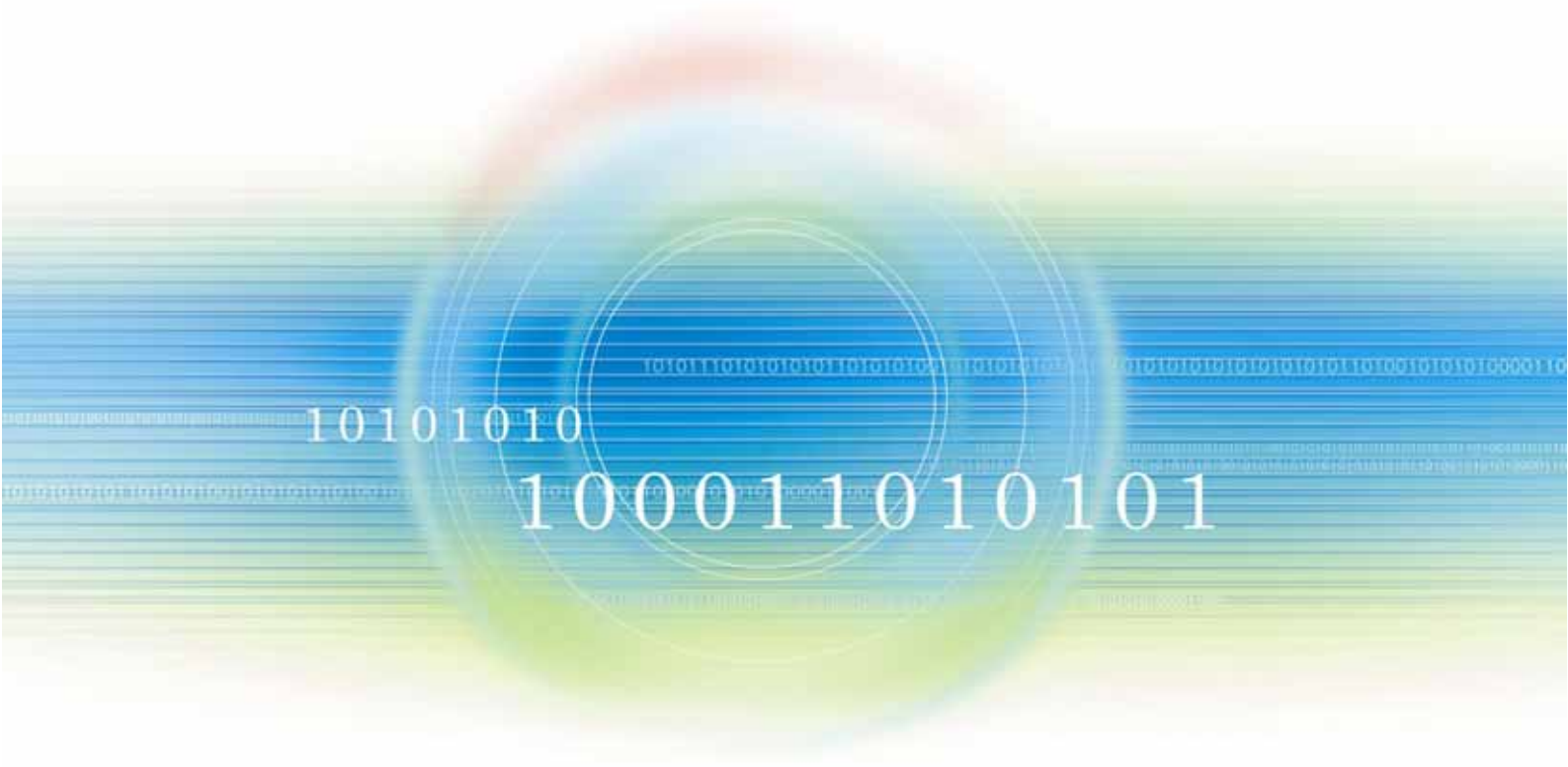
北京华泰科信科技有限公司

项目管理杂志

（第三十四期）



北京华泰科信科技有限公司

Beijing Huatai Information Technology Co., Ltd.

目　　录

发现项目经理

作/转载者：游丽娟

一、发现项目经理的必要性

在美国似乎有一种流行的看法，即如果你擅长做什么事，那么你就可以管理其他人做同样的事。自从 1981 年我进入培训咨询行业以来，曾与数千名被任命到管理工作岗位而未接受过任何管理培训的人交谈过。事实上，我也曾遇到同样的问题，因此我从根本上知道这其中所隐含的假设的谬误。

我个人认为，技能的缺乏是项目失败的主要原因之一。因此，本文的目的是告诉项目经理他们需要什么样的技能，以及如何获得这些技能。

二、项目经理需要的技能

以下是项目经理需要的基本知识领域和/或技能。也许还有其他一些特别针对项目经理工作的技能，但这些技能涉及面很宽。例如，如果你负责承包，你应该知道在当地如何承包工程和如何管理工程的细节。

计划编制	决策
解决问题	冲突管理
设定目标	数据分析
谈判技巧	领导技巧
口头沟通	书面沟通
面谈	指导/建议
团体动力	团队建设
质量功能展开	聆听技巧
全面质量管理	进度计划方法
协同设计	挣值分析
时间管理	

三、项目经理：什么是最重要的？

常常有人问我，什么技能最重要，我总是毫不犹豫地回答：“对人的技巧。”一位不能与人打交道的项目经理将遇到很多麻烦。一个大建筑集团的总裁告诉我，他不得不把他的一位项目经理调到一个不必与人打交道的岗位。这位项目经理懂建筑，也知道如何计划，但他常常使人感到愤怒，因此这位总裁要花费很多时间去平息这些怒火。

正如我所说，在表示项目管理系统的金字塔中，对人的技巧放在最底部，这是因为它们是其他任何事情的基础。可是，大多数组织对这些技巧的价值认识不足。每年，有关计划、进度和控制的三天课程我要教大约 40 次，而有关领导、管理和促进项目组的课程只有大约 4 次。看来，许多公司仍然认为这些“软技巧”没有底线，因此它们不会花钱让自己的员工参加这类培训课程。

我想我还没有遇到过哪个项目因为项目经理不会制作 PERT 进度图而导致失败，我倒是看到很多

项目由于人际问题而陷入严重危机。也许在某一天遇到这种情况时，项目经理们会醒来并认识到人们需要在这一领域的培训，因为这些技能不在学校教授。

四、项目经理的个人特征

过去数年中，我问了许多项目组成员他们对其项目经理期望什么。下面是我收到的反应：

是好的聆听者	相互所有
能提供支持	是稳定组织的缓冲器
有组织性	有领导能力
扫清路障	有技术知识
相互尊敬	公正
是团队建设者	有灵活性
知道自己的局限	虚心
有幽默感	放权
有反馈	诚实/可靠
是好的决策者	理解他人
提供后援	能调动项目组把工作做好
分享经验	知道项目组成员的优点/缺点

显然，任何一个人要满足所列的要求都不容易。我的建议是利用库存资源。要认识到，在你感到比别人弱的方面需要你的项目组成员支持你。

五、思维方式

毫无疑问，谁都听说过思维中的左脑取向和右脑取向。左脑取向的人在思维时比右脑取向的人更有分析性、逻辑性和连续性，这些人的思维更趋向类比、直觉和全面。

内德．赫尔曼（1995）在研究人类如何思维时，发现左右脑分法不足以解释思维的差别，他假设了另外一个基于脑边沿思维的坐标轴。增加这一维度后，出现四个象限，产生四种不同的思维方式。

测量这些思维方式的仪器是赫尔曼大脑优势仪（HBDI）。回答者会收到一个如图所示的轮廓图。在轮廓图中，有 1（最先选择的）～3（最后选择的）的不同打分。没有 0 分，因为每个人都会在某种程度上使用所有四种思维方式。还应注意，该仪器测量偏好，而不是技巧或能力。赫尔曼相信，对于某种思维方式的偏好是基于大脑化学或遗传的。无论这种观点是否正确，偏好的确存在。迄今为止，超过 100 万人做过 HBDI 测试，大多数人认为测试结果教好地代表了他们的思维方式。很少有人说：“那才不是我呢！”

思维方式的HBDI图

有很强 A 象限偏好的人，趋向于较强的逻辑性、分析能力、技术性、数学能力和解决问题的能力。他们常常被那些能够以此方式思维的任务和工作所吸引。工程师、金融分析家等趋于聚集在该象限。

B 象限包含的人一般以组织、行政、保守、克制和计划的方式思维。有这种思维偏好的人经常是经理、行政助理等。

C 象限的思维者倾向于人际关系、感情、音乐、精神和健谈。他们中的很多人成为教师、法律顾问和其他人类服务工作者。

最后，D 象限的人比较喜欢艺术的、整体的、富于想象力的、综合的和概念化的方式。显然，这一象限的人属于创造性的类型：艺术家，雕刻家等等。趋于 AB 偏好的人认为 D 象限的思维者有点“古怪”，不脚踏实地。

有最好的思维轮廓图吗？没有。

有适合某项特定工作的最好的思维轮廓图吗？也许有。

HBDI 是为个人选择职业提供咨询的很好仪器，但它只能在认证的专业人员指导下使用。之后，写出所得结果的意思，思维轮廓图即告完成，但最后要经过专业人员电话或亲自审核。然而，赫尔曼小组特别强调 HBDI 不是作为一台选择仪器设计的，也反对用于此目的，除非该详细记录通过了熟练的心理测量专家的验证。

现在有完善文档记录的有关 HBDI 的应用是在团队中。一个团队应集体表现为一个“整脑”，这意味着假如你将该团队所有成员的轮廓图叠加起来，将形成一个在四个象限都存在偏好的复合轮廓图。假如一个团队“失去平衡”——假如整个团队对四个象限的某个象限强烈厌恶，那么，需要用那一象限进行思考的问题也许就得不到很好解决。

返回目录

项目管理释疑:谁是最理想的项目经理

作者：www.csai.cn

好几次在开会的时候，当遇到一件事情与会者争执不下，主持会议的人作不了决定的时候，通常都以将此事交由项目处理来作结论。但这个项目该交由谁来负责呢？当然是在争论中意见最多的那个人。对议程的进展，我不得不承认这是一个屡试不爽的好办法，但对问题的解决，这种太极拳式的推法，十之八九祇把问题推掉，而不能解决问题。

在项目管理中最常遇到两个问题：

项目的组织和隶属怎么安排最有效？

谁是最理想的项目经理？

在谈第一个问题之前，让我们先回头看看公司的基本组织有哪些。

一开始，公司的组织是按业务的功能不同而采取一种金字塔式的组织。这就是所谓传统式的组织结构。这种组织结构的最大优点在容易控制，对预算和成本的计算也比较精确。但这种垂直式的组织对专案的管理却是一大障碍，因为它缺少一个能对项目负起全责的人。同时，垂直式按功能划分的组织虽然利于上下沟通，但对项目管理视为必要的横向沟通却无能为力。在这种情形之下，很多公司在原有的组织结构下，创造了一些新的头衔，像项目领导人、项目行动小组、专案联络人等等，名目繁多，不一而足。但经验和事实告诉我们，这种治标不治本的改变不是很有用，因为这些改变仍然不能解决两个基本的瓶颈：项目主持人没有实权；项目成员仍然效忠于其原属单位。因为他们都知道，项目的形成本来就有点像“乌合之众”，项目完成后，各人都要回到各人自己的部门。在这种情形下，谁都不会把项目的任务当成“本业”来看待。

和传统式组织相反的另一种结构，那就是不按业务功能，而按产品而形成的组织。换句话说，相同的功能，往往出现在不同的组织中。这种按产品而区隔出来的组织结构，对产品从研发到上市的时间上，当然可以提高不少效率，但在资源和人力的运用上，因不能彼此支援而造成极为浪费的现象。同时，它还是解决不了项目完成后，人员何去何从的问题。我曾经在一家软件公司做过事，那家公司原来祇有一项产品，大家相安无事。后来产品增多了，每个产品的推销，都被视为一件特殊的项目，结果每项产品都有自己的业务和支持干部，产品与产品间彼此竞争，不相往来，造成公司极大的浪费，以致于两败皆伤的局面。

近来流行一种矩阵式的组织。关于矩阵(Matrix)最简单的一种解释，就是在传统垂直式的组织中，一个新的实体(Entity)从侧面横向切过去，这个新设的实体叫做项目。

从表面上看起来，矩阵式的组织和项目管理应该是“天作之合”，因为项目的目的是协调不同的部门，而矩阵式组织的目的是协调公司各部门以支持某项目。但根据许多公司的经验，矩阵式的管理说比做容易，因为每个公司的环境和面对的情况都不一样，所以天下没有两个完全相同的矩阵。

矩阵式组织最大的问题出现在工作流程，它不但要从上到下运作，并且要横向平行运作。在这种繁杂的运作下，项目的管控显然是一大问题。举个划龙舟的例子，来说明矩阵组织画虎不成反类犬的可能性。传统式的组织就像我们端午节看到的划龙舟一样，一个人打鼓，其它的人听鼓声用力划。如果船上人人可以打鼓，人人又要划船，船会往前走才怪哩。

不可否认地，在所有的组织结构中，矩阵式的组织对项目管理最有帮助，因为在管控上，它可以弥补以产品为中心式组织的缺失，在负全责上，它又可以弥补传统式组织的不足。但项目经理究竟应该直接向谁负责呢？换句话说，项目的隶属权究竟如何呢？

我认为：

如果是小公司，项目经理应比照部门主管，直接向总经理或执行副总经理负责。

如果是大公司，可以另外设立一个项目管理部，不同的项目经理向一个部门主管负责，部门主管向总经理或执行副总经理负责。

也可以考虑成立一个项目技术支持中心。在功能和组织上，这种做法和第二种做法不同，技术支持中心祇有协助权没有领导权。成立此中心的主要好处，是将有限的资源做最有效的运用，并且对项目管理人才的培育有很大的帮助。

当然，我们也可以采取将项目必需的技术人员，像工程师、程序师、化学师等人才，直接划归于项目之下，至少在项目进行期间，这些项目必需的技术人才，直接受项目经理的指挥。

我们提到过，矩阵式的管理谈起来容易，但做起来却挺难的。克服此一困难的唯一办法，是对有关人员进行有条理有计划的训练：矩阵式组织如何运作、如何沟通、如何解决问题、如何赏和罚。最重要的还是在矩阵式或项目式的组织结构下，各成员扮演的角色一定要订得很清楚。名不正则言不顺，言不顺则事不成。

至于什么样的人选是项目经理的最佳人选呢？是该选管理技巧比较好的人来领导呢？还是选技术能力强的人来领导？可不可能一个项目有一个以上的项目经理呢？关于这些问题，我相信不同的人有不同的看法，孰是孰非，见仁见智。

我个人认为一个理想的项目经理，他或她的管理技巧和技术能力都不是最重要的，最重要的是经验，哪怕是失败的经验。如果我有权在两个项目经理中选一个来替我做事：一个毫无经验但学识能力都很好，不但能管人，自己也能放下身段来办事；另外一个曾经有过类似项目的经验，但那项目在他领导之下，缺点很多，成绩并不好。在两者之间，我仍然会挑后者来做我的项目经理，理由很简单：我如果要找一个向导带我去一个新地方玩，我一定会找一个去过那个地方的向导，而不是没去过但书和图片都看得烂熟的向导。当然，我们假设这位向导(项目经理)能在失败的经验中吸取教训，否则的话，那我们岂不是自找麻烦？

项目经理的管理技巧和技术能力哪样比较重要？以前，大家都认为管理技巧比技术能力重要，近年来，这个想法有转变的现象。转变的主要原因，乃是由于越来越多的技术人员感觉到，如果没有管理技巧，对事业生涯的发展是极大的障碍。基于此种认识，在技术专业外再追求管理技巧的训练，几乎已经变成了“胸怀大志”的年轻人必经的过程。反过来看，先有一般管理技巧再去追求技术的专业，这种例子毕竟比较少。由于一般的项目，在完成的期限上压力会很大，我们发现，如果一个项目经理在技术层面上不能与人沟通，当然并不是说一定不行，但在决策的速度上，却要大大地打个折扣。因此，如果我非要为经验、技术能力和管理能力三者来订一个百分比的话，我个人认为一个项目经理，他的经验应该占百分之五十的重要性，剩下来百分之三十是技术，百分之二十是管理能力。当然这三者往往是交互运用，分不出彼此的。至于一个项目是否能有一个以上的项目经理呢？我的答案是“能”，但“不应该”。为什么？除了政治上的理由外，我实在看不出这样安排的原因何在。

谈一个好项目经理应具备什么条件的文章很多，我现在想用另外一个角度来看同样的问题。我要问：

“在什么样的情形下，会让我们选‘错'项目经理呢？”

误信年龄大的人一定比较成熟。大家都承认项目经理的成熟与否，影响项目的成败很大，但殊不知成熟与否与年龄的大小或白头发的多少没有直接关系。我曾经见过很多公司的项目经理，往往从一伙人中找年纪最大者来做。这不单是不了解成熟和年龄无关，并且更不了解真正成熟的项目经理，他手上经历过的项目绝不能局限于某一类型。所以成熟和经历有关，和年龄无关。

正好有空。选项目经理又不是买电影票，不能说这个人目前正好有空，就叫他来做项目经理好了。太专横独断。专横独断的人只有两种：一种是对自己太有信心，认为自己比其它人都行。这种人做项

目经理，项目效果再好也有限。专横独断的另一种人，就是对自己毫无信心但又要硬充内行，项目请这种人来做经理，比找到上一种人更惨。

超级明星。找超级明星来做项目经理，注定不会有什么好下场。如果此超级明星属于好名派，那其它的成员只有干瞪眼，看风头都被项目经理一个人出尽。如果此超级明星属实干派，那在他累死其它成员之前，先把自己累死。找超级明星来做项目经理，不是一个聪明的选择。好的项目经理，宁愿让他的成员成为超级明星，而自己退居幕后。

不重视顾客所需。这种项目经理完全犯了本末倒置的毛病，忘了谁是项目的后台老板。项目经理扮演的角色，很大一部份是公关兼行销的角色，具有这种身分的人能够得罪后台老板吗？在行销这行，我们常说：“我们卖他需要的，但他买他想要的。”在“需要”和“想要”的两者间，我个人认为项目经理没什么筹码好讨价还价。

在职训练。如果一个项目经理，可以用来作为在职训练的试验，那这个项目的重要性也有限得很。我们常常看到，某些公司为了特别培养某人，而将其调派到各处去做不同项目的经理，这对当事人来说，增加经历当然很好，但对公司来说，险冒得太大了点。

不了解公司的情况。一个项目经理最容易忽略的一件事，就得对公司的长处短处，以及个人的长处短处不愿意花功夫去探讨。在这种情况之下去领导一个项目，岂不是瞎子摸象兼盲人骑瞎马！

总之一句话，选一个理想的项目经理，公司的负责人一定要注意以下的“三不原则”：

不要因为某人的薪水已经达到某一级的最高点，就升他为项目经理。

不要因为某人的手下有最多的幕僚，就选他做项目经理。

不要因为某人是项目经理，所以他的薪水就一定要比别人高。

最后，项目经理还有两个问题也是常常被人问到的：

在项目管理中，如果项目经理只能在两个部属中留下一人；而这两个人中的一个非常能干，但工作态度非常不好；另外一个则是非常不能干，但工作态度非常好。如果你是这个项目经理，你留谁？

一个项目经理，如果只能在两种管理方式下选一种：要就是把“人”看得比什么都重要，另一种就是把成效看得比什么都重要。在这两者之中必须选一个的情况下，哪一类型的项目经理比较容易成功？

这两个问题都不容易回答，而我也没有胆量说我知道绝对正确的答案，但有几点个人的意见可以提供读者做参考：

◆如果是项目的前期，我会留下工作态度好的人，因为工作态度不好的人会影响其它项目成员的士气。但如果项目已进入后期，我会留下能力比较强的人，因项目的压力比较大。但项目结束后或项目情势稳定后，我仍然会将其“整肃掉”。要注意的是在项目完成前，我一定会全力安抚他。

◆项目经理的责任是成功地完成项目，而不是交朋友。项目小组所追求和重视的，不是成为兄弟会或俱乐部的一员，而是成为一个足以自豪的项目成员。在这种情形之下，一个我说跳你就跳的项目经理，要比只重视人际关系的项目经理要有用。覆巢之下岂有完卵。如果项目不能生存，哪有个人生存的空间。做一个好的项目经理，心还是不能太软的。

返回目录

一个有效管理者的 11 面镜子

作/转载者：苏苑　中国项目管理资源网

许多许多的历史才可以培养一点点传统，

许多许多的传统才可以培养一点点文化。

在今天看来，管理中有些传统的做法是一种错误，不管是过去还是现在都应该避免。任何人都会犯错误，不犯任何错误的人，也必定是一无所成的人。问题是我们要如何避免犯错误？一般来说，经理人（主管）容易迷失和容易犯的错误有 11 条，应是我们可贵的检讨与反省的指标，就像一面无情的镜子，需要我们勇敢地面对它，从错误中学到宝贵的教训与经验，使自己成为一位成功而有效的主管。

拒绝承担个人责任

有一次一件事情出了差错，董事长把我叫去骂了一顿。我对董事长说这是我的错，我在董事长面前从来不会说这是谁的错。等我回到办公室，把几个副总叫过来，第一句话就告诉他们，刚才我被董事长骂的时候，可没有讲你们任何一个人的名字，我在他面前一人挑起全部的责任，可你们给我犯下这样子的错误。

如果我在董事长面前说，这是徐副总的错，这是赵副总的错，这是王副总的错，董事长只讲一句话：余总经理，我白请你了，如果今天都是他们的错，你在干什么？我要是被他这样子说，我就没脸再呆下去。所以在董事长面前，我一肩挑起，这叫负起责任，错就是错嘛，干嘛要把责任推卸给别人呢？大胆地承认错误，然后想办法解决问题，吸起经验教训，这才是最重要的。

有效的管理者，总是会为事情的结果负起个人的责任，不轻易把麻烦传给别人。美国总统杜鲁门曾在自己的办公室门口挂了一条醒目的标语："buckets stop here！"意思是问题到此为止，不再传给别人。每一位主管都应该把这句话当作自己的座右铭。

世界上有两种人，一种人在努力的辩解，一种人在不停地表现。做主管的要尽量地表现，少去辩解，要敢于负起责任。当出现问题时，看看是不是自己的原因？当准备去请教你的上司时，先自问一下，有没有负担起自己的责任，是不是非进上司的门不可？总之，要时刻记住美国著名管理顾问史蒂文·布朗的一句话：管理者如果想发挥管理效能，必须得勇于承担责任。

不去启发下属

所谓启发，是指随人、随时、随地的教育。不管什么时候，只要看到下属不对，都可以启发，连门卫都可以启发，但是许多当主管的往往不愿意开启尊口。有一次我看到一个文员在写信封，写错了，就马上把其他人一起叫过来说，各位请看，这信封的写法“刘总经理”4 个字要一样大，“总经理”3 个字不能小写，好像人家不配当总经理似的；名字反而大写是一个错误，名字要避讳，要小；后面写敬启，是错的，敬启是恭敬地打开，人家凭什么要恭敬地打开？写给人家总经理要写大启、君启或亲启，这样才有礼貌。

这叫机会教育，我在公司花很多时间在教育上，但这都是辛苦在前，轻松在后。千万别忘记，你的下属有 70%的教育都是靠你。但有人说教育是人力资源部的事情，这样的想法是错误的。一个主管应负七成的责任去教育他的下属，只有三成的责任是靠人力资源部门，而且人力资源部门只管基础性教育。真正的主管要做专业教育，不能“放牛吃草”，要抓住任何机会去启发你的下属。

只强调结果，不强调思想

人首先要有思想，然后才有想法，产生触动，最后变成行为，久了就变成习惯。行为要变成习惯是很难的。文学家陈之藩在《剑桥倒影》一书中有句话：许多许多的历史才可以培养一点点传统，许多许多的传统才可以培养一点点文化。可见文化是多年的习惯，要养成习惯不是简单的事情。比如准时观念是一种思想，一种行为，如果把它变成一个习惯，就会形成文化。世界强国的时间观念都是非常强的。

很多老总都喜欢讲一句话：不要告诉我过程，我只需要结果。这个话听起来很帅，很有个性，有风度。如果你是军长、师长，你可以命令部下带着兄弟们把那个山头给我攻下来，不要给我讲流多少血，我对血没有印象；不要跟我说死多少人，我不在乎，我只要求今天中午之前把那个山头给我攻下。不错，在军事上都强调这种观念，很多老总就喜欢有这种派头。可是，今天我们是干事业，不是叫下属去死。我们应该强调思想，你不扭转他的思想，讲 100 遍也没有用。

如果你的下属跟着你而思想没有长进，进来的时候和离开的时候都是一样的，你就不可能是一个成功的主管，他对你也会有怨恨。下属的思想是主管教育和灌输的，做主管的应该像教育自己的孩子一样去教育他们。如果你没有教给他思想，他就没有想法，就不可能产生触动，没有触动就发展不出行为，没有行为就更不可能产生习惯。

一视同仁的管理方式

每个人都要经历家庭、学校和社会教育。他父母怎么教育的，我们已经管不着了；他的老师当初告诉他什么，我们也来不及追究了，主管应成为他们的“父母”，有责任去教育他们。问题是，每个人的背景不同，个性不一样，经历不一样，世界上没有两个人是完全一样的，那应该怎么教育呢？那就是拒绝一视同仁的管理方式。一把钥匙只能开一把锁，不能用一把钥匙开所有的锁。比如批评人时，对脸皮厚的人，可以当众批评；对爱面子的人，要叫到办公室单独谈。

做主管的要费些心思，要去研究你的下属，从他们的行为、动作、眼神、语言、思想上去了解，去判断。如果他非常喜欢钱，就让他去做销售；他做事很仔细，可以让他干设计工作；看东西只看地上的人，适合守仓库；吃饭都用计算器的人应该做会计；婆婆妈妈的人去搞客户服务；坐不住的人就让他去做外勤。这叫用人用长处。最糟糕的是把人都看成一样，不同的家庭、不同的学校培育出不同的人，每个地方有不同的文化、亚文化。作为主管，应该关注这些问题。我手下有个管理人员，他把权看得很重要，对钱没什么大的要求，我就把公司的印章交给他，每天在办公室“嘣嘣嘣”的盖章，而且把他的位置放在办公室的中间，让大家都看着他，让他有一种权力感，结果他非常高兴。

对思想比较单纯、服从性比较高的人，我们可以给他工作指示，给他效率要求，给他预算控制，可以实行从上到下的直线管理；对于受过高等教育、敏感、见过世面、经历复杂、强调团队精神的人，要让他参与，要注重双向管理，不能搞单行道。一个公司到底用什么方法，没有完全的定式，有的强调制度，有的重视人性管理。一个公司凭一本人事规章是没什么用的，每个公司都有人事规章，而且大同小异。所以，管理要适应对象，不能一视同仁。

忘了公司的命脉：利润

有一天，一家公司的总裁在餐厅吃午饭。吃到一半，有 4 个熟悉的声音由隔壁的厢房传出。那些人的讨论相当热烈，他忍不住偷听，发觉是手下的几位高级主管在得意地谈起自己的部门。

总工程师说：“没人能跟我比，对一家公司的成功，贡献最大的部门就是生产部门。如果你们没有像样的产品，那等于什么也没有。”销售经理抢着说：“错了！世界上最好的产品一点用都没有，除非你有强大的销售部门把它卖出去。”

主管公司内部及公共关系的副总裁也有意见：“如果公司没有良好的形象，惨败是绝对的，没人会向一家他不信任的公司买产品。”

“我认为你们的观点都太狭窄了”，主管人力资源的副总裁展开攻击：“我们都知道公司的力量在于它的员工，去掉强有力而且工作意愿高的员工，公司立刻陷于停顿。”

4 位雄心勃勃的年轻人继续讨论，为他们的部门力争。直到总裁吃完午饭，讨论仍未结束，他离开餐厅时顺便在那间厢房门口停下。“诸位，”他说，“我忍不住听了你们的讨论，很高兴你们能为自己的部门感到自豪，不过我不能不说，经验告诉我，你们没一个说得正确。在任何公司里，没有哪个部门能对公司的成败负责。如果你追究到问题的核心，你会发现管理一家成功的公司就像玩特技的人维持 5 个球在空中，其中 4 个球是白的，分别写着：产品、销售、企业与公共关系、员工，另外一个是红球。在任何时候，玩特技的人一定要记住，无论发生什么事，绝不能让红球掉到地上，因为红球上写着两个字：利润。”这位总裁的话绝对正确。没有利润，公司即使有最完美的产品，最好的形象，最有能耐的员工，最引人注目的财务基础，它还是很快就会陷入困境。

做主管的有四大责任，为股东创造利润，为社会谋求就业，为员工谋求福利，为消费者谋求品质。最重要的是第一个，创造利润，让公司有发展，是所有主管的首要责任。总公司考核你，最高主管评价你，只问一件事：有利润吗？当然，我们所谓的追求利润不是说要不择手段地去赚钱，而是要把追求利润看成是责任是目标，并且始终牢记心中。

只见问题，不看目标

作一个主管，要注意目标，就像游泳一样，要一边游，一边看前方，不要一头撞到池壁才知道到了。不要花太多时间在小问题上，要多花时间在目标上，如果一个主管把精力放在小问题上，就会忘记自己的目标，会丧失创造力，或者至少会逐渐枯竭。很多主管好像很忙，其实常常都是空忙，他们每天花 90%的时间去做对公司只有 10%的贡献，这种缺乏效率的一个主要原因是他们只注意小处。

做事要看大原则，每天上班先做最重要最紧急的事情，其它做不完的事要放下，一个人不可能做完所有的事，永远都有做不完的事。

我们强调要看目标，并不是说不要看问题，问题一定要看，而且要看得仔细，因为问题就是机会。但只有站在目标的高度上看问题，问题才可能变成机会。所以，主管不要说我遇到了一个问题，要说我面对一个机会，这样的意义就不一样。如果专注于琐事，就难以看到真正的问题，也看不到机会。做不到这一点，你的竞争对手就会抢先一步，因为行销学上有一句名言：凡是你想不到的，你的对手会帮你想到。

不当主管，只做哥们

做主管的要有自己的威严，在公司里不要坏了规矩，下属的脚一旦踩到你的肩膀上，接下来就是踩到你的头顶上。我们对下属要爱，要支持要奖励，但是他站在你的头上讲话就不可以，这叫没有伦理，坏了规矩。一个主管如果纵容下属最后会很难管理，他第一次破坏规矩就要开始处罚。如果没有处罚，他就会成为一个“榜样”，公司的标准将被破坏，以后的事就难办了。

不少主管很难做到这一点，原因是他常常希望在公司获得大家的支持，自己不够稳时，常常在公司套交情，用哥们义气把大家拢在一起，但这样他讲话就会没有威严。原因何在？因为他没有把公司的要求和纪律看得非常重要，却把私人感情和个人功利看得更重，结果动之以情，把大家通通看成是哥们。人有时是这样子的，你对他好，他反而不感恩，反而认为你这个人可以非常的随便。

在我们公司，男职员拍女职员的肩膀我是有意见的，男职员如果讲黄色笑话也是不允许的，这叫

坏了规矩。有一次，在仓库我无意中听到一个男职员讲黄色笑话给一个女职员听，那女职员笑了笑。我马上走过去告诉那位男职员：“这种笑话回去讲给你妈妈和你的姐妹听，如果你觉得不能讲，那么我们的职员也不可以，这叫公司伦理。”我这话一说，他们显得非常困惑和尴尬。我的意思是，公司也要有伦理，如果这个时候我跟着他们一起笑，那就等于把他们当成了哥们，就会破坏规矩，给管理带来麻烦。

所以我的下属和我在一起，没有人可以随便和我开玩笑。我裁人时不用顾及太多，非常的凶狠。敢于这样做，就是因为平时我没有把他们当哥们，不欠他们的人情，裁人时可以不流下一滴眼泪。我今天很轻松，因为实际上我十分恩爱下属，只是不希望他们把我当作难兄难弟，在公司就得遵守公司的规矩。

没有设定标准

英国有家公司，专做世界有名的杯盘，他们的产品摔坏的比合格出品的还要多。有个设计师专门摔盘子，每 10 个摔坏 6 个，合格的只有 4 个，但是订单订到 3 年以后，没货。公司不怕摔，摔坏的盘子全都计入其它盘子的成本。就这样还难以买到，为什么？那是精品，公司有高标准，具有尊严。

一个公司设定行为标准就是让公司有尊严，让公司的员工有尊严。公司没有标准，一个经理没有设计出标准，公司就会不成体统。如果你进了一家公司，人家问你是怎么进去的。你说，进去不容易啊，要笔试，要口试，要扒一层皮。反过来，如果你说，进来很容易啊，随便就进去了。人家会认为这公司不值得呆，没有经过筛选，没有严格的标准。人都有这种心理：你越是有一种行为标准，越是有一种绩效要求，他觉得越有尊严。

所谓标准，其实是一种誓约、一种尊严、一种品质。像德国的奔驰一样，在街上看到奔驰，你会想到什么？那是一种尊严。为什么有钱的人都喜欢买奔驰，如果制造商没有那样的标准，你会买它的车吗？一样的道理，谁能像奔驰那样有一种标准，谁就有尊严。公司有了标准，可以让员工感到在这种公司工作是一种荣耀。当一切有关的人把标准视为一种誓约，一种品质的要求，自尊心在公司就会变得愈来愈强，管理也就变得愈来愈轻松。因此，主管不仅要执行标准，更要设立标准，只有具有管理标准，才会有高的管理绩效。

纵容能力不足的人

有的主管喜欢在办公室寻找爱，寻找下属对他的爱。其实错了，管理不是比赛，看看谁的爱最多，不要当老好人。我在公司常常讲一句话：做人（是指不讲原则只和稀泥这样的人）就不要做事，做事就不要做人。如果你做不到这一点，你就把位子让出来，让那些愿意当黑脸的人来当主管。今天公司交给你一个任务，是希望你能完成。而你怕得罪这个怕得罪那个，那么干脆就不要做。

古代的法家韩非子在这个问题上有个精辟的论述，用今天的话来说就是：一个主管只会压制自己，那叫怕；一个主管只会纠正自己，那叫乱；一个主管只会节省自己，那叫贱。主管没有必要告诉自己不要做这个不要做那个，纠正这里那里，总是为自己节省。有本事，自己乱七八糟，手下一切正常，每天在外面应酬客户，公司平安无事，这叫厉害。如果你一天到晚穿得标标致致，台灯照着你孤独的背影工作到深夜，最后还口吐鲜血，积劳成疾，这叫犯贱。为什么？管理层就像金字塔，如果只是顶上有点烂，下面稳固，不会有什么大事；如果底下坏了，顶上再好，也会摇摇欲坠。所以，你要严格管理你的部属，纠正你的下手，叫你的组织去节省。如果主管只要求自己，等于是纵容能力不足的人。

还有的老总喜欢找一个能力比自己差的人作副手，副手也找一个能力比他差的人作部下，这样下去，能力越来越差，于是主管就总是说自己的部下不行，其实都是他当初自己造成的。中国人用人有一种地方观念，喜欢用与自己地域关系或人缘关系较近的人，哪怕他的能力差些。这都是纵容能力不足的人。过度纵容能力不足的人，让那些没有能力、不求上进的人留在组织里，对其他人来说不公平，

于是大家都没有劲，结果是差的拖垮好的，最终拖垮了组织。

眼中只有超级明星

不要眼中只有超级明星，要强调团队精神。就像一个球队，如果只强调超级明星，不强调全体的努力，是难以取胜的。麦当劳有一句话，我们公司没有店长，店长是叫给外人听的。麦当劳的店长也要替客人点餐，这是公司总部的规定，全世界麦当劳的员工不分职位都要替客人点餐。他们体会到，公司能有今天的成功，靠的是全体员工，不是哪一个超级明星的功劳。

如果你把你那一行最顶尖的人全请到你公司去，那一年结束，还是只有一个人能挣得排行第一的位置。为什么？因为这么多顶尖好手根本不存在，而且就算他们存在，也只有一个"第一"，而其他的人得到的是"落选者"的头衔。一个公司真正的超级明星是很少的，公司大部分的业务都是那些一般的人做的。只重视超级明星，唯一的结果是降低管理绩效、减少公司业绩。

即使公司有超级巨星，也要淡化他的贡献，如果自己是超级巨星，更要有这种胸怀。主管要把 90%的爱放在 90%的人身上，不要把 90%的爱放在 10%的人身上，那样对另外 90%的人不公平。凡是为公司作出贡献的人都应看成公司的英雄，这样公司就成了一个 TEAM。

很多的公司去挖一些有名气的人，把他们当成超级明星请来，可我们常常听到没多久就是他们分道扬镳的结局。为什么？因为一些超级明星不会感恩，他们认为能有今天是自己努力的结果，不是公司栽培的结果；他们不合群，认为自己在公司是鹤立鸡群，而且还不妥协，碰到公司有难，还常常不愿意委屈自己；他们除了要求高薪以外，对公司没什么贡献。所以，做老总的不要眼中只有超级明星，要重视栽培部下，让部下变成明星。

在公司内部形成对立

有一次，我问董事长：他们在干什么？董事长严肃地望着我：他们是谁？我们楼下的维修工啊，我说。当时他这样子问我，我没有反应过来。又有一次，我问：他们那个工程……，“他们是谁？”我还没说完，他就打断我的话问。这时我想起了上次同样的经历，立即意识到了是什么问题。我对董事长承认，我错了。董事长说：“余总，这里只有我们，没有他们。”这件事情给我很深的教训。在公司内部，在顾客面前，不要说“他们”，要说“我们”。

作为主管，千万别小看这一字之差。举个例子，我在日航公司工作时曾到东京蛏田机场受训，有一次经过附近的一个超市，买了一盒杏仁豆腐，回去一吃，坏了。第二天我经过那里，进去跟营业员小姐说，我昨天买的杏仁豆腐是坏的。坏的？有没有带来？那个小姐问。我说，那又不值什么钱，我把它扔了，没关系，不要误会，我不是来要钱的。不不不，这是大事，你等一下。她说完就咚咚咚地跑到楼上，没多久，咚咚咚地跑了下来，旁边还有一个男士，手里拎着一个袋子，走到我面前说，先生，这里有 5 盒杏仁豆腐，保证是新鲜的，您拿去吃，这是您昨天买杏仁豆腐的钱，我们退回给您。我们店里卖出这样子的豆腐是我们的羞耻，但是我们已经打了电话，供应商下个礼拜要来开会，我们要研究一下为什么会发生这种事情。先生，如果下个礼拜一您还经过这里，您有兴趣的话，可以来找我，我会告诉您我们哪里犯了错误。这以后，我经过那里时都会去买东西。为什么？我相信它，我这辈子在那里买的任何东西，他们都会负起责任。其实，当时那个小姐不是卖给我东西的那位，那个店长也不是，可他们没有说：这个不是我经手的，这是供应商的错，这是昨天那个小姐的错，这是你自己的错。而他们只说，这是“我们”的错！

主管要常常强调“我们”的观念。如果有谁做错了什么，就是我们的错，然后去检讨是哪里出了问题。这个观念，应该从你的职业生涯开始就建立起来，久而久之就会形成一种习惯，最后在公司才不会形成对立，公司或你的部门就能真正团结成一个整体。

返回目录

项目经理成长曲线与角色价值

作/转载者：刘正高

项目经理是干出来的，不是学出来的；是带出来的，不是教出来的。一个人要成长为一名合格的项目经理主要不是靠学，而是靠干，当然学也很重要。靠干，完全不学，可以出项目经理。但靠学不靠干，是绝对出不来项目经理的。光干不学，有可能会出现的情况是，你的能力本来可以做一个大项目经理，但现在可能只能做一个中的或者小的项目经理，因为你没有理论指导，有些问题可能处理得不够好。

项目经理是怎么成长起来的呢？比如说一个人适合做项目经理，已经有了一定的基础，有可能去做项目经理了。现在有一个项目，这个项目可能很大，里面要有第一项目经理，还要有几个帮忙的项目经理，这时候把你放进去给别人帮忙，其实你并不是真正意义上的项目经理。不论大到什么样的项目，项目经理只有一个，即所谓的第一项目经理，他是真正集成败于一身的项目经理。别人都会把矛盾集中到他这里，由他来做判断，承担所有的后果。但你可以去做一个不是第一项目经理的项目经理，通过这个过程，别人来带你，然后再有一个比较小的项目，把你放进去做一个独立的项目经理。在开始时，有人会协助你，对于一些刚成型的文档、重要的会议，项目启动的时候会有一些有经验的上级经理或项目经理进去帮助你。原因很简单，项目是不允许失败的，它和科研是两个概念。目前国内在这方面有一定的欠缺，很多公司的研发和专业服务是不分的，或者都是项目驱动性的"研发"，没有真正面向产品的研发。其实研发和专业服务是两码事，研发是一种投资行为，既孕育着巨大的商机，也存在战略和技术上的失败风险；但专业服务或者说做项目是一种商业行为，是不允许失败的，所以不可能让一个完全没有项目管理经验的人做项目经理。

一个合理的公司是不可能让没有项目管理经验的人去干赔本赚吆喝的事情，但正如"先有蛋还是先有鸡？"的问题一样，项目经理又必须通过干才能被证明和成熟起来，这样就必须有人去带，去把关。

麦肯锡写的《软件业成功的奥秘》书中的统计，大多数的软件开发项目都是失败的，"软件开发就是不可能完成的任务"。对于一个公司来讲，在打算去做一个项目开发的时候，一开始如果你就想过成功与失败的概率分别是多少，一开始如果你就意识到你很可能是在做一件赔本赚吆喝的事情，那么你在做这件事情的时候就会比较谨慎。一个项目在开始做的时候，会有很多矛盾、争论，很难说哪一个是对的，哪一个是错的，大家都会有合理和不合理的地方，但最终总得有一个方案，项目经理的威信就是在这个时候发挥作用的。做项目管理的人往往处于矛盾的焦点上。国内目前对于项目经理的理解是很差的，一说做项目，大家都想到具体的事情，认为做项目管理是应用开发中最没有用的事情。他们认为，从做一个应用开发的项目来讲，需求分析需要业务人员，设计需要资深的技术人员，编码需要程序员，测试要由大家一起来做，什么事情是由项目经理干的呢？没什么事情！然而，并非如此。项目经理的活动是在这些工作之外的，没有项目经理，大家都在那里闷着头干，干完之后，凑不到一起去，白干了。所以，项目经理的工作就是去解决这些问题的。就像是打仗，有冲锋的，有掩护的，连长、团长不是做这些事情的，但是没有他们肯定不行。如果说打仗，大家能理解，但说到做项目，大家就不理解，认为项目经理就是打杂的，或者认为项目经理应该是写程序最多的人。事实上不是这样的。项目管理的核心是'三角平衡'，即范围、成本、进度三个方面保持平衡，但是这三点往往是互相打架的。这三点从根本上来说是矛盾的，在矛盾的状况下，一个项目经理会面临的挑战是，你如何去把它们处理好，使它们达到平衡，这是所有的项目经理都会面临的挑战。因为，并不是说有一个公式一套，就可以把问题都解决了，如果是这样，那就不叫管理了，管理是一门艺术而不是技术。你拿着这么一套方法论的东西，凭借自己的经验，在做不同的项目的时候，处理问题的方式是不一样的，每

一个项目的具体做法都是不一样的。没有哪一个案例可以把项目中所有的问题都包括进来。要想做一个成熟的项目经理，不光要经过一些专业的项目管理培训，还要有大量的项目管理理论的学习和在做项目过程中总结的大量经验的相互结合。如果只是通过听几堂课，学了几招技巧和方法，马上就能用，而且还能用对，那么业内资深的项目经理就不值钱了。

返回目录

软件项目如何估计规模大小

作/转载者：51CMM

软件项目的规模估算历来是比较复杂的事，因为软件本身的复杂性、历史经验的缺乏、估算工具缺乏以及一些人为错误，导致软件项目的规模估算往往和实际情况相差甚远。 因此，估算错误已被列入软件项目失败的四大原因之一。

软件工程师经常会被问到，编一个什么什么样的软件需要多长时间、多少钱。面对这个问题，有不少人很犯难，因为，第一用户的需求太不具体，第二，自己缺乏一个科学的估计方法。这里向大家介绍几种软件项目规模的估计方法。

概念介绍

先介绍一个衡量软件项目规模最常用的概念--LOC(Line of Code)，LOC 指所有的可执行的源代码行数，包括可交付的工作控制语言（JCL：Job Control Language）语句、数据定义、数据类型声明、等价声明、输入/输出格式声明等。一代码行（1LOC）的价值和人月均代码行数可以体现一个软件生产组织的生产能力。组织可以根据对历史项目的审计来核算组织的单行代码价值。

例如，某软件公司统计发现该公司每一万行 C 语言源代码形成的源文件（.c 和.h 文件）约为 250K。某项目的源文件大小为 3.75M，则可估计该项目源代码大约为 15 万行，该项目累计投入工作量为 240 人月，每人月费用为 10000 元（包括人均工资、福利、办公费用公滩等），则该项目中 1LOC 的价值为：

（240×10000）/150000＝16 元/LOC

改项目的人月均代码行数为：

150000/240=625LOC/人月

方法一、Delphi 法

Delphi 法是最流行的专家评估技术，在没有历史数据的情况下，这种方式适用于评定过去与将来，新技术与特定程序之间的差别，但专家"专"的程度及对项目的理解程度是工作中的难点，尽管 Delphi 技术可以减轻这种偏差，专家评估技术在评定一个新软件实际成本时通常用得不多，但是，这种方式对决定其它模型的输入时特别有用。Delphi 法鼓励参加者就问题相互讨论。这个技术，要求有多种软件相关经验人的参与，互相说服对方。

Delphi 法的步骤是：

1、协调人向各专家提供项目规格和估计表格；

2、协调人召集小组会各专家讨论与规模相关的因素；

3、各专家匿名填写迭代表格；

4、协调人整理出一个估计总结，以迭代表的形式返回专家；

5、协调人召集小组会，讨论较大的估计差异；

6、专家复查估计总结并在迭代表上提交另一个匿名估计；

7、重复 4-6，直到达到一个最低和最高估计的一致。

Delphi法规模估计迭代表

项目名称：_____
估计日期：_____
估计者：_____
估计伦次：_____
结果：
代码行_____LOC；周期：____月；工作量：_____人月；费用 _____元。

理由：

图1 Delphi法规模估计迭代表样例

方法二、 类比法

类比法适合评估一些与历史项目在应用领域、环境和复杂度的相似的项目，通过新项目与历史项目的比较得到规模估计。类比法估计结果的精确度取决于历史项目数据的完整性和准确度，因此，用好类比法的前提条件之一是组织建立起较好的项目后评价与分析机制，对历史项目的数据分析是可信赖的。

其基本步骤是：

1、整理出项目功能列表和实现每个功能的代码行；

2、标识出每个功能列表与历史项目的相同点和不同点，特别要注意历史项目做得不够的地方；

3、通过步骤 1 和 2 得出各个功能的估计值；

4、产生规模估计。

软件项目中用类比法，往往还要解决可重用代码的估算问题。估计可重用代码量的最好办法就是由程序员或系统分析员详细地考查已存在的代码，估算出新项目可重用的代码中需重新设计的代码百分比、需重新编码或修改的代码百分比以及需重新测试的代码百分比。根据这三个百分比，可用下面的计算公式计算等价新代码行：

等价代码行 = [(重新设计% +重新编码% +重新测试%)/3]× 已有代码行

比如：有 10,000 行代码，假定 30%需要重新设计，50%需要重新编码，70%需要重新测试，那么其等价的代码行可以计算为：

[(30% + 50% + 70%)/3]× 10,000 = 5,000 等价代码行。

意即：重用这 10000 代码相当于编写 5000 代码行的工作量。

方法三、功能点估计法

功能点测量是在需求分析阶段基于系统功能的一种规模估计方法。通过研究初始应用需求来确定各种输入、输出、计算和数据库需求的数量和特性。通常的步骤是：

1、计算输入，输出，查询，主控文件，和接口需求的数目。

2、将这些数据进行加权乘。下表为一个典型的权值表。

功能类型	权值
输入	4
输出	5
查询	4
主控文件	10
接口	10

3、估计者根据对复杂度的判断，总数可以用+25%、0、或-25%调整。

据发现，对一个软件产品的开发，功能点对项目早期的规模估计很有帮助。然而，在了解产品越多后，功能点可以转换为软件规模测量更常用的 LOC。

方法四、PERT 估计法

PERT 对各个项目活动的完成时间按三种不同情况估计：一个产品的期望规模，一个最低可能估计，一个最高可能估计。用这三个估计用来得到一个产品期望规模和标准偏差的 Pert 统计估计。Pert 估计可得到代码行的期望值 E， 和标准偏差 SD.

详细的估计方法，读者可参考笔者所写的《应用 PERT 进行项目工期估计》一文，这里不再赘述。

返回目录

如何制定软件项目测试计划

作/转载者：钟华

摘要 随着测试走向规范化管理，测试计划成为测试经理必须完成的重要任务之一，本文根据实践经验结合理论，探讨如何制定软件项目测试计划。

关键字 测试计划 变更

软件测试计划作为软件项目计划的子计划，在项目启动初期是必须规划的。在越来越多公司的软件开发中，软件质量日益受到重视，测试过程也从一个相对独立的步骤越来越紧密嵌套在软件整个生命周期中，这样，如何规划整个项目周期的测试工作；如何将测试工作上升到测试管理的高度都依赖于测试计划的制定。测试计划因此也成为测试工作的赖于展开的基础。

一个好的测试计划可以起到如下作用

1. 避免测试的“事件驱动”

2. 使测试工作和整个开发工作融合起来

3. 资源和变更事先作为一个可控制的风险

测试计划的模板在各个公司中都大同小异，在个人实践中发现，测试计划制定中存在的问题具有相似性，下面重点就这些相似的问题谈谈如何制定软件项目测试计划。

问题一：测试阶段划分

就通常软件项目而言，基本上采用“瀑布型”开发方式，这种开发方式下，各个项目主要活动比较清晰，易于操作。整个项目生命周期为“需求－设计－编码－测试－发布－实施－维护”。然而，在制定测试计划时候，有些测试经理对测试的阶段划分还不是十分明晰，经常性遇到的问题是把测试单纯理解成系统测试，或者把把各类型测试设计（测试用例的编写和测试数据准备）全部放入生命周期的“测试阶段”，这样造成的问题是浪费了开发阶段可以并行的项目日程，另一方面造成测试不足。

合理的测试阶段应遵循下面划分方法：

<table>
<tr><th>阶段
测试类型</th><th>需求</th><th>设计</th><th>编码</th><th>单元</th><th>集成</th><th>系统</th><th>确认</th></tr>
<tr><td>确认测试</td><td>计划</td><td colspan="5">设计</td><td>执行</td></tr>
<tr><td>系统测试</td><td>计划</td><td colspan="4">设计</td><td>执行</td><td></td></tr>
<tr><td>集成测试</td><td></td><td>计划</td><td colspan="2">设计</td><td>执行</td><td></td><td></td></tr>
<tr><td>单元测试</td><td></td><td></td><td>计划/设计</td><td>执行</td><td></td><td></td><td></td></tr>
</table>

照上图所述，相应阶段可以同步进行相应的测试计划编制，而测试设计也可以结合在开发过程中实现并行，测试的实施即执行测试的活动即可连贯在开发之后。值得注意的是：单元测试和集成测试

往往由开发人员承担，因此这部分的阶段划分可能会安排在开发计划而不是测试计划中。

问题二：系统测试阶段日程安排

划分阶段清楚了，随之而来的问题是测试执行需要多长的时间？标准的工程方法或 CMM 方式是对工作量进行估算，然后得出具体的估算值。但是这种方法过于复杂，可以另辟专题讨论。一个可操作的简单方法是：根据测试执行上一阶段的活动时间进行换算，换算方法是与上一阶段活动时间 1：1。1~1。5 左右。举个例子，对测试经理来说，因为开发计划可能包含了单元测试和集成测试，系统测试的时间大概是编码阶段（包含单元测试和集成测试）1 到 1。5 倍。这种方法的优点是简单，依赖于项目计划的日程安排，缺点是水分太多，难于量化。那么，可以采用的另一个简单方法是经验评估。评估方法如下：

1. 计算需求文档的页数，得出系统测试用例的页数

需求页数：系统测试用例页数 ≈ 1：1

2. 由系统测试用例页数计算编写系统测试用例时间

编写系统测试用例时间 ≈ 系统测试用例页数×1 小时

3. 计算执行系统测试用例时间

编写系统用例用时：执行系统测试用时 ≈ 1：2

4. 计算回归测试包含的时间

系统测试用时：回归测试用时≈ 2：1

注：以上比值是个人工程经验值，需要更正比值的测试经理可以在具体实践中收集数据

基于以上方法优点是需求为已知的，可以利用已知来推算未知，适用于需求是已知且相对稳定的情况下；缺点是处于研发状态的项目，需求不清晰的时候比较难计算。现套用一个例子加于说明：需求文档页数为 500，系统测试用例页数推算为 500，则编写系统测试用例时间为 500 小时，执行系统测试用例时间为 1000 小时，回归测试需要 500 小时，加起来总共为 2000 小时，按一天 8 小时计算，共计 250 个工作日/人；假如一个月为 22 个工作日，则共计约 11 人/月，即投入 4 个人需要 3 个月左右时间工作量完成。当然，这是系统测试需要的全部时间。根据测试阶段划分原则，设计用例时间可以和开发同步进行，只需在测试阶段中安排的时间为 1500 小时即 4 人 2 个月工作量。

（测试经理在编写测试计划时候，测试进度中的计划开始/结束时间往往用如 20050101-20051201 的具体时间划分方式，这样引起的问题是当项目计划进行变更的时候，测试计划时间不得不随时调整，这种变更可能是频繁而琐碎的，可以替代的办法是取消这种方式，采用 30 工作日/2 人或者 2 人月这种工作量记录方式，这样一来，只需在项目计划中跟踪阶段的具体开始时间即可，不必反复修改测试计划。）

值得注意的是：国内大多数公司的测试时间都是不足的，不可能按照这样的理想比例进行运作，因为测试执行的时间实际上不可能占据整个项目周期的 1/2，甚至要短于其中任何一个项目阶段时间。

即使是微软的测试结束原则也并不是完成所有必需的测试，而是测试在按计划结束的那一天结束！在测试时间不足的情况下，可参考下面项目计划变更时的做法，因为计划变更也涉及到测试时间不足的情况。

问题三：变更的控制

测试计划改变了已往根据任务进行测试的方式，因此，为使测试计划得到贯彻和落实，测试组人员必须及时跟踪软件开发的过程，对产品提交测试做准备，测试计划的目的，本身就是强调按规划的测试战略进行测试，淘汰以往以任务为主的临时性。在这种情况下，测试计划中强调对变更的控制显得尤为重要。

变更来源于以下几个方面

1. 项目计划的变更
2. 需求的变更
3. 测试产品版本的变更
4. 测试资源的变更

测试阶段的风险主要是对上述变更所造成的不确定性，有效的应对这些变更就能降低风险发生的几率。要想计划本身不成为空谈和空白无用的纸质文档，对不确定因素的预见和事先防范必须做到心中有数。

对于项目计划的变更，除了测试人员及时跟进项目以外，项目经理必须认识到测试组也是项目成员，因此必须把这些变更信息及时通知到项目组，使得整个项目得到顺延。项目计划变更一般涉及都是日程变更，令人遗憾的是，往往为了进度的原因，交付期限是既定的，项目经理不得不减少测试的时间，这样，执行测试的时间就被压缩了。在这种情况下，测试经理常常固执的认为进度缩减的唯一的方法就是向上级通报并主观认为产品质量一定会下降，这种做法和想法不一定是正确的。由于时间不足，不能“完美”的执行所有测试，为了保证质量，第一种办法是调整测试计划中的测试策略和测试范围，实践中测试经理常常忽略测试计划的这个章节。调整的目的是重新检查不重要的测试部分，调换测试的次序和减少测试规模，对测试类型重新组合择优，力求在限定时间内做最重要部分的测试，可以把忽略部分留给确认测试或现场测试。其他应对办法包括减少进入测试的阻力，例如降低测试计划中系统测试准入准则；分步提交测试，例如改成迭代方式增量测试；减少回归测试的要求，例如开发人员实时修改，在测试计划中对缺陷修复响应时间和过程进行约定；和公司 QA 商量进行简化配置管理，跳过正式发布环节；缺陷进行局部回归而不是重新全部测试等等。

第二：项目进行过程中最不可避免的就是需求的变更。那么，测试计划中就不能进行控制和约束的吗？答案是未必。当制定计划时，如果项目需求处于动态变化时，在测试用例章节就要进行说明。许多测试经理在编制测试用例时往往没有把测试用例和测试数据进行区分，因此，造成的问题是当需求变化时辛辛苦苦设计的数据就作废了。在这时，假使面临一个需求动态的项目，必须在计划中对需求变更造成的测试（设计）方式变化进行说明，例如采用用例和数据分离、流程和界面分离、字典项和数据元素分离的设计方式，然后等到最终需求确定后细化测试设计；另一个方面是最好制定一个变更周期的约定——尤其在执行测试阶段发现需求的变更——定义变更的最大频度和重新测试的界限，

计划从一定程度上能够降低不可预期需求变化造成的投入损失。值得注意的是：需求发生变更时测试经理额外的工作是记住要在需求跟踪矩阵上做记录。

对于测试产品版本的变更，除了部分是由于需求变更造成之外，很有可能是由于修改缺陷引发的问题或配置管理不严格造成。众所周知，测试必须是基于一个稳定的“基线”进行，否则，因反复修改造成测试资源和开发资源的浪费是可观的。合理的测试计划在章节中应增加一个测试更新管理的章节，在此章节明确更新周期和暂停测试的原则。例如，小版本的产品更新不能大于每天三次，一个相对大的版本不能每周大于 1 次，规定紧急发布产品仅限于何种类型的修改或变更，由谁负责统一维护和同步更新测试环境。测试计划通常制定了准入和准出准则，这是不够的，要考虑测试暂停的时候，产品错误发布或者服务器数据更新就是一个例子，暂停的时候如果测试经理不进行跟踪，可能发生测试组等待测试而没人通知继续测试的情况，所以，增加更新周期和暂停测试原则是很有必要的。

最后，测试资源的变更是源自测试组内部的风险而非开发组风险，当测试资源不足或者冲突，测试部门不可能安排如此多的人手和足够时间参与测试时，在测试计划中的控制方法与测试时间不足相类似。没有测试经理愿意承担资源不足的测试工作，只能说公司本身是否具备以质量为主的体系或者项目经理对产品质量的重视程度如何决定了对测试资源投入的大小，最终产品质量取决因素不仅仅在于测试经理。为了排除这种风险，除了象时间不足、测试计划变更时那样缩减测试规模等等方法以外，测试经理必须在人力资源和测试环境一栏标出明确需要保证的资源，否则，必须将这个问题作为风险记录。规避风险的办法可能有：

一，项目组的需求和实施人员参与系统测试；

二，抽调不同模块开发者进行交叉系统测试或借用其他项目开发人员；

三，组织客户方进行确认测试或发布 β 版本。

尽管上面尽可能的描述了测试计划如何制定才能“完美”，但是还存在的问题是对测试计划的管理和监控。一份计划投入再多的时间去做也不能保证按照这份计划进行实施。好的测试计划是成功的一半，另一半是对测试计划的执行。对小项目而言，一份更易于操作的测试计划更为实用，对中型乃至大型项目来看，测试经理的测试管理能力就显得格外重要，要确保计划不折不扣的执行下去，测试经理的人际谐调能力，项目测试的操作经验、公司的质量现状都能够对项目测试产生足够的影响。另外，计划也是“动态的”！不必要把所有的因素都可能囊括进去，也不必要针对这种变化额外制定“计划的计划”，测试计划制定不能在项目开始后束之高阁，而是紧追项目的变化，实时进行思考和贯彻，根据现实修改，然后成功实施，这才能实现测试计划的最终目标——保证项目最终产品的质量！

返回目录

软件项目中的“敏捷流程”

作/转载者：吴昊

1991 年秋，在美国勒海大学亚科卡学院的一份研究报告《21 世纪美国制造业的战略：一个工业主导的观点》中，首次提出了敏捷竞争的概念。何谓敏捷（Agility）？对于企业而言，敏捷意味着企业能够在顾客机会不断变化、难以预测的竞争环境中赢利运营；对于个人而言，敏捷指在企业对难以预测的顾客机会做出反应，不断重组其人力和技术资源的过程中，个人能够对赢利底线做出贡献，提高企业的净收入。因此，敏捷可以看作是对变化和不确定的全面反应。

变化和不确定，对于软件业来说，是多么熟悉而又让人烦恼的名词。软件工程自诞生以来，一直试图通过技术和管理的手段来降低软件项目的不确定性。在这个美好的愿景指导下，专家们发明了结构化、发明了面向对象、发明了 CMM，这些新的技术和方法的确有助于“软件危机”的解决，促进了软件业的发展；然而，超支、超时、低质量的老问题并未得到根本解决。为了对抗不确定，软件开发越来越复杂，越来越庞大，传统的重量级（Heavy Weight）方法的副作用也越来越明显——组织臃肿、办事低效、官僚主义...

相对于重量级方法，软件业一直有另一种声音在，那就是轻量级方法（Light Weight），其目标是以较小的代价获得重量级相当的效果。

最负盛名的轻量级方法是 XP。XP 是 Extreme Programming 的缩写，从字面上可以译为极端编程。但是，XP 并不仅仅是一种编程方法，也不是中文中理解的那种不可理喻的“极端”化做法。实际上，XP 是一种审慎的（deliberate）、有纪律（disciplined）的软件生产方法。XP（Extreme Programming）植根于上个世纪 80 年代后期的 Smalltalk 社区。90 年代，Kent Beck 和 Ward Cunningham 把他们使用 Smalltalk 开发软件的项目经验总结和扩展，逐步形成了一种强调适应和以人为导向的软件开发方法。

XP 的核心是四大价值，即改善沟通（communication），寻求简单（simplicity），获得反馈（feedback）和富有勇气（courage）。在此基础上，XP 总结出了软件生产的十余条做法（practice），涉及软件设计、测试、编码、发布等各个环节。与其它轻量级方法相比，XP 独一无二的突出了测试的重要性，甚至将测试作为整个开发的基础，每个开发人员不仅要书写软件产品的代码，同时也必须书写相应的测试代码；所有这些代码通过持续构建和集成（Continuous Build & Integration）为下一步的开发打定了一个高度稳定的基础平台。有了这样的基础平台的保证，XP 就可以实施软件设计的再造（Refactoring）。XP 的设计理念是，在每次迭代周期仅仅设计这次迭代所要求的产品功能，上次迭代周期中的设计通过 Refactoring 形成此次的设计。

2001 年 2 月，在美国犹他州的一个滑雪场，17 位轻量级软件开发方法的创始人和专家，包括 Kent Beck（Extreme Programming）、Alistair Cockburn（Crystal Methodologies）、Jim Highsmith（Adaptive Software Development）等等，共同发布了“The Manifesto for Agile Software Development”（敏捷软件开发宣言）。这表明，在软业经历了无数次的项目失败之后，人们开始反思软件开发的工程特性，反思计划和控制的有效性，反思过去对于不确定性的态度和反应。敏捷终于为这个行业，以及这个行业中的一些人所认识、理解和推崇。

与会者之一 Martine Fowler 在其后来的文章“The New Methodology”中这样解释重量级、轻量级和敏捷：

轻量级与重量级的差异来自于人们对两种方法的文档数量的直观感受，即轻量级方法较少产生和依赖于庞大的文档，但这只是一个表面现象；在诸多的轻量级方法之间存在着很多相通的地方，敏捷更恰当的表达了这些轻量级方法的最根本之处。首先，敏捷强调适应，而非预测。重量级方法花费大量的人力物力，试图制订详细的计划来指导长期的工作，而一旦情况变化，那么计划就不再适用。因此本质上重量级方法是抵制变化的，而敏捷方法则强调适应变化。其次，敏捷方法以人为中心，而非以流程为中心（即以事为中心）。敏捷方法强调软件开发应顺乎本心（work with people's nature ），软件开发应带来乐趣。

敏捷流程（Agile Process）汲取众多轻量级方法的“精华”，更加强调对变化的适应和对人性的关注。除了上面介绍的 XP 以外，其他知名的敏捷流程包括：

1. Crystal

Cystal 事实上不是一种开发方法，而是一系列的方法。因为 Crystal 的发明人 Alistair Cockburn 认为，不同类型的项目需要采用不同的方法。Alistair Cockburn 从两个维度来划分项目，一是项目规模，以人数计算；二是产品发生错误的后果，以严重性计算。

Crystal 方法集也形成于 90 年代，当时，Cockburn 接受了 IBM 的任务去写一些关于开发方法的东西。相对而言，Crystal 的个人色彩要淡些，因为它不仅来源于 Cockburn 的个人经验，而且也来源于 Cockburn 走访的很多不同的项目；而 Cockburn 本人也较为思想开放，乐于接受“异见”。

在以人为导向上，Crystal 和其他敏捷方法有些差异。在 Cockburn 看来，自由是人之本性，要人遵守纪律总是有难度的；因此，要在工作生产率和流程的纪律性之间作一权衡。流程要易于遵照执行，而这是以牺牲生产率为代价的。Cockburn 认为，XP 的流程规范仍太复杂和难于执行，而采用 Crystal 虽然生产率不如 XP，但开发人员更乐于采用。

Cystal 也强调在每个迭代后的 Review，并以此进行 Cystal 方法的自身改进。

2. ASD

ASD（Adaptive Software Development）的发明人 Jim Highsmith 本来是一个传统开发方法的工作者，他有多年的预测型方法的研究、教学和实施经验，但后来，他发现这些预测型方法根本就存在很大缺陷，尤其不适合当前的软件业务。

ASD 强调开发方法的适应性（Adaptive），这一思想来源于复杂系统的混沌理论。ASD 不象其他方法那样有很多具体的实践做法，它更侧重为 ASD 的重要性提供最根本的基础，并从更高的组织和管理层次来阐述开发方法为什么要具备适应性。

3. SCRUM

SCRUM 同样也包括了很多具体做法，这些做法并无多少特别之处，但多数有一个“怪异”的名称。比如，SCRUM 将开发过程划分为 30 天的迭代周期，每个迭代周期叫做一个 Sprint；每天有一个 15 分钟的短会，用来决定第二天的任务安排，这样的短会就叫做 scrum。

SCRUM 较为有特色的，是它特别强调开发队伍和管理层的交流协作。每天，开发队伍都会向管理层汇报进度，如果有问题，也会向管理层要求帮助解决。

4. FDD

FDD（Feature Driven Development）的发明人是 Jeff De Luca 和 Peter Coad。FDD 在 OO 社区较为人所知。

FDD 定义了 5 个流程，分别是 Develop an Overall Model、Build a Features List、Plan by Feature、Design by Feature 和 Build by Feature。其中前 3 个流程是在项目开始就进行的，而后两个则出现在每次迭代周期中。FDD 的迭代周期是两周。每个流程被划分为不同的任务和相应的验证标准。

开发人员被归为两种，一种是主程序员，另一种是 class 所有者。主程序员不作具体的编程工作，但要负责将 Feature 和 Class 对应起来，并充当开发协调者、设计者、技术支持和指导者等；class 所有者则进行实际的编程。

在软件业，敏捷流程还犹如星星之火，特别是在国内，敏捷流程还鲜为人知。在即将到来的未来，敏捷流程将何去何从，中国的软件从业者又将在其中扮演何种的角色，套用一句中国的古话，“路漫漫其修远兮，吾将上下而求索”。

返回目录