**[解释传统与敏捷方法最贴切的故事：大象与猴子](http://www.cnblogs.com/henryhappier/archive/2010/07/05/1771127.html)**

很久以来我都对传统的软件工程方法和盛行的敏捷方法的联系与区别郁闷不已，因为每一个领域都是那么庞大，作为一个非专业人士，要能清楚地把握其中的关系，确实不是件容易的事情，直到我看到下面的这个故事（摘自《平衡敏捷与规范》Balancing Agility and Discipline: A Guide for the Perplexed，邓辉 孙鸣译 清华大学出版社）。在敲字的过程中我自己对一些名词对照着两个软件开发方法中的常用概念进行了注解，不当之处，还请大家指正。

很久以前，在一片充满隐喻的土地上，住着一头大象（喻传统开发方法）。很多年来，这头忠实的大象一直都是他所居住村庄主要的食物采集者，并且非常清楚这个村庄需要什么（喻工程前的事先计划性）。他在丛林中修建了一条路（喻最佳实践），这条路总是能指引他找到最好的根茎、蔬菜、坚果和水果等食物（喻项目的功能）。他知道哪种水果用鼻子可以够得着，也知道哪种水果需要鼻子去晃动才能摘取。他很强壮，能够一次带回足够好几天吃的食物，所以他总是预先估计这个村庄的需要（喻典型的项目需求），并提供恰当的供给。他恪尽职守，整个村庄的人都很感激他，并认为他的工作很有价值。

唉！就像生活和寓言中常常发生的那样，情况发生了变化（喻需求变更）。村庄里的厨师（喻项目干系人）想要一些奇特、罕见的原料来烹饪（喻个性化需求）。这些原料大象曾经听说过，但是不在他每天要走的路上（喻传统工程方法不适合涉及的领域）。他还在为存储那些无从需要的食物（喻开始的需求已经过时）疲于奔命，却没有时间去修建新的道路以满足新的需求。村庄里的人渐渐地对这头无法满足他们需求（可以理解为RUP等传统方法不适合网站开发）、越来越气馁的大象失去了耐心。

与此同时，附近村庄有一只猴子（喻敏捷方法）做着和这头大象类似的工作。但是，和这头大象不同，这只敏捷的小猴在这片丛林中快速地穿梭（喻XP等编程方法），采摘它所看到的果实。他只去寻找那些挂得很低的果实（喻无法满足企业级应用）并迅速把它们送回到厨师那里。这只猴子不像大象那样依赖于一条久经考验的道路，他依赖于自己的记忆和本能去找到食物，并且只带回当天所需要的食物（喻迭代编程、每日构建等）。有时，他会跑出去寻找那些日益珍奇的食物（喻新奇的功能，如AJAX功能），偶尔还会迷路（喻新的方法太多）。但事实证明，他的速度和敏捷完全能胜任村庄分派给他的任务。和那头大象一样，他也颇受爱戴（喻在自己的领域内，如Web开发）。

很不幸，这只猴子的生活也发生了变化。他所生活的村庄一天天地扩大起来（喻企业级Web应用）。这只小猴需要满足太多的需求，以至于他总是疲于奔命，努力记住在每个地方的所有需要。他不得不来来回回地跑很多次，因为他没有足够的力量一次运回人们需要的所有东西（喻功能太多）。同样地，这个村庄渐渐对这只猴子失去了耐心，而这只猴子也开始怀疑自己是否能够胜任这项工作。

碰巧有一天，这只疲倦的猴子遇到了那头气馁的大象。这只试图在背负重物的情况下快速奔跑的猴子发现大象背上的筐可以装得下那么多的食物。而大象也被猴子的速度震撼了**（喻双方开始正视对方的优点），**他能跑得那么远，他能那么轻而易举地拿到大象竭尽全力也无法获得的食物。这两只动物，虽然都以自己的技能而自豪（喻从前的论战），但还是承认了对方在某些方面技高一筹。

大象和猴子认识到一起工作的好处（喻一个控制大的流程，一个处理软件开发细节），于是决定联合起来。猴子利用他的敏捷去满足对于远距离水果的新需求，并让大象带回到他的村庄。大象则把充足的食物运到猴子的村庄去满足那里逐渐增多的需要。虽然他们花了一些时间去找出协作的方法（喻两个不同方法之间的磨合），但是很快他们就能够很好地满足两个村庄的需要了，从此，他们过着快乐的生活，彼此互相信任，从此丰衣足食的村民们又开始看重他们了（喻实用才是第一：））！

转自：<http://blog.csdn.net/futurelight/archive/2006/09/21/1262383.aspx>

另：看到以下说法：

\* 敏捷方法则是将规范的、大规模的软件开发再拉回到最初的“软件小作坊”，是一种新的倒退。

\* 小作坊的特征就是团队小、天天可以见面、大家比较熟悉，主要靠口头交流，采用工具也比较原始（白板、贴纸条等），挺像敏捷方法的实践。

\* 互联网的许多应用需求不确定、变化迅速，互联网的不少应用又是免费的，对质量的要求不是很高。这一切适合快速迭代，呼唤着一种更灵活的开发方法，那就是敏捷方法。

\* 小团队以人治为主，法治为辅。大团队以法治为主，人治为辅。敏捷方法适合小团队，主要停留在人治的阶段，但说得好听点则是“以人为本”。

\* 敏捷方法还是一种倒退，对角色定义越来越不够明确，测试人员和开发人员的界限越来越模糊，而成熟的工业，角色细分，责任明确。

\* 敏捷方法违反传统的质量观点“第一次就将事情做对”，而是打着拥抱变化的旗帜，想怎么做就怎么做，然后再不断修改、重构。