
step365思步网.com
会员自己的 精品 读物
记录我们的足迹 分享成长的点滴
电子期刊
作者：思步网会员
主编：cecilia
编委：step365he Jane
深海 xiaoyi830426
校对：fishred
第八期
封面設計：steplv
发布：2008.07.18

著作权说明

本文档为免费电子文档，任何人可以在思步网http://www.step365.com 免费下载。**著作权属于思步网及思步网会员共同所有。**

在不对本文档做任何修改的前提下，任何人都可以在互联网上自由下载、传播本文档，也可以放在自己的站点供他人下载。

若通过互联网在线转载其中部分内容，或者通过其他媒体转载本电子书及其中部分内容，必须注明文档来源“思步网 http://www.step365.com ”和文档作者。

本文档仅供学习交流之用，未征得思步网及思步网会员的同意，本文档不得用于商业用途。

由于编者水平所限，在文档中难免会出现错误，欢迎读者对本文档提出批评建议，意见请反馈至：service@step365.com ，谢谢！

阅读注意事项

在阅读本电子文档时，请留意以下字符（字母）所代表的含义。

本电子文档第一部分中：

Q（user name）：Q 代表问题 Question

A（user name）：A 代表解答 Answer

C（user name）：C 代表结论 Conclusion

括号中内容 user name 为思步网会员名

例如：**Q**（steplv）xxxxx，**A**（sungubbi）zzzzz，**C**（fishred） yyyyyy

表示：

提出问题者为会员 steplv，问题内容为：xxxxx

解答问题者为会员 sungubbi，解答内容为：zzzzz

做总结者为会员 fishred，总结内容为：yyyyy

依次类推。

思步在线交流

同时欢迎广大读者加入我们的 IM 群，与众多朋友在线交流：

- MSN群：mgroup24287@hotmail.com
- QQ群：19894513（申请加入时，请注明职业）
- WebSite：www.step365.com

目　录

第一部分：思步简讯

思步网首次线下活动成功举办

6 月 28 日，思步网首次线下活动——思步沙龙“北京站”，在北京中国科学院软件研究所顺利举行。

这是思步网自成立以来，所举办的第一次线下活动，因此凝聚着太多的期许与关爱。此次活动由思步网主办，中科方德软件有限公司协办。

举办本次活动，旨在促进同城朋友的广泛接触：从“不认识——接触——认识——了解——熟悉——交心”，我们希望这是一条不会间断的闭环线，我们思步网愿成为这中间的连接线。

活动前，主办方与协办方通力协作，精细筹划、积极准备；活动中，参与人员积极互动，活动精彩纷呈；活动后，大家联系热络，彼此分享心得体会。

总之，这是一次尝试，同时，更是一种责任的开始，希望通过更多此类活动的举办，不但可以提升个人竞争力、促进行业快速发展，更能让活动成为大家相互了解、认识的桥梁。

第二部分：你问我答

1 全员改进系统

★ 持续集成与 build 的周期

Q（Scott）：

【转载自网络】敏捷软件开发中一个重要的实践是“持续集成”，而不同敏捷程度的开发团队在实现持续集成的方式上有很大不同，而集成形式的问题有时候会使问题本末倒置，使开发者迷失了“持续集成”本来的意图。持续集成是持续地维护一个进化中的软件产品。为什么要维护这样一个产品呢？用敏捷开发的一个基本原则来解释就是：一切对与错，只有通过可操作的产品才能最终评判，所以，持续集成的目的是为了尽可能早的获得有效的反馈，反馈周期越短，反应速度越快，开发过程驾驭变化的能力就越强。如果用 monthly build，反馈信息的规模是以 change request/requirement packet 来计算的；如果

用 weekly build，反馈信息以 feature 来计算；如果用 daily build，反馈信息规模以 task 来计算；如果随时创建，反馈信息的规模则可以以代码行来计算。试想，如果产品中出现一行不恰当的代码即可引起开发者立即反应，这样的效率一定是一个已经达到非常高的敏捷程度的开发团队才能实现的。

Q（iamredeye）：

大家 integration 的周期一般是多久？

A（lily_014）：

看项目的阶段来规划，大规模的编码和集成测试阶段采用 daily build。否则报告里面也没有什么改变的东东。

★ 解答一下产品集成的问题？

Q（sungubbi）：

我所指的产品集成，有几种情况：

1）软件产品本身各模块或功能的集成，最后形成该产品可执行的产品包。

2）多个软件产品的集成，此时涉及布署问题。

3）（多个）软件产品＋硬件（如小型机）＋支持软件（如 ORACLE）的集成，形成最终可正式使用的产品，此时涉及布署问题。这种情况可能更多时侯叫做系统集成吧！

Q（iamredeye）：

软件产品＋环境（比如编译，运行环境）？硬件环境会一起集成吗？还是仅用 BOM（bill of material）来描述？

A（scott）：

被大家说糊涂了，就我理解说两句吧，我没参加过 CMMI 认证，所以没法揪这个“产品集成”这个字眼。就我的理解，产品集成，就是通过某种方式集成为产品，这里应该不涉及到部署，但涉及到软硬件集成。

对于纯软件来说，我们完全可以每日构建集成；如果系统复杂，可以分为多个模块，分别 Build，这个例子我举过，比如 A,B,C,D，A 和 B 先 Build，C 和 D 再 Build，最后 ABCD 合在一起 Build，这种 Build 的策略、环境和各模块之间的接口都属于产品集成。对于软硬件的也是一样，只要把软硬件看成模块就可以了。

A（timlq）：

粒度的问题，取决于项目与产品的集成方式，据项目自身来定义。

A（iamredeye）：

刚好这几天在关心持续集成，我感觉很关键还是 unit/itegration test cases 的有效性，因为看到有些人也在比较频繁的集成，但 testing 不到位，实际上没办法发现并解决更有风险的问题。

2 最佳实践

★ 敏捷该如何管理？

本实践摘自“全员改进系统”

Q（Scott）：

这里面很多都是搞过程改进或QA的人员，对于CMMI绝大多数人是通过企业过级学习的，CMMI书籍少之又少。对于敏捷，看上去书籍好像非常多，资料也很丰富，但是大家有没有发现这里面很多资料都是敏捷技能，比如结对编程、每日构建、全程建模等，至于敏捷管理提得很少。CMMI抓里程碑，敏捷抓迭代周期，本质是一样的，结果也是一样的，我们遇到很多实施敏捷的企业进度一样延期，而且有时很难控制。最近出了一本新书叫做“敏捷估计与规划”，终于谈到了一些敏捷管理的内容。实际上在此之前是有几本敏捷管理的书，只不过我当时看时一头雾水，可能是能力有限，不知道有没有高手吸收了精华，比如ALISTAIR COCKBURN的敏捷软件开发（http://www.china-pub.com/38010）。

最近看了另一本敏捷管理的书籍，感触很深，非常想和一些敏捷实践者们沟通一下。希望有实践 FDD、XP、Scrum 或对敏捷有些了解的人士一同讨论。愿意参加的请直接回帖，并请注明所擅长的敏捷方法学。

等有几个人之后我结合群和论坛发议题讨论，大概议题包括但不限于以下内容：

什么样才算敏捷管理？或者说什么是敏捷管理？

敏捷管理的本质是什么？或者说敏捷管理到底关注什么？快速交付？客户需求？产生价值？

敏捷管理是否天生混沌？拥有不确定特质？

如何度量敏捷管理？

敏捷管理的典型角色？

敏捷管理中的估计与计划？

敏捷管理中的监控与跟踪？

企业中各部门如何应对敏捷管理？

敏捷管理在企业中适用的层次？产品？项目？

如何使用各类敏捷方法学中的管理内容？

本讨论暂不关注敏捷技能，只关注敏捷管理。举例暂时考虑 FDD，如果组内人员对其他方法较为熟悉，可考虑更换。

较为熟悉 FDD，研究过 FDDPMA，很早以前实践过 XP。之前关注敏捷技能，目前关注敏捷管理。

A (iamredeye):

不太明白你说的这个敏捷技能和敏捷管理的区别。这里的“敏捷管理”大概是指针对实施敏捷方法的项目所进行的项目管理吧？

我想不管用哪种方式实施项目，项目管理关注的东西应该没有区别。就像 CMM 给的那个框架，PM area 都一样，engineering area 被不同的方法所填充。

另外相对比较沉重的一些 engineering 方法，敏捷关注的就是相对独立的 practices

敏捷并不是很成体系的理论。但正如 Iva Jacobson 现在极力推销的思想，也许这样的思路更为实用和有效。Jacobson 下周会到我们公司搞个座谈会，有啥新鲜东西我会告诉大家。

A(Scott):

For Example:

安装 CMMI 的管理做计划：明确的需求->估计->进度

按照 RAD 和其他的敏捷方法：进度->估计->筛选确认的需求，进度一般为迭代周期。

那么这两种管理做计划的方式是不同的。由此我们推测，XP,FDD,Scrum...应该有自己的管理方式。

敏捷里强调一种自适应性，这来自系统工程，所谓自适应性也表现出了混沌的特质。如果敏捷真的就是混沌，那就说我们对敏捷是不可控制的，不可以管理的，那实施敏捷的项目延期也是很正常的。请问，如果是“实施敏捷就等于对项目失控”，你还会愿意实施敏捷么？

我们平常研究的只是敏捷技能，比如如何在大项目中做重构，如何做每日构建，这些只是敏捷的技能，也就是你所说的 Practice。

我记得 05 年时 Jacobson 一直在推行面向方面编程，不知道最近又在推行什么。

★ 敏捷！敏捷！

本实践摘自“全员改进系统”

Q（iamredeye）：

好几年前就看到有极限编程的书在卖，单看封面，还以为是本编程方面的书，也就没有留意。

到今天已经被充斥大街小巷的一系列敏捷名词不停地轰炸到头晕，却对敏捷仍一无所知，突然间从一个 PM 身上发现了敏捷的魅力，对它一下好奇了起来。

CMM 提升了我们看问题的视角，RUP 用工具和可操作性充实了迭代理论。敏捷呢？它一系列的实践太具有吸引力了！你觉得敏捷是中国软件的一贴良药吗？

大家都在用什么方法呢？交流一下吧……

A（xixiaojing666）：

不懂，不知道什么是敏捷！

A（iamredeye）：

我也不懂啊，略微接触之下发现眼前一亮，所以才想和大家交流一下。

A（step365he）：

如果楼主把 Cowboy coding 等其他几个较为陌生的名词解释下，可能效果更好。

我在楼下放了篇文章，里面谈到几种方法，给大家参考。

A（sungubbi）：

觉得敏捷是过程中的一些最佳实践，是一些可以提高工作效率的方式或方法。

A（step365he）：

敏捷中的极限——极限编程创始人谈论敏捷开发：

极限编程（XP）的创始人和“敏捷软件开发宣言”的执笔者之一，Ken Beck，一直是敏捷编程的倡导者，他认为敏捷编程可以缩短软件开发项目的开发周期。近日他参加了旧金山举行的 Qcon 大会，并接受了媒体的采访，谈论了敏捷开发的相关问题。

记者：你能谈论一下敏捷宣言和极限编程（XP）吗？

Ken Beck：在极限编程（XP）后面的基本设想是，肯定有一些对软件开发有帮助的敏捷行为。我曾经问过这个问题，“假若我们尽最大的热情来进行软件开发，会发生什么？”这就是“极限（Extreme）一词的由来。结果证明，如果你采取其中的一些做法，例如技术上的协作、测试，并比以前更加努力的推行这些做法，至少在一般的软件开发项目中你会获得比较好的结果。例如你的错误率会比较低，这意味着你可以更频繁的发布新

版本软件。你能够以非常低的成本让项目启动进入最佳的可部署状态，与其他风格的开发模式相比，你获得了更低的开发成本和更短的开发周期。而且如果你将这种理念融入到后续的设计中，持续的对设计进行认真的投入，你可以在很长一段时间内以一种非常稳定的速率来继续部署新的功能。

这就是敏捷编程如何启动的方式，事实证明为了实现所有上述目标，无论是作为一个团队还是一个个体，你还必须学习许多新的社会技能：诚实、无私、有责任心。在获得一定的开发速度后，接下来的目标是，如果你想更快的进入下一步，你所要做的是能够更清晰和透明的交流正在进行的开发工作。

记者：你提到了推动敏捷编程发展的风格：可靠性、变化的低成本、增加的投资回报率。为什么软件开发市场正在从瀑布风格转向敏捷开发风格呢？或者现在是否只有一小部分开发者正在使用这个方法？

Ken Beck：是的，现在是只有一小部分开发者正在使用敏捷方法，但是我认为这一数字正在非常迅速的增长。我虽然没有明确的数字来证实我的观点，但是如果你关注一下敏捷开发者大会的增长，你会清晰的感觉到这一点。我认为现在推动敏捷开发的因素是这种开发风格更加贴近真实环境、更透明和更有责任感，如果你的开发周期比较短你会决定它就是你希望实现的开发方式。对于那些尊崇真实至上的软件开发，存在着巨大的潜在市场。

记者：为什么使用者还比较少？

Ken Beck：因为它存在的时间还不是很长。

记者：你今天早晨提到这个问题：用户不得不自己来确认你的软件是否可以正常使用。这本来是一件理所当然的事情，但是事实并非如此。为什么会出现这种情况？

Ken Beck：嗯，我认为造成这种现象的原因比较复杂，是技术和社会因素的综合原因导致了在已部署的软件中存在所有的缺陷。一部分原因是人们抱有软件天生就是不可靠的态度，客户习惯了这种状况。开发者习惯了接受这种观点。测试者也对此习惯了。我们只是觉得它像天气一样，对它无能为力，但这不是一个负责任的态度，因为实际上开发者有很多关于它的措施可以采取。从技术上来说，可以通过测试驱动开发、自动化集成测试、持续集成等手段；从社会学的角度来讲，端正自己的态度，不想推出具有缺陷的软件。以及加强开发团队成员之间的交流和关系等。

记者：你所熟悉的这些研究也说明了现在有如此众多失败的软件项目的原因。敏捷方法能拯救它们吗？还是只是一个临时的解决办法？

Ken Beck：敏捷方法是失败软件项目的救星吗？我认为许多软件项目依然会失败，问

题是除非你非常深入的了解这些软件项目，你不知道会失败的是哪一个。那么敏捷方法是它的救星吗？不。我认为它依然会失败，因为好的想法并不一定在实践中产生好的结果。敏捷开发可以带给你的一件事情是：让这些项目失败的更快、损失的更少，因为你可以将时间和精力用于开始做下一件事情。

记者：对于极限编程，你提到隔几个星期进行一次迭代（iterations），而不是六个月到一年，你如何定义这个迭代周期，以及软件发布的周期？

Ken Beck：在极限编程中的基本周期，基本的计划周期是一个星期，这是非常合理的，因为它是根据人们工作的时间来定的。在每个星期结束的时候，这个软件应该比这一星期开始的时候具有更多的功能。

极限编程（XP）与其他敏捷方法相比如何？比如 Scrum。

关于极限编程我比较喜欢的一件事情是其全面性。它从关于与好的软件开发一致的观点的全面讨论开始。介绍了一些基本原则，介绍了获得我们讨论的某种目标的具体的实践。因此我认为极限编程区别于其他方法的特点是它具有这种全面性。

记者：你能指出一个使用极限编程完成的比较突出的软件项目吗？

Ken Beck：例如 Carfax。

记者：我们曾经听过“牛仔编程（cowboy coding）”的叫法，敏捷方法和牛仔编程有什么区别？

Ken Beck：我偶尔会体验一下牛仔编程方式，尽管它在我的工作中不占有重要地位。在牛仔编程中，你可以像一个勇士一样去编程，你会对自己的工作感到很满意，因为你认为这个世界上再没有别人可能作出和你的一样的东西。

极限编程风格的开发也是需要勇气的；这一点它与牛仔编程类似。但是牛仔编程是完全不透明的，而极限编程则是透明的。牛仔编程是一种个体行为，而极限编程是一种团队行为。不仅仅是程序员一起工作，还包括用户、管理者和测试者。因此我认为它们还是有差异的。

在极限编程风格中，早早结束编码是因为可以更快的看到真正的用户反馈，从而继续改进。一个牛仔式的程序员也会做同样的事情。或许你刚刚介绍了一半问题的定义，它们会说，“好的，我明白了”，然后就去开始编程。是的，这一点看上去和极限编程团队有些类似，但是两者实际上是不同的，因为牛仔式程序员已经结束了这次交谈，而 XP 团队则是才刚刚开始这次交谈。

记者：牛仔编程有成功的时候吗？

Ken Beck：当然，在短期内它具有巨大的风险和巨大的成本、巨大的隐形成本。我有过使用它的经历，但是它有时候会效果不错。

记者：有的软件项目是由分散在不同地区的团队成员在不同的时间来进行开发的，在这种情况下的开发情形如何？敏捷方法可以解决这个问题吗？

Ken Beck：是的，这就是我大多数时间的工作方式，因为我住在南俄勒冈州，出于生计，我的大部分时间仍在编程。因此我一直在进行着跨地区的开发工作。时区是一个挑战，但是最大的挑战是保持团队成员之间的关系。这是一件非常困难的事情。如果你能做到这一点，如果你的团队之间有良好的关系，那么你就能制定出技术的详细信息，例如谁什么时候进行了重构，谁加入了什么代码和已经作出的程序。

记者：对于 Java 和.Net，敏捷方法和极限编程适合吗？

Ken Beck：一个技术平台如果可以让你以较低的成本进行修改，它就更适合敏捷开发。你当然可以在 Java 和.Net 中使用敏捷方法。

记者：在敏捷编程中有什么比较重要的事情吗？

Ken Beck：是的，那就是你在一个技术中需要的东西。在一个技术平台中它真的很有帮助。我的父亲一直在从事敏捷开发的技术方面工作，以汇编语言为微处理器编程。他使用自动化测试、增量设计，他频繁发布小版本，他写绝对可靠的代码。

记者：那么像 AJAX、Ruby 和 PHP 等脚本语言呢？它们适合敏捷编程吗？

Ken Beck：当然。举个例子说，Ruby 来自于以人为中心的语言。你可以通过一种非常具有可维护性的风格编写 Ruby 代码，在很长时间内可以非常轻松的修改。我担心的是当你使用数据库的时候，这种修改可能会突然变得非常困难。

记者：和 Ruby 语言有关吗？

Ken Beck：一般来说是和关系数据库有关。但是敏捷社区正在推出解决这个问题的技术。厂商也正在让数据库接受增量修改。

记者：Web 开发如何？它是否与敏捷开发特别适合？还是它只是另一个应用程序领域？

关于 Web 开发的最伟大的地方之一是其开发非常省钱。而且如果你一直在做部署的话，简单的把它放到一些服务器上就可以了。传统应用软件的刻录成光盘然后进行分发，部署的成本要大很多。

A（iamredeye）：

没什么难懂得词呐。

唯一可能没听过的就是 cowboy coding，上面贴出来得文章里也提到了。其实就是没有软件工程，需求还没弄清楚就可以直接写代码，非常牛 b 的一种方式。我知道很多非软件公司的软件部门实际上就是这么做的。

A（Scott）：

敏捷也是我最近比较关注的问题。因为 CMMI 得确有一些骨子里的问题难以解决，比如必须假定某个版本有一系列相对固定的需求，然后按照需求做估计，在做进度计划，再去跟踪；而实际上这样做很难适应快速变化的市场。

我们现在讨论的敏捷多数只停留在开发阶段，实际上敏捷的领域已经扩展到整个项目开发，甚至整个产品开发各个环节，包括市场。这个领域微软走的比较快，VSTS 里面介绍了一些很好的方法。精益生产、约束理论、系统思维、丰田生产方式、全面质量管理，这些感觉遥远的东西好像突然间都和敏捷挂上了勾。

现在的敏捷已经绝非 03 年我看到的 XP、FDD 那样简单了。最近正在研究，希望能够融入到我们的全员改进系统中，也希望能在实践中不断应用。

A（iamredeye）：

现在各种方法体系很多，每个公司的实际情况和需求又不尽相同。所以不妨什么方法都研究一下，有好用或有启发的地方就坚决拿来主义，反正不掏钱。

每个方法都有它的有缺点，或者适用的情形，其实他们都只关注到了这个世界的某一个局部。经历过不同特点的项目，大到上千人几年，小到几个人几个月，背景文化甚至完全不同， 我的感受是按照项目的特点去抓药方就好了，管它黑猫白猫，必要的时候找只黑猫换条白腿也行啊（其实要换腿或加条机械手臂的情况很常见）。

Btw，最近刚好看到有人批评说微软的敏捷还是不敏捷，其实也许都是外人说歪话罢了。另外正好 Scott 提到了 CMM，有个问题我一直没弄清楚，因为我对 CMM 的了解毕竟没有那么深入。我感觉 CMMI 这个框架并没有限制一定要按 waterfall 的方式实现呐？iterative 当然也可以。但为什么我在很多地方都看到大家评论 CMMI 是 waterfall 的框架呢？

这里没人投“cowboy coding”很正常，因为上这个论坛的大多都是 QA 或流程相关的朋友，这些人在以 cowboy coding 方式的公司混不下去的， 但是目前好像还没有人投 Agile 的票，这个好像不合适嘛，本来我想揪几个 agile 专家出来的。

让弟兄们继续投吧，过一段时间（一年？）我们再回头看看结果会有什么不同。

由于目前尚未有太多成型的应用，所以不好在这里乱说。仅仅提出一个新词，供大家

揣摩：价值驱动。

A（Scott）：

听 Introduction to the cmmi 时，老师得确说过 CMMI 没有限制一定要用瀑布，但是为啥我忘记了：（另外，我最近感觉敏捷提到的迭代实际上也有问题，和我们的实际情况环境不同，我们更多的是用瀑布式增量开发。

最近看了一本书，我觉得讲得很有道理，不论哪种生命周期，其实最根本的是看周转的周期，就像生产里面的周转率，迭代里的时间盒。如果这种周期足够的短，那么增量式开发可能是最好的，同时也是现实中最多被采用的。迭代也足够短，但迭代和增量本质区别在于迭代带来了更多的不可预测性，或者说混沌。

经验也有显性的和隐性之分，迭代就是靠隐形的，即每个人的特殊技能；而增量还是可以依赖显性经验来更加合理的“预估”。

to iamredeye:

我 1，2 个月之前也是这样认为，就像“建立自优化流程框，推动全员过程改进”中提到的一样http://www.step365.com/bbs/thread-64-1-1.html 如果是这样，那我们需要的就是有选择的“拿来主义”。

但我现在觉得这种思路有点问题，问题在于两点，第一这样的组合方式太多了，而且随着新优秀实践的演进，组合会爆发式的增长；第二这样的组合可操作性不强，试想哪个开发人员在进入编码阶段时还要先评估一下哪个优秀实践更适合我，然后再去做，即使是 QA 很了解并且给他们培训，那怎么保证他们把优秀实践做到原汁原味或者更加优秀。

HW 曾经并且一直在沿用优秀实践 DNA，即优秀实践具备哪些可复制特性，这样更易于复制。这种方式和“敏捷与秩序”一书中思想基本一致。但结果呢，据我了解实践并不是很理想，具体数字我现在不得而知了。

我现在更倾向于 Humphrey 的 PSP，让每个开发人员形成自己的技能集，在统一的 TSP 下高效运作。但是 Humphrey 的 PSP 感觉太学术化，实际中存在困难，尤其是在中国职业化严重落后的情况下，我真不知道中国是否有应用 PSP 成功的企业？曾经参加过 2007SEPG 大会，当时联想的一个外包公司曾经做过这样的演讲，公司使用 PSP/TSP 效果很好，假设这种情况真的存在，那接下来该如何做呢？继续实施 CMMI？那不是又回到了我们的老路上。“一堆牛粪插了一朵鲜花，远处看上去很美，但实际上还是一堆牛粪。”“长了翅膀的不一定是天使，可能还是个鸟人。”--我一直不太看好 CMMI，虽然我一直在大公司里不停的在讲解着如何应用 CMMI 在项目中，可我每次都要耐着心思去和

项目经理解释，为什么估计有用，为什么要做一个详细的进度计划，引导了 2，3 年，结果问题依然存在，项目复盘的生产率仅仅能作为规模到工作量的一个除数，而规模我们依然要拍着脑瓜子去猜测；进度计划依然在写，可写完了基本上也没有人再去更新了。

也许 iamredeye 会问，那你看好什么：我不说，还是那个词：价值驱动。

A（iamredeye）：

你误会我的意思了。我说的拿来主义并不是让开发人员按他的想法随意选择或组合，而是针对公司的流程管理人员或所谓的 SEPG。SEPG 针对公司的情况制订最合适的流程，这个过程中可以从任何地方选择任何有价值的框架或经验来参考。

另外再聊两句上面 CMM 的话题。CMM 是个非常空的框架，规定了一些针对流程改进需要关注的点。按 cmm 来实施改进的公司需要针对自己的情况用流程来填充这个框架，用 RUP 也好，waterfall 也好，应该都是可以的。我在网上也看到有人用迭代的方式实现 CMM 框架，另外我个人感觉应该也是可以的。

A（Scott）：

我的意思恰恰与你相反，不是 SEPG 组合一个公司级的流程，而是开发人员自己组合自己的流程（也就是 Humphrey 说的 PSP）。不由 SEPG 来制定的原因就是 SEPG 有点脱离项目组甚至产品线。

SEPG 应该有较少的牛人组成，主要是围绕公司高绩效做文章，比如引入 TL9000，建立标杆管理等。你说网上有人说用迭代方式实现 CMM，CMM 没有限制生命周期。

A（iamredeye）：

了解你说的 developer 自己的“psp”。这个对开发人员的 maturity 要求不低。敏捷成功实施的基础实际上也是成员的 maturity（当然也对项目特点有要求）。如果能达到这种自管理自组织的水平，当然很好。现在很多公司也都引入了一些 bottom up 的方式。

你说的对，一般所谓的 SEPG 是个单独的组织，脱离了项目，这种方式我也觉得有点问题。我说的“所谓的 SEPG”只是一个名字而已，可以有多种实施方式，可能比较好的一种是各种相关开发人员组成的 virtual team，实际上他们主要工作就在项目里，了解真正的需要。也许还需要少数人专门来做一些协调或总体工作，比如你举的例子。当然如果整个公司或项目的成员不够成熟，也没办法实行这种方式。不同公司的情况不同，没有最好的和唯一的，只有对他们最合适的方式

A（oceanchip）：

最主要的还是要根据自身项目以及 PM 来考虑问题会比较合适，不要 XP 与 XP 。

3 项目管理

★是需求规范重要，还是调研人员经验重要？

Q（清水至真）：

需求是项目的纲，做过项目的人都知道，但是如何获取有效的客户需求呢？不少公司在如何获取需求上有各自的方法。有些强调需求规格说明书的结构，讲究需求的所谓“无歧义”“一致性”“可追溯性”等；但是做过需求调研的人都知道还有其他更加多的内容包含在里面。

A（sungubbi）：

能够调研到用户的真正需求并进行合理的归纳、总结、分析，这是最重要的。好的开始是成功的一半。但是需求的描述规范性同样会埋下隐患，如果需求前后矛盾、阅读起来意思模棱两可，也无疑会给产品质量埋下定时炸弹。需求文档毕竟不像代码，有些文字阅读起来不同的人会有不同的理解，所以即使有了文档，有些内容也是只可意会不可言传，所以适当的沟通是很有必要的。

正确地获取、描述、传递需求最重要。

形式上的统一、规范则有利于企业财富的积累、沉淀、传播及复用。

A（jaedonger）：

按照 CMMI 的理论，还是“需求规范”重要，需求要是不明确，产生变更，项目的后期会让你感到很难受。至于经验，如果过分看重的话，那何须 CMMI 呢？还是要和相关人员多沟通，明确需求最重要。

A（lily_014）：

两者都重要，相辅相成，事半功倍。

需求人员的经验能够更好的去获取用户需求，好的需求人员在用户也不明确需求的情况下，会以很好的提问方式，不断的去引导用户。最终明确需求；而需求规范是为了后期的开发，维护等等。

A（cecilia）：

各人对相同文档的描述会有着不同的理解，需求规格说明书描述若存在歧义，那对于后期的设计说明书及测试用例也都是有影响的，需求管理变更及维护也不方便，不好管理。所以说需求文档及调研（经验）同样重要，只有将你的需求调研的经验转化成文字描述，

才能让更多的人明白你的想法，那样做出来的才是有价值的。

A（iamredeye）：

这两个东西并没有矛盾，如果有经验的需求人员知道需求应该包含更多内容，为什么不想办法把他们的想法通过模板或 guideline 体现出来供其它人参考呢。这个公司没有“流程改进”吗？

A（lily_014）：

又不是需求、设计、开发、维护自始至终都是那一个有经验的人在孤军奋战。

C（fishred）：

需求规范和调研人员经验都很重要。需求规范了，可以减少必要的歧义，需求文档也具有可维护性，即使需求人员流动了，设计、开发能够顺利进行下去，降低风险，也利于企业相关经验的积累、沉淀；调研人员有经验，可以顺利诱导客户的需求。

4 QA/EPG 专区

★如何进行质量分析？

Q（xixiaojing666）：

我们目前的质量分析做的很简单，只是按照 PA、过程域、NC 的严重程度进行简单的统计。

大家是如何进行质量分析的？听说可以使用工具、图表、数据等等？

A（fishred）：

1、一方面根据度量数据进行分析（度量数据来源度量计划）；

2、另一方面，针对发现的项目问题进行分析，一般根据 PA 来分析；

至于工具嘛，我们目前用 Excel 自带的统计、图表功能；只有缺陷管理用工具（TD）进行统计了；

A（stop-start）：

乍一看这个标题，我的第一个反应是项目的质量分析，再细看一下原来 LZ 想交流的是 QA 所做的质量分析，如果只是 QA 所做的质量分析的话，感觉与“全员过程改进”版块中，sungubbi 最近提出的“一起聊聊：如何量化评价项目执行情况”这个话题类似，她这个话题涵盖了“QA 对过程执行情况的评价”与“如何使用测量的结果”两个问题。LZ 可以参考一下那边的一些交流内容

A（stop-start）：

另外我很赞同 fishred 的“质量分析分为两方面，一方面是根据项目度量数据进行的，一方面是针对 QA 发现的问题进行的”这个观点，我想再把根据项目度量数据进行的质量分析这个话题引申地具体一点，当然，以下都是个人的一些观点，希望得到各位的指正和补充：

根据项目度量数据进行的质量分析，主要是针对项目实际数据超出计划预估值规定偏差范围时所做的的分析，比如规模、进度、缺陷数等，有的要求比较严的项目还会开展缺陷的根原因分析，这一般都是在项目阶段结束及项目结项的时候实施。分析的主体责任人是项目负责人，QA 在这时的职责主要是核对数据，如果有异常数据出现，提醒或协助项目负责人进行分析。这种分析一般用 exl 文档就可以实现，根原因分析一般会运用到鱼骨图这种工具。

A（lily_014）：

商业目标不同，侧重点也不同。

进行质量分析，我理解的是度量（不知理解是否正确），根据不同的目标定义相应的质量指标。至于如何进行质量分析，同意 stop-start 说的结合 sungubbi 的贴子讨论。要形成了相应的性能基线才能够分析。否则也只是进行了数据的采集工作。即也就是进行了针对某个具体项目的进度、成本等估算的偏差。个人意见。

C（fishred）：

1、 一方面在条件允许的条件下建立性能基线。

2、 根据度量计划进行度量数据收集，并进行分析，并适当地使用图表等表示。

3、 针对项目问题，可以按过程进行分析，并提出改进意见。

★老板的力量——一次还算成功的 EPG 会议

Q（sungubbi）：

一字一句的念问题、昏昏欲睡、嚷嚷着要退出EPG，每次参加EPG的会议都是这个样子。所有兼职EPG成员对正在讨论和分析的问题都似乎漠不关心，要么加入批判的队伍，要么极力排除与自己的关系。这就是我参与过的EPG会议。

担任 EPG 组长后的第一个 EPG 会议已经被我拖延了一段时间，面对新上任后各部门的不配合，实在是不适合匆忙地开这样一个会议。我把这个会议的时间定在了季度测量分析之后。

完成了一季度的测量与分析，来自产品质量、服务、过程执行、效率各方面的数据，以及暴露出来的问题，有了用事实说话的基础。我找到了老板，希望他能够参与到EPG的工作中来。老板瞪大了眼睛：这个事情要你们去做啊，怎么让我来做呢？进行了一翻解释和沟通，老板答应先参加一下EPG的会议。

一切都那么顺其自然，时间到了，大家稀稀拉拉地来了，一副课间休息的样子，提早发出去的会议资料，几乎没人看过，我到老板办公室把他请来，顺便把与他一起的另一副总一起叫上。气氛MS有点认真起来，每个人都不再是斜靠在椅子上。从数据的展示开始，就有了讨论和交流，老总们不时插上几句追问原因，并且在争执中帮助确定措施责任部门，计划中的改进事项，也顺利地分配到了相关的成员或部门身上。我得意地笑，我得意地笑～～～将要结束的时候，把EPG成员的工作任务进行了确认，老板主动提出要把骨干的员工轮流送进来学习和参与改进————老板的领悟力就是高！！

一次还算成功的EPG会议，要传递的信息传递了，要完成的工作分配了。会后，老板请吃饭，一致认为这种行式很好、要坚持。我也从中体会到了让决策层参与改进活动是多么的有效。不过，我深知这会艰巨步伐才刚刚开始，如果不能把即定的目标完成，一次两次，如果连决策者都对它失去信心，丢饭碗的时候就到了。

A（chfyx）：

不错，我认为你这次成功的 EPG 会议关键点在于两点：

1、用数据说话

2、请高层参加

A（Scott）：

如果不能把即定的目标完成，一次两次，如果连决策者都对它失去信心，丢饭碗的时候就到了

赞成，所以如果不是员工自己主动改进，而是 EPG 推动的改进，责任就落到 EPG 的头上，做好了受益的是大家，做不好惩罚的是 EPG。建议你把改进的工作职责逐渐分摊下去（你们的 EPG 好像是兼职，应该是来自产品的人吧，并且以 Top 任务的形式进行跟踪，抄送老板，这样他们的动力就更大了。

A（iamredeye）：

感觉瓶颈是你的老板们

A（sungubbi）：

是的，都是兼职的方式参与工作，除了质控们的几人是专职之外（其实他们还兼着 QA 的工作），其他人员来自生产部门。

一定要让他们意识到这些工作需要大家一起来改进，抛出问题，强调要解决的问题和目的，让他们来提供方案。

我接触过的几位老板都是不错的，可能他们不太懂，但是一旦领会了，就会去做，而且在各个场合都会强调质量、管理、改进的重要性，如果能够与他们多做一些交流，也能为我们的改进带来更好的思路、了解公司的需要。但是太少人与他们做这样的沟通。

A（iamredeye）：

我不是说你老板的 personality，我是说 top mgmt 的 maturity

A（Scott）：

那就建议你的大老板们去参加一些高级管理的培训，提升一下自己：）

A（冰封伯爵）：

我现在也是公司临时 EPG LEADER,我们的 EPG 成员,都是由经理级别的同事组成,我们老总也是我们 EPG 小组中的一员,我们平常 EPG 开展工作一般都比较顺利.

目前我们这种做法也有一个弊端,基层配合不是很好,配合到是"配合",给你磨时间,"拖";提也提不出什么好的建议;可能是我们推广不够.

希望大家多给点建议,分享下一些成功的做法.

A（lily_014）：

你们的 EPG 会议除 EPG 组成员外，组织级和项目组 Q A 要参与吗？或者是其它部门的总经理要参加么？ Q A 要作各个项目的情况汇报吗？

A（stop-start）：

先为一次成功的 EPG 会议庆祝一下，然后问一句：可不可以把测量分析的报告共享一下，让大家学习一把

A（cecilia）：

领导力和执行力

A（rainshadow96）：

EPG 组长由高层担任比较好。

C（fishred）：

1. 用数据说话比较明智；
2. BOSS 支持了，这样以后大家对数据就会重视起来；
3. 过程改进的工作 EPG 只是组织，改进的来源还是来自项目组本身。

5 Test 专区

★ 测试方案和测试计划的区别

Q（MSN 群某网友）：

测试方案和测试计划的区别？

A（lee_huo）：

一、测试计划：全过程对测试组织、资源、原则等进行规定和约束，并制订各个阶段的任务以及安排时间进度，提出各项任务的评估、风险分析和需求管理。

二、测试方案：描述需要测试的特性、测试的方法、测试环境的规划、测试工具的设计和选择、测试用例的设计方法、测试代码的设计方案。

三、测试计划是组织管理层面的文件，从组织管理的角度对一次测试活动进行规划。

四、测试方案是技术层面的文档，从技术的角度对一次测试活动进行规划。

五、测试计划要明确的内容：

1、明确测试组织的组织形式；测试组织和其他部门关系，责任划分。测试组织内的机构和责任安排。

2、明确测试的测试对象（明确测试项，用于后面划分任务，估计工作量等）。

3、完成测试的需求跟踪。

4、明确测试中需要遵守的原则。测试通过/失败标准，测试挂起和回复的必要条件。

5、明确测试工作任务分配是测试计划的核心：

- ✓ 进行测试任务划分
- ✓ 进行测试工作量估计
- ✓ 人员资源和物资分配
- ✓ 明确任务的时间和进度安排

✓ 风险的估计和规避措施

✓ 明确测试结束后应交付的测试工作产品

六、测试方案的具体内容：

1、明确策略

2、细化测试特性（形成测试子项）

3、测试用例的规划

4、测试环境的规划

5、自动化测试框架的设计

6、测试工具的设计和选择

七、测试方案需要在测试计划的指导下进行，测试计划提出“做啥”，而测试方案明确“咋做”。

C（step365he）：

测试计划的第一步工作是建立一个高层次的测试策略。而这个策略中最关键的部分是确定测试范围。考虑这个产品有多少是可测试的。我总结了几点帮助大家更好的确定测试范围。

第一个是考虑设置测试优先级

第二个是考虑对新功能以及修改旧功能的代码进行测试

第三个是使用等价划分和边界值分析技术来减少测试工作量

第四个是测试大家觉得最有可能出问题的地方

最后一个是关注用户最常使用的功能或者最常用软件配置下的功能。

★ 如何划分缺陷【类型】【等级】【原因】？

Q（lqpz）：

高质量是所有项目追求的目标，而评审和测试是发现缺陷的最重要手段。为了更好的评价工作产品的质量，同时预防同类错误以后再次发生，必须对缺陷进行度量、分析。那么，你们是如何划分缺陷的“类型”“级别”“原因”的呢？

是根据缺陷等级：致命，严重，一般，建议？或者根据缺陷类型？还是根据缺陷原因？

为了建立各项目工作产品质量的比较，是不是得有一个统一的缺陷基准？如 1 致命缺陷=2 严重缺陷=4 一般缺陷=？

A（stop_start）：

首先要说的就是缺陷等级、类型、原因的划分会因公司或项目管理需求的不同、业务类型的不同而不同，公司 EPG 在参考其它公司分类情况的基础上，关键任务是要根据自身的需求和业务情况总结出适合自己的分类情况。

接下来聊聊缺陷等级，个人认为缺陷等级的划分要看公司或项目组关心哪些数据，比如关心缺陷的严重程度，那等级可能可以分为致命、严重、中等、一般等几个级别；如果比较关心缺陷影响范围，那等级可能可以分为很大、大、中、小等级别；如果比较关心缺陷的处理的紧急程度，那等级可能可以分为紧急、高、中、低等级别……。当然，纬度还有很多种，关键是看你关心什么数据，而且这些纬度也可以结合在一起，以加强对问题的管理。

虽然说等级的划分很多是定性的，但最好也能够有个比较清晰的定义，以方便推广和选择，减少歧义。

再有就是上面提到的“为了建立各项目工作产品质量的比较，是不是得有一个统一的缺陷基准？如 1 致命缺陷=2 严重缺陷=4 一般缺陷=？？？”这样做的目的是为了项目绩效考核吗？一般很少有公司会这样做吧，个人认为等级划分是为了更好地对问题进行管理和分析，而不适合用定量的方法去转换各不同级别缺陷的数量。

缺陷类型：

缺陷类型在不同的阶段划分是会有所区别的。

比如需求、设计阶段，主要是以评审作为发现缺陷的手段，缺陷数据主要来源于评审的报告或记录等，缺陷类型可能会划分为：对客户需求理解错误、客户需求不明确/错误、数据结构/数据库设计问题、参数设定问题、接口设计问题、异常分支处理错误……等等。

比如到了编码后的阶段，主要是以各种测试作为发现缺陷的手段，缺陷数据主要来源于测试或者里程碑报告，缺陷类型可能会划分为：极限值、最大值、最小值错误、Retry 处理错误、指针错误、初始设置遗漏/错误、判定分支遗漏/错误……等等。

缺陷类型有可能需要分级划分，比如组织级定义一个比较初略的分类，项目组根据自己管理的需要，可以对组织级的划分进行细化，形成二级分类等。

缺陷原因：

缺陷原因的分类可以参考制造业运用很成熟的鱼骨图分析法，这种方法将引起缺陷的原因归纳为人、机、料、法、环这五个类，在我们软件行业，机与料的影响相对

较少，而沟通这个因素会比较突出，可以单独列出。从上面的这个思路展开，缺陷的原因可能就可以这样分：人员某方面技能不足、与顾客的沟通不足、项目组内沟通不足、开发环境不合适、使用的工具/技术不合适等等。另外，建议可以运用鱼骨图对缺陷进行逐层分析，以定位缺陷。

C（step365he）：

感谢 stop_start 精彩的总结。提醒大家在运用这些数据时，对事不对人，这样才能保证获得好的产品质量。

6 CM（配置人员）专区

★【讨论】配置项粒度 & 组织标准工作环境

Q（lqbz）：

1. 不知道大家的配置管理用什么工具，要不要手工维护【配置项状态清单】？

如果有 50 本需求文档，50 个画面原型，你会如何划分配置项呢？

如果每个文件一个配置项，你将维护一个庞大的配置项状态清单；如果 10 个文件为一个配置项，你将在其中一个文件发生变更进入基线时，对 10 个文件进行审核——选择哪个比较合适？

什么配置管理工具做这个最好用？能最好的实现【需求追溯】和【变更管理】？

2. CMMI1.2 要求的【组织标准工作环境】，关于这个的过程文件或规范中，里面都包含了些什么内容呢？

A(iamredeye)：

怎么会那么多需求文档呢？如果有关联并且 owner 是一个人，不如合并到一两个文档来维护。

虽然不知道你说的【需求追溯】和【变更管理】具体到什么程度，不过我想任何工具都是来支持流程的。工具之上的流程一定要有(有些情况下甚至可以不需要太多工具支持)。任何工具自己“可能”都没有办法实现这些功能。

比如我发现一些公司尽管有了配置管理工具，但是实际上并没有真正执行变更管理流程。说明还是有人对变更管理及 CM 工具没有理解。

你说的【需求追溯】如果我理解的正确的话，也许可以试试 requisite-pro。

A(rebeccazxy)：

1. 关于配置管理工具，我们使用的是 CVS，虽然功能不像 CC 那么强大，但是一般也够用了。

关于手工维护【配置项状态清单】：我们是手工维护的，目前还没发现 CVS 能有这个统计归纳的功能，即使有，也不如自己定制来的方便。

关于配置项的划分：与 iamredeye 一样，我也奇怪怎麼会有这么多需求文档呢？需求文档也好，界面原型也好，如果是相对独立的，那就可以划分为一个配置项。具体来说，50 个需求文档，若隶属于不同项目，那先按项目划分，然后按照需求调研书、用户需求、软件需求划分开来。一般来说，描述用户需求的文档可以放在一起作为一个配置项，当然能合并成为一个《用户需求说明书》文档是最好不过的了，没必要分散放置。

其实配置项的划分细度，是根据实际情况来的，最主要的目的还是为了方便管理和版本控制。举例来说，对于一个小型项目，或中型项目，可能你会把所有的界面原型放在一起作为一个配置项；而对于一个大型项目而言，你可能会把所有的界面原型划分为若干个相对独立的配置项。所以还是看你的实际情况而定。

关于能最好的实现【需求追溯】和【变更管理】的配置管理工具：我接触过的配置管理工具只有 VSS、SVN 和 CVS，这些都是比较基础的配置管理工具，能实现最简单的版本演化记录，而【需求追溯】和【变更管理】我们并不是使用配置管理工具实现的，而是使用《需求跟踪矩阵》和《配置项变更记录单》来管理，所以具体什么配置管理工具能实现你的要求我不太清楚，同问。

2. 关于【组织标准工作环境】：你说的是 OPD 的 SP1.6 的要求吗？我是这样理解的，组织建立工作环境标准，主要是指组织能提供的基本工作环境条件，以便项目在确定工作环境时选择使用。在我们实际的操作中，主要从硬件、软件等多个方面列出组织能提供的，从而制定出【组织标准工作环境】，项目在建立项目工作环境时，从【组织标准工作环境】里面选择，若里面没有的，那想办法解决（借用、购买等）。不知理解是否正确，请高手指正。

A(guanxb)：

个人觉得，作者混淆了一些概念：

配置项的颗粒度 与 产品部件的颗粒度

环境 与 资源

Q(lqbz)：

谢谢 iamredeye and rebeccazxy ！

1．需求文档是同一个项目的，项目组习惯每个功能模块一个需求文档，合并的可能性不太大；而且合并的话，文档太大也不好维护。如果多个需求文档作为一个配置项的话，单个文档的版本怎么管理和控制，变更时会审核这个配置项里所有的需求文档吗？

2．ebeccazxy 说的好像是资源，我理解的是开发环境，目前只包括了硬件配置要求，基本安装的软件及版本等，不知道大家是怎么理解的，还有什么内容？

A(iamredeye)：

不管分成多少个文档，其实本质上对变更管理和需求追溯也没任何差别。在工具里多 check in/out 几下呗，流程上没差别。就是感觉比较奇怪，不想合并也行，rar 压缩一下。

一个文档变动了另外一个是否有影响？问 owner 不就得了，不用那么机械。如果你们的 SCM 真执行的那么机械，问题就大了！

另外你问的这两个变更管理和需求追溯的功能与工具基本上都无关。我相信只要定义好流程，VSS，CVS/SVN，ClearCase 都不妨碍其功能的实现。不要把流程和 CM 工具混淆了。这两个功能一个是项目管理的需要，一个是需求管理的需要。

C（深海）：

在配置管理中，配置项的颗粒度划分，以方便管理和版本控制为目的，并没有一个绝对的划分标准，根据项目实际情况而定。对于配置项状态清单，大部分实践还是由手工维护的。不管配置项粒度如何，本质上对于变更管理和需求追溯是没有差别的，与所选用的 CM 工具无关，只要定义好流程，任何工具都不妨碍变更管理和需求追溯的实现，重要的是要定义好工具之上的流程，真正的理解流程而非机械的执行流程。

第三部分：会员原创

1 思步 PTG 小组组建问题之我见

作者：思步 Scott

1、假如你刚刚进入一家CMMI3 级的软件公司做QA，但是很快你发现公司实际上连 2 级都没有，有些项目甚至都没有立项，而是以紧急任务的形式在开发，需求直接从各个部门下来，没有需求分析与设计，直接编码和测试。而你的领导恰好要让你负责这个项目。请问你在接下来的 3 个月打算如何做？

这种情况也是比较典型的，但个人感觉要根据个人以及公司的情况来处理。现在 QA 的水平一般做不到让企业达到真正 CMMI3 级的水平，即使有这样的能力公司的实际情况也可能做不到。“冰冻三尺非一日之寒”，应该承认这个现状，那么解决这个问题也绝非一朝一夕的事情。

第一个月你应该处于试用期，你需要做的是：了解公司的造成这个问题的原因，是文化？是咨询公司能力？是特殊情况？同时，尽量按照项目对 QA 统一的要求去做，而不是把自己之前的想法试图应用在项目中；如果没有一个统一的要求，那就向其他 QA 学习一下如何做的。

第二个月你可以抓住其中的一个问题来解决，但这个问题最好是比较典型但可以解决的问题，你可以考虑 2-3 个议题，比如统一需求入口、设立项目管理里程碑，同时给出解决方案（改进目标、实施计划、试点、实施，也就是 PDCA 的方式），然后要和高一级的人员沟通，比如项目主管（项目经理的经理）、QA 经理、EPG 成员，然后让项目主管来选择一个议题，因为改进的主体永远是他们。这里试点最好选择一个月左右。

第三个月就是执行这个试点，同时注意总结试点成效，做到有始有终。

不要把自己能力看的太高，公司比你牛的人可能有很多，他们没有能力改变公司现状，就说明这种存在有其原因，你需要的是深入了解这个背后的原因，然后逐步开展改进工作。少说多做对于这时的你是最重要的。

2、假如你刚刚进入上面的软件公司，有一天，你收到另一个已经工作了 1 年的 QA 发的一封邮件，邮件的大体内容是：你刚刚统计的数据存在问题，而这些数据刚刚用来考核了几个项目组。同时，这封邮件还被抄送给了你的领导以及所有被考核的项目经理。请问你打算如何处理这件事？

首先，应该回复所有被考核的项目组以及相关人员，请暂时不要使用此数据，你会尽快了解问题并解决这个问题，同时感谢大家的支持！

然后要和这个 QA 了解具体统计错误的原因，如果真的是存在问题（这种情况几率较大），那么先改正此数据，然后再次征询此 QA 的意见。没有问题后，自己要总结原因。最后，再次给所有人发送正确的数据，同时说明此事发生的原因以及避免此类事情发生的办法，并对带来的麻烦表示致歉，对此 QA 表示感谢，希望大家一如既往的支持 QA 工作！如果数据没有错误，也要了解误解的原因，最后回复给所有的人邮件，说明这时一个误解，数据依然正确，但同时要感谢此 QA 对你的协助，也希望大家一如既往的支持 QA 工作！

也许这对你是一次公关危机，但危机的背后也是一次机会，不要把得失看得太重，尤其是对于一个新 QA。心有多大，舞台就有多大。

3、假如你刚刚进入上面的软件公司，你的领导让你负责一个项目。在你和项目经理初次沟通时，你了解到这个项目由于进度非常紧张，没有及时立项，而目前已经在进行编码工作了，并且只有一个用 Excel 记录的工作任务分配表。请问，在项目管理方面你会给项目经理开出哪些不符合项或者提出哪些建议？

这个问题要根据项目经理（PM）的能力而定。如果 PM 是技术高手，那你需要了解此项目的目标和范围（需求）、进度计划，以及 PM 如何跟踪项目（Excel 工作任务分配表、周例会等）。接下来先从需求进展入手，要从项目组成员的角度来了解 Excel 工作任务分配表是否有效，如果存在不能及时跟踪进展的问题，即存在进度延迟的风险，你需要找出原因以及解决办法反馈给 PM。如果 PM 不接受，你可以邀请高层领导参加项目例会，在例会中用数据说明这个风险，以及建议的解决方案。范围明确后，要和 PM 讨论项目的目标，到底是要保证进度还是保证质量（我们不是故意分开此二者，而是相对来说有一个侧重），如果是保证进度，那要和项目经理一起分析对于进度零偏差存在什么样的风险，制定缓解方案（比如设定里程碑）和应急方案（对需求排序，裁剪项目范围），并在状态报告中进行跟踪。最后，一定要让项目组进行（双）周例会，以及经验教训的收集与分享。如果人员存在离职风险或者新手，一定要建议 PM 考虑人员培养与备份。如果你的公司需要在 EPG 立项，那么在以上事情忙完后，可以让项目经理进行立项。

项目管理领域，最重要的也是这几方面：范围、风险、沟通、人力、时间、成本、质量其实在这简单的做法中已经涵盖了，但它并没有“僵化”。没有强制 PM 去补充估计、进行立项，这些在此时都已经不是能够推动项目成功的内容。换位思考是很重要的，但也需要丰富的经验去充分理解什么必须做、什么可以舍弃。如果你的经验不够丰富，可以寻求其他人的帮助，但不要变成项目经理的助理。

4、假如你在软件公司做软件QA，有一天领导突然告诉你，公司目前需要做硬件开发，需要你去做硬件QA。假设，你不想做硬件QA，但也不想失去在这个公司工作的机会，请问你会如何做？

当你从事QA这个行业时，我相信有很多人是认为它很有挑战或者它能学会很多的知识，比如软件工程、项目管理、质量管理等等。有变化才有发展，这句话每个人说起来容

易，但自己做起来总是很难。我曾经和自己说要专注，只关注软件其他的忽略，但当我不断学习和成长的时候，我发现我和硬件、产品、甚至市场、财务都有接触，我开始逃避不去接触，但总感觉像在原地打转转，而当我真正放开心态去学习后才发现自己的能力又有所提升。

就这个问题来说，你不想做硬件 QA 如果有更多是主观的原因，那么我建议你要说服自己，毕竟 QA 需要更多的知识。但如果是客观原因但可以短期克服，建议你和领导坦诚客观原因，并尽量接手，同时希望考虑继续招聘适合人员接替你的工作。如果不能克服，也要坦诚和领导说清原因，并请他谅解。

工作不是创业，即使是创业，有时也不能太自由、随意。年轻时应该多承担些责任、勇于接受挑战。年轻就是资本，的确，年轻是你积累丰富经验的资本。

5、请试着翻译以下内容：

Do CMMI appraisors assess customer satisfaction or PA goals only when reviewing projects? Is meeting PA goals in practices correlated to degrees of customer satisfaction? In my view it is not. A project with customer satifaction level 5 could be assessed at any level rating. And an organization has a corporate wide process institutionalized and be consistent in all projects with customer satisfaction level 5 but could be rated at C/M L 2 or even L 1.

If customer satisfaction is not measured in CMMIappraisal, an organization can create practices for CMMI purpose only to reach ML 5 and fail all projects. I am talking about possibility of such scenario if there is such a company to do such experiment. Meeting a goal of a PA is not something objectively determined. In the world of complex behaviors where software projects are, you can not separate the observer from what is being observed. The interpretation of what is observed changes from observer to observer and from time to time.

这段翻译来自国外论坛上的一个激烈的讨论，我拿出来有两个本意，第一是 QA 需要具备熟练阅读并理解英文的能力，毕竟这个行业的新鲜知识多数来自海外，通过英文你能够更多的接触国外的先进思想。第二，做为 QA 不能太像 CMMI 的 PPQA，只关心产品和过程的审核，那样就被 CMMI 化了。CMMI 对于 QA，就像新老七工具对于 QA 一样，仅

仅是一种工具、一种方法，你要了解这些工具的优缺点，到底可以在哪方面帮助你解决问题。就像文章所说，“如果 CMMI 评估不能度量客户满意度，即使一个组织能够创建实践从而达到 CMMI5 级，但可能所有项目都是失败的。”

知其然也要知其所以然，在软件领域没有银弹。

2 一个 QA 的成长—送给那些想要离开 QA 岗位的人

作者：思步 Scott

很久没有上网，也很久没有写东西了，最近一直在看书，在工作，也一直在生气。今天突然间特别想写，那就再多写一些。

这几天又和以前 HW 的同事探讨了很多问题，包括质量文化的宣传、标杆管理等，可能是因为自己更多的从产品的角度来关注问题，很多以前学习的东西又一次变得清晰了一些，而不是那么的模糊。昨天晚上睡觉时，我觉得就像 zhangcb 说的，在这个网上并不是有很多很牛的 QA，他们可能都没有时间在这里闲谈，我一开始也是这样认为的，但现在感觉这是另一种收获，一个从无到有的收获，看到大家遇到的很多问题，又多么的似曾相识，可为什么大家会有这么的区别呢？

我觉得这里缺少一种精神，一种聪明的执着精神。执着的原因是因为任何高层次的达成都不是一朝一夕的，需要长期的坚持，我从 99 年开始看书，从 Windows 98 一直到最近的敏捷管理，一路走来我看了太多的书，人说“熟读唐诗三百首，不会作诗也会吟”，我觉得是这个道理，很多书我看得时候都是一头雾水，可后来不断的在工作中思考、应用，总时常有让人惊喜的收获；三年前，我老婆指出了这个问题，应该 宠辱不惊 ，现在逐渐明白这个词背后的意思，那就是全面的考虑，有人也叫它成熟。QA 需要这种执着，也需要这种成熟，执着会让你影响项目组中的每个人，会让你坚持做每一件改进的事，成熟会让你能应对各种人和各种事，与其说 QA 是在影响着项目，不如说 QA 是在影响着项目组中的人。

在执着前面加上聪明，这是非常重要的。机会是用执着创造的，可一旦你变得不聪明而是冲动，那机会也就自然消失了。就像我们的工作，本来是一个很好的岗位，可就是由于看不惯主管、或和同事不合，结果一冲动便只能离开另辟公司。这些都是我的亲身经历。还记得那句话么，冲动是魔鬼。真的应该时刻提醒自己。千万不要和项目经理吵架，千万不要在背后骂你的客户，千万不要对你的老板指手画脚。。。你需要做的只是做好本职工

作，而这点做好了也是非常非常困难的。热爱你的工作，聪明的执着至少 5 年以上，你一定会有机会的。否则，你永远在起跑线上。

也许看到此，你会有些冲动，也会有些不懈，不管你怎样想，你还需要的就是具体的如何做。

如果你只是一个刚入行的 QA，你需要的就是学习，学习各种各样的东西，CMMI,XP,FDD,ISO9000,TQM。。。一切你感觉能用上的东西都要去慢慢学，记住是慢慢学！同时要踏实的工作，记住是踏实的工作，最好能在一个稍微好的环境中，如果环境不能保证，那一定要相对自由，因为你学要更多学习和体会，也需要一定的锻炼；你需要收获知识、金钱，但你更需要收获失败。这可能会花去你至少 3-5 年的时间。

如果你已经有了些基础，自认为有些经验，你应该试图去梳理你的知识，让它系统化，因为当你站在更高层面去关注问题时，你应该更加谨慎，你的一言一行，无论对于你还是对于产品、项目都是非常重要的，你需要的尽量减少失败，尽量保持少量的从头到尾的成功。也许你更需要的是名声，这个名声会让你更好的活下去。

如果你真的已经经验丰富，那就不需要我在说什么了，因为你已经学会了思考，学会了抉择。

QA 只是个叫法，一个岗位，是一个充满智力挑战的角色，有时候我觉得它像古代的军师，可以指挥千军万马，连大将军都要听你的；有时你又像一个说错话的谏官，被无情的扔在了一边，好像所有的人都可以骂你两句。但那只是感觉，QA 只是 QA，是存在于我们这一代的特殊产物。Tomorrow is another day.

第五部分：思步沙龙

1 北京线下活动

思步沙龙启动

活动全体人员合影留念

祝贺思步网首次线下活动取得圆满成功!

首先代表思步网发表感言。在这里要感谢的人太多，感谢一直给予思步支持的会员们、同行朋友们，感谢协办方中科方德给予我们这次活动的大力支持，感谢思步特邀嘉宾 Jetor 给这次活动前来助阵，感谢思步会务组人员为这次活动的付出和努力，感谢……

总结几点：

总体来说，是一次举办较成功的活动，给思步的首次活动打响了一个好彩头，为长期的作战活动奠定了基石，无论是活动前期准备还是活动期间的协调事宜都相当充分和有序，以及会议期间互动的也非常不错，气氛也相当活跃。最后 Jetor 还给大家讲解了 L4 的相关知识，即建立性能基线模型。

根据北京下线活动参与人员所提供的资料，主要有以下几点：遗憾、不足及益处。

遗憾：

1、 启动时间比原计划定的延迟了近 1 小时，以致于后期的时间安排较为仓促并且还省去了多个环节；

2、 由于某些原因，以前的一同事签到后没能参加就返回了，对此感到非常遗憾；

3、 会议室的布局，有个讲台似乎更好；

4、 思步介绍过多，由于时间原因进行了压缩，以致 lily_014 在作思步介绍过程中非常紧张；

5、 第二个主题讲座，即协办方主讲时再控制或压缩时间似乎会更好；

6、 案例讨论时，选择第二个case进行探讨似乎会更贴切本次的主题以及提升讨论价值；

7、 既然称沙龙活动，感觉没必要搞得太正规化，如果供更多的时间让大家交流效果会更佳，因本次讨论的是QA的角色与价值以及QA的发展，而且参与的人员有不同的角色：PM、EPG、DEV、QA等，不仅仅是介绍，更有讨论价值。若留更多的时间让大家发表自己的见解那就太完美了；

8、 由于天气原因，6 点多就暗下来了，以致于户外感受软件园优雅的办公环境合影留念取消；

9、 最后不知是否属行业的整体状况，美女们居多啊，大声呼喊帅哥们在哪里……

10、因为需要抓拍PP，所以没能跟大家一块参与游戏环节，会后时间也不早了，聊得都不尽兴；

不足：

1、没有充分考虑大家的交通状况，导致活动延迟半小时开始。以后在准备的时候，提醒大家充分考虑交通，以保证准时到达；

2、思步网介绍的PPT过于冗长，需要精简变更为 10 分以内为好；

3、发表人员有点紧张，需要多加锻炼（下次就好了）；

4、需要控制协作方的发表时长；

5、需要增加更多的交互及团队游戏内容，本次因为时间缘故，最后的团队游戏没有得以实施；

益处：

1、思步网组织方与协办方中科方德，协作顺利；

2、讲座内容、游戏、案例准备充分；

3、人员参与度高，活动气氛不错；

2 案例讨论

案例一

有如下一个 CASE：

“有一群小朋友在玩耍，而那个地方有两条铁轨，一条还在使用，一条已经停用，只有一个小朋友选择在停用的铁轨上玩，而其它的小朋友全都在仍在使用的铁轨上玩。很不巧的，火车来了，小朋友们已经来不及躲闪，而你正站在铁轨的切换器旁，因此你能让火车转往停用的铁轨，这样的话你就可以救了大多数的小朋友，但是那名在停用铁轨上的小朋友将被牺牲，你会怎么办？”好了，我们一起从 SPI 过程中涉及的不同角色、角度、方法等方面来探讨，到底我们走那一条路是正确的？

可以从以下多个方面来考量：

1、这个 CASE 本身，是不是还存在其他的可能，为什么旧铁轨没用了？作废了？还是旧铁轨更危险？（当然，这个不同的人有不同的看法）

2、抛开 CASE 本身，联想企业实施 CMMI 过程中的一些策略与选择，是不是我们只站在自己的角度考虑问题？有没有站在市场的角度？或者 BOSS 的角度？（当然，可以有很多的角度与策略）

思考 1 ： 大局上面仍然有另一个大局

思考 2 ： 公平永远有不同角度的公平

思考 3 ： 这个游戏只有站在切换器旁边的人可以决定结果

C(lily_014):

1. 选择停用的铁轨

理由:无论是什么角色，做任何事情都要站在 BOSS 的立场，站在公司的利益考虑问题。倘若这次选择相当于公司来说是生存问题，对公司来说 1 个小朋友的价值比多个小朋友价值少，所以 对公司来说，会选择目前的利益，同时可能会面临很多未知的风险。作为过程改进来说 也是持续改进的过程，如果这次的选择会带来更大的风险，那么也将形成组织的经验库。回到企业来说 说明了 QA 在实际的工作中还是要根据实际情况随时应变。

2.选择正常的铁轨

选择这个的理由是做风险的评估：

1）设备的检验

2）人员和时间的风险

3）游戏规则，虽然说不论是废弃的还是正常的铁轨 这些小孩都违背了游戏规则。但目前考虑的责任度的问题。回到企业来说 要按照企业的规则来办事，不能背违了企业的规章制度。