PEARSON

Visual Studio Team Foundation Server 2012

Adopting Agile Software Practices: From Backlog to Continuous Feedback

敏捷方法与Visual Studio工程实践

[美] Sam Guckenheimer 著
Neno Loje
梁春艳 译

清华大学出版社

敏捷方法与 Visual Studio 工程实践

[美] Sam Guckenheimer 著
Neno Loje
梁春艳 译

清华大学出版社
北 京

内 容 简 介

本书以 Visual Studio Team Foundation Server 2012 为软件开发生命周期(ALM)的平台，着重从 Scrum 等敏捷方法和 Visual Studio 工程实践的角度诠释.NET 开发人员如何充分利用敏捷方法和 VS 管理工具更快交付产品。书中提供最真实的开发技巧与最先进的敏捷实践，旨在帮助解决软件工程所面临的困境与挑战，有系统地终结浪费、改善透明度，让软件开发成为一种轻松愉快的工程过程。

本书适合.NET 程序员阅读和参考，可以帮助他们更快构建更多价值的软件产品，提高用户满意度。

图书在版编目(CIP)数据

敏捷方法与 Visual Studio 工程实践/(美)古根海默(Guckenheimer，S.)，(美)洛耶(Loje，N.)著；梁春艳译.
--北京：清华大学出版社，2015

书名原文：Visual Studio Team Foundation Server 2012: Adopting Agile Software Practices: From Backlog to Continuous Feedback

ISBN 978-7-302-41481-0

Ⅰ.①敏… Ⅱ. ①古…②洛…③梁… Ⅲ. ①程序语言—程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2015)第 212482 号

责任编辑：文开琪
封面设计：杨玉兰
责任校对：马素伟
责任印制：杨 艳
出版发行：清华大学出版社
网 址：http://www.tup.com.cn, http://www.wqbook.com
地 址：北京清华大学学研大厦 A 座 邮 编：100084
社 总 机：010-62770175 邮 购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市春园印刷有限公司
经 销：全国新华书店
开 本：185mm×230mm **印 张**：16 **字 数**：304 千字
版 次：2015 年 9 月第 1 版 **印 次**：2015 年 9 月第 1 次印刷
印 数：1~3000
定 价：49.00 元

产品编号：045191-01

推荐序 1

能够为 Sam 的书写序是我的荣幸。Sam 既是软件开发的从业者，又是一名学者。在过去三年中，我与 Sam 一起工作，将 Scrum 融入现代工程实践与微软的 VS 2010。我们都感谢微软的 Aaron Bjork，他开发了 Scrum 模板，通过 Scrum 模板在 VS 2010 中实例化 Scrum。

我不希望 Scrum 是规定性的。我留下了许多空白，比如产品积压工作的语法和结构形式，将产品积压工作项变成潜在可交付增量的工程实践，创建自组织团队的魔法。在本书中，Sam 很好地描述了填补这些空白的方法。他描述了技术和工具制作及其原理。他描述得很详细，主题广，行文也很幽默。我很高兴与 Sam 自 2009 年以来针对这些实践和工具与微软的合作。我们首先发布的课程是 Professional Scrum Developer .NET 课程，教开发人员如何在 VS (在自组织和跨职能的团队中工作)中使用现代工程实践的稳健增量。Sam 的书是这门课程以及更多课程的权威著作，细节丰富，逻辑缜密。如果你的团队使用.NET 技术来构建软件，那么这本书就是为你写的。如果你正在使用 Java，这本书也足以吸引你，让你觉得也许值得切换到.NET。

在 2001 年设计并签署敏捷宣言时，我们的第一个价值是“个人和交互重于过程和工具”。嗯，我们的过程和工具都是针对微软开发环境的。在 Sam 的书中，有一些开发人员能够理解过程和工具的方法和价值。现在，真正努力工作的人，经过 20 多年的时间已经被看作资源，但很难让他们成为有创造力、负责任的人。我们的第一个挑战是管理开发人员的人。他们可以使用 VS 工具的指标对过程和开发人员进行微观管理，挤出最后一点创造力并放弃敏捷。或者，他们可以使用这些工具的指标来了解开发人员所面临的挑战。然后，他们可

以指导并带领他们做得更好，更具创造力，更具生产力。这是对所有工具的挑战。它可能是出色的，但用好它才能决定它的成功。

感谢你们的书，Sam 和 Neno。

肯·施瓦伯(Ken Schwaber)
Scrum 共同创始人

推荐序 2

Sam 和我在 2003 年遇到过一个事情。我们所在的新团队使命是承接 Visual Studio(这个全球世界领先的个人开发环境)并将其转为世界领先的团队开发环境。他刚刚加入微软并想让我相信我们的成功将不仅仅是打造最好的工具，而是创建最好的端到端集成，展现这个思想。[①]他说服了我和团队其他成员。

他也说服我们应该从一开始就考虑发挥敏捷过程的作用。一个关键思想就是我们把花在迁移上的时间最小化。我们将打造工具使所有数据都透明并尽可能压缩软件过程中的浪费和管理。这样，团队可以聚焦于交付价值流给客户。

这一愿景即我们现在所说的敏捷共识，Team Foundation Server 和 Visual Studio Team System 的第一个版本被宣告为我们 2005 年的产品线。本书第一版阐述了我们做出这个飞跃的原因。

Neno 是我们第一个客户和咨询师。他很快成为我们最强大的拥护者和批评家，抓住这个愿景并识别出一个接一个版本中我们需要弥补的差距。他现在是产品线当之无愧的专家之一，比任何人都清楚我们的用户是如何使用 Visual Studio 产品线的。

自从我们发布了 V1 版本，我们就已经开始了跨微软开发者部门的敏捷转型。我们的产品已经帮助这个改变成为可能。首先，在 VS 2008 的浪潮中，我们就开始规模应用敏捷软件工程实践。在我们今天的产品能力中，比如单元测试、封闭签入、并行开发以及测试实验室管理等，都能识别出这些实践。这些实践帮助我们减少了浪费，并创建了值得信赖的透明度。一旦达到这些目标，我们就能在开发 VS 2010 的第二波浪潮中开始着手真正增加客户价值流。最近，随着 VS 2012 的发布，我们已经真正解决了周期时间问题。最明显的

① 在 VS 2012 中，我们也把故事板纳入了产品。

例子就是托管的 Team Foundation Service，每三周就可以部署到客户那里。第 9 章对此进行很好的描述。

Sam 和 Neno 的这本书解释了为什么要推行现代软件实践，并将其具体化到 Visual Studio 产品线中。这本书并没有试图取代简单的产品文档。相反，它全面覆盖了软件生命周期，从需求积压到部署，并展现了如何应用现代最佳实践来开发正确的东西的例子。如果你是一个业务依赖于软件的管理者，那么你会阅读这本书。如果你是一个试图提升团队速度或让软件满足客户需求的团队领导，那么也请阅读这本书。如果你是一个开发人员或测试人员，并且希望在工作上更出色，将更多时间聚集于任务上并有更多乐趣，请阅读这本书。

布莱恩·哈利(Brian Harry)

Microsoft Technical Fellow，Team Foundation Server 总经理

前言

七年前，我们将 Microsoft Visual Studio 进行了扩展，以包含应用程序生命周期管理(ALM)，使我们的几十万用户以及数万名微软同事的生活更轻松，生产力更高。2010 年，当我们发布了 Visual Studio 2010 高级版、旗舰版、专业测试版以及 Team Foundation Server 之后，我们已经实现了成为广泛公认的行业领导的目标。2012 年，我们给服务器补充了托管团队基础服务的公共预览，并开始更频繁地给软件团队提供更多价值。

过去七年中，我们从客户身上学到了很多东西。Visual Studio 使得高性能的敏捷软件开发团队能够更频繁地发布更高质量的软件。它被广泛公认为提供给软件团队的市场领先的创新解决方案。我们系统地深入研究了浪费应用程序生命周期的主要根源，提升了团队广泛参与的透明度，并专注于最终用户的价值流。我们已经消除了角色之间不必要的筒仓，专注于建设一个多学科的自我管理的团队。下面是一些例子。

- **没有更多的无法再现**。软件开发中浪费的最大一个来源是开发人员无法重现报告的缺陷。传统上，这就是所谓的“无法再现”bug。测试人员或用户提出一个 bug，稍后收到“无法重现”或“它在我的机器上工作”或“请提供更多信息”或类似的响应。通常，这是 Bug Ping-Pong 持久战的第一次迸发，虽然软件没有得到改进，但产生了巨大的挫折感。

  Bug Ping-Pong 对地理分布的团队尤其困难。正如第 1 章和第 8 章详细介绍的那样，VS 2012 缩短或消除了这场没有赢家的游戏。

- **不再等待构建建立**。很多开发团队已经掌握了持续集成的实践方法，从而一天中能多次产生软件的定期构建，甚至是高度分布式的基于 Web 的系统。虽然如此，测试人员经常等待数天得到一个新的构建来测试，因为将此构建部署到现实生产实验室中很复杂。通过虚拟化测试实验室并将自动化部署作为构建的一部分，VS

2012 使得测试人员能够每日或当日没有中断地获取新鲜构建。第 7 章将描述如何构建和实验室自动化。

- **没有更多的 UI 回归**。最有效的用户界面(UI)测试往往是探索性的，无脚本的手动测试。然而，当 bug 修正后，往往很难说它们是实际已修复，还是仅仅没被再次发现而已。VS 2012 通过捕获测试人员探索的操作日志并允许它转换成自动测试，从而消除了这种不确定性。现在，修复可以重新测试，并且自动化能够专注于实际观察到的 bug，而不是猜想的 bug。第 8 章涵盖探索和自动化测试。
- **没有更多的性能回归**。大多数团队都知道失去客户最快捷的方式是缓慢的应用程序或网站。然而，团队不知道如何量化性能需求，因此，直到发布前才测试负载能力，这时修正发现的 bug 为时已晚。VS 2012 使得团队能够尽早开始负载测试。性能量化并不需要提前进行，因为测试能够回答这个简单问题“什么变慢了？”从端到端结果，VS 能够分析代码中的热路径，并直接为开发人员指出故障点。第 6 章和第 8 章涵盖性能分析和负载测试。
- **没有更多遗漏的变更**。软件项目有许多活动部件，迭代越多，部件变动得越多。开发人员和测试人员很容易误解需求或者忽略变更的影响。为解决这个问题，Visual Studio 专业测试版引入了测试影响分析。此功能比较任意两个构建之间的变更，通过查看构建之间完成的工作，并基于先前覆盖情况分析哪些测试覆盖了变更代码，从而建议运行哪些测试。第 3 章和第 4 章分别介绍了产品待办工作和变更管理，第 6 章到第 8 章从单元测试显示测试影响分析以及相应的安全网，构建自动化，和验收测试。
- **不再计划黑盒**。过去，团队往往不得不猜测他们的历史速度和未来容量。VS 2012 直接从 Team Foundation Server 数据库绘制这些并建立一个 Excel 工作表，允许团队查看冲刺中个人的负载有多重。然后，团队可以根据需要透明地转移工作。敏捷计划的例子在第 2 章和第 4 章讨论。
- **没有更多迟来的意外**。敏捷团队，迭代递增地工作，经常使用燃尽图来评估他们的进展。不仅 VS 2012 自动化燃尽图，项目仪表板也从很多维度超越了燃尽图：需求、任务、测试、bug、代码改动、代码覆盖率、构建健康和障碍提供质量和进度的实时视图。第 4 章将介绍运行项目的“快乐路径”并讨论如何解决项目“臭味”。

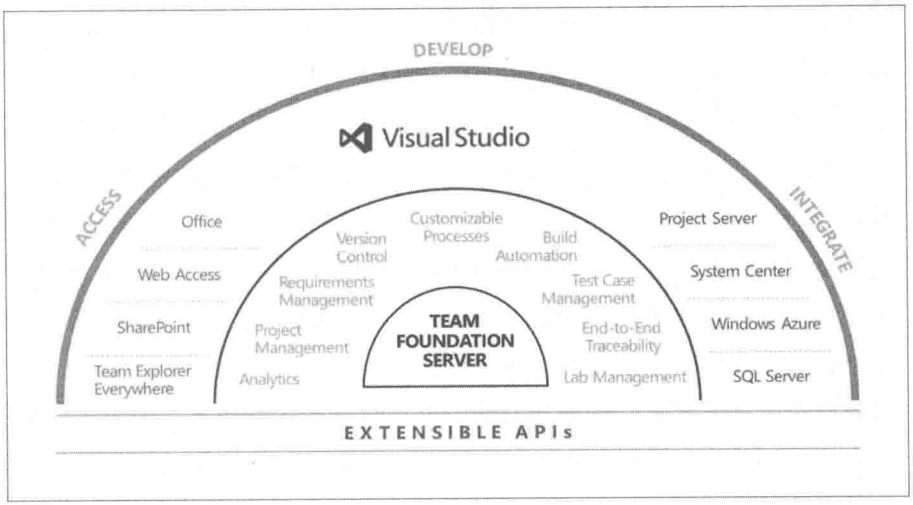
- **不再担心旧版本**。很少有软件项目是真正的“新建”，在新项目上开发全新的软件。更经常的情形是，团队扩展或改进现有系统。不幸的是，较早版本的工作人员往往已经无法解释他们留下的资产。VS 2012 通过引入架构发现工具使得现有代码的使用更加容易。VS 2012 在软件中显示这种模式，使我们能够自动执行减少或消除不必要依赖性的规则。这些规则可以成为签入政策的一部分，确保团队的“完成”定义能够防止意外的架构漂移。架构变化也能够被捆绑到 bug 或工作以保持透明度。第 5 章介绍现有架构的发现，第 7 章告诉我们如何自动化“完成”的定义。
- **没有分布式开发的痛苦**。分布式开发是必需的，原因有很多：地理分布、项目复杂性和版本演变。VS 2012 前瞻性回顾性地去除了分布式开发过程的痛苦。门控式签入在接受签入之前强制执行带有验证测试的干净构建。分支可视化允许回顾性地观察已应用的变更。代码变更和描述变更的工作项更新(例如，bug 修复)都可见。可以直观地发现哪里已经变更以及哪里仍然需要提升。第 6 章和第 7 章告诉我们如何在分布式团队中使用源、分支和积压。
- **没有更多的技术筒仓**。越来越多的软件项目使用多种技术。过去，团队往往不得不根据他们的运行时目标选择不同的工具。因此，.NET 和 Java 团队无法跨筒仓共享数据。Visual Studio Team Foundation Server 2012 通过分别为.NET 和 Java 在 Visual Studio 和 Eclipse 集成开发环境(IDE)中提供客户端集成了二者。这样就将二选一的选择变成了兼容并蓄，双赢了。同样，第 6 章和第 7 章包含使用 Java 资产以及.NET 的例子。

这些场景并不是详尽的清单，而是 VS 2012 开发动机的代表。所有这些都说明了简单优先级：减少浪费，提高透明度，加快终端客户的价值流。本书是为考虑使用 VS 2012 运行软件项目的软件开发团队而写的。本书更多的是关于根源而非方式。

本书总体上是为团队写的。它以一种帮助所有团队成员了解对方观点的方式展示信息。我一直试图保持主题能够吸引所有团队成员。我喜欢爱因斯坦的名言“尽量简单，而不是更简单。”我试着这样写。我希望你会同意并且在看完本书后向你的同事(也许是老板)推荐。

关于 Visual Studio 2012

我所指的 Visual Studio(或 VS)，指的是完整的产品线。如下图所示，VS 2012 系列由一台服务器和一小部分的客户端工具组成，VS 旗舰版上都有。

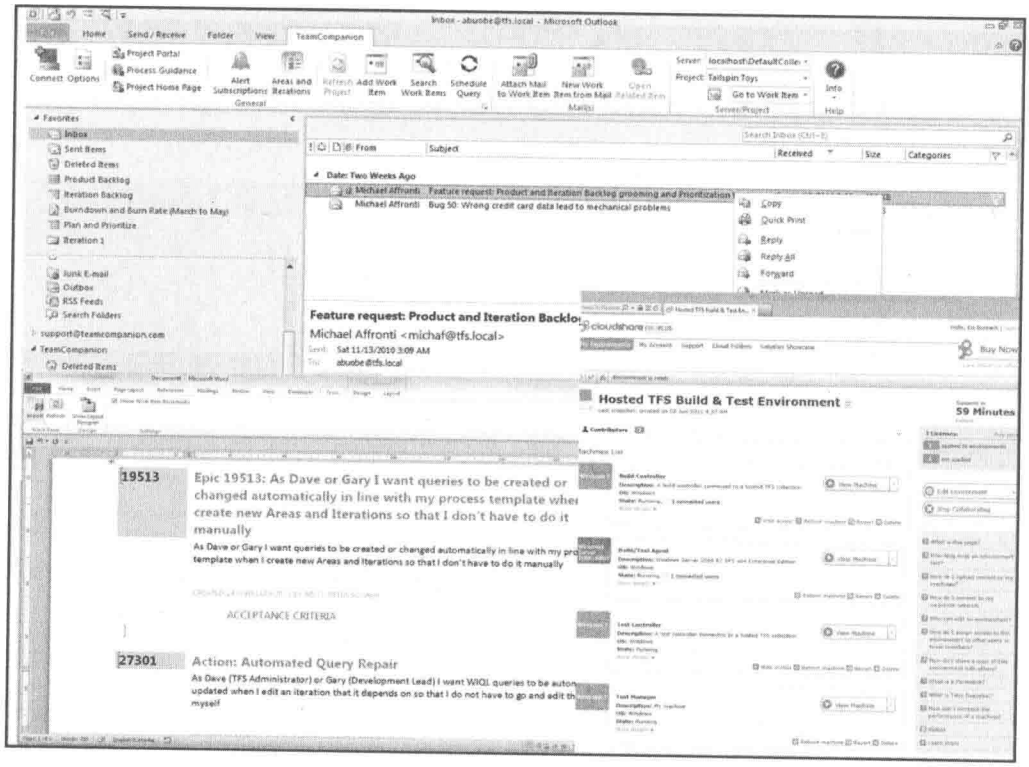


Team Foundation Server(TFS)是 ALM 的主干，提供源代码控制管理、构建自动化、工作项跟踪、测试用例管理，报告和仪表板。TFS 的一部分是实验室管理，扩展 TFS 的构建自动化，将物理和虚拟测试实验室集成到开发过程。

如果你只有 TFS，那么可以得到一个叫 Team Explorer 的客户端，独立或者作为插件启动 Visual Studio Professional IDE。Team Explorer Everywhere，一个用 Java 写的类似客户端，作为 Eclipse 插件启动。你还可以得到 Team Web Access 以及允许连接到 Microsoft Excel 或 Project 的插件 SharePoint 托管仪表板。

Visual Studio 高级版添加了第 6 章中描述的围绕使用代码的场景。Visual Studio 专业测试版尽管具有 VS 的名字，但实际上是一个 IDE 之外的独立应用程序，是为测试人员设计的。可以在第 8 章看到大量专业测试例子。VS 旗舰版包括专业测试，增加了架构建模和发现，在第 5 章讨论。

还有丰富的合作产品，使用可扩展性在 TFS 上提供额外的客户端体验。下图显示一些第三方扩展的例子，启动 MindManager、Microsoft Word 以及 Microsoft Outlook 作为 TFS 的客户端。可以在 www.visualstudiowidgets.com/找到一个目录。



当然，所有的客户端读取并输入数据到 TFS，仪表板上的趋势面通常托管到 SharePoint。使用 Excel Services 或 SQL Server Reporting Services 可以自定义这些仪表板。仪表板示例是第 4 章的重点。

当然，在开发者中心 http://msdn.microsoft.com/vstudio/还可以进一步了解 VS。

作者的话

Sam Guckenheimer

写这本书之前，我已经在微软工作将近三年了。我像下面这样描述自己的历史。

我在 2003 年加入微软 Visual Studio 团队系统(VSTS)的工作，一个在 2005 年年底刚刚发布的新产品线。作为产品群策划师，我扮演着自己喜爱的首席客户支持的角色。我从事 IT 行业已经有二十多年了，我的职业生涯主要是测试人员、项目经理、分析师和开发人员。

作为测试人员，我一向明白先进开发方法，比如单元测试、代码覆盖率、静态分析、内存和性能分析的理论值。同时，我一直不明白怎么会有人有耐心去学习晦涩的工具，你需要遵循正确的做法。

作为项目经理，一直困扰我的是我们只能得到 bug 方面的数据。单独从 bug 数据驱动项目就像闭着眼睛驾驶汽车，只在碰到什么问题才会转动方向盘。我们真正需要的是正确方向上的正确指标，而不是在偏离它时感觉到碰撞。同样，在这里，我总是能理解指标的价值，比如代码覆盖率和项目速率，但我一直不明白怎么会有人真的收集所有这些东西。

作为分析师，我喜欢建模。我通过视觉方式思考并发现图形化模型可以强制记录和沟通。但是模型总是在该实现时过时。

模型处理不好开发人员、测试人员和操作的关键问题。

在所有这些情况下，我很失望的是很难为整个团队把这些事情串起来。我喜欢 Scrum(敏捷过程之一)的这种想法“单一产品积压”(在这里，你可以看到所有工作)，但是人们实际使用的工具使工作无法聚焦。这些需求与任务、模型元素、测试有什么关系？而且，源代码在哪里？

回顾历史，我认为 IT 走出了困境，它停止尝试手动过程自动化，而是提出这样的问题：

“自动化后，我们如何重新设计核心业务过程？”这是它开始提供真正商业价值的时候。

他们说，鞋匠的孩子不穿鞋。对 IT 来说，这也是真的。当我们一直忙于其他业务过程的自动化时，已经在很大程度上忽略了我们自己。几乎所有的工具都针对 IT 专业人员，且团队似乎仍然在自动化旧的手工过程。自动化之前这些过程需要很高的开销，自动化后它们仍然具有很高的开销。如果你去参加一个 1 小时的项目会议，前面 90 分钟肯定都在争论谁的数字正确？

现在，有了 Visual Studio，我们严肃地问：“自动化后，我们如何重新设计核心 IT 过程？我们如何遵循良好过程以消除这些开销？我们在将所有这些不同角色整合成一个高性能团队时，如何使他们具有更高生产力？”

Neno Loje

我从一名软件开发者开始我的职业生涯，开始作为一种业余爱好，后来作为专业。高中一开始，我就爱写软件，因为它能使我将想法变成对别人有实际价值的东西，以此来创造一些有用的东西。后来，我才知道，这就是产生客户价值。

然而，其影响力和价值受限于这样一个事实，即我只是一个供职于一家小公司的单个开发人员，所以我决定专注于帮助并教其他开发人员怎么做。我一开始只提供单纯的技术培训，但主题很快扩展到包括过程和人员，因为我意识到，只引入新工具或技术本身并不一定能使团队取得更大成功。

过去 6 年，作为一个独立的 ALM 顾问和 TFS 专家，我已经使用 VS 帮助许多公司建立起团队环境和软件开发过程。看到消除不必要的手动工作如何使开发人员和整个项目更具生产力，一直都令人着迷。每一个团队都是不同的，都有其自身问题。我惊讶地看有很多方法(包括过程和工具)都能实现同一个目标，即更快交付客户价值，虽然软件功能很强大。

当团队回头看看之前没有 VS 时他们的工作方式，他们通常会问自己没有现在使用的工具他们如何生存。不过，与过去不同的不只是工具，还有他们作为团队的工作方式。

敏捷共识的应用程序生命周期管理和实践帮助团队专注于重要的事情。VS 和 TFS 是实现 ALM(即使是小的，非分布式的团队)的实际方法。如果你还是不相信，我建议你尝试一下，自己判断。

目录