极限编程

- 实践原则
- 极限编程过程

四、极限编程

极限编程（eXtreme Programming，简称XP）是敏捷方法中最显著的一个。它由一系列简单却相互依赖的实践组成。

“如果你的组织准备好了要改变开发软件的方式，有缓慢的增量方法：一个一个地解决问题；同样也有快速的途径：跳进XP来。不要被名字吓到，它根本不是那么极限。大部分是多年积累的老处方和常识，被很好地整合起来，去除了这些年来积累的多余脂肪”

---Philippe Kruchten，加拿大UBC大学教授

极限编程

- 实践原则
- 极限编程过程

1、极限编程包含的实践

(1)客户作为团队的成员

含义：客户与开发人员一起紧密的工作，相互了解所面临的问题，并共同解决之。其中，客户（人或团队）的主要责任是定义产品特征、并对这些特征进行优先排序。

注意：“如出现不能一起紧密工作的情况，应该寻找可以代表真正客户的、并可一起工作的人”。

北京大学

- 实践原则
- 极限编程过程

（2）“用户素材”(user stories）

含义：为了了解与项目需求有关的内容，采用“用户素材”(user stories）——一种规划工具，作为在进行关于需求谈话时所使用的助记工具。通常，在客户的索引卡上记录认可的一些词语， 与之同时，在该卡上写下关于需求的估算。

（3）短的交付周期

含义：短的交付周期是指每隔两周，就交付一次可工作的软件。这意味着每两周的迭代都实现了“涉众”的一些需求；并在每次迭代结束时，可给“涉众”演示由迭代所生成的系统，以得到他们的反馈。

显然，这一实践涉及迭代计划和交付计划的制定。

极限编程

- 实践原则
- 极限编程过程

(4）验收测试

- 作用： 通过验收测试，捕获用户素材的有关细节。
- 编写时间：在要实现该用户素材之前或实现该用户素材期间进行编写。
- 编写工具：编写验收测试，使用能够让它们自动、反复运行的某种脚本语言。

(5)结对编程

结对编程的含义是：共同设计、共同编写、功劳均等，以促进知识在全队的传播。

注：Laurie Williams和Nosek的研究表明，结对不但不会降低团队的开发效率，而且还会减少缺陷率。

北京大学

极限编程

- 实践原则
- 极限编程过程

(6) 测试驱动的开发

含义：首先对产品的某一功能编写一个单元测试。由于该功能还不存在，因此它一定会运行失败。然后编写这一功能的代码，使该测试得以通过。

益处：为了测试用例通过而编写代码，这样的代码称为可测试的代码。优点是：由于要独立对它进行测试，因此可激励解除模块之间的耦合，使模块之间具有低的耦合。

北京大学

- 实践原则
- 极限编程过程

(7) 集体所有权

集体所有权的含义：

①编程中的每一结对，都具有“检出”（check out）任何模块的权力；

②没有程序员对一个特定的模块或技术单独负责；

③每一个人都参与GUI工作，每一个人都参与中间件方面的工作，每一个人都参与数据库方面的工作。

没有人对结对所编写的模块和技术具有更多的权威。

注意，这并不意味着开发人员没有自己的专业知识。

极限编程

- 实践原则
- 极限编程过程

(8) 持续集成

持续集成的含义是：

程序员每天可以多次检入（check in）他们的模块进行集成。

其中，最重要的是，要确保所有的测试都能通过：

①可以把新的代码集成到代码库中，可以对代码进行合并；

②必要时，可以和检入的程序员进行协商。一旦集成了他们的更改，就构造了新的系统，从而要运行系统中的每一个测试，包括当前所有运行的验收测试。一旦所有的测试都通过，这才算完成这一次的检入工作。

极限编程

- 实践原则
- 极限编程过程

应用极限编程实践中应注意的两个问题

(1)应将这组简单的实践融为一体,相互依赖。

(2)可增加一些实践或对其中一些实践进行修改。

北京大学

极限编程

- 实践原则
- 极限编程过程

2、极限编程过程

XP使用面向对象方法作为推荐的开发范型，它包含了策划、设计、编码和测试4个框架活动的规则和实践。

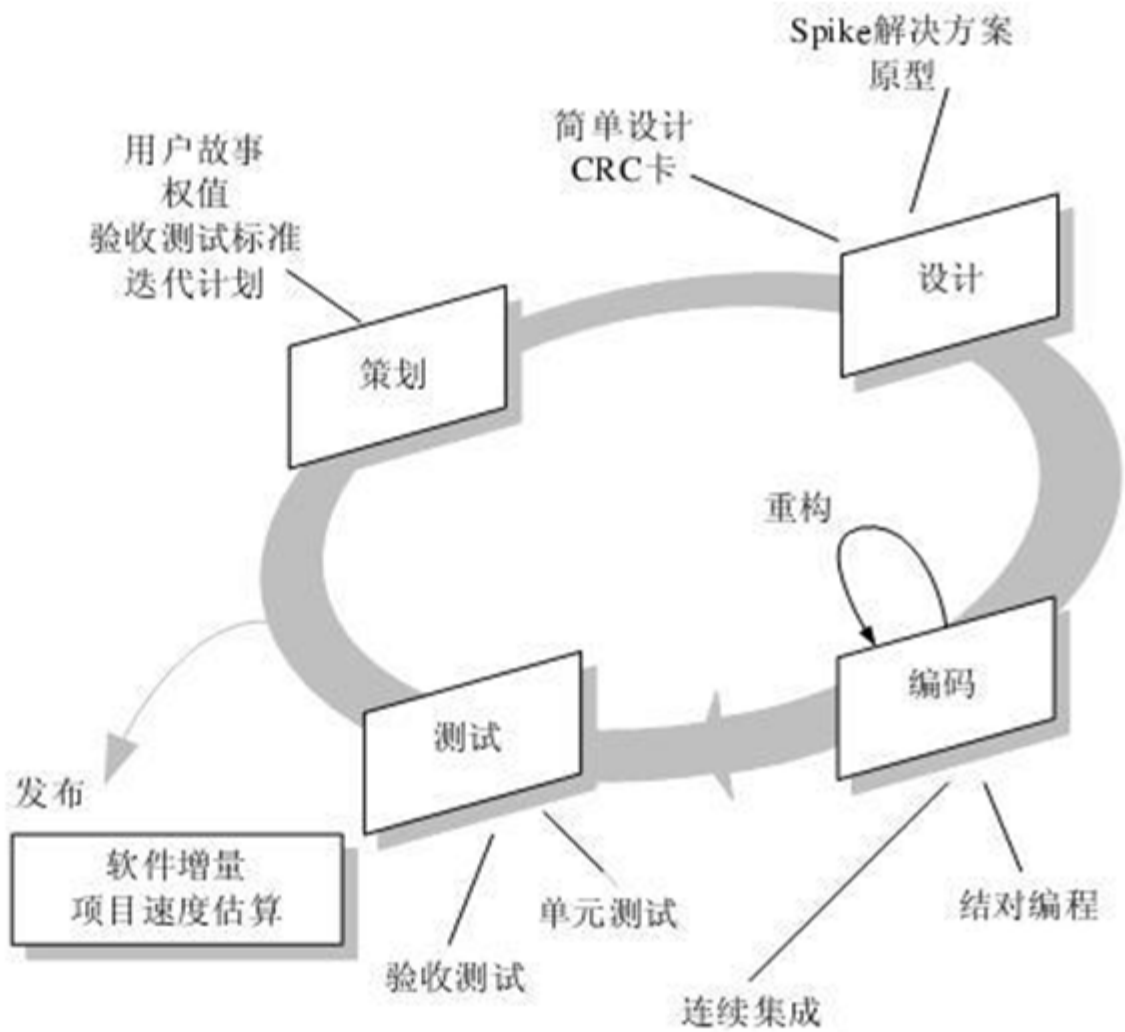

图　极限编程过程