SCRUM 简介

使用Scrum的 Agile（敏捷） 项目管理介绍

作者： **Pete Deemer** 与
Gabrielle Benefield

1.04版本

goodagile>
scrum 培训在印度和亚洲| www.goodagile.com

Pete Deemer 是Yahoo! Emerging Markets Group的首席产品执行官。Gabrielle Benefield是负责Yahoo!公司敏捷开发的高级总监。他们共同致力于Scrum在Yahoo!

公司全球范围的大型推广工作。

读者提示：目前在互联网上有很多关于Scrum的简短介绍，本简介旨在提供更深入一步的研究。此简介并不是学习Scrum的最终阶段；我们建议那些考虑采用Scrum的团队借鉴Ken Schwaber的著作《Agile Project Management with Scrum》和《Agile Software Development with Scrum》，并积极利用和参与目前众多优秀的Scrum培训和咨询活动；详细情况请参考scrumalliance.org. 在此对Ken Schwaber, Jeff Sutherland博士和Mike Cohn 所做的帮助和贡献表示感谢。

传统的软件开发

各类大中小型企业所运用的传统软件构建方法，即是众人皆知的“瀑布”型开发方法。此模型存在很多变体，但其典型性是在开发初期制定详细的计划，在计划中最终产品已被仔细研究，设计，并且一切详细资料都记录在案。任务已设计制定，并且在工作中使用如Gantt (根特)图表等工具和Microsoft Project项目管理软件。开发团队预计开发项目的时间是以累计其相关每一步骤而得出的。当项目管理者(stakeholder)全面审核开发计划并表示赞同，开发团队即时开始工作。团队成员完成他们所专长的部分工作，即刻转交给其他成员，形成生产流水线的形式。当全部工作都已完成，成品将会转交给产品质量控制部门，在这里将会进行在产品到达客户手中之前的全面检测。在

整个流程中，所有相对于原始计划的分歧都将受到严格的控制，以保证生产的成品与原始设计保持一致。

此种模式有优点但也有不足之处。它的最大的优点是非常有逻辑性：在构建之前思考，并全部记录下来，按照计划实施，保持各项事务尽可能的有组织性。它有一个最大的弱点就是：人的参与。

比如：此种方式要求所有的建议和想法都要在开发周期的最启始阶段提出，此时建议和想法将可以被容入开发计划之中。但是众所周知，好的想法和建议的出现会贯穿整个开发过程——
在开发最启始阶段，在开发中期，有时可以在产品交付使用的前一天，如果整个开发过程中不容许变化的产生，那么将会遏制创新的产生。在使用“瀑布”
型开发方式时，开发末期出现的创新将不是好的征兆，而是对整个产品开发产生的危机。

瀑布型开发方式特别注重将事件记录在案，以此作为传递重要信息的首要方法。因此最合理的假定是如果我可以把我的想法尽可能都记录下来，这样将会使信息更可靠的传输到团队每一个成员的头脑中；另外，因为所有东西都记录在案，这将是我完成任务的最实际的证据。但是实际上，在大多数情况下，没人会读这些50页左右的详细规格要求资料。同样，如果有人读取该资料，那么将会产生许多的误解。记录的资料只是我头脑中想法的抽象提取；当你阅读此资料时，你将会产生另一个抽象的概念，此时与我的真正想法已经相距甚远了。所以严重误解的产生也就不足为奇了。

当人参与时,还有一种情况会发生——“实际应用中产生的灵感”——
在第一次实际应用产品时，你可能会立刻产生20种使产品更加完美的灵感。但是遗憾的是，这些非常有价值的想法通常会在产品开发的末期产生，这时创新是最困难和最具有分裂性的——换句话说，是做正确抉择最昂贵的时刻。

人的预知未来的能力是有限的。比如，某场比赛的最终结果是出人意料的。不可预测的技术问题的突然出现，会强制开发方向的转变。此外，人们往往非常欠缺对于未来作出长远计划的能力——今天猜测未来八个月中每周你如何工作将如幻觉般渺茫，这正是Gantt
(根特)图表的衰败表现。

另外，瀑布型方式将会促使团队成员在交接工作时产生敌对的关系。“他要求我构建在规范要求中不存在的东西。”“她经常改变主意。”“我不可能对我不能控制的东西负任何的责任。”这些都引起我们对瀑布方式的另一种思考——
在此种方式中工作毫无兴趣可言。事实上，进一步说，瀑布型方式是造成产品开发人员苦恼的根源，并且最终产品将不会体现出他们的创造力，技能和开发创造者的激情。人不是机器人，此种将人认为是机器人的流程将会造成人们工作激情的丧失。

一个僵硬的，拒绝变革的流程也会造成低劣产品的产生。客户可能会得到根据他们第一需求所生产的产品，但是产品在形成阶段时还是客户真正需要的吗？在开发之前收集所有的需求并将它们嵌入毫无改变可言的顽石般的计划中，此产品将被宣告只会和最起始的想法保持一致，
而不是开发团队在实践中不断发现可能性而使其尽善尽美。

许多使用瀑布型方式的团队不断的体验到了它的缺陷，但是它看起来是如此付有逻辑性的方式，因此第一的自然反应就是对内部工作的谴责：“如果我们尝试更好的使用它，它将会发挥作用”——
如果我们计划的更详尽，记录更详细，更严格的拒绝任何变革，一切将会很顺利地进行。遗憾的是，许多开发团队只得到的相反的效果：他们越竭尽全力尝试此方法，得到的结果越差！

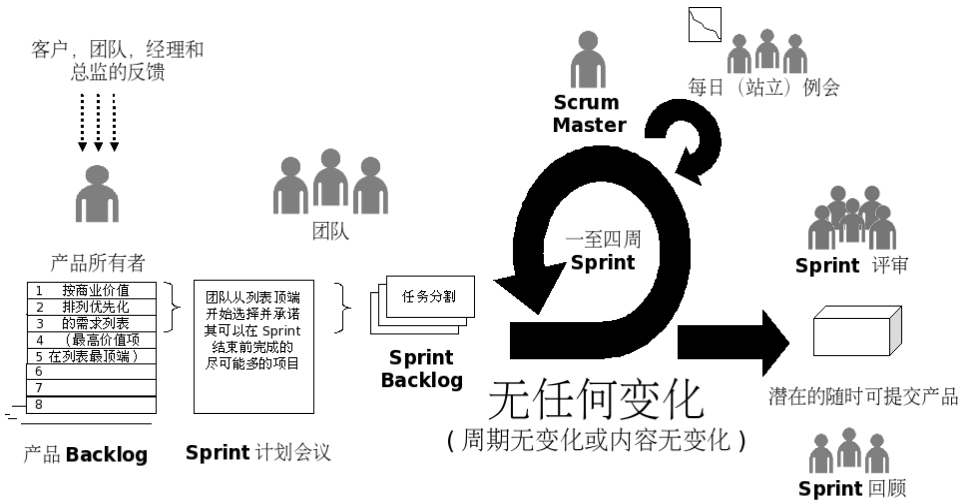
Agile (敏捷) 开发和Scrum

Agile (敏捷)开发整体概念的产生是基于一种方法更接近人类活动现实情况,
以便取得更好效果的信念上。Agile
(敏捷)强调构建可以即时操作的软件，相对于在构建前花费许多时间记录规格要求。Agile
(敏捷)注重授于小型多职能的团队决策权，相对于大型层次和部门职能的划分，Agile
(敏捷)注重快速迭代，并且其中结合尽可能多的客户反馈。在学习了解Agile
(敏捷)的时候，经常会有这样的公认——
感觉非常像回到了开发启始的阶段，我们曾经”做过”。

Scrum是众多快速发展的Agile (敏捷)方式之一。Scrum是在十多年前由Ken Schwaber 和Jeff Sutherland博士共同提出的，现在此方式已被众多大中小型企业使用，其中包括Yahoo! , Microsoft, Google, Lockheed Martin, Motorola, SAP, Cisco, GE , CapitalOne和 US Federal Reserve.
许多使用Scrum的团队取得了重大的改进，其中更有个别在生产效率和职业道德方面得到了彻底地改革。对于产品开发者——许多曾受到“管理层每月的一时狂热”的伤害——
这是非常重要的。Scrum是简单的，强有力的，并立足于常识之中的。

Scrum基础知识

Scrum是迭代的，增量型的流程。Scrum构造的产品迭代周期为**Sprints**,
工作的迭代时间一般为一到四周，并且是相互衔接的。Sprints是有固定的周期——
结束于固定明确的日期，无论该工作完成与否，从不延长。在每一Sprint的启始阶段，一个多职能的团队从已优先化的要求列表中挑选若干项目，并承诺在Sprint的末期完成这些项目；在Sprint中，任务的内容不会变化。每一工作日，团队成员互相通告工作进度，并更新简易的剩余工作量直观表示图表。在Sprint的末期，团队将对这一阶段工作结果作一展示并取得相关的反馈，为下一Sprint做好准备。Scrum强调生产可以使用的产品，意指在Sprint的末期产品的“完成”；在软件方面，是指编码已经被检测并可以随时交付使用。



图示1 Scrum

Scrum中的角色

在Scrum中有三个基本的角色：产品所有者 (Product Owner)，开发团队和ScrumMaster。**产品所有者（Product Owner）**负责取得产品最大的商业价值，收集相关于产品的所有信息——从客户或产品的终端使用者，开发团队成员和项目管理者中获取——并将信息转化为优先权项目列表。在一些情况下，产品所有者 (Product Owner)正是客户本人；在另一些情况下，客户可能是有不同需求的成百上千的人。产品所有者 (Product Owner)这一角色在许多企业中是由产品经理或产品市场经理担任。

开发团队构建客户将会购买的产品：比如软件或网站。Scrum团队是“多功能”的——它包括交付每一Sprint中的随时可交付产品所需的各类专门人员——并且它是有很高自律性和责任性“自我管理”的团队。团队决定承诺完成哪些任务和完成承诺任务最好的方法；在Scrum范畴中，团队被称为“ Pigs猪“，企业组织中其他人员被称为” Chickens鸡“（这些称谓源于一个笑话，一头猪和一只鸡打算一起开一个餐馆叫”火腿和鸡蛋“，但是猪重新考虑了一下，因为”他将需要自我献身，而鸡只需要参与就可以了。“）

Scrum团队通常包括五到十个成员，然而团队大到15个成员和小到3个成员也有很好的收效，一个软件项目的开发团队包括程序员，界面设计师，检测员和研究人员。开发团队不仅构建产品，他们也向产品所有者 (Product Owner)提供让产品尽善尽美的建议和想法。团队成员可以将其时间划分给Scrum项目和其他的项目，但是如果团队成员能专注于Scrum项目开发则效率更高。团队内部成员也可以在不同Sprint中变化，但是这样会减少整个团队的生产效率。大型项目开发通常会组成几个Scrum团队，每一个注重产品开发的一方面并且互相保持紧密的沟通。

ScrumMaster的任务是以任何方式帮助整个团队取得成功。ScrumMaster不是团队中的经理；ScrumMaster是服务于整个团队，帮助团队铲除壁垒而取得成功。协助团队会议，并支持Scrum的实践。在一些团队中会有某一人专心致力于担任ScrumMaster，而另一些小型团队可以采用其中一个成员兼职担任（此人会适当减少日常工作量）。一个好的ScrumMaster可以来自不同的背景

和学科：项目管理，工程技术，设计，检测。ScrumMaster和产品所有者 (Product Owner)不应是同一人；有时，ScrumMaster可能会号召拒绝产品所有者 (Product Owner)（例如，他们有时会在某一Sprint中期试图加入新的条件）的要求。不同于项目经理，ScrumMaster不会指示和分配工作——他们只是协助流程的实施，推动团队自我组织和管理。

除了以上三个角色之外，还有其他对于项目成功作出重要贡献的人员：可能其中最重要的是**经理**。他们的角色在Scrum不断的发展, 他们仍保持了相当重要的位置——
他们支持开发团队，尊重Scrum规则和精神，帮助团队铲除壁垒，为整个项目的开发提供知识，技术和各种必要的协助。在Scrum中，这些人转化了以前“保姆”式的角色（布置任务，收取进程报告，和其他一些谨小慎微的管理方式），取而代之的是承担起更多的“指导“作用（指导职业发展，在职辅导培训，协助铲除障碍，帮助解决问题，提供创新的建议和指导团队成员的技能发展）。为了能更好地实现这一变化，经理们需要改进他们的管理方式方法；比如，运用苏格拉底哲学提问方式来帮助开发团队找寻问题的解决办法，而不是简单地决定解决方法并分配给开发团队。

Scrum启始

Scrum的第一步是产品所有者 (Product Owner)清晰地展示产品的未来景象（vision）。这些是以按需求的优先列表展示的，按客户和商业价值排序，最高价值的项目排在列表顶端。这就是**Product Backlog,**
它存在（并发展）于产品的整个生命周期（图示2）。在项目开发的任何时候，Product Backlog是唯一具有权威性的“以优先权排序为准，需要完成的所有任务”的概况。只可以存在唯一一个Product Backlog；这就意味着产品所有者 (Product Owner)需要在所有的工作范围中作出优先项的决策。

描述	粗略估计大小
授权所有使用者可以在购物车中添加书籍（模拟示例和补充细节在此）	5
升级交易流程模块（必须可以支持每秒 500 个交易）	13
研究加速信用卡确认的方法（见目标绩效度量在此）	20
升级所有服务器为 Apache 2.2.3	8
判断并修正订单流程编译错误 (bugzilla ID 14823)	3
授权所有使用者可以创建／保存购物单	40
授权所有使用者可以增加和删除他们的购物单	20

第一项是当前的
最高优先权项，
第二项是次高优先权项，
以此类推……

↓

图示2 Product Backlog

Product Backlog
包括许多的不同项目，例如功能（“使用户可以把所选书籍放入购物车”），开发需求（“重新改进处理流程模块，使其可以升级”），探索式的工作（“研究关于加速信用卡确认过程的方法”）

，和已知的bugs（“判断并修复定单流程中的错误”）。很多人喜欢依照“使用者经历”来关连需求：依照产品对最终使用者价值来简明清晰的描述其功能。

Product Backlog是由产品所有者 (Product Owner)随时更新，以反映客户需求的变化，新的想法和见识，竞争对手的发布，出现的技术障碍等等。团队提供给产品所有者 (Product Owner)在Product Backlog中每一项所需的相对工作的粗略估计，这可以帮助产品所有者 (Product Owner)作出优先权项的决策（一些项成为非优先权项，是当产品所有者 (Product Owner)了解到其交付需要花费大量的工作）。以上这些估计是相对的，因此他们可以用“点”来衡量，而不是用现实的工作单位如人员－星期；一段时间之后，团队收集对于其工作速率的数据（在一段时间内有多少这些相对的“点”可以被完成），可以使用这些数据计划发布日期和进行其他长期的计划。

Product Backlog中的项目在规模上会相差甚远；有些大的项目通常在Sprint计划会议上被划分为许多较小的项目，而小的项目有些会被合并为一。关于Scrum的误解之一是它会阻止你记录详细的规范说明；而现实中，这是由产品所有者 (Product Owner)和开发团队共同决定详细资料的多少，这些从其中一个Product Backlog到另一个有可能存在不同。一般的建议是，在所需的最少的空间中说明最重要的事情——
换而言之，不需要描述某一项目的所有可能的详细资料，只需要阐述清楚产品被认为是完成品所具备的要求。在Product Backlog列表中越底端的是更大和粗略的项目；当它们接近被开发时，产品所有者 (Product Owner)会添加更多的详细资料。

Sprint计划会议

Sprint计划会议在每一Sprint的启始阶段进行。

在Sprint计划会议的第一部分，产品所有者 (Product Owner)和Scrum开发团队（在ScrumMaster的协助下）共同评审Product Backlog，讨论Backlog中各项目的目标和背景，并提供Scrum开发团队深入了解产品所有者 (Product Owner)想法的机会。

在会议的第二部分，Scrum开发团队从Product Backlog中挑选项目并承诺在Sprint的末期完成任务，从Product Backlog的顶端开始（换而言之，从对产品所有者 (Product Owner)具有最高优先权的项目开始）并按列表顺序依次工作。这是Scrum的重要实践之一：开发团队决定承诺完成工作量的多少，而不是由产品所有者 (Product Owner)安排工作量。这就使任务的交付更可靠；第一，因为开发团队承诺工作量，而不是其他人代替其“承诺”工作量；第二，开发团队自己决定所需要的工作量，而不是其他人决定工作量“应该”是多少。产品的所有者对于开发团队承诺任务多少没有控制权，他或她只知道开发团队负责的项目是由Product Backlog中按顺序从上至下进行的——

换而言之，是从他或她选择的最重要的项目开始。开发团队可以从列表底层选择项目提前完成，如果其对于整个开发具有意义（比如，提升和快速完成较低优先权的项目，作为完成较高优先权项目的一部分）。
Sprint计划会议通常会持续几个小时——
开发团队对于承诺完成的任务作出认真地抉择，这些责任要求通过仔细地考虑以达到成功的目标。团队将从估算每一成员拥有的完成Sprint相关工作的时间开始——
换句话说，是团队成员平均的工作时间减去他们花费在检修bug和其他必要的维护工作，参加各种会议，回复电子邮件，午休等的时间。大多数人会有4到6个小时每个
工作日可以完成Sprint相关的工作。（图示3）

Sprint 周期	2 星期
Sprint 中包含的工作天数	8 天

团队成员	**Sprint** 周期中可利用的天数 *	每日可利用的时间	**Sprint** 中总共可利用的时间
Tracy	8 天	4 小时	32 小时 [=8 x 4]
Sanjay	7 天	5 小时	35 小时
Phillip	8 天	4 小时	32 小时
Jing	6 天	5 小时	30 小时

*除去假日，其他
不在办公室的天数

图示3 预测可利用时间

当可利用时间确定下来，开发团队会从Product Backlog的顶端第一项开始工作——
换句话说，从产品所有者 (Product Owner)的最高优先权项开始——
团队共同工作，将其分配为小块任务，并记录在**Sprint Backlog**中（图示4）。当任务确定后，团队成员可以自愿承担任务，在考虑任务间依赖关系和次序的情况下，给每一项任务估算时间，并保证每一个人的工作量合理。在此流程中将会和产品所有者 (Product
Owner)有往复交流，阐明要点，核实交易，将较大型Backlog分割成小块，并保证开发团队真正全面了解开发的需求。开发团队将按Product
Backlog序列继续计划，直至消耗所有可利用时间。在会议的末期，开发团队将提供所有任务的列表，并写明每一项工作由谁完成和他们的估算时间（一般以小时计算或每天的一小部分）。
许多开发团队也使用可视任务跟踪工具，利用墙体面积大小的任务板，任务（写在便签上）在Sprint中跨越栏目，“未开始”，“在进行”，“需校验”，和“已完成”。

				Sprint 中每日所剩余的工作时间									
Backlog 项目	任务	任务所属者	初步估算	第一天	第二天	第三天	第四天	第五天	第六天	第七天	第八天	第九天	第十天
授权所有使用者可以在购物车中放置书籍	设计商业逻辑	Sanjay	4										
	设计用户界面	Jing	2										
	执行后端编码	Philip	6										
	执行前端编码	Tracy	4										
	完成单位检测	Sarah	4										
	完成回归检测	Sarah	2										
	编写文档	Sam	3										
升级交易流程模块（必须可以支持每秒500个交易）	合并 DCP 编码并完成分级检测	Jing	5										
	完成设置 pRank 机器命令	Jing	4										
	更改 DCP 和阅读程序 使其使用 pRank http API	Tracy	3										
		总计	**50**										

图示4 Sprint Backlog

Scrum的重要支柱之一是当Scrum开发团队确认承诺任务后，产品所有者 (Product Owner)在此Sprint期间不可以添加新的需求。这就意味着即使在Sprint中途，产品所有者 (Product Owner)想要添加新要求，他在下一新的Sprint开始前不可能作出任何变更的决定。如果一个外部情况出现致使项目优先级的变化，意味着如果开发团队按原计划工作将会是浪费时间，产品所有者 (Product Owner)此时可以结束该Sprint；意味着开发团队停止一切工作，重新开始Sprint计划会议等等。此种决断会产生很大的影响, 除非在非常特别情况下，不会采用这种方式。

保证开发团队在Sprint中目标不被变换有着正面影响。首先，开发团队确信在工作开始时的承诺是不会变化，这样会集中开发团队的注意力。第二，这样也可以培养产品所有者 (Product Owner)在安排Product Backlog中项目的优先权时，适时作出正确的抉择。他们知道任务的承诺是在整个Sprint中不可变更的，使其在决定从哪一项目开始工作更为细心。

作为回报，产品所有者 (Product Owner)也可以得到两个好处。首先，他或她有充分的信心，开发团队对所承诺任务强烈负责，并随着时间推移，Scrum开发团队将会做的很好。第二，产品所有者 (Product Owner)可以在下一Sprint开始前，提出他或她希望Product Backlog的项目变动。在这一点上，增加条件，删减条件，更改条件和重新排列优先权都可以被接受。但产品所有者 (Product Owner)不可以在Sprint进行中提出任何变更，他或她只是需要等待一个Sprint的完成时间或更短。不再僵持于是否变化——方向的变更，条件的变更，或只是简单的想法的变更——可能正式此原因，使得产品所有者 (Product Owner)成为Scrum拥护者中最热衷的成员。

每日（站立）例会

每当Sprint开始,
Scrum开发团队将会实施另一个Scrum的重要实践方法:**每日（站立）例会**。这是在每个工作日特定的时间举行的短小（15分钟）的会议，Scrum开发团队的每一成员都将参与；为了保证其短

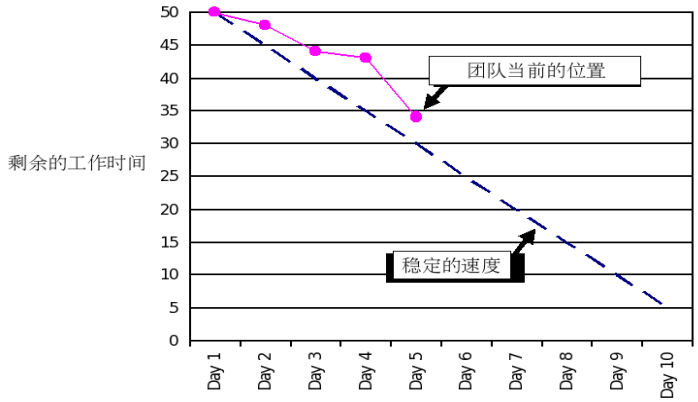
小精悍，与会成员都保持站立（因此为“站立会议”）。以此提供给开发团队机会来汇报交流成果和阐述任何存在的障碍。一个接一个，每个团队成员只可以向其他人汇报三件事情，并且只这三件事情：从上次会议之后完成了哪些工作，在下次会议之前准备完成哪些工作，在工作进行中存在哪些障碍。ScrumMaster将会把障碍内容记录下来，在会后协助团队成员铲除障碍。在每日的（站立）例会中不容许讨论，只是将以上三个重点信息做一汇报；如果需要讨论，将在会后进行。产品所有者 (Product Owner)，经理和项目管理者可以参加会议，但是他们应该在会议结束以前避免问问题或提出讨论——每一与会者应该清楚的是开发团队是在互相汇报和交流情况，并不是向产品所有者(Product Owner)，经理或ScrumMaster汇报。有些团队邀请产品所有者 (Product Owner)参加并将其每日工作做一简要阐述，这是团队可以自由选择的。

在会议结束后，开发团队成员将对其负责的Sprint Backlog中的项目做剩余时间的更新（图示5）。

				Sprint 中每日所剩余的工作时间									
Backlog 项目	任务	任务所属者	初步估算	第一天	第二天	第三天	第四天	第五天	第六天	第七天	第八天	第九天	第十天
授权所有使用者可以在购物车中放置书籍	设计商业逻辑	Sanjay	4	4	3	3	1	0					
	设计用户界面	Jing	2	2	1	1	1	1					
	执行后端编码	Philip	6	6	2	5	2	0					
	执行前端编码	Tracy	4	4	3	2	2	2					
	完成单位检测	Sarah	4	4	3	3	3	3					
	完成回归检测	Sarah	2	2	3	3	3	3					
	编写文档	Sam	3	3	4	2	0	0					
升级交易流程模块（必须可以支持每秒500个交易）	合并 DCP 编码并完成分级检测	Jing	5	5	2	2	1	0					
	完成设置 pRank 机器命令	Jing	4	4	2	0	0	0					
	更改 DCP 和阅读程序 使其使用 pRank http API	Tracy	3	3	3	2	2	2					
		总计	**50**	**50**	**48**	**44**	**43**	**34**					

图示5 每日估算关于在Sprint Backlog中所剩余的工作时间

根据更新情况，ScrumMaster将团队剩余工作时间作一总结并绘制在**Sprint Burndown Chart**图表中（图示6）。它是用来显示每日直至开发团队完成全部任务的剩余工作量（以小时或天计算）。理想的情况下，抛物线轨道在Sprint的最后一天应该接触零点。有些时候会是这样，但是大多数情况不是这样。重要的是它体现了团队在相对于他们的目标的实际进展情况——并不是目前花费了的时间的多少（对于Scrum来说这是不太相关的事项），而是仍剩余多少工作量——
开发团队仍距离完成任务多远。如果此曲线的轨道在Sprint末期不是趋于结束，那么开发团队应该加快速度，或简化和削减其工作内容。此图表也可以使用Excel表格管理，许多团队认为在他们工作室的墙上用图纸标明更为简单和有效，并可以用笔随时更新；这个技术含量不高的做法比电子表格更快速，简易和更可见。



图示6 Burndown Chart

Scrum的核心原则之一是Sprint的周期不可以延长——
其结束于指定的时间不论开发团队完成任务与否。如果开发团队未完成其Sprint目标，他们将在Sprint的末期声明他们没有完成预期的任务。这个方法创造了反馈循环的可视性，并强制开发团队在Sprint预期估算时做出更好的判断，并确保可以按时完成任务不会失败。开发团队通常在前几个Sprint时试图完成许多任务，但实际上并不可以达到Sprint目标；然后他们可能会过度小心而挑选较少的任务，结果会提前完成；在第三或第四个Sprint时，开发团队通常会了解自己的工作效率，他们会按时完成Sprint的目标。鼓励开发团队在某一Sprint中挑选一个周期（比如两周）并且不经常变化——
固定的周期可以帮助开发团队了解他们的实际工作效率(这样可以帮助预算和长期发布的计划），并且可以帮助其达到工作的节奏性（此指团队的统一“心跳”）。

Sprint评审

在Sprint结束后，将进行Sprint评审，团队在此期间展示他们所构造的产品。出席此会议的有产品所有者 (Product
Owner)，开发团队成员，ScrumMaster，加上客户，项目管理者，专家，高层人士和任何对此感兴趣的人。这不是开发团队做成果“演讲”——
会议上不会有PowerPoints图片文件，通常会议不会需要超过30分钟的准备时间——
这只是简单的展示工作结果，所有与会人员可以提出问题和建议。会议可以持续10分钟，也可以是两个小时——会议目的只是对工作结果的展示和听取反馈。

Sprint回顾

在Sprint评审之后,开发团队会进行Sprint回顾。有些开发团队会跳过此过程, 这是不合适的,
因为它是Scrum使潜在的改进可视的重要技巧，并可以将其转化为结果。这是提供给开发团队的非常好的机会，来讨论什么方法能起作用而什么不起作用，并一致通过改进的方法。Scrum开发团队，产品所有者 (Product
Owner)和ScrumMaster都将参加会议，会议由外部中立者主持；一个很好的方法是由ScrumMaster互相主持对方的回顾会议，可以起到各团队间信息传播的作用。

组织Sprint回顾的最简单方法是在白板上画两个栏，分别注明“哪些项工作顺利”，“哪些项可以做的更好”——
让与会者在每一类别下增加些项目。当项目重复时，可以在该项旁边记正字累计，这样一些比较普遍出现的项目就一目了然了。然后团队成员共同讨论找寻这些项目出现的根本原因，同意在下一个Sprint中的改进计划，并负责在下一个Sprint回顾会议上评审项目结果。

一个非常实用的方法是在回顾的尾声，让团队成员在每一类下的项目中，用“C”标记如果其根源是Scrum，或用“E”标记如果其是由Scrum显现出来的（换句话说，无论Scrum存在与否该项目都会发生，但是Scrum使开发团队注意到了该项目的产生）。或者用“U”表示，如果其产生与Scrum无关（如天气情况）。开发团队会在“哪些项目工作顺利”下发现许多的C标记，在“哪些项目不成功”下有许多的E标记；这是个非常好的现象，即使“哪些项目不成功“是比较长的列表，因为解决问题根本原因的第一步就是让其显现出来，Scrum正是此作用的强有力的促进因素。

开始下一个Sprint

在Sprint评审会议之后，产品所有者 (Product Owner)将提取所有建议，和在Sprint中产生的新的优先权项目，并将这些项目合并于Product Backlog之中；增加新的项目，现有项目进行了更改，重新排序或删除。当Product Backlog的更新完毕，循环周期可以再次开始，以下一个Sprint计划会议为开端。

许多开发团队感觉在每个Sprint末期进行优先化的会议很有意义，和产品所有者 (Product Owner)一起对下一Sprint中的Product Backlog项目进行评审。除了给开发团队的一个机会提醒产品所有者 (Product Owner)其未注意的事项——技术维护，比如——
此会议也开始了在Sprint计划会议之前所需的初步想法。

在各Sprint之间没有间隔期——
开发团队通常在下午时间进行了Sprint评审，第二天上午就进行下一Sprint计划会议。Agile (敏捷)开发的价值观之一是“可持续性“，只有在正常的工作时间和劳动强度下，开发团队才可以继续此周期的持续。

产品发布计划

Sprint持续直至产品所有者 (Product Owner)决定产品已经可以准备发布，此时会有“发布Sprint“来进行最后的整合和发布产品前的检测。如果开发团队一直贯彻很好的开发方法，不断地重构和持续集成，在每一Sprint中的有效测试，就不会存在许多遗留问题需要清除。

有这样一个问题有时会被提到，是怎样在一个迭代的模式中产生长期的发布计划。在一个项目的起始阶段，开发团队会作出粗线条的发布计划；他们不可能预先得知工作的结果，其重点是创建一个大体的计划提供给项目发展一个大体的方向，并阐明交易决策如何形成（比如范围相对于进度表）。以此作为路标来指引你向目标迈进；在行程中你实际挑选的路程和所做的决策

都是途中决定的。

有一些产品发布是以日期界定的；比如：“我们会在11月10日的产品展示会上发布我们项目的2.0版。“在此种情况下，开发团队会在现有的可利用时间内完成尽可能多的Sprint（构建尽可能多的功能）。有些产品要求某部分构建完成才可以说明整个产品的完成，产品不会在这些要求满足前被发布，无论周期长短。Scrum强调在每一Sprint中都生产出可以随时交付的编码，开发团队可以进行中间的发布，使客户可以更快的收到产品的效益。

大多数产品所有者 (Product Owner)会选择一个发布方式，但是会通知其他的——
比如，他们会决定发布日期，他们会与开发团队成员一起对Backlog中项目的完成日期做一大体的估算。在“固定价格/固定日期/固定交货期”的情况下——比如，合同制开发——
在这些变量中至少有一个内部存在缓冲区，可以容许不确定因素和变更；在此方面，Scrum于其他的开发方法并无区别。

普遍存在的挑战

Scrum不是流程－而是提供给团队可视性的框架，并且容许他们相应的“检验和适应”的技巧。Scrum试图让许多存在于开发团队中的问题显现出来。比如，大多数的开发团队并不擅长于估算在固定期间内完成的工作量，因此在第一个Sprint结束时不可能交付他们预计完成的工作。对于开发团队来说，这个是工作的失败。事实上，这个经验正是能更好预计工作量所需的第一步，并促使其对承诺的任务更加负责。这种模式－Scrum可以使机能失调更易显现出来，让团队切实可以解决问题－是使用基础的技巧产生出最有意义的收益，是团队使用Scrum的经验之一。

一个开发团队普遍犯的错误是，当他们在Scrum实践中遇到挑战，他们会改变实践方法，而不是改变自身的问题。比如，开发团队有困难完成Sprint任务时，会延长Sprint的周期，这样他们就会有充足的时间完成任务——
在流程中，确保自身不用提高工作预测的技能和更好的管理时间。这样，在没有经验丰富的Scrum培训师的指导下，开发团队只会将Scrum作为自身弱点和机能失调的映射，并破坏了Scrum真正的意义：使优点和缺点都显现出来，给开发团队自我的提高提供机会。

另一个普遍存在的错误是，人们以为某一实践方法是被禁止或不提倡的，只是因为Scrum没有明确的提出此方法。比如，Scrum并不特别要求产品所有者 (Product Owner)对于他或她的产品作出长远的目标计划；也没有要求工程师请教更有经验的人员关于比较复杂的技术问题。Scrum将这些问题留给个人作出正确的处理；在很多情况下，以上两个方法（和其他许多的方法）会被建议。做个正确的比喻：“因为Scrum没有提到早餐的问题，并不意味着你就得挨饿！”

另一个需要留意的问题是有时经理会强制开发团队使用Scrum；Scrum是为开发团队提供自我组织的空间和工具，由上层强加于开发团队恐怕不是取得成功的良方。比较好的方法是让开发团队从同事或经理处了解到Scrum，并通过专业系统的培训，在开发团队实验此方法（比如90天）后作出决定；在实验周期后，开发团队通过经验的总结来决定是否继续采用此方法。

在第一个Sprint之后，结果往往会对开发团队产生很大的挑战，采用Scrum的好处也会在其尾声显现出来，使得许多新的Scrum团队宣称：“Scrum是困难的，但是它比我们以前采用的方法要好许多！”

Scrum的成果

Scrum产生的收益在开发团队经验中体现在不同的方面。在Yahoo!公司，在过去的30个月中我们将近90个开发项目采用Scrum，一共近900人参与，采用Scrum的团队数量在快速发展。这些项目从客户界面，网页设计如Yahoo! Photos, 到后端基础构造服务如服务上百万客户的Yahoo! Mail；从全新的产品如Yahoo!
Podcasts，其利用Scrum作为从构思到发布的流程（并获得该年度同类产品的Webby奖），到一些增量的项目，包括开发新的部件并维修bug和其他的维护工作；我们也利用Scrum来分配分布式项目。每年多次，我们对Yahoo!公司每一位Scrum的使用者进行调查（包括产品所有者 (Product
Owner)，开发团队成员，ScrumMaster和这些人员的经理），并让他们将Scrum于以前的开发方式做一对比。以下是相关的资料显示：

- **生产效率：**68%回复显示采用Scrum后生产效率提高（4分或5分在以5分为衡量标准上）；5%显示采用Scrum后生产效率降低（1分或2分在以5分为衡量标准上）；27%显示采用Scrum后无变化（3分在以5分为衡量标准上）。

- **团队精神：**52%回复显示采用Scrum后团队精神加强；9%显示采用Scrum后团队精神减弱；39%显示采用Scrum后无变化。

- **适应性：**63%回复显示采用Scrum后适应性加强；4%显示采用Scrum后适应性减弱；33%显示采用Scrum后无变化。

- **责任性：**62%回复显示采用Scrum后责任性加强；6%显示采用Scrum后责任性减弱；32%显示采用Scrum后无变化。

- **协作能力：**81%回复显示采用Scrum后协作能力加强；1%显示采用Scrum后协作能力减弱；18%显示采用Scrum后无变化。

- **在产品所有者 (Product Owner)估算下，开发团队的生产效率平均提高了36%。**

- **85%的开发团队成员表示如果其拥有决策权的前提下，将会继续使用Scrum。**