

从一个项目谈XP在国内的应用 PDF转换可能丢失图片或格式
，建议阅读原文

https://www.100test.com/kao_ti2020/149/2021_2022__E4_BB_8E_E4_B8_80_E4_B8_AA_E9_c41_149683.htm 目前国内对于XP方面的研究和应用此起彼伏，各种关于XP的书籍争相出版，对于以XP为代表的"敏捷软件工程"方法的争论也在网络上随处可见。之所以出现这样的情况，是因为国内的用户在软件项目的实施过程中遇到了很多问题，例如项目的交付时间推迟、用户需求变更频繁等，我们的软件工程师迫切的希望能够找到解决问题的"银弹"。对于高度动态、通过非常短的迭代周期来应对需求变化的极限编程方法论来讲，确实能够从一定程度上解决问题。但是，对于国内的软件开发项目来说，XP并非"银弹"，它的一些最佳实践不是都适合国内的情况。本文结合一个具体的软件开发项目，讨论一下XP在国内的应用情况。

XP简介传统软件开发方法 在最近的数十年中，很多企业的CEO们都面临着增加盈利的压力，因此，他们采用各种方法，例如裁员、业务外包、BPR、ERP、CRM等等。以上种种，使得世界500强的大部分企业在20世纪90年代的后半期一直保持者二位数的利润增长。但是很多迹象表明，在传统的企业业务模型中已经没有什么可供削减开支的地方，因此，需要进行彻底的改革。在软件开发领域，情况更是如此。自上个世纪60年代以来，软件工程思想逐渐形成与发展，也出现了很多软件开发模型与方法，例如瀑布模型、快速原型、增量模型和螺旋模型等[1]。而在90年代以后，卡耐基梅隆软件学院推出的CMM，更是对于软件开发的过程管理，提出了确切的衡量指标。但是，最近的研究表明，有50%的项

目会拖延交付，有30%以上的项目会超出预算，软件开发领域的项目情况比软件工程刚刚提出的时候相比，只是有很小的提高。详细的数据[4]（数据来自美国GSM研究机构，Michael Mah）如下表所示：传统的软件开发过程，以RUP为代表，强调项目的可控性，是一个用例驱动的基于UML和构件式架构的迭代增量式开发过程。RUP定义了初始、细化、实现和部署4个阶段，分别对应着关键里程碑的划分。RUP对于角色、流程、工件和活动的要求是灵活、可配置的，所以它广泛的适用于各种类型的项目。但是，在RUP的各个里程碑，都规定了要交付的成果，尤其是对于需求的变更以及文档，它强调及时的更新与同步。以上这些都决定了RUP是一种重量级的软件开发方法，比较适合大中型的项目和产品开发。XP以及其核心价值最近，出现了很多轻量型的软件开发方法，例如水晶模型、适应模型以及极限编程等。它们都强调，软件开发是人与人合作进行的过程，因此成功的软件开发过程应该充分利用人的优势，而弱化人的缺点，突出了人在软件开发过程中的作用[2]。 Kent Beck在XP的开篇之作

《Extreme Programming Explained - Embrace Change》中提出了极限编程这一创新的软件过程方法论。极限编程是一种高度动态的过程，它通过非常短的迭代周期来应对需求的变化。在极限编程中，包括四个基本活动：编码、测试、聆听与反馈，XP项目的状态变迁如下图所示[3][4]： Kent Beck指出，XP有四个核心价值是我们应该注意的[3][4]：沟通：项目中发现的问题往往是由于开发人员与设计人员、设计人员与客户之间的沟通不畅造成的，因此，在XP项目中没有沟通是不可能的。简单：XP认为应该尽量保持代码的简单，只要它

能工作就可以。Kent Beck指出与其实现一个复杂的系统，不如设计一个能够满足目前需要的、简单的系统，因为你所考虑的情况可能永远都不会发生。 反馈：尽快获得用户的反馈，并且越详细越好，使得开发人员能够保证自己的成果符合用户的需要。 勇气：这是最重要的核心价值。因为XP强调要"拥抱变化"，因此对于用户的反馈，要勇于对自己的代码进行修改，丢掉坏的代码。 下面我们将要谈到的XP的最佳实践就体现了上述四个核心价值。实际上，XP中并没有多少新的观点，它的一些最佳实践也都是长久以来都在使用中的。

XP的适用环境 从XP项目状态图以及它的核心价值中我们可以看到，XP弱化针对未来需求的设计，非常注重当前的简化。它的实践，有一个非常关键的假设就是开发人员只注重眼前需求而依赖重构来适应需求的变动所带来的风险与开销要小于需求变化使得事先充分设计失效的代价；反之，实施XP就是不明智的[5]。因此，XP适合规模小、进度紧、需求变化大、质量要求严的项目。它希望以最高的效率和质量来解决用户目前的问题，以最大的灵活性和最小的代价来满足用户未来的需求，XP在平衡短期和长期利益之间做了巧妙的选择。我们可以看到，XP并不是解决问题的"银弹"，它也有不适合的情况。Beck曾经建议在以下情况下不宜采用XP： 中大型的项目（项目团队超过10人）； 重构会导致大量开销的应用； 需要很长的编译或者测试周期的系统； 不容易进行测试的应用； 团队人员异地分布的项目； 不能接收XP文化的组织和团队；

项目概况及背景 我们公司是亚洲领先的电子商务解决方案供应商，在J2EE架构的项目执行方面有丰富的经验，结合RUP形成了自己的一套电子商务项目实施方法论，并在多

个项目中成功进行实施。同时，由于具体项目时间和成本的限制，也出现了许多问题，主要有以下两点：项目交付后，用户提出很多的修改意见，有些甚至涉及系统架构的修改：出现这种情况的主要原因是很多项目虽然是采用增量迭代式的开发周期，但是在部署前才发布版本，用户只是在项目部署后才看到真正的系统，因此会发现很多界面、流程等方面的问题；对于用户提交BUG的修改周期过长：开发人员在作开发的时候，对于单元测试的重视程度不够，模块开发结束后就提交给测试人员进行测试，而测试人员由于时间的关系，并不能发现所有的问题；在用户提交BUG后，开发人员由于项目接近尾声，对于代码的修改产生惰性，同时又没有形成有效的回归测试方法，因此，修改的周期比较长。针对XP的核心价值，可以看到，如果我们能够加强与用户的沟通、增加项目中测试实施的力度、提高开发人员的勇气，就可以从一定程度上解决上述问题。从2001年开始，公司内部展开对于XP等敏捷方法的研究，希望能够借鉴一些做法，来完善项目方法论。2002年5月，我们决定在公司的一个新的项目中启用XP的一些最佳实践，来检验其效果。该项目是为一家国际知名手机生产厂商的合作伙伴提供手机配件订购、申请、回收等服务，项目的情况如下表所示：从上表中可以看出，该项目是一个小型项目，而且项目小组成员对于XP在项目开始之前都有一定的了解，另一方面，客户要求的项目周期比我们预期估计的时间有一定的余地，因此我们决定利用这个项目进行XP的试验性实践。XP的最佳实践以及在项目中的应用在项目执行过程中，我们基本上还是采用RUP的软件过程，而没有死板的套用XP的做法，例如：在需求分析阶段，我

们还是采用Use Case来对需求进行描述，而不是XP规定的CRC卡片；在系统分析与设计阶段，首先进行系统的架构设计，而不是简单的套用XP的"简单设计"实践。下面我们结合项目的具体情况，讨论一下XP的12个最佳实践。

现场客户 (On-site Customer) XP：要求至少有一名实际的客户代表在整个项目开发周期在现场负责确定需求、回答团队问题以及编写功能验收测试。
评述：现场用户可以从一定程度上解决项目团队与客户沟通不畅的问题，但是对于国内用户来讲，目前阶段还不能保证有一定技术层次的客户常驻开发现场。解决问题的方法有两种：一是可以采用在客户那里现场开发的方式；二是采用有效的沟通方式。
项目：首先，我们在项目合同签署前，向客户进行项目开发方法论的介绍，使得客户清楚项目开发的阶段、各个阶段要发布的成果以及需要客户提供的支持等；其次，由项目经理每周向客户汇报项目的进展情况，提供目前发布版本的位置，并提示客户系统相应的反馈与支持。

代码规范 (Code Standards) XP：强调通过指定严格的代码规范来进行沟通，尽可能减少不必要的文档。
评述：XP对于代码规范的实践，具有双重含义：一是希望通过建立统一的代码规范，来加强开发人员之间的沟通，同时为代码走查提供了一定的标准；二是希望减少项目开发过程中的文档，XP认为代码是最好的文档。对于目前国内的大多数项目团队来说，建立有效的代码规范，加强团队内代码的统一性，是理所当然的；但是，认为代码可以代替文档却是不可取的，因为代码的可读性与规范的文档相比合适由一定的差距。同时，如果没有统一的代码规范，代码全体拥有就无从谈起。
项目：在项目实施初期，就由项目的技术经理建

立代码规范，并将其作为代码审查的标准。每周40小时工作制 (40-hour Week) XP：要求项目团队人员每周工作时间不能超过40小时，加班不得连续超过两周，否则反而会影响生产率。评述：该实践充分体现了XP的"以人为本"的原则。但是，如果要真正的实施下去，对于项目进度和工作量合理安排的要求就比较高。项目：由于项目的工期比较充裕，因此，很幸运的是我们并没有违反该实践。计划博弈 (Planning Game) XP：要求结合项目进展和技术情况，确定下一阶段要开发与发布的系统范围。评述：项目的计划在建立起来以后，需要根据项目的进展来进行调整，一成不变的计划是不存在。因此，项目团队需要控制风险、预见变化，从而制定有效、可行的项目计划。项目：在系统实现前，我们首先按照需求的优先级做了迭代周期的划分，将高风险的需求优先实现；同时，项目团队每天早晨参加一个15分钟的项目会议，确定当天以及目前迭代周期中每个成员要完成的任务。

100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com