

1、持续集成是什么

　　大师Martin Fowler对持续集成是这样定义的： 持续集成是一种软件开发实践，即团队开发成员经常集成它们的工作，通常每个成员每天至少集成一次，也就意味着每天可能会发生多次集成。每次集成都通过自动化的构建（包括编译，发布，自动化测试）来验证，从而尽快地发现集成错误。许多团队发现这个过程可以大大减少集成的问题，让团队能够更快的开发内聚的软件。

　　在敏捷开发与测试中，持续集成是极限编程十二实践之一（1999年Kent Beck编写的《解析极限编程》），最初被使用极限编程方法的开发人员所推捧，并在过去的几年中得到广泛应用，成为业界广为人知的软件开发实践。该实践用于解决软件开发过程中一个具体且重要的问题，即“确保当某个开发人员完成新的功能或修改代码后，整个软件仍旧能正常工作。”

　　简单来说，持续集成是频繁、持续的在多个团队成员的日常工作中进行集成、验证并反馈。一个典型的持续集成周期基本包含如下几个步骤：

　　1）持续集成服务器不断从版本代码库的服务器上检查代码状态，看代码是否有更新；

　　2）若发现代码有最新的提交，集成服务器就会从版本代码库服务器下载最新的代码；

　　3）等代码完成更新结束后，持续集成服务器调用自动化编译脚本进行代码编译；

　　4）运行所有的自动化测试；

　　5）进行代码分析

　　6）输出可执行的软件，提高给测试人员进行最后的测试与验证

　　7）通过测试工程师的测试与验证，最后发布集成到发布

　　通过上面的持续集成构建步骤我们不难知道，其实持续集成就是一个循环、多次运用、统一检查的过程，如图-2所示描述

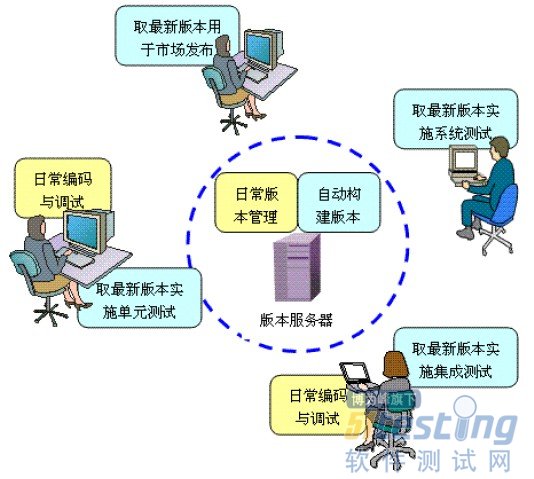
[[](http://www.51testing.com/batch.download.php?aid=40514)](http://www.51testing.com/batch.download.php?aid=40514)

图-2

2、持续集成模式

　　根据持续集成在项目中的运用与分析，从基础搭建模式到企业级的解决方案模式，基本可以分成下面三种模式，它们分别是递增的关系，在软件开发中随着系统的复杂度和测试件的可测性不断改进的过程，最理想的持续集成达到是统一化、流程化和服务化的过程。

　　1）基础模式

　　目前，有很多种持续集成工具，其中不乏开源产品，如Maven、Hudson平台。项目伊始，我们可以建立自己的持续集成服务器，整个项目的持续集成基础结构如图-3所示：

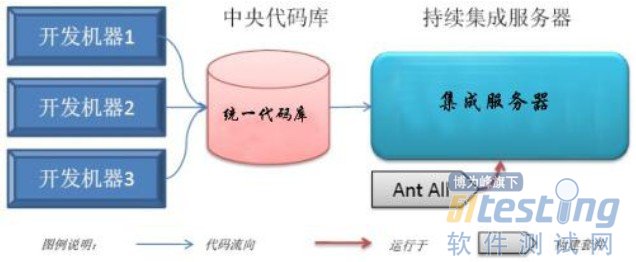
[](http://www.51testing.com/batch.download.php?aid=40515)

图-3

　　并发多个工程师进行代码开发，每个工程师有独立的开发环境和开发分支，然后统一提交到中央代码库，最后进行统一集成编译。

　　2）阶段式模式

　　在基础模式的框架基础上，我们增加软件开发过程单元测试、静态代码检查、UI功能自动化检查，如图-4

　　阶段模式的持续集成较集成提升了很多，在集成server中增加了很多构建套件，综合利用持续集成的特点进行统一管理。

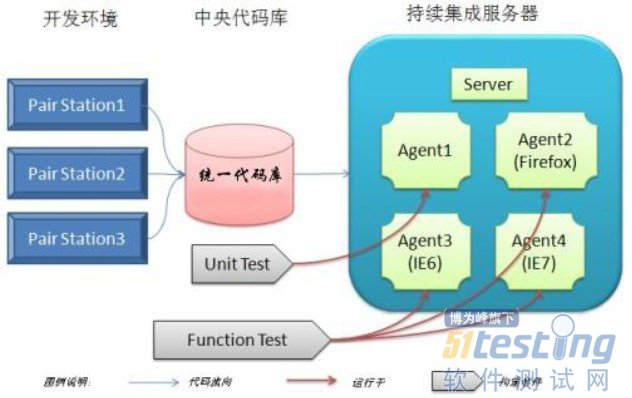
[](http://www.51testing.com/batch.download.php?aid=40516)

图-4

　　3）管道模式

　　阶段式持续集成重复任务多，而过程化集成的管理复杂性太高了，任何过程化上的变化都要修改已经写好的脚本，而这些脚本维护比较困难。既然以上两种模式都不灵了，所以就引出了高级模式就是管道式的持续集成模式。

　　管道式持续集成形式上与过程化持续集成相类似，但却在概念上有显著不同。在管道式持续集成中，所有的过程单元都运行在同一管道的上下文中，即各单元所使用的原材料都是完成相同的，即代码基线相同。当持续集成服务器发现有新的代码时，会创建新的一个管道，所有的过程单元都在这一个管道中运行，包含编译打包、单元测试、功能测试、性能测试和自动化测试。而每个单元产生的产物也在该管道中有效。如图-5所示：

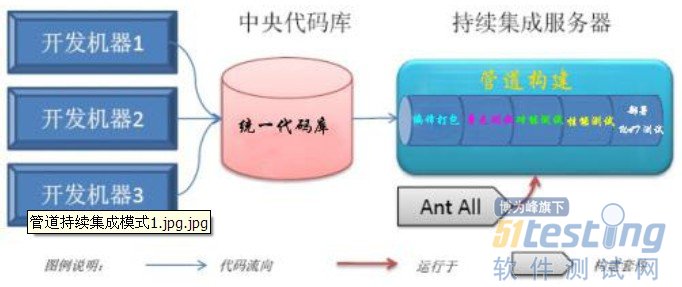
[](http://www.51testing.com/batch.download.php?aid=40517)

图-5

　　不难看出管道模式的持续集成综合了基础模式和阶段模式的优点，在管道式中，每次构建都会试图从管道的一端走到另一端。因此，你不会遗漏任何一个版本的成功产品代码，基本上可以使项目研发过程全部自动化了。

　3、持续集成流程

　　一般在互联网软件开发和测试过程中，增加了持续集成构建，在开发和测试环节会进行多次集成与构建，做的比较好的公司，如google，微软等，可以集单元测试、功能自动化测试等集成在一起构建，做到分支代码变更脚本通知一条龙智能流程化和服务化，可见下图-6流程所示。

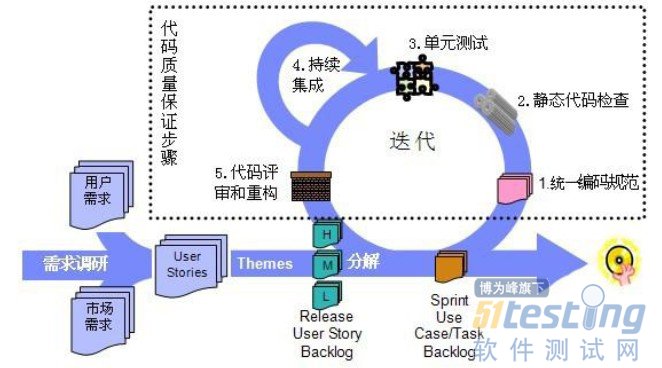
[](http://www.51testing.com/batch.download.php?aid=40518)

图-6

　　4、持续集成优点

　　通过上述概念和模式的阐述，让读者很容易发现持续集成最大的优点就是降低风险，提高项目研发过程的效率和质量，迎合在互联网时代信息快速更新的现象。持续集成本身并不能帮助开发工程师找到bug，它是通过不断的测试和反馈来尽早的发现缺陷，问题发现的越早处理问题的成本就越小越容易解决，由于无法证明通过了测试的代码是无bug存在的，所以持续集成中的测试非常重要，好的测试能够更多更快的发现当前版本中的错误。

　　往往在开发和测试过程中，软件的缺陷是累积性的，当缺陷很多时，就很难发现它们，利用持续集成构建的思想，在项目过程中可以尽早的发现缺陷，最大限度的降低了我们在项目后期发现缺陷的可能性和偶然性。

　　每成功构建运行一次就意味着之前做的代码提交可以成功集成，没有与他人提交的分支代码发生冲突，没有带来新的缺陷，有利于开发人员对项目保持自信心和提高工程师的工作激情。

　　持续集成在项目中的频繁部署将会使最终部署的难度降到最小，用户能够看到频繁上线的软件，并做及时的沟通反馈，有助于增强互联网模式下用户的信心和动力，也有助于Web产品化和服务化朝着正确的方向快速发展。

　　5、持续集成构建分析及工具

在敏捷的软件开发和测试团队中，我们所要做的只不过是：不断的回顾、找出问题与瓶颈、不断地重构。通过不断重构持续集成基础结构以及自动化构建脚本，使其达到我们对“反馈时间”和“判断质量准确性”的要求。另外，我们已将“持续集成”扩展到整个软件开发周期，涵盖了持续部署及发布。在上面的配置文件中，不难看出在管道模式中的两个Stage分别名为 “UAT”测试和交付的“Production”，它们一个用于部署新版本到我们自己的持续集成服务器，另一个用于部署新版本到一个公用的持续集成服务器。部署 ‘UAT’的频率为两天到一周之间，‘Production’的频率基本都是一周。这样，我们可以得到快速反馈，改进自己的产品，同时其它团队可以尽早地使用我们开发的新功能。

6、Jenkins 简单介绍

Jenkins 是一个开源项目，提供了一种易于使用的持续集成系统，使开发者从繁杂的集成中解脱出来，专注于更为重要的业务逻辑实现上。同时 Jenkins 能实施监控集成中存在的错误，提供详细的日志文件和提醒功能，还能用图表的形式形象地展示项目构建的趋势和稳定性

