Extreme Programming with Ant

Building and Deploying Java Applications with JSP, EJB, XSLT, XDoclet, and JUnit

Ant 极限编程

——利用 JSP、EJB、XSLT、XDoclet 和 JUnit 构建和部署 Java 应用程序

（美）Glenn Niemeyer 著
Jeremy Poteet
孟浩文 译

SAMS

清华大学出版社

Ant 极限编程

——利用 JSP、EJB、XSLT、XDoclet 和 JUnit 构建和部署 Java 应用程序

(美) Glenn Niemeyer 著
Jeremy Poteet
孟浩文 译

清华大学出版社
北 京

内 容 简 介

本书通过开发一个真实的项目，全面深入地介绍了如何使用 Ant 进行极限编程的过程。读者只需要将书中的代码和示例稍作修改，即可应用于实际的工作中。全书共 11 章，内容涉及到 Ant 应用的方方面面，包括如何编写 buildfile，如何使用 Ant 的内置功能，以及如何开发定制 Ant 构件等。

本书面向软件开发工程师，要求读者对软件开发过程有大致的认识，并且熟悉 Java 语言，最好有一些实际项目开发经验。同时，本书对高等院校计算机软件方向的教师和学生也具有重要参考价值。

Glenn Niemeyer Jeremy Poteet

Extreme Programming with Ant: Building and Deploying Java Applications with JSP,EJB,XSLT,XDoclet,and JUnit

EISBN：0-672-32562-4

北京市版权局著作权合同登记号　图字：01-2003-7184

图书在版编目(CIP)数据

Ant 极限编程——利用 JSP、EJB、XSLT、XDoclet 和 Junit 构建和部署 Java 应用程序/(美)尼米尤(Niemeyer , G)，(美)珀提(Poteet , J)著；孟浩文译. —北京：清华大学出版社，2004.10

书名原文：Extreme Programming with Ant: Building and Deploying Java Applications with JSP,EJB,XSLT,XDoclet,and JUnit

ISBN 7-302-08825-X

Ⅰ. A… Ⅱ. ①尼… ②珀… ③孟… Ⅲ. 软件工具，Ant—程序设计 Ⅳ. TN311.56

中国版本图书馆 CIP 数据核字(2004)第 082337 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　　**邮　　编**：100084
社总机：010-62770175　　**客户服务**：010-62776969
组稿编辑：曹　康
文稿编辑：王　军
封面设计：康　博
版式设计：康　博
印 刷 者：北京市世界知识印刷厂
装 订 者：北京市密云县京文制本装订厂
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：22.5　**字数**：576 千字
版　　次：2004 年 10 月第 1 版　2004 年 10 月第 1 次印刷
书　　号：ISBN 7-302-08825-X/TP・6261
印　　数：1~4000
定　　价：45.00 元

前　言

Ant 极限编程

本书的主要目的是指导读者如何应用极限编程(XP)方法以及使用 Apache Ant 实现极限编程过程。在开发项目的过程中，虽然可能会遇到一些典型的或不同寻常的问题和挑战，但还是可以开发出适用于很多项目的优秀解决方案。本书包含大量的实例，比如用来扩展 Ant 的 Java 源代码。即使不使用极限编程方法，本书也可指导您应用 Ant 创建和部署应用程序。

读者对象

本书面向应用程序开发人员、技术小组组长和系统架构师。在阅读本书之前，读者应该能熟练地使用 Java 语言进行开发，并且具备创建和部署应用程序的基础知识。

内容提要

本书主要包含以下几个方面的内容：

- 获取和安装 Ant
- 使用 Ant 实现极限编程过程
- 使用 Ant 建立基本的创建和部署过程
- 实现单元测试自动化
- 动态生成 Web Application、Enterprise JavaBeans 和 Java ServerPages Tag Libraries 的部署描述文件
- 实现功能测试自动化
- 自动生成项目报告(project metrics)，包括各种版本控制系统的报告、单元测试报告以及其他报告
- 自动检测没有经过单元测试的类
- 使用 Java 程序编写定制任务(custom task)和使用 Ant 提供的 Bean 脚本框架引擎(Bean Scripting Framework Engine)，以扩展 Ant
- 提供大量关于生成项目报告、UML 图，集成 Oracle SQL 加载器以及生成测试报告等定制任务的示例
- 创建定制的记录器(logger)来发送包含创建结果的加密邮件
- 如何在企业级和部门级规划实用的创建和部署过程
- 与 NetBeans 集成开发环境一起使用

极限编程(Extreme Programming，XP)方法是一个轻量级的开发过程，它强调使用迭代过程来提高软件质量，简化开发过程以及不断提供关于开发过程状态的反馈信息。通过不断收集反馈信息，并根据这些反馈调整下一步的工作计划，极限编程方法可以帮助开发团队很好地控制应用该方法的项目，从而取得成功。因为极限编程的迭代特性，所以实现许多过程的自动化更为可取，而 Apache Ant 正好可以满足极限编程方法的这种需要。

Ant 允许用户以 XML 的形式定义并建立创建、测试和部署过程。Ant 具有很多内置的功能，同时还可以通过多种途径对它进行扩充。用户可以使用 Java 语言来创建自己定制的任务，并加入到 Ant 中。Ant 还支持 Bean 脚本框架(BSF)，允许使用 JavaScript 和 Python 这样的脚本语言编写的代码加入到创建过程中。

在本书中，我们借助了一个利用 Ant 实现极限编程过程的项目。这个项目存在着典型的甚至不常见的问题和挑战，同时也包含了适用于大多数项目的解决方案。本书首先介绍如何建立一个适用于很多项目的创建、测试和部署的典型过程。然后，介绍扩展 Ant 的多种途径。本书提供了大量的用于扩展 Ant 的定制任务的实例，包括 Java 源代码和用 XML 编写的 buildfile。这些定制的任务包括能够检测到遗漏的单元测试的开发方法，在创建过程中建立 UML 图以及集成像 Oracle SQL Loader 和 NoUnit 这样的工具。接着，本书介绍如何创建定制的记录器，该记录器可在一个不安全的网络中创建加密邮件。另外，还介绍如何在 Ant 中使用 BSF 引擎，以便将 JavaScript 或 Python 结合到 Ant 的 buildfile 中。Ant 也可以与 IDE 集成。最后讨论了如何建立一个能广泛使用的(在部门甚至整个企业使用)Ant 创建、测试和部署过程。本书中的所有例子都是基于 Ant 1.5.3 编写的。

即使不使用极限编程方法，本书仍然可以指导读者应用 Ant 来完成创建和部署过程。

读者对象

本书面向开发人员、项目技术主管和系统架构师。它包含了关于如何用 Ant 自动过程来实现极限编程项目的一些实用性建议。作为一名开发人员，应当能够熟练地使用 XML 和 Java 并且对创建、测试和部署软件的面临的挑战有一定的认识。作为一名程序员、项目技术主管或系统架构师，应该乐于学习和应用新技术和新方法。

本书的组织结构

本书共 11 章，它们互相关联。书的内容是基于一个名为 eMarket 的、用 XP 实现的虚拟项目展开的。我们在每一章中都先对开发小组遇到的困难和需求进行概述，然后再讨论如何解决在程序的创建、测试和部署过程中遇到的问题。最后，每一个章节都详细地介绍了所考虑的技术的实现，并配有实例。这些实例包括用 XML 编写的 buildfile 和用来扩展 Ant 的 Java 代码，以及加入到开发过程中的第三方工具。

第 1 章讨论了什么是极限编程方法，以及为什么 Ant 是极限编程项目的一个不错的选择。

第 2 章论述了 Ant 的基本概念以及如何使用 Ant 来建立一个初始的 spike，它是极限编程过程的一部分。同时还介绍了如何使用 Ant 的内置功能实现基本的极限编程过程。

第 3 章阐述了关于单元测试、修订版本控制以及如何把它们集成到极限编程过程中等内容。

第 4 章介绍了一个完整的创建过程。首先介绍 Javadoc 和 doclet 的使用方法，然后介绍如何自动生成文档。另外，还讨论了单元测试和修订版本控制方面的一些高级问题。

第 5 章介绍了一个夜间创建过程(nightly build process)。介绍了如何使用记录器和监听器，如何建立一个非常清楚的创建过程，它包括修订版本控制、编译、单元测试，以及为小组成员和组长准备的关于创建结果的自动报表。另外，本章还介绍了如何集成 CruiseControl 工具进行无人值守的连续创建。

第 6 章介绍部署过程的建立。在本章中，对项目进行部署以测试服务器，并检验 Ant 完成的基本任务。同时还将学习如何使用第三方工具自动生成 Web 应用程序、企业级 JavaBeans(EJBs)和 JavaServer Pages Tag Libraries(JSP Taglibs)的部署描述符。

第 7 章讨论了当更多的开发人员加入到项目中时，极限编程方法所面临的挑战。本章主要关注如何通过修改创建过程来提升开发人员责任心的问题，重点介绍了当开发人员大量增加时，如何防止项目陷入混乱。另外，本章还介绍了一些可以集成到 Ant 的创建过程中来加强项目标准的工具，包括 Icontract、Jalopy、Checkstyle 和 PMD。

第 8 章介绍把一个应用程序转变为产品时必须要解决的一些问题。主要包括如何自动生成一系列 CVS 报表和版本注释。同时还介绍了一些技巧：如何提高代码生成效率，怎样执行另外一台服务器上的远程创建，以及在部署产品时如何把数据导入数据库中。另外，还开发了一个定制任务将 Oracle SQL Loader 集成到部署过程中。

第 9 章讨论公司重组方面的问题，开发小组必须适应一些新的要求。本章讨论了如何在 NetBeans 集成开发环境中使用 Ant；如何使用 Styler 工具来完成高级的多级流水线代码生成工作；介绍了一个基于 Ant 开发出的功能测试工具——Anteater。另外，本章阐述了一个定制任务的开发，可以使用它检查没有经过单元测试的类。最后，本章还介绍了如何使用 Ant 提供的 Bean Scripting Framework(BSF)引擎向 Ant buildfile 中插入 JavaScript 脚本代码。

第 10 章讨论了一个商业软件开发团队打算采用 eMarket 项目使用的极限编程过程，并讨论要作哪些改动来满足他们的需求。例如，他们要发布 JAR 文件，所以要对 JAR 文件进行混淆编译，以防止被逆设计，这些工作都要加入到 Ant 的创建过程中。最后还介绍了如何使用 SourceForge 公司提供的 NoUnit 工具来了解单元测试的代码覆盖率，并且增加了一些功能以提高创建的效率。

第 11 章讨论了如何在企业级环境中使用前面开发的 Ant 过程。所有的 buildfile 都被重构以适应新的需求。本章创建了新的定制任务用来实现根据 Java 代码库自动生成 UML 图的工作，介绍了 Cactus Web 应用程序测试框架，以及一些用以启动和停止 Weblogic 服务器的 Ant 任务脚本。另外，我们还创建了一个名为 EncryptedMailLogger 的定制邮件记录器，它使用 Gnu Privacy Guard(GPG)加密性在发送邮件前对创建结果加密。这主要用来帮助那些需要通过 Internet 发送重要信息的开发团队避免被窃听。

附录 A 介绍了获得 Ant 的方法：如何决定是选择源版还是二进制版，以及下载 Ant 后该如何安装。

附录 B 讨论了扩展 Ant 的途径，并包含了大量的实例。读者将会学到如何编写定制的任务、记录器、监听器、映射器、选择器、过滤器和输入处理器，并可以参考示例程序。还可以学习创建定制组件的一些基本原则、任务的生命周期、以及如何编写接受嵌套元素的任务；如何在定制组件中使用 Ant 的日志系统。

附录 C 展望了 Ant 2。包括 Ant 1 和 Ant 2 之间的兼容性问题以及一些保证能够尽快从 Ant 1

平滑过渡到 Ant 2 的做法。

附录 D 总结了本书中提到的所有 buildfile。本书的内容是基于一个极限编程项目展开的，这个项目使用 Ant 不断地改进创建和部署过程。本附录提供了一个独立的空间，它可以集中查看 buildfile 所发生的全部更新过程。

附录 E 总结了本书中使用到的所有工具及它们的版本号。

特殊段落

本书还包含了一些特殊段落，贯穿本书的“用户故事”段落记录了 eMarket 开发小组使用极限编程的全过程。“工具栏”说明了如何安装 Ant 的辅助工具。最后，提供了大量的实例和代码清单来说明如何实现本书中提到的各种概念，其中，大部分的例子和源代码只需要做很小的改动就可以直接应用到很多实际的项目中。

源代码和更新

读者可以从 Sams 公司的网站(www.samspublishing.com)上下载书中提到的所有代码文件和例子。只要将本书的 ISBN 号(0672325624)输入到“search”框中，按回车键，就可以进入下载代码文件的网页。在那里，可以找到本书的更新和勘误。

目　　录

第1章　XP和Ant

本章内容

- XP 的定义
- XP 过程
- Ant 和 XP 过程
- 本书的内容

极限编程(Extreme Programming，XP)是一种软件开发方法，使用它可以保证开发出高质量的软件，同时满足客户的需求。Ant 是 Apache 项目的一个开放源码(open-source)工具，用来实现构建和部署过程。在本书中，读者将学习如何使用 Ant 来实现高效的极限编程过程。

1.1　极限编程的定义

极限编程本质上是迭代的过程，它通过连续的自动测试和集成来开发高质量的软件产品。极限编程提倡开发过程保持简单，这意味着需要通过重构代码来减少重复或者冗余代码。它强调开发人员要随时随地满足客户需求，而不是在一段时间以后。极限编程强调客户和开发人员以及开发人员之间要不断沟通。事实上，在极限编程中，客户是开发团队的一部分。极限编程与制造业中的实时库存管理有些相似之处。

1.1.1　极限编程的特点

极限编程以客户满意度为重点。在极限编程中，客户是整个开发团队的一部分。这意味着客户在整个项目的规划过程中要发挥重要作用。客户通过编写用户故事(user story)和参加发布计划会议等方式来推进项目的实现和发布工作。因为在极限编程中软件开发应用迭代过程，所以一个实用的系统可以提供不同层次的功能。客户可以随时查看运行的系统，对当前状态进行评估，并且还可以对系统的需求(即用户故事)进行修改。这意味着如果客户对项目的用户故事进行修改，很快就能看到这些改动带来的价值。

极限编程强调团队合作。开发人员两两结对进行编程并进行交流。客户或客户代表将和开发团队一起工作并随时准备回答他们的问题。开发团队中的每个人都负责系统中的不同部分，并且对这部分代码具有所有权。极限编程方法简化了团队合作方式，前提是团队成员之间必须互相信任。

极限编程可以增加项目的灵活性。因为极限编程强调简单、质量和测试，所以极限编程比其他任何一种严格的开发方法都更容易适应不可预见的情况。实际上，极限编程预料到了程序开发中各种不可预测的情况，这正是其强调根据需求实现功能，并且不断进行测试和集成的原

因，而开发人员也正是这样做的。

在极限编程中，可以尽早发现集成问题，因为极限编程是基于迭代过程的。创建、单元测试、集成和验收测试等环节连续完成。开发团队不需要先用几个月的时间完成编程工作，然后再进入可怕的集成周期。在极限编程中，集成工作要容易得多，因为开发团队总在不断地进行集成。如果有人在开发过程中偏离了预定的轨道(在结对编程中很少会出现这种情况，这里主要是指在实现用户故事时可能会发生的情况)，在对任务计划产生重大影响之前就能够很容易被发现。

1.1.2 极限编程的核心价值

极限编程的关键是它的核心价值：

- 交流
- 简单
- 反馈
- 勇气

交流

交流对于 XP 项目能否成功至关重要。这里的交流是指开发人员之间以及开发团队和客户之间进行的高效的交流。极限编程要求客户或者客户代表能够和开发团队一起工作，这样可以鼓励和增进交流，也符合极限编程及时的特性。客户和开发人员要在发布计划会和迭代计划会上进行正式的交流。

在结对编程(pair-programming)期间开发人员之间需要交流。开发人员和客户之间也需要交流。当开发人员遇到问题时，应该和客户进行面对面的交流，从而得到及时的回答。这样能够提高项目的质量和成功率。如果开发人员必须等待一份文档或者一个回复，他们通常会猜测结果，或者去做其他一些并不重要的工作。这显然都不是所期望的。尽管可以通过 email 进行交流，但是面对面交流的效果要好得多，因为这样可以更容易地传递信息，并且更容易确认对方是否真正理解了自己表达的意思。

简单

简单通常比复杂更难实现，它需要精益求精的精神。在极限编程中，这意味着开发人员不用为以后需要的功能编辑，而只为现在的需求编程。开发团队应该尽量使设计保持简单。这意味着必须要重视可重用代码以减少冗余的代码。如果在完成了一段代码后发现了更简单的实现方法，那么就应该重写这段代码。因为软件的大部分费用花在维护上，所以从长远的眼光来看，这种做法是省时省钱的。同时，因为极限编程强调持续测试，所以不用担心改变系统的某个部分或某些类会产生负面影响。

实现简单性的另一方面就是保证代码的标准化。如果代码是用通用的格式编写的，那么任何人都可以轻易地挑出一段代码，并理解代码的意义。还要保证变量名、方法和类的标准化，大小写、括号和缩进格式的使用也要保持一致。这样有助于降低复杂程度，减少建立与维护软件的相关成本。

反馈

在极限编程中，通过进行频繁而彻底的自动测试来得到反馈。测试的内容包括单元测试和验收测试。极限编程强调先编写测试代码，再编写程序代码，并对其进行单元测试。这样能够尽快完成大量的单元测试。随着项目不断地进展，这种做法得到的回报远大于付出的代价，它可以避免灾难性的错误，而修复这种错误的代价是极高的。测试代码可以帮助开发团队更好地理解如何使用类，从而只编写满足需要的代码。

现在介绍一下反馈的重要性。打个比方，没有人会打开文字处理程序后关闭显示器，然后再输入一篇文档。因为人们要通过显示器得到动作的反馈结果，以此来减少错误。得到这些反馈并不能完全消除错误，但能够防止过度偏离正题。在极限编程中主要通过如下方法得到反馈：每完成一次创建就进行一次自动单元测试，这样可以大大地减少错误。频繁地进行自动测试可以尽早发现错误。在很多项目中，这种方法可以减轻集成和测试阶段的工作压力。

进行频繁的测试还有另外一个原因。开发人员在编写一段代码时，这段代码给他的印象最深。所以测试代码的最佳时机是在代码刚刚编写完的时候，因为这个时候开发人员还能清晰地记得所做的一切。如果测试安排在开发完成后一两个星期进行，而开发人员已经转到其他的工作领域，测试和修正代码的工作将会变得困难得多。通常开发人员只能在短时间内清楚地记得所做的工作，因此在进行单元测试和验收测试的过程中使用迭代方法可以有效地减少遇到的麻烦。

勇气

如果以前没有接触过极限编程方法，通常第一次应用它时会遇到很多困难。所以在开始阶段要求开发人员要具有一定的勇气。极限编程中的勇气是指：作为用户在发现用户故事不满足需求时果断地抛弃它，并重新写一份；作为开发人员，当发现一段代码尽管可以使用，但却显得很笨拙时，应该有勇气对代码进行重构；当知道有更简单的方法时，应该毫不犹豫地抛弃旧代码并重新设计算法。

勇气还意味着当遇到困惑时敢于提出疑问。当开发人员需要更深入地了解用户故事时，能够主动地去找客户交谈。在编写代码时，不要对极限编程的结对编程方法产生恐惧。这都是极限编程过程中需要具备的勇气。

1.1.3　更高的生产率

极限编程是一种高效的开发方法。首先，开发团队实现对客户最有价值的用户故事(用户故事的价值由客户确定)。他们总是优先完成最重要的任务，不在价值不明确的功能上浪费时间。在极限编程中，很少出现在开发完成后，一些功能由于是数月之前完成的或者需求发生变化而导致其丧失价值的情况。其次，使用极限编程可以开发出高质量的代码，因为这些代码被不断地测试和改进，这样将大大减少花在修正错误上的时间。

1.2　极限编程的过程

在极限编程中，客户控制着开发的优先级。开发过程从客户定义用户故事开始，这些用户

故事描述了系统的运作过程。用户故事通常写在索引卡上，这样可以方便地混合、标记，甚至撕毁。用户故事应该像下面这样："当客户发现需要购买的物品时，可以将它放入购物车，然后继续购物。"一个用户故事应该由一位开发者在大约一周的时间内完成。当然，客户可能并不知道开发人员一个星期可以完成多少工作，所以应该由开发人员指导客户把大的用户故事分解为一些小的用户故事，或者把几个小的用户故事组合成一个大的用户故事，总之要使用户故事保持合适的大小。

用户故事形成系统需求，并且客户还要决定这些用户故事的优先级与完成顺序。当开发人员查看这些用户故事时，通常只需要关注如何实现它们，或者用什么技术来实现它们。如果存在一些不明确的问题，或者存在一定的风险，就需要建立一个 spike。spike 是一些可抛弃的代码，在开发它时，不需要像开发最终代码一样严格。编写 spike 的目的是验证某些特定的概念，使开发人员能够更准确地估计完成一个用户故事的时间。客户书写用户故事和开发人员开发 spike 的这个阶段，称为迭代的探索阶段(exploration phase)。

迭代中的阶段

发布版本是由多次迭代组成的。它通常包含 3 次迭代，一次迭代的周期通常是 1～3 个星期。极限编程中把一次迭代中的活动划分为 3 个阶段：探索阶段、提交阶段、调整阶段。

探索阶段的主要目的是收集信息。它包括收集用户故事，决定系统的需求以及建立 spike。

在提交阶段，开发人员查看用户故事卡，并评估实现每一个任务的难度。在这个过程中可能要分解或合并一些用户故事，使其达到合适的大小。然后召开一次发布或迭代计划会议，以制订工作计划，并将根据商业价值的大小选择下一次发布或者迭代要实现的用户故事。

调整阶段主要处理迭代和发布剩余的工作。在这个阶段要收集项目的反馈，并根据这些反馈采取措施。这是一个不断修正的过程，就像调整汽车一样。调整工作在极限编程过程中占有重要地位，要在开发过程中不断进行。一些传统的软件开发方法也许会隔几个星期，甚至几个月才根据反馈做一次调整，这样很容易使项目走入歧途。在极限编程中，软件每天都被测试和集成很多次。通过不断进行调整能够保证项目始终朝着正确的方向发展。

下一个步骤是召开发布计划会议。客户和开发团队共同检查客户提出的用户故事。开发人员分成不同的小组分别检查用户故事，然后评估出这些用户故事的难度等级，或给它们打分。用户故事越难实现，它的得分就越高。同时，这个会议还为开发人员提供了向客户提问的机会。开发人员估计在一次迭代中能完成多少分的工作，然后估算出在第一次迭代中可以完成哪些用户故事。如果不能在第一次迭代中完成所有客户希望的用户故事，就让客户排定这些用户故事的优先级。客户根据每个用户故事的商业价值来决定它们的优先级，并估计它们的难度。规划小型的、频繁的发布是很重要的，这样可以尽快地交付给客户他们所需要的新功能。

极限编程项目通常把 3 到 4 个星期作为一个周期，并把它称为一个发布(release)。在每个发布结束时，以新添加的发布形式把新功能交付给客户。每一个发布都分解成多次迭代，每次迭代通常历时 1 周。在每次迭代开始时，客户和开发人员都要共同召开一次迭代计划会议。在会上，客户先详细地说明每一个用户故事，然后开发人员对一些疑点进行提问。开发人员还将讨论关于如何实现这些故事的技术问题，并公布于众。在会议的最后，开发人员与客户签订合同，承诺要完成哪些任务。

1.2.1　迭代

在每次迭代计划会议后及在每一次迭代过程中，开发人员都要举行简短的设计例会。需要用到类责任协作卡(Class Responsibility Collaborator CRC 卡)或者 UML(Unified Modeling Language)图，从而更好地了解有哪些对象，及对象间如何进行协作。

当开发团队着手编写代码时，要以结对的方式组织团队成员。结对并不是固定的，但它在开发过程中的某一个特定阶段是固定的。在整个开发过程中，每个开发人员都可能和多人结对。编写代码时，一对开发人员首先要编写一段单元测试代码，用它来测试将要编写的类。然后开始编写、编译并测试这个类。在编写单元测试和类时可以进行迭代。举个例子，首先编写用来测试某个类中一个方法的单元测试代码，并在这个类中实现该方法。再开发出测试这个类中下一个方法的单元测试代码，并在这个类中实现第二个方法。关键是要经常进行渐进式的改进和测试。由于要频繁地进行单元测试，所以实现单元测试的自动化势在必行。本书后面的内容将介绍如何进行自动的单元测试，并且让它成为整个创建和部署过程的一部分。实现单元测试自动化后，可以很快发现由于修改而引入的错误代码。

无论何时，当代码被编译并通过单元测试后，就可以被提交到修订控制库中。这时，这些代码将被整合并在整个系统范围内进行测试。XP 依赖于一个持续整合的过程。这意味着整个系统建立于一个有规律的基础之上，也许每天要进行多次整合。这样做的目的是要尽快找到错误和不一致的地方，以便于更容易地修改。作为集成创建的一部分，还需要使用工具检查是否符合编码规范，并生成代码的规格说明。本书还将介绍如何使用第三方的开放源码工具来完成这些任务，以及如何作为创建和发布过程的一部分与 Ant 集成。

在向客户交付发布前，需要进行自动的验收测试。如果出现问题，就要修正错误，重编代码并再次进行测试。如果制定了小规模发布计划，就可以以渐进的方式增加功能，并能保证软件的质量，不断地将新功能交付给客户。

1.2.2　追求速度

使用像极限编程这样的迭代方法进行开发，意味着开发团队几乎每天都要执行多次如下活动：创建、单元测试和集成，并要求尽可能准确、快速地完成这些任务，同时，这些活动必须是可重复的。这就是为什么要将这个需要重复的创建、测试和部署的过程自动化的原因。

建立自动化过程还带来另外一个好处：增加了思考过程的严谨性。在建立自动创建过程来执行一个任务时，程序员将会比在手工记录过程的情况下更加仔细地思考每一个细节。用手工记录的过程不需要被编译和执行，所以很容易遗漏某些步骤，或者出现一些错误。当然，用户也可以测试和修改记录下来的过程。相反，实现自动化的过程虽然并不能保证完全没有错误，但也比任何一个手工记录的过程要可靠得多。

总而言之，实现了自动化的过程意味着较少错误，并且可以简化程序员的工作，同时也能提高开发速度。极限编程提倡尽可能多地实现自动化，这样在减轻程序员负担的同时，也减少了开发过程中的错误。任何需要经常重复的任务都应该实现自动化。

1.2.3　知识共享

过程自动化还有一个好处：整个团队可以共享知识，并使开发人员的知识成为整个团队的