之，请记住，这篇文章不是在给模式社群的人们看的。我想，绝大多数的模式专家都会同意：下面这些都是很常见的误解。但是他们也许不会同意我消除这些误解的方式。

这几年以来，我听到过许多人谈论模式。把听到的这些东西整理一下，对模式的误解大概有三类：对于“模式是什么”的误解，对于“模式可以做什么”的误解，以及对于支持模式的人群的误解。我的“十大误解”都落入这三类之中，所以我就把这些误解都组织起来。首先，我们来看看关于“模式是什么”的误解。

误解之一：“模式是某种场景下某个问题的解决方案”

这个定义来自于 Christopher Alexander[AIS+77]，所以如果把它称为“误解”，也许有人会把我目为异端。但是下面这个简单的反例就能让你看清它的缺点：

问题：我获奖的奖券就快过期了，我怎样拿到奖金？

场景：离截止时间只有一小时，可是我家的小狗把奖券给吞了。

解决方案：把狗开膛破肚，掏出奖券，然后飞奔到最近的兑奖地点。

这是“某种场景下某个问题的解决方案”，但它不是模式。还缺了什么？至少缺三样东西：

1. 可重复性。解决方案应该对应于外部的场景。
2. 可传授性。一个解决方案应该可以移植到问题的不同情况上。（绝大多数模式的可传授性都建立在“约束”和“效果”的基础上。）
3. 用来表示这个模式的名称。

确实，很难找到一个令人满意的定义。“模式讨论”邮件列表（patterns-discussion@cs.uiuc.edu）上正在进行的讨论也证明了这一点。难以定义的一大原因是：模式既是一个事物，也是对类似事物的描述。要区分这两者，有一种办法：“模式”这个术语只用来指代模式的描述，同时用“模式实例”来指代模式的具体应用。

但是，术语的定义很可能是白费力气，因为一个定义可能对一个人（比如程序员）有意义，但是对另一个人（比如只能看到书面材料的项目主管）却毫无意义。当然，我不打算在这里提出什么最终定义。但是，任何对模式要素的规定，除了必须包括问题、解决方案和场景之外，都必须提及可重复性、可传授性和名称。

误解之二：“模式就是行话、规则、编程技巧、数据结构……”

我通常把这些误解统称为“蔑视”。如果你试图把某些不熟悉的东西简化为熟悉的东西，产生这种想法是很自然的，尤其是当你没有特别的兴趣去钻研这些不熟悉的东西时。另外，某些人经常拿新瓶装陈酒，然后大吹“创新”、“革命”一类的口号。保持警惕也是好的。

但是，这种蔑视通常不是来自亲身体验，而是来自肤浅的认识和一点点冷嘲热讽的。而且，没有什么东西是真正“全新”的。人们的脑海中一直都存在着各种各样的模式，只不过我们现在刚开始为模式命名、将模式记录下来。

来逐个说明这些看法：的确存在着模式的行话，例如“模式”这个词，例如“约束”，例如 Alexander 的“无名质量”，等等。但是模式是不能简化为行话的。与其他计算机科学领域相比，模式引入的新术语实在是少得可怜。对于听众来说，好的模式本来就很容易接受。在说明一个模式的时候也许有必要引用问题领域的行话，但是几乎没有必要使用什么特定于模式领域的术语。

没有哪个模式是能让你不假思索就使用的规则（组件则正好相反），同时模式也不仅仅是“编程技巧”，尽管模式中的“方言（idiom）”部分是特定于语言的。在我听来，“技巧”这个词带有轻蔑的意思，并且过分强调解决方案，而忽视了问题、场景和名称的重要性。

毫无疑问，你也曾经听说过一次创新被接受的三个阶段：首先，它被当作垃圾扔在一边；然后，人们会认为它不具可实施性；最后，大家都明白它的意义时，它也没有意义了——“我们一直都这样做”。现在，模式还没有完全走出第一个阶段。

误解之三：“理解一个，就理解了全部”

一刀切的规则往往是不公平的，而人们对模式却更加一刀切。模式的领域、内容、范围、形式、质量……这个领域大得吓人，只需要随手翻翻 PLoP 的书[CS95, MRB98, VCK96]就能看出。不同的人写出不同的模式，而模式的变化甚至比作者还要多。象 Alistair Cockburn、Jim Coplien、Neil Harrison 和 Ralph Johnson 这样的作者已经超越了最初大量记录不同领域、不同形式模式的阶段。只看几个例子就对模式做一个笼统的结论，这样的结论必定是错误的。

误解之四：“模式需要有工具或方法学的支持才会有效”

在过去的五年中，我记录、使用并且帮助其他人使用模式，还至少帮助完成了一个基于模式的工具[BFV+96]，所以我可以很有把握的说：模式的绝大多数好处都来自于原封不动的使用——也就是说，没有任何形式的外部支持。

在谈到这个主题的时候，我通常会指出模式带来的四大好处：

1. 它们记录了专家的经验，并且让非专家也能理解。
2. 它们的名称构成了一份词汇表，帮助开发者更好的交流。
3. 它们帮助人们更快的理解一个系统——只要这个系统是用模式的方式描述的。
4. 它们使系统的重组变得更容易，不管原来的系统是否以模式的方式设计的。

过去，我一直认为第一项是模式最大的好处。现在我认识到，第二条起码也同样重要。请想一想，在一个开发项目的过程中，有多少字节的信息在开发者之间流动（包括口头的和电子的）？我猜就算没有 1G 也有好几兆。（在推出了《设计模式》之后，我已经收到了好几十兆给 GoF 的电子邮件。而且我们所描述的还都是小型到中型的软件开发项目。）由于有如此之大的信息交流量，所以效率上任何微小的提升都能大量节约时间。在这个意义上，模式拓宽了人们交流的带宽。我对第三、四条的评价也在逐渐提高，特别是在项目越来越大、软件生存周期越来越长的今天。

至少在短期内，模式主要存在于大脑中，而不存在于工具中。如果有了方法学或自动化工具的支持，应该还有其他的收益，但是我相信这些都只是蛋糕上的奶油，而不是蛋糕本身，甚至都不能算蛋糕的一层。

* * *

到目前为止，我说到的误解都是关于“模式是什么”的。现在来看看关于“模式可以做什么”的误解。这里有两种不同的风格：夸大其词的和心存疑虑的。

误解之五：“模式可以保证软件的复用性、更高的生产力、世界的和平……”

道理很简单，模式什么都不保证。它们甚至有可能无法给你带来利益。在创造新事物的过程中，模式无法取代人的位置。它们只是带来一种希望，有可能让一个缺乏经验的、甚至是未入门的，但是有能力、有创造性的人尽快获得设计的能力。

人们经常说：好的模式会让读者有“啊！”的回应。实际上，只有当模式恰好拨动了读者的某一根心弦时，他们才会有这样的反应。如果没有，模式就只象森林里普通的一棵树。因此，什么是好的模式？不管它写得有多好，如果不能引起读者的共鸣，它又怎能算是好的模式？

模式只是开发者的工具箱中的另一件工具，对模式寄以过高的期望只会适得其反。准备充分，然后做最坏的打算——这就是防止受骗、消除对抗最好的方法。

误解之六：“模式可以‘生成’整个软件体系”

这个误解与上一个有点相似，不过侵略性稍微少些。

每过一段时间，模式论坛上就会讨论一次模式的生成能力。按照我的理解，“生成能力”是指模式创建最终行为的能力。很多人认为，模式可以帮助读者解决模式本身没有明确提出的问题。我读过的一些书里也有同样的观点。

对于我来说，“生成能力”的关键是模式的可传授性——约束及其解决，或者对于效果的讨论。在你定义、精炼一个体系结构时，这些观察是特别有用的。但是模式本身并不制造任何东西——任何东西都是由人来制造的。而且，模式不可能覆盖体系结构的每个方面。给我任何一个有实际价值的设计，我都能找出其中模式没有涉及的设计问题。也许这些问题不常见、不具可重复性，或者只是还没有以模式的形式记录下来。不管怎样，你都必须负责填补模式之间的空白——用你自己的创造性。

误解之七：“模式是针对（面向对象）设计或实现的”

另一个极端则是过分的限制，就象这个误解。坦白说，任何人都完全可能相信它，我不会有丝毫惊讶。无数的人问过我这个问题，所以它也进入了“十大误解”之列。如果你觉得这种观点实在天真幼稚，跳过这一部分吧。

如果模式不能记述专家的经验，那它们就什么都不是。对于模式的作者来说，任何形式的专家经验都是可记载的。当然，在面向对象软件设计中有值得记载的经验，但是在非面向对象设计和分析、维护、测试、文档、组织结构……这些方面也同样有值得记载的经验。现在，不同领域中的模式开始浮出水面。至少已经有两本关于分析模式的书[Fowler97, Hay96]，并且每次 PLoP 大会都会收到新的模式类型。（特别有趣的是 1996 年的 PLoP 大会甚至关注了音乐作曲的模式！）

和大多数的误解一样，这个误解也是有原因的。看看人们用来描述模式的格式，有两种基本的风格：描述高层结构的 GoF 风格（用于《设计模式》中）；接近文学的 Christopher Alexander 风格——叙述性的，尽量少的结构图。由于已经用模式描述了面向对象设计之外的东西，所以我现在认识到 GoF 风格造成的偏差。对于我研究过的某些领域的专家经验，这种风格根本无法描述。为什么结构图看上去总是让人联想到 C++？对于音乐作曲的模式，“实现”应该是什么？“协作”部分真的有意义吗？

很明显，一种格式不能适应所有的需求。比较具有普遍意义的是模式的概念：模式是记载并传达专家经验的工具——不论是哪个领域的专家经验。

误解之八：“没有证据表明模式帮助过任何人”

这种观点曾经有一定的说服力，但是现在已经没有了。在 Software-Practice and Experience [Kotula96]这样的杂志上、在 OOPSLA[HJE95, Schmid95]和 ICSE[BCC+96]这样的会议上，人们不断的报告自己从模式得到的利益。Doug Schmidt 已经清楚的说明了在传授计算机科学时使用模式的收益[PD96]。尽管绝大多数的证据都只是定性的，但是据我所知，至少有一个组织得到了一些定量的结论[Prechelt97, PUS97]。

随着时间推进，我们需要更好的掌握使用模式的好处和缺点。尽管最初的反馈已经让我们看到了希望，但是我们还需要更多的实践经验来做全面的估计。但是，如果仅仅因为模式的好处还没有完全证实就拒绝使用模式，那你就真的是个大傻瓜了。

* * *

关于“模式能干什么”的谬论还真不少。下面，最后两种误解不是关于模式本身，而是关于支持使用模式的人们的。

误解之九：“模式社群是精英的小圈子”

我很想知道这种观念到底是从哪里来的。如果说模式社群的人们有什么引人注目的地方，那就是他们之间巨大的差异。从 PLoP 大会的与会者名单就能看出——人们来自世界各地，有大公司的也有小公司的，有分析员、设计者和实现者，有学生也有教授，有大名鼎鼎的作者也有初出茅庐的菜鸟。更让我惊讶的是，竟然有不少与会者都不是计算机科学工作者！这个社群仍然在不停变化，每年的与会者名单都与前一年不同。

如果将模式社群的背景公开出来，别人也许会觉得奇怪：很少有学院派人士。实际上，绝大多数 PLoP 的与会者都是实践者。软件模式早期的推崇者——包括 Kent Beck、Peter Coad 和 Ward Cunningham——都没有学院背景。GoF 中只有一个人（Ralph）是学院派，而且他也是我所认识的最实际的学院派。很明显，模式社群从根本上反对任何可能的宗派主义和精英论。

误解之十：“模式社群是自私的，甚至是阴谋家”

我不止一次的听人指责说：模式最大的用处就是为写模式书籍的人带来了源源不断的收入。他们甚至还暗示模式的发展有着某种邪恶的目的。

胡说八道！

作为 GoF 的一员，我可以肯定的告诉你：对于《设计模式》引起的反响，我们和其他任何人一样吃惊。对于它在 OOPSLA 94 上的初次亮相引起的暴风骤雨，我们也同样没有准备——甚至出版商都没有预料到会有这么大的销量。在整个写作过程中，我们最关心的就是尽量写出一本高质量的书，至于销售上的问题，我们根本没有时间去想。现在“模式”已经成了一个时髦的词汇，不可避免的会有一些人将这个词用来谋取私利。但是，如果你认真读过顶尖的模式作者的作品，你就会感觉到他们共同的目标：将难以获取的经验、最佳的实践、甚至是竞争中的优势——多年实践经验的累积——分享给读者，而且讲解得一清二楚。正是这种帮助读者的热情激励着每一个真诚而踏实的模式作者。如果没有这种热情，作者就会弄巧成拙——这种不负责任的作者往往正是对模式所有误解的根源！