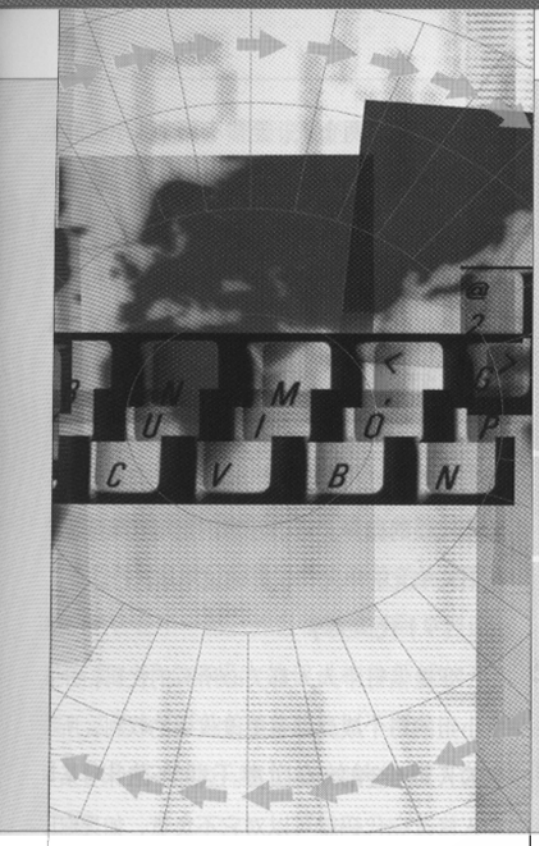

极限编程

Extreme Programming

王辉 余雪丽（太原理工大学计算机系 030024）

摘 要：本文首先介绍了极限编程(XP)软件方法，分析了其核心思想，进而论述了XP相关的十二种实践方法及其价值，以期对提高软件生产率有所帮助。

关键词：极限编程 软件工程 迭代

1 引论

我国的整体经济正在高速发展，但我们却不得不面对整个IT行业的低迷状态。过去几年中，软件生产商通过压缩公司规模、再工程、企业资源规划(ERP)等手段保持了其利润的高速、稳定增长，但今天的现状迫使我们要大幅度的压缩开发成本，寻找一个可以提高开发速度和效率的高效软件开发方法便是达到这一目的重要途径之一。

极限编程(XP)是由Kent Beck于1999年提出的一种“轻量型”的软件开发方法，它与Crystal一样是被视为“敏捷方法”中的一种。与传统的软件开发方法不同，XP摒弃了大多数重量型过程中的中间产物（诸如干特图、状态报告，以及多卷需求文档等）来提高软件开发速度；由于XP的迭代特性使得其能够在开发过程中对需求的变化有很好的适应性，从而XP的目标便是：在最短的时间内将较为模糊、变化较大的用户需求转化为符合用户要求的软件产品。

2 XP核心思想

传统软件工程中的瀑布模型要求需求分析人员一次性的从用户那里精确地得到用户需求，然后进行设计、编码和测试。所有步骤都必须被很好的执行，否则错误发现的越迟，所处的阶段越靠后，它带来的危害越大，甚至要将项目完全推翻。然而所有的步骤都不可能尽善尽美：用户经常对自己的需求不太确定，有时还会自相矛盾，他们还会不断提出新的或修改以往的要求；设计人员会犯错；编码中会存在BUG；测试时会存在遗漏。由于不能适应各种变化，长开发周期已经日益显得力不从心了，我们需要更短的开发周期，这恰恰是XP为我们提供的。

XP将传统瀑布模型中分析、设计、编码和测试四个核心步骤以迭代的形式在不同的迭代版本中得以实现，若在某个迭代版本中出现问题，我们可以就地将问题解决，从而保证了最后软件版本的可靠性和完整性。比如一个软件共含有6个功能点，XP会将这6个功能点分为3个迭代阶段，每个迭代阶段实现2个功能点。在每个迭代阶段，这两个功能点的实现也分为分析、设计、编码和测试等四个阶段，进而保证了每个迭代版本的高质量（如图1所示）。

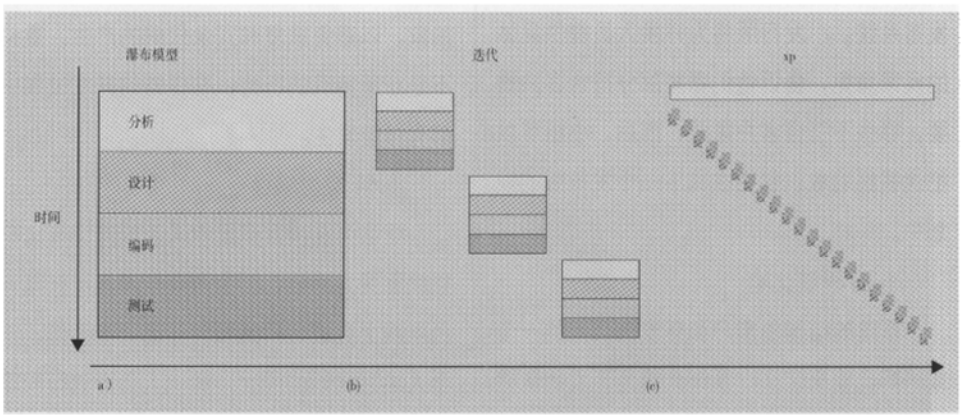

图1 迭代过程

3 XP的十二种实践方法

XP是一种以促进实际软件开发为目标和核心的方法学，它不明确区分设计者和编程者，编程者与设计者一般是统一的，从而，程序代码就成了比较方便的架构表达工具。当然，这并不意味着分析、设计、编码和测试人员之间没有其他方式沟通，XP认为口头交流是最为有效的，从而架构描述工具在此就不再需要了。以下给出XP认为在实践中最为行之有效的十二种方法：

（1）计划制定

用户根据开发人员提供的估计来决定各迭代版本的发布时间，开发人员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。项目计划在建立起来以后，需要根据项目的进展来进行调整，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。根据实践，建议如下：系统实现前，首先按照需求的优先级确定迭代周期的划分，一般将高风险的需求优先实现。开发团队最好每天早晨开一个简短的项目会议，确定当天以及目前迭代周期中每个开发成员要完成的任务；每天下班前也有一个简短聚会，查看当天任务的完成情况，以及时调整开发计划。

（2）小型发布

强调在非常短的周期内以递增的方式发布新的迭代版本，从而可以很容易的估计每个迭代周期的进度，便于控制工作量和风险；同时，只要开发人员觉得有意义就可以发布系统。小发行版将为开发人员提供具体的反馈意见，告诉他们哪些部分符合客户需要，哪些不符合客户需要，而后，小组可以把这些经验教训包括在其下一个发行版的规划中。

（3）系统比喻

系统的结构由用户和开发人员间的一个或一组比喻来定义。系统比喻为开发团队提供了一个一致的映射，它们可以用来描述现有系统的工作方式、新部件适合的位置，及其应该采取的形式。系统比喻的优劣不是看其外表是否美丽，而是在于它是否可以使开发人员明了新的代码部分适合放在何处。

（4）简单设计

XP并没有忽略了设计过程，不同的是重量型方法建议开发人员在编写代码之前完成大部分琐碎的设计任务，这是以需求不变化，或者很少变化为前提的。XP则认为需求是会经常变化的，因此不存在完美的设计，也就是说设计应该是一项持续进行的过程。简单的设计应该拥有以下特性：运行所有测试；不包含重复代码；让程序员对与其相关的所有代码及其内在关系有清晰的理解；尽可能包含最少的类和方法。

（5）测试

XP中有两种测试：单元测试和功能测试。程序员要经常编写单元测试用例，而且应该将单元测试尽量自动化，并提供测试成功或失败的明确结果。(xUnit测试框架是大多数XP小组喜欢使用的一种自动化测试工具)。在每一个迭代版本推出后，应该让用户协助提出功能测试用例。

（6）重构

XP中的重构是在不更改系统功能性的前提下对代码加以改进。重构的两个重要时机是：一个功能点的实现前、后。开发人员尝试修改现有代码以让新功能点的开发更容易；他们查看现有代码，尽可能将其简化和抽象，以避免重复和冗余代码的产生。重构不是XP所特有的行为，在任何的开发过程中都有可能发生。

（7）成对编程

XP认为在项目开发中采用成对编程比单独编程更为有效，即由两个开发人员在同一台电脑上完成代码的编制与测试等工作，一个人编写代码的同时，另一人负责进行同级评审。成对编程有如下有点：所有设计决策都牵涉到至少两个人；至少有两个人小组相关系统的每一部分；可以尽量减少单独编程时经常出现的疏忽测试等情况；重新组合各组成员可以在整个开发团队中传播知识和经验；代码总是由至少一个人进行复查。

（8）持续集成

XP认为应将新编的代码不断集成到已有系统中，并进行测试，测试通不过的单元要义无反顾的抛弃。应该注意的是每次集成应在相关单元的单元测试都通过以后执行。

（9）代码共享

XP提倡每个人人都关心所有的代码，开发小组的每个成员都有更改代码的权利，所有的人对于全部代码负责。这并不意味着某个人修改完代码就可以对它不负责，如果你修改完的代码出现问题，就有责任将它修补、完善，至少要做到将代码复原。

（10）现场客户

XP中最理想的情况是在开发现场有一位客户来明确素材，并协助做出重要决策。开发人员纸面客户进行交流，可以大大减少产生误解的可能。但现实中不能保证一位有一定技术层次的客户常驻开发现场，XP要求至少客户必须在需要回答开发团队提出的疑问和为团队提供指示时有通畅的交流平台，用户的反馈信息可以及时的到达开发团队。

（11）每周40小时

XP要求开发团队的人员每周工作时间不得超过40小时，不得存在超过连续两周加班的情况，因为连续加班会影响开发团队的工作效率，疲劳的开发人员也会犯更多的错误。XP希望开发人员“每天早晨都感到有活力有激情，每天晚上都感到疲惫而满足”。

（12）代码标准

XP要求每个开发人员要严格遵守统一的代码规范，这样它们就可以顺畅的进行沟通，也可以尽可能减少不必要的文档。与此同时，规范的代码还是以上提到的代码共享

原则的前提条件。需要说明的是，代码标准在开发团队内有效即可，经过小组成员讨论后可以对标准进行修改。

4 XP的价值

虽然如XP的创始人之一的Beck所说“XP中没有一个想法是全新的，大多数想法产生的时间实际上和编程一样古老”，但XP对软件开发提出了一种崭新的思路。它将这些“古老”但实践证明最为行之有效的方法有机的融合在了一起，并且将其提升到了理论的高度。XP必将极大的促进软件领域的开发，这是由于XP的价值体现在以下两点：

4.1 XP将有助于我们有效的缩短开发工期

从以上对XP的叙述可以看出：首先，XP对实际开发者提供了强有力的指导。开发者在理解了XP的核心思想后，可以应用其十二中实践方法，改善软件的开发过程。其次，XP属于轻型方法，利用其迭代特性可以使实际的软件开发有很强的适应性。基于以上两点，按工期完成软件产品的开发将被得以保障。

4.2 XP将使开发成本大幅缩减

XP的核心思想是迭代，小型发布和简单设计等便是其在实践中的具体体现。与传统开发方式相比，XP的使用将使开发成本大幅下降，这在图2中可以看出。

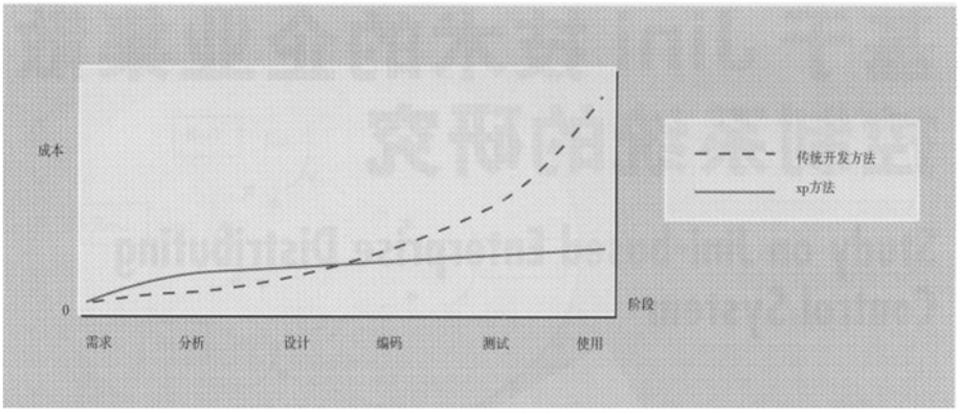

图2 开发成本比较

5 结束语

自XP提出以来，它尤如一缕新鲜的空气为软件工程领域带来了不同的思路。我国对XP的实际应用还处于起步阶段，幸运的是以上提到的这十二个实践方法作为XP思想在实践中的体现，它们之间大部分是相对独立的。在实际开发中，我们可以先取其中几个方法对我们的开发过程进行指导，在逐渐适应以后，再逐步对XP的方法进行较为全面的应用。

参考文献

1 Kent Beck, Extreme Programming Explained: Embracing Change, Addison Wesley, 1999.

2 Ivar Jacobson, Grady Booch, James Rumbaugh: The Unified Software Development Process, Addison Wesley Publishing Company, 1999.

3 Steve Hayes, An Introduction to Extreme Programming, http://www.khatovartech.com, 2001.

4 XP 精华－如何使 Java 项目获得更大成功，IBM DeveloperWorks，2001.