顾客

由于大部分的软件购买行为都是顾客对既有软件购买新版本（repeat business），因此绝大部分的软件公司都必须维持与顾客的长期良好关系，然而这并不容易。我们不讨论买过一次就了事的那种软件购买行为。基本上，身处在这快速进步的信息科技之中，顾客总会需要再去更新软件的。这种顾客-供货商之间的“持续关系”(ongoingness)，在软件业中特别密切、复杂，也特别重要。

在软件业中的顾客-供货商关系之所以复杂到惊人的程度，主要是因为顾客花很长的时间在使用软件，而且研究得很深入。就拿我的Visual C++来说，我敢说我的顾客每天与它相处的时间超过8小时，而且我的顾客待在虚拟世界的时间也比在现实世界要长。他与我的产品相处的时间超过他与家人相处的时间，这是很值得注意的事情！

第二个使顾客-供货商关系非常复杂的原因是：软件不只是耗费他的时间，还对他的潜力造成限制。软件为了做到自动化和执行效率，必须事先定义很多的限制条件，而这也就同时限制了使用者的表现自由。

被软件俘虏的顾客

如果顾客迷恋你的产品，请对以下的讨论特别

注意。你的顾客选择你的软件，是基于心理和情绪性的因素，而不是基于做事的需要。顾客购买你的产品时，好像跟自家人买东西一样，就像是对偶像的崇拜一般。这时候你得特别小心，这种顾客－供货商关系特别容易出问题。

顾客是非理性地选择你，并没有事先比较过其他的产品。一旦热情消退，他就会觉得挫折和憎恨，而且会不公平地感受你的软件的优缺点。你必须补偿他们因为缺乏选择所造成的遗憾；你必须让他们心甘情愿地选择你。即便现在是被你的软件俘虏，但不是已经误上贼船，不得不选择你。在这种情况下，一开始你是处于负面情况，你得让顾客觉得不后悔他的决定。

软件的一切动作都是经过设计的（多多少少都找过专家咨询）。设计者是找出最正常的程序，归纳出一组标准动作，然后选择要自动化的功能，限定它的输入和输出应该遵循什么规则，应该在什么机器上执行，要用什么软件来做。设计者的目标使用者是大多数的普通人，设计者希望能让最多的人获得最大的满足。这种做法是没错，但却令少数人觉得受到挫折。

虽然使用者花在软件上的时间那么多，但他对设计的影响力却少得可怜。软件的开发就像是黑箱作业，有一大堆的信息放进去，建筑师（产品设计师）从其中找出人数最多数的类型，考虑其住房需求，而不直接与其中的住户对谈。这种产品是为了服务市场，而不是服务顾客，这种做法实际上剥夺了个人无穷的潜力。

这种不平衡是必要的，但确实使得客商关系复杂化。没错，每个人都可以说，软件公司不去开发足够的软件工具（也许是技术和时间因素），而专注在开发自己的应用软件，软件公司总是做一个自己认为的典型，却不让顾客有使用上的自由。自制软件的变化性，对照于市售软件的笨拙，使个人更增添挫折感。虽然每个人都喜欢以自己的方式做事，但大部分的软件都要求使用者用特别的方式才能满足一点个性。好吧，只要有办法做得到，使用者勉强觉得比较满意。应用软件厂商当然可以对于这一点加强努力，让那些不满足设定标准的人，有自己的弹性空间。

没有它万万不行，但有了它我还是
不能做我想做的事。

然而，因为个人化的差异而在软件工具或应用软件中加入多样化的组态，也可能反过来使软件复杂到无法被大多数人接受的地步。无法接受的复杂度不一定是人工操作组态所造成，但二者常常伴随发生。因为组态是很难设定的，人为的尝试设定常常失败。说得更具体些，就是提供弹性的结果，通常是让很多人在有意或无意间碰到了组态的问题，然后搞得是一团乱，大家都不满意。“容易使用”和“便于修改”两者之间在本质上就存在着冲突和对立。如果你能舒缓这个问题，就可以满足更大范围的顾客需求。让软件容易使用是一大学问，而要让软件能够动态运用更是难上加难。如果你能让它变得简单些，就很容易吸引那些“改变需求就是我的需求”的顾客。

在这个时间和资源有限的世界中，软件厂商必须选择适当的功能特色来开发，如此就必然限制了产品能够给予使用者挥洒的空间。愈来愈多的使用者对软件上的限制觉得懊恼，也许可以应付每天例行性的工作，但绝不能算是得心应手。软件使用者最后只能很痛苦地迁就现实，每天都在用，这么拘束难受，但不靠它根本行不通。就好像每次穿鞋时都用鞋拔硬挤一下把脚塞进去——姑且称它为“鞋拔现象”吧。

我经常以软件开发为题发表演讲，每当谈起软件与顾客的关系时，总是会放一张投影片，上面写着：“大部分的软件都烂毙了”（Most Software Sucks），每一次都使听众大笑不已，最后掌声不断。我不得不承认，现在的软件虽然令人如此不满，但总比没有好，所以才会创造出数十亿美元的财富吧！但是有多少人称赞它呢？

我买软件，也喜欢它们（刚开始的时候），只因为有总比没有好。但是愈用愈觉得恨透了这些限制，比我自己写还不适用。我自己重新安排一下设定，还是不行，于是我很自然地决定更换软件，结果全世界都改变了，这个新软件变得更糟，我却得倚靠这个无名孤儿般的软件。我真是烦死了这些！亲爱的软件厂商们，求你们让我有机会升级吧，没有它我做不成事情，但有了它我还是不能做我想要做的事情。

顾客购买模式

从顾客的角度来看，一个简化的软件购买心理和情绪过程分为几个阶段，大致上是依序的：首先，他们会收到厂商的信息，引起对产品的注意（attentive），然后开始对软件发生兴趣（interested），想对它多了解一些，

最后他相信（convinced）买下它是个正确的决定。当然啦，如果他一开始就对产品信息充耳不闻，他也就不会有后面的心理变化。但即使顾客在心理上已经相信，但他还不一定会买，他一定要对产品有渴望（desire），才会促成购买行为。

你要让顾客渴望你的产品，从一开始设计时就必须考虑这一点。想象一下你的软件有什么特质“properties”是能够让使用者觉得满足，把它们具体化一些，你自己先“渴望”它们，把它们鲜明地表现出来。个别的功能可以妥协，但这个中心特质一定要强化。对这些特质思考再思考，回顾一下软件的各个版本对这些特质做到什么程度，设法使它深入到程序的每一个角落，一定要让顾客能够明显感受到这些特质才行。你得让任何人在初次使用时都能体会它，并且说得出这个特质是什么。

软件的美学

让我们在这里稍微离题一下，谈谈美学（esthetics）。大部分的人都常常听到这个字，但是只有很模糊的概念，觉得这大概是有关艺术或设计的一种知识。也许从反面来谈美学（anesthetic，麻醉

之意）会比较容易掌握它的基本精神，那些让我们提不起劲或想睡觉的东西，就不是美的东西。所以，美的东西会让人的精神为之一振，感觉变得敏锐，或是神清气爽。

什么东西会使人振奋呢？答案是容易感受得到的东西，如形状、颜色、声音、动作、和谐感等等，这些特质在无形中都会吸引人们的注意力。设计时加入了美感的软件，能够让人的心情愉悦，让人在使用时感受比较舒适。

美感的提振作用虽然是纯心理方面的，但会影响到使用者实际的行为。让你的软件创造出美的感受，会有更大的使用价值和市场潜力。

法则 15

Enrapture the customer
给顾客惊喜

大部分的顾客会购买既有软件的新版本，经过一段长时间与软件相处，他知道这个软件一切的优缺点。而且由于行销部门不断地宣传你的产品和服务，市场非常了解你的公司。总而言之，你的顾客认识你。

如果顾客对于自己必须倚赖你那不太能满足他的软件，他会感到愈来愈不舒服，他每天的软件生活都是处在“鞋拔现象”中，他一定殷殷盼望你在下一版能够改善并解救他，哪怕鞋子只是宽一点点也好。

成功地符合顾客的期望是和那些怒发冲冠又穷凶极恶的顾客沟通，了解他们对每一版的不满在那里。如果你对顾客忠实，愿意倾听他们的声音，采纳他们的意见，他们就会愿意与你走在一起（虽然他们可能仍怀着愠怒的心情）。譬如说，如果你的系统中有一个执行太麻烦，使得顾客痛苦不堪，那就在下一版把它改善。顾客最低的希望是你能够理解他所感受到的痛苦经验，并且你必须在下一版中表现出来。

顾客最低的希望是你能够理解他所感受到的痛苦经验。

前面所述是正常的成功，伟大的产品当然不止于此，而是应该把技术的轴心放在顾客心底最深处的需要，不是他们最低的期望。你必须用创新的方式，满足顾客那些难以表达的需求，用你的产品告诉他们你深深了解，包括那些萦绕在潜意识里的念头，给他们惊喜和感动，扫除那些令人愤怒的缺点，让软件完美地配合他们的需求。

伟大的软件在市场上会得到证明。伟大的软件一版又一版地淬炼她的完美，你会藉由她告诉顾客你多么了解他们，她会吸引无数顾客的忠心。你会被顾客当成力量的来源，顾客因此而狂喜。

问题自然又来了，你如何知道是什么令顾客困扰？更重要的是，如何了解他的内心深处的困扰？答案很简单，直接去问他们，任何时候任何地方，以任何形式，通过任何渠道。

市场调查和统计数字会告诉你市场的偏好，帮助你决定未来的产品方向，给你明确的数据。然而，如果你想要和顾客建立良好的关系，是比较艺术性的，而不是科学性的，而且你想知道重要的事情，那么以下的讨论会告诉你几个比较容易的方式可以了解顾客。

套句广告词儿，就是“ just do it ”。寄信给你的顾客问

他们喜欢什么或讨厌什么；打电话给他们；在信息展的会场与顾客聊天；组织一个“使用者联谊会”(focus group)。没有一种方式极复杂或很花钱。使用者联谊会只要有个场地，邀请一些不太远的顾客，就可以坐下来谈很多事，你做点笔记整理一下，就可以得到深入的意见。做一次市调，研究一下我们需要知道什么问题，用问卷做调查，就可以得到广泛的意见。

你不必完全照上述方法执行。在现实世界中的软件开发团队，一定对收集到的顾客意见争论不完（到底顾客的实质意义是什么，我们如何将顾客意见落实在软件中等等）。这很正常，也是好现象，可以鼓励他们这么做。你可以继续追踪顾客的需求，记住市场是愈来愈复杂的，了解顾客需求是一件持续性的工作，不是做完就结案的短期任务。

了解顾客对软件的需求是需要技巧、创意和持续不断的努力。你必须认识市场的核心需求，将技术和沟通等资源集中来满足它。了解核心需求对软件产品有下面的效果：

- 产品将带给顾客安全感。
- 产品会让顾客能掌握得更多。

- 如果你的产品在其他方面不如竞争者，但在核心需求上绝不辜负使用者，你的产品还是很有希望会赢。
- 你的产品诉求信息非常清楚。
- 你的产品在使用上更简单易学。

法则 16

Find the sweet spot
寻找靶心

我相信任每一个市场都有它的靶心，靶心就是各方人士价值观的交集。靶心永远随着市场的成长而移动，有些时候还移动得飞快。如果你的产品能够抓住大部分人所认定的重点，也就是正中红心，因此只要发表得当，就一定会大受市场欢迎。

当我们开发Visual C++ 1.0时，大家都很顺理成章地认为C++一定会成为市场主流。因为主要的经销商都把C和C++一起搭售，所以无法在市场的统计数字中得知：纯就C++而言它被市场接纳的程度；我们只能假设，由于这一组产品主要是以C++ 编译器做诉求，因此这个庞大的销售数字代表了C++被大量的顾客采用。

有趣的是，当时的分析师、媒体和顾客都一致告诉我们，市场需要我们把“模板”（template）和“例外处理”（exception handling）放在产品中。这在当时对C++而言还是相当新的功能特色，有非常多的理由显示这是很不容易设计，而且我们在那样的时间限制之下，几乎没办法满足这项要求，而且我也怀疑顾客是不是真的需要我们做这些。我不断问自己：“为什么我要把这么复杂的特色纳入产品中，而我敢打赌几乎没有人会用到这么复杂的功能？”事实上这正是行销的理论基础，追求独特，如此才能摆脱传

统的束缚。

传统思维的束缚

> 在收集到的市场情报中，找出奇特或不甚合理的地方，由此切入，摆脱传统思维的束缚。

我们做了一些研究，结果是只有不到15%的人从C转移到C++，虽然有80%的人计划要，或是想要，或是希望

在明年转移。所以我们开始与顾客面对面沟通，企图了解他们真正的需要。通常顾客不会告诉你他真正想要什么，特别是答案与传统背道而驰的时候。因为他们欠缺安全感，他们会告诉你他以为他要的。这也是行销的理论之一：顾客恐惧新科技，而开发人员也一样恐惧新科技。

如果你走进软件开发者的工作室，问他们：“你们之中有多少人学不会C++？”我想一定没有人敢举手站起来大声说：“我！我学不会，它实在太难了。”他们一定会这样说：“哦，我们还在等待模板和例外处理。”或是其他的借口。所以，你必须更深入挖掘顾客内心真正的需要，而不是这种表象式的纯技术方面的功能特色。

真正有效的做法是，私下问他：“是不是在使用C++时遇到了困难呢？”

这时候顾客可能会说出心底话：“是啊，我实在没空学C++，这玩意儿太复杂了，我真的是被它给打败了。我连Windows程序都不是很会，虽然我已经试着读一两本这方面的书，我也看过入门录像带，那太粗浅了，我还是没办法对它有概念；但是所有的人都会这东西，我落后了，怎么办？我真的好害怕。”

法则 17

It's a relationship, not a sale

与顾客建立关系，而不是卖产品

在当时，因为C++的观念太新，显然不太可能用这么复杂的程序语言来把我们的市场整个开展起来。C++实在太难了，顾客不知道该从何着手来学习它，而Windows程序也是，很多人都在为此挣扎。他们觉得真要弄出一个不错的应用软件，非得付出大量的努力不可。

于是本小组中有一位聪明人士，想出了一个聪明点子：精灵（wizard）。现在看来它也没什么了不起，它不是什么仙女棒，只不过是一个可以产生一小段程序代码的按钮罢了。在概念上来说，精灵差不多是我们所做过的功能特色中最容易的（虽然它的基础MFC，相当困难）。但是精灵却是正中靶心，它让无数的程序设计师只要按一下按钮就产生出一堆基础程序代码。

我们把精灵放进产品之后，Visual C++的市场占有率明显窜升了数十个百分点。由Visual C++的例子可以看出来，高深的技术应用并不是重点，重点是帮助你的顾客，找出他们真正的需要，他们的内心在为什么事情挣扎，然后把所有的注意力转向这里。不必管模板和例外处理，不必管其他的高科技名词，只要全心全意开发大多数人真正需要的东西就够了。

顾客需要被了解，虽然他很可能说不出口，你应该体

会他真正的需要，并且以软件产品表现出你对顾客的了解。将技术的轴心放在这里，做出顾客梦寐以求的东西。在Visual C++的例子中，顾客的愿望是：“我想成为C++的程序设计高手，成为年薪9万美元的高级工程师。如果你让我实现这个愿望，而不必学得那么累，我一定花钱买你的产品，并且叫我的亲朋好友一起用你的产品，我一定要得到它！如果公司不买我就自掏腰包，如果我的另一半不同意我就跟她争论到底，我会不惜任何代价买到它。”

“我一定要买这个软件！甚至不惜和老婆吵架,我愿意付出一切代价买它。”

身为软件厂商，一定要牢牢记住顾客需要安全感，要能够控制软件，要软件帮助他事半功倍。顾客不愿意承认他无法跟上新科技，顾客不会说：“你的软件烂毙了。”他会想：“一定是我太笨了。”一般人除非相信自己聪明到足以掌握科技，否则对科技都怀有畏惧和逃避。你不能等着竞争者使顾客聪明。

你和顾客的关系就像是舞伴。你向前一步（推出新产

品和传达新信息），然后顾客响应一步，然后你跳下一步；你必须注意整体发展，而不是刚刚那一步。这就是我在法则3中提到的多版本计划。如果顾客知道你会按时出新版，而且在这漫长的科技旅途中你都会忠实地伴随他，那他就会在下一步时比较能够配合你，跟着你的脚步舞得更流畅优美。

你和顾客的关系也像情侣。如果你忽略他们太久，或是表现出兴趣缺乏或者是拒绝，当然你们的关系就没法子融洽。你的忽视会使顾客觉得受到背叛、伤害，觉得生气，恨不得给你一点报应。所以，一定要对顾客“一路走来，始终如一”。

我一定是太笨了

不久前我在飞机上与一位女士聊天，她问我在哪儿工作。通常人家听到我在微软工作，不外乎两种反应，一是露出很崇拜的样子，一是直接问我认不认识比尔·盖茨。

这位女士问过比尔·盖茨的问题之后，开始和我谈计算机，告诉我她学计算机的计划。她说她要去一间社区大学上课，要学所有的计算机软件，包括数

据库、电子表格和文字处理器。谈到这些，她脸上流露着兴奋的光芒：“计算机实在太棒了，我要学会所有的东西，Windows啦、RAM啦，反正所有的东西，要花几年也无所谓。”

听到这位女士的梦想，我对她面对计算机革命的态度有两种感想。我钦佩她的勇气、决心和远见，她不愿被科技抛在后头，错过这本世纪最重要的文化与技术核心。她愿意投注大量的时间、金钱和努力去学习计算机，不怕承认自己不会计算机的事实。

咦？等等，这位女士真的认为计算机很难，计算机真的那么难吗？她需要回到学校，才能够学会操作PC的软件。而事实上这些软件实在非常地容易。这么说，这位女士一定很笨□罗，还得重新回去做老学生，忍受数不清的挫折，只为了学计算机。嘿！这可真不错，这就是我们软件厂商的大生意，把软件弄得很难表示我们智商高人一等，然后顾客会付我们白花花的银子来学软件。即使我们的软件很烂，顾客也只会觉得自己太笨或计算机太难。

既然你是本书的读者，应该已经具备相当程度的专业知识，而且你的亲朋好友都知道你懂计算机。

请回想一下，你是不是常常遇到有人困窘羞怯地告诉你自己实在是计算机白痴呢？这是相同的现象。大部分的人都觉得不懂计算机一定是自己太笨的缘故，而感到极度不安。人们的错觉是：计算机是对的，自己是错的。

法则 18

Cycle rapidly
加速产品推出的周期

我们已经谈过了准时推出软件产品，兼顾进度和品质。除此之外，你还得加快新版本推出的速度，这么做的理由有几个：第一，无论你做得多快，市场、科技的进步和竞争者都会比你更快。第二，你与顾客之间的关系是维系在你多勤快地与他们沟通，沟通的主要媒介就是产品的新版本。你要告诉顾客的每一件事都包含在软件产品之中：你对顾客的细心体贴、你的热情、你的信念、你对顾客的看法，往往都由软件表现，软件是不会说谎的，她是你（和你的公司）的代言人。

我从来没有听说过有任何的关系会被太多的诚恳沟通而破坏。借着频频推出新版，不只是使你在市场中的曝光率增加，也等于是回馈顾客，拉近与顾客之间的距离。钞票也不会说谎，常常推出新版会使你与顾客的交易机会增加，也使得双方的关系更密切。

Visual C++团队具有在短时间内推出产品的丰富经验，所以我们在这方面做得很成功，以后还要继续。这是我们强大的竞争武器，也表示我们能与市场维持良好的互动关系。现在我们甚至能够用预订的方式卖软件，一年推出三次新版本。我认为这是贩卖软件的最好方式，但是有两个问题：第一是很少有厂商能够保证自己规律而快速地

推出软件；第二是软件更新往往需要厂商付出大量的成本，而且很难管理得当。不过我相信现在的软件厂商愈来愈能够处理这个问题，再加上PC和网络的进步成熟，我认为这种方式会逐渐普及的。

站在顾客的立场，他不一定每一个版本都得使用，他可能已经习惯于旧版本的环境，或是采购预算的限制，或是已经在旧版本上做了一些投资（比方说开发了一些应用软件）。不论理由如何，顾客永远有权说要或不要。顾客需要做自己的技术、预算或时间规划，他需要预估你何时推出新版本，而届时他就可以使用市面上最先进的技术。不幸的是，现在的软件大多是非常不定期地推出新版，而在新版中会有重大的变革或是大幅度的除虫。

经常而准时地推出新版（修改规模比较小），比不定期的一次出个革命版本效果要好得多，而且利润也比较高。开发软件最重要的是对市场能够迅速反应，而不见得要有革命性的突破（responsive but revolutionary）。定期推出新版让你有机会直接响应顾客的需求，同时很快地扫除令顾客烦恼的障碍（也许只是个小瑕疵），而顾客会对你心存感激。我发现消除顾客的烦恼（Annoyance fix）实在是一本万利，通常不需要太多的技术难度，有时候只不过是一

点点使用习惯的问题。这会让顾客相信你会替他解决问题的，大大增加了他的安全感。

我发现顾客比较喜欢新产品摆在他眼前，而不喜欢厂商的承诺，有少部分的顾客是两者都要。但一般来说，持续推出新产品才是对顾客最好的保证。送顾客纪念品只能让他高兴一下，不会让他想买你的软件。

完成很多个小成就，比完成一个大计划，还要受人欢迎。因为这种方式比较容易切合使用者的需要。按时推出新版软件会让你设定比较小的目标，对这个目标更清楚而且容易掌握，比起很大而不确定性甚高的目标来说，也比较好管理。