敏捷世界中的 TMMi

版本 1.4

由 TMMi 基金会制作

编辑：Erik van Veenendaal

贡献者

Asim Ali（阿拉伯联合酋长国）

KatalinBalla（匈牙利）

Clive Bates（英国）

Jan JaapCannegieter（荷兰）

Vahid Garousi（荷兰）

Alon Linetzki（以色列）

Fran O’Hara（爱尔兰）

Jurian van de Laar（荷兰）

Leanne Howard（澳大利亚）

Poonam Jain（印度）

Tim Moore（英国）

Alfonsina Morgavi（阿根廷）

Meile Posthuma（荷兰）

Matthias Rasking（德国）

Chaobo Shang（中国）

Erik van Veenendaal（博内尔岛 – 荷兰加勒比）

Blaine Webb（英国）

Karolina Zmitrowicz（波兰）

致谢

中文翻译参与者（按姓氏拼音排序）

曹栋、郝毅（组长）、霍嘉、李知益、吕纬、孙建成、汪丹唯、肖烨、苑普光、张桂伟、张烨宜

中文评审参与者（按姓氏拼音排序）

陈晟、戚大伟、商超博、唐淼淼 、郑文强（组长）

致谢企业（按企业名称拼音排序）：

新华三集团

中国工商银行业务研发中心

版本历史

本节仅提供信息之用

版本	日期	内容
1.0	30/06/2017	介绍章节和所有 TMMi 2 级过程域
1.1	16/05/2018	添加了所有 TMMi 3 级过程域的敏捷情境中的解释。
1.2	10/12/2018	添加了所有 TMMi 4 级过程域的敏捷情境中的解释。
1.3	03/07/2019	添加了所有 TMMi 5 级过程域的敏捷情境中的解释。
1.4	24/12/2019	添加了 6.4 及 6.5 章节显示 TMMi4 级和 TMMi5 级精确适用性。

目录

1. 介绍

1.1 目标

本文阐述了如何将软件开发的敏捷方法和 TMMi 测试过程改进模型相结合，使之成为实现业务目标的可能途径。它解释了在敏捷环境中如何有效地使用和应用 TMMi，为实现 TMMi 实践的传统测试方法提供了经过验证的替代方案，同时保持并甚至提高了其灵活性。本文通过提醒人们注意随着组织成长和项目压力增加而常常失去可见性的关键测试实践，涵盖了 TMMi 如何帮助敏捷组织，无论该组织在提高敏捷性的过程中多么成熟。每个 TMMi 过程域及其特殊目标都将逐一讨论。它描述了这些与敏捷的关系，以及这些实践通常是什么样子。如果由于实践没有附加价值，而不希望在敏捷上下文中得到执行，则也将在本文明确说明。

许多（小型）敏捷组织都取得了成功并不断壮大。但是他们几乎没有文档化的过程，也很少为相关人员提供正式的培训课程。为了在组织成长过程中保持成功，他们需要更富有纪律性的过程原则。但同时他们也担心失去成就当前成功的敏捷文化。这也是在敏捷环境中启动 TMMi 测试过程改进计划时面临的挑战之一。

这份文档有许多目标读者，其中两个主要群体是：

- 希望在保持过程成熟的同时转向敏捷的成熟传统组织。
- 取得成功并不断扩大规模的敏捷组织。因此他们需要在保持敏捷收益的同时，获得一定的过程成熟度。

本文档不打算独立于原始 TMMi 模型使用。它旨在为在敏捷环境中开展测试过程改进的人员提供各种有关 TMMi 目标和实践的解释和含义的指导。本文档是对原始 TMMi 模型的补充。

1.2 TMMi 和敏捷

有一种错误观点认为，TMMi 和敏捷方法是不一致的。其实，敏捷方法和 TMMi 不仅可以共存，而且成功整合将会带来实质性的收益。其中的一个挑战是以不同方式看待测试，如何与敏捷开发集成，以及这在“测试”改进方案中意味着什么。请注意，TMMi 的“i”是指测试应该成为软件开发的一个组成部分，而不应作为完全独立的内容。关于敏捷项目测试的文献和介绍往往侧重于单元测试、测试自动化和探索性测试，但还有更多！在敏捷环境中使用 TMMi 模型可以提醒人们注意那些经常“被遗忘”的关键测试实践。本文将通过示例展示 TMMi 和敏捷方法可以有效协同工作。这其中面临的挑战是应用精益原则来增强敏捷实践并促进 TMMi 实践。

实施 TMMi 时必须考虑到 TMMi 模型的目的不是强加给组织一套实践，也不是将其作为一个必须“证明遵循”的标准。如果使用得当，TMMi 有助于定位到所应用的特殊测试过程域，其变更可以为给定的业务目标带来价值。无论使用什么样的（开发）生命周期

模型，情况都是如此。切记，TMMi 实践是一个期望组件，可以通过“替代”实践来替换 TMMi 中已定义的实践。试想一下，实践的目的是什么，理由是什么，它如何为企业增加价值？通常在敏捷文化中，目标已经通过另一种实践来实现了。通常，“任何”解决方案只要受到业务需求的驱动即可符合要求！在使用 TMMi 时不要太过刻板，教条并不是 TMMi 的初衷。始终根据您的情况解读 TMMi 的目标和实践。一般情况下，首先在您的特定业务环境中确定过程需求，然后可以决定如何聚焦并推动过程改进的优先级。

TMMi 和敏捷之间产生的大多数冲突，要么是基于 TMMi 的历史观点，即“良好实践”在实现时应该是什么样子，要么是基于敏捷实践应该如何支持敏捷价值观而对敏捷实践基本原理的误解。TMMi 专家，包括（主任）评估师，将需要重新思考，对无意中传递的关于哪些实践看起来更像是“与 TMMi 标准一致”这样的信息进行反思。当敏捷方法与 TMMi 过程一起实施时，这有助于测试实践的有效实施，而不是它们的缺失。请注意，除了 TMMi 中的特定（测试）实践之外，还有一些通用实践。通用实践的目的是支持过程域的制度化，这意味当新成员进入或组织内发生其他变化时，组织有一个基础来支持该过程域。

从基于传统的软件开发转向敏捷，往往会提出对当前定义的过程进行修改和精简。通过这种方式，基于 TMMi 的组织将从敏捷的思维中获得收益。人们往往过分解读 TMMi 模型，导致不必要的非增值过程和工作产品。通过回归根本和改进目标并按照预期使用 TMMi 模型，将得到与实际过程需求和目标一致的，且有实际价值增加的过程。敏捷思维模式下的裁剪和精益过程将反映人们真正所做的过程，并确保只收集有用的数据。敏捷思维也将注意力放在尽可能保持简单性，这通常不容易，但会为 TMMi 实施者带来好处。敏捷内部的改进通常由小型的授权团队进行，这些团队可以采取快速行动，这也是 TMMi 可以从敏捷中受益的另一种方式。另外请记住，TMMi 3 级和 TMMi 4 级及 5 级之间存在自然跨度。对敏捷组织而言，只有当 TMMi 4 级和 5 级确实重要并且带来增加值时，才推荐实施，同时需要精心选择其中的实践。虽然 TMMi 是全面的，但成功的组织必须确定关键的测试实践并聚焦需要的改进。

1.3 测试成熟度模型集成（TMMi）

TMMi 框架由 TMMi 基金会开发，作为测试过程改进的指导和参考框架，解决那些对测试经理，测试工程师，开发人员和软件质量人员来说很重要的问题。TMMi 中定义的广义的测试涵盖所有与软件产品质量有关的活动。

TMMi 使用成熟度级别的概念进行过程评估和改进。此外定义了过程域，目标和实践。应用 TMMi 成熟度标准将改进测试过程，并显示出对产品质量，测试生产力和周期时间成效的积极影响。TMMi 的开发旨在支持组织评估和改进其测试过程。

TMMi 是一个过程改进的阶段型架构。它包含阶段或级别，组织可以通过它们使组织的测试过程从临时的和未管理的，进化为已管理、已定义、已测量和优化的过程。实现

每个阶段，需要确保有足够的改进，使其成为下一阶段的基础。

TMMi 内部结构有丰富的测试实践可以被系统的学习和应用来支持质量测试过程,这些过程是通过增量的步骤来进行改进的。在 TMMi 中有五个级别，规定了成熟度级别和测试过程改进的路径。每个级别都有一组过程域，组织需要实施这些过程域来达到对应的成熟度级别。图 1 显示了 TMMi 每个成熟度等级的过程域。

TMMi 的一个主要基本原则：它是一种适用于各种生命周期模型和环境的通用模型。TMMi 定义的大多数目标和实践已证明适用于顺序和迭代生命周期模型,包括敏捷。然而，在模型的最底层，根据应用的生命周期模型，提供的许多子实践和示例是（非常）不同的。请注意，在 TMMi 中，只有目标是强制性的，而实践则不是。

TMMi 可在 TMMi 基金会的网站上免费获取。该模型已被翻译成西班牙文，法文和中文。TMMi 也以出版的书籍格式提供。

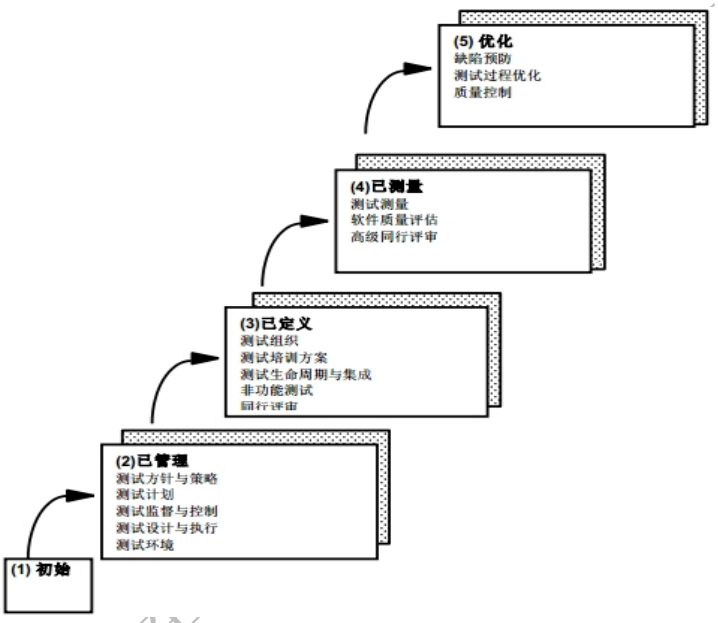


图 1：TMMi 成熟度级别和过程域

1.4 敏捷

2001 年，代表最广泛使用轻量级软件开发方法的一群人，就一套共同的价值观和原则达成了一致，成为了众所周知的的敏捷软件开发宣言或敏捷宣言。敏捷宣言包含四个价值观：

- 个体与交互 胜过 过程与工具
- 可用的软件 胜过 完备的文档
- 客户协作 胜过 合同谈判
- 响应变化 胜过 遵循计划

敏捷宣言认为，虽然右边的概念有价值，但左边的概念更有价值。敏捷本身并不是

一种方法论，但是同时开发了不同的方法，提供了将这些价值付诸实践的途径。支持敏捷方法的三个重要代表是极限编程（XP），Scrum 和 Kanban（看板）。

1.5 敏捷环境下的测试过程改进

敏捷重点关注在有自主权的团队上，他们通过频繁的回顾来开展自身的持续改进。其中一些改进可能关注在测试上。改进可能是更高级别的基于 TMMi 过程改进项目，但也可能是解决与测试有关的过程问题。需要特别强调的是，测试过程改进环境中的行为不应该被解释和/或被用作夺取敏捷团队的自我主导权。

在考察敏捷对改进上下文的影响时，需要考虑的一些基本原则是：

- 改进周期频度
- 组织因素
- 改进范围
- 改进来源
- （测试）文档级别
- 改进方法

在使用敏捷的项目中，通常会在频繁的反馈循环中进行改进，因此可以频繁地考虑测试过程改进（例如，使用 Scrum 时的每个 Sprint 的结束阶段）。由于范围往往局限于上一个周期或迭代，因此小型而频繁的改进主要集中在解决具体的项目问题上。这些改进的重点往往不在于跨项目学习和改进的制度化。

至于如何组织和管理测试过程改进，可能不太关注组织级别的测试过程小组，而是更重视项目内部团队的自我管理。这些团队通常有权改变项目内用于满足其需求的测试过程，从而使过程高度定制化。当然，也有一些组织使用每周测试站立会，推动改进提升到更高的跨项目水平。

由于焦点更多地集中在针对项目（测试）过程改进，因此更广泛的影响测试跨组织的问题可能不太受重视。例如，基本测试问题可能无法完全解决，因为它们超出了这个以项目为中心的上下文。这里的一个典型例子是用来测试某些质量属性的测试方法，例如性能和可靠性。这些问题可能会从一个迭代延伸到下一个，因为它们通常需要比项目团队更多的技能和资源。这些领域很难在没有重大投资的情况下有明显的进步。仅在项目层面解决问题也很容易导致局部优化，看不见全局。

与非敏捷生命周期模型相比，敏捷环境需要考虑的替代改进思路的范围和数量可能会大大增加。由于所有成员都在项目中进行了一些测试，这些想法可以来自任何团队成员。这更强调评估并考虑改进建议的优先级，这更多是团队的任务，而不是分配给一个测试过程改进者。当然，由于这可能需要测试过程改进者的具体测试知识，因此他们也可以根据请求担任团队的顾问。

对测试文档的要求，不要期望在使用敏捷方法的项目中能达到和使用顺序生命周期的项目相同的级别。可能有一个集成的 “测试文档”涵盖了测试方针、测试策略甚至高级别测试计划的基本要素。测试过程改进者应该避免提出需要更严格和更全面的测试文档的“改进”建议。文档只有在对它有明确无误的需求时才会创建，这是敏捷的一个主要原则。不仅测试文档需要减少详细程度，过程文档也是如此。在敏捷环境中，“不规范也合法”的策略通常是成功的。如果某些工作正常，则不需要为了 TMMi 而改变它。然而，把不规范的内容文档化并与其他人分享，这当然是有益的。“不规范也合法”策略的意义在于，如果有一个过程可行，但在某些方面它不那么规范，那么就按它是怎么执行的去传授和记录它。轻量化过程并不意味着每个可能的使用场景都需要描述出来。这些过程因而必须得到导师的指导和工作过程中的辅助，特别是在初始部署期间。因此，在保持敏捷文化，将测试过程成熟度带给敏捷组织的同时，还需要更多的培训。同样，在评估期间，收集证据的重点将转向采取更多访谈而不是研究工作产品。

在使用敏捷时，用于提出测试过程改进的方法将倾向于将重点放在评估问题根本原因的分析方法上，如因果图。这些对于解决问题的思维方式来说是非常有用的方法，这在迭代结束时非常重要。

2. TMMi 2 级已管理

2.1 过程域 2.1 测试方针与策略

测试方针与策略过程域的目的是开发和建立测试方针，以及组织范围或项目群范围的测试策略，其中测试活动是明确定义的（例如测试类型和测试象限）。为了衡量测试性能，引入了测试活动的价值以及需要改进的地方，以及测试性能指标。

2.1.1 SG1 制定测试方针

任何开展测试改进项目的组织都应该从定义测试方针开始。测试方针定义了组织的整体测试目的、目标和有关测试和测试人员的战略性意见。测试方针必须与组织的整体业务（质量）方针保持一致。测试改进应该由明确的业务目标驱动，而这些目标又应该在测试（改进）方针中记录。测试方针对在组织内的所有干系人之间获得测试及其目标的共识来说是很有必要的。这个共识可以使整个组织的测试（过程改进）活动保持一致。请注意，测试目标不应该是一个仅限于自身活动的目标，它应源于工作软件和产品质量的更高层次的目标。

上述对于实施敏捷软件开发的组织也是如此。事实上，在许多组织中，关于测试角色的变化、测试的独立性、测试自动化和专业测试人员在敏捷软件开发中的作用有许多讨论。这些条目和其他条目通常是与管理层和其他干系人进行讨论时应解决的问题，并记录在测试方针中。任何组织，包括那些实施敏捷的组织，希望启动一个测试改进项目，

需要识别和定义它的业务驱动因素和需求。为什么要开始一个改进项目？通过花时间捕捉真实的业务需求，人们可以提供一个上下文来决定将（测试）改进优先级集中在哪里，例如哪个过程域。请注意，测试方针通常是组织级的单页精简文档、网页或挂图，而不是项目级别的文档。

TMMi 中制定测试方针的特殊目标，包括其特殊实践，完全适用于应用敏捷软件开发的组织。当然，测试方针的要素也可以纳入开发方针中。TMMi 并不特别要求它成为一个单独的文档。例如，应用敏捷软件开发的组织中的开发方针可以将 Scrum 作为其管理框架，将 XP 作为使用的主要方法，并以实现敏捷价值作为主要原则。另一个可以提及的重要原则是团队中的每个人都对一切负责，产品质量也是团队责任。

但是，对于测试方针以及特别定义的测试（改进）目标，敏捷还有一个重要的考虑因素。虽然可能存在与组织中测试过程改进相关的总体目标，但这需要将单个项目及敏捷团队负责改进自己的过程进行平衡。敏捷过程改进的挑战在于既在组织级给出指导和改进架构，同时又不降低单个敏捷团队对自己过程的所有权。

2.1.2 SG2 建立测试策略

测试策略遵循测试方针，并作为项目内测试活动的起点。测试策略既在组织范围定义，也在项目范围定义。典型的测试策略基于高级别的产品风险评估，并包括将要执行的测试类型、测试象限和测试级别的描述，例如：单元、验收、回归和性能测试。仅说明在迭代内进行单元测试和验收测试是不够的。我们需要定义单元和验收测试的含义，这些通常是如何进行的以及他们的主要目标是什么。经验表明，当定义和遵循测试策略时，各种测试活动之间的交叉可能会减少，从而带来更高效的测试过程。此外，由于各种测试类型和测试级别的测试目标和测试方法保持一贯的，因此遗漏会更少，从而带来更有效的测试过程，进而提高产品质量水平。为了建立这种协调关系，强烈建议建立一个“伞型”测试策略覆盖同一个项目下的敏捷测试和非敏捷测试。但是，如果有单独的敏捷项目和非敏捷项目，通常有两种测试策略。

测试策略是敏捷环境中的关键文档。它在高层定义了敏捷团队（迭代团队）要完成的测试：执行什么样的测试类型、测试象限和测试级别，以及执行测试的方式。该文档描述了如何组织测试，例如，敏捷团队内部做哪些测试以及团队之外做哪些测试。它将定义从敏捷团队到那些在其范围之外执行的测试级别的关系，例如硬件/软件集成测试，系统集成测试或 beta 测试。测试策略确保所有参与测试的人员都了解大的测试图景。

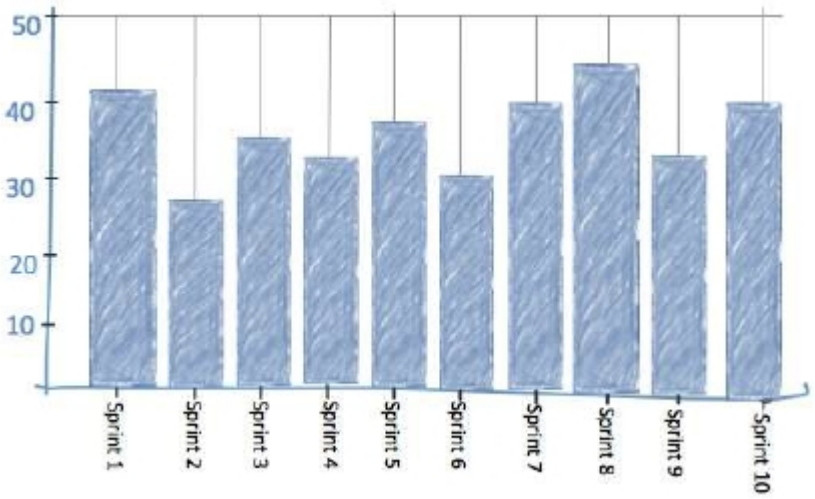
精益的测试策略文档通常是敏捷组织的一个好的解决方案，也是详细的（项目）测试计划之外的一个方法。在发布计划期间，会讨论并确认可用的组织范围或程序范围内的测试策略，或专门为项目创建衍生测试策略。确认或创建的测试策略提供了跨越所有迭代的测试框架。

TMMi 的特殊目标：建立测试策略，包括其特殊实践，完全适用于应用敏捷软件开发

的组织。请注意，例如对于测试策略文档的“分发”活动，在敏捷中可能不那么重要。作为整体团队方法的一部分，那些作为利益干系人应该已经参与了早期的讨论。然而，测试策略不是在团队层面上，而是在更高层次上。整个团队在自身层面运作，因此组织级或项目文档仍然需要分发给团队，甚至其他利益干系人，例如，如果存在混合模型，则包括在团队之外执行测试活动的人。

2.1.3 SG3 建立测试性能指标

测试方针中定义的测试改进目标需要转化为一组关键测试性能指标。测试策略和附带的性能指标提供了一个明确的方向，并提供了交流预期和已达到的测试性能水平的方法。性能指标必须向利益干系人提供测试和测试过程改进的价值。由于过程改进投资需要长期的管理层支持，因此定义衡量改进项目的收益以保持其动力至关重要。请注意，TMMi 的特定目标是定义有限数量（例如 2 或 3 个）的测试性能指标。这不是建立和实施完整的测量程序，而是定义一组核心指标，告诉您测试的价值随着时间和交付环境的变化而变化。



在敏捷中，重点将是更多基于团队和系统思维。这可能导致相应的指标扩大到团队和整体系统，而不仅仅局限于测试本身的细节。TMMi 2 级的指标主要与迭代的最终结果相关。例如包括逃逸缺陷、速度、客户满意度评级、努力/浪费、测试自动化比例等。面临的挑战是如何定义与基于团队的方法和系统思想相关的指标的适当组合，同时很好地指出基于 TMMi 的测试改进项目的绩效成果。

TMMi 中建立测试性能指标过程域的特殊目标，包括其特殊实践，是完全适用的，尽管选择和应用的性能指标可能具有更大的范围，并且比仅与测试相关的范围更广。当然后者会使性能指标的分析和说明更具挑战性。实际上，所使用的性能指标可能不是所谓的测试性能指标，而是团队性能指标或系统性能指标。在 TMMi 的上下文中，只要它具有与测试相关的元素并且正在用于评估正在进行的测试改进过程，这仍然是可以的。

2.2 过程域 2.2 测试计划

测试计划的目的是基于识别的风险和已定义的测试策略而定义一套测试途径，并为

执行和管理测试活动建立和维护有良好基础的计划。

请注意，成功的测试计划的关键在于先行思考（“活动”），而不是在于定义相关的测试计划（“文档”）。

对于敏捷生命周期，通常会出现两种计划，即发布计划和迭代计划。TMMi2 级的测试计划过程域既关注发布计划相关的测试活动，也关注迭代计划相关的测试活动。发布计划将在项目开始时预见产品的发布。发布计划需要定义产品待开发列表，并可能涉及将较大的用户故事细化为一组较小的故事。发布计划为跨越所有迭代的测试方法和测试计划提供基础。发布计划是高级别的。发布计划制订完成后，开始制订第一次迭代的迭代计划。迭代计划将贯穿单个迭代直至结束，并关注迭代产品开发列表。

请注意,测试计划过程域有许多特殊实践。TMMi 没有说明何时或如何进行这些实践。它也没有说明不能以增量方式制订计划。传统的方法是尽可能地预先固化决策，因此相关的成本和时间表也可以得到固化。这种方法的基本原理是更好地评估工作并降低范围蔓延的风险。敏捷方法通常采取的立场是，我们通过基于最新信息持续改进计划并与客户进行持续合作，从而获得更好的计划。

2.2.1 SG1 执行风险评估

完全测试是不可能的，需要经常做出选择，并且需要经常确定其优先级。这一 TMMi 的目标也适用于敏捷项目。对于敏捷项目，高级别产品风险评估应根据产品愿景文档或发布计划中的高级别用户故事进行。对于每次迭代，都应根据用户故事或该迭代的其他需求作为迭代计划工作的一部分执行更详细的产品风险评估。与采用顺序生命周期模型的传统项目中进行产品风险评估相比，敏捷项目中的产品风险评估过程将具有更轻量级的形式。风险卡片是一个轻量级产品风险评估技术的例子[Van Veenendaal]。

在发布计划制订过程中，了解发布中功能的业务代表可以对要开发的功能进行高级概述，包括测试人员在内的整个团队将协助进行风险识别和评估。

在迭代计划制订过程中，敏捷团队根据即将到来的迭代中要实现的用户故事来识别和分析产品风险。最好是敏捷团队的所有成员以及可能的其他一些利益干系人参与产品风险评估。结果是确定测试关键区域的产品风险项的优先级列表。这反过来将有助于确定适当数量的测试工作分配，以便用足够的测试覆盖每个风险，并以优化测试工作的有效性和效率的方式对这些测试进行排序。任务板上的估计任务可以根据与其相关的产品风险水平来定义不同的优先级。与较高风险相关的任务应该尽早开始并涉及更多的测试工作。与较低风险相关的任务应该稍后开始并涉及较少的测试工作。

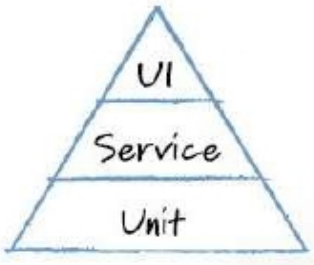
2.2.2 SG2 建立测试途径

测试途径的定义是为了缓解已识别和确定优先级的产品风险。对于特定迭代，在迭代计划过程中需要确定要测试的条目和特性。此活动也基于产品风险活动的结果。要测试的条目的优先级列表通常涉及要在此迭代中测试的用户故事。除其他外，这些特性通

常与要测试的各种软件质量特性相关。新产品风险可能在需要额外测试的迭代期间变得明显。通常会在每天的站立会议上讨论需要额外测试的新产品风险等问题。

在迭代级别确定测试途径有助于降低风险，包括，例如可以覆盖用户故事和验收标准的额外评估，与风险级别成比例的测试工作，根据风险水平和风险类型选择合适的测试技术。在版本发布级别的测试途径将处于更高的层级，应基于组织或项目级别上定义的测试策略。测试途径常常在团队/项目 wiki 上进行控制或显示。

迭代开发中一直存在的重要风险是回归风险。测试方法需要定义回归风险的管理方式。通常这将通过创建特定的回归测试集来完成，该测试集最好



是可自动化。在这种情况下自动化测试金字塔[Cohn]很有帮助。它展示了如何最大化回归测试自动化的价值，从金字塔最底层的单元测试开始，继续进行服务层级测试。用户界面测试位于最顶层。单元测试快速而可靠。服务层允许在 API 级别或服务级别测试业务逻辑，而不受用户界面（UI）的阻碍。层级越高，测试越慢，也越脆弱。

测试入口准则通常是定义测试途径的一部分（特殊实践 2.3），很可能与敏捷开发不相关。在敏捷软件开发中，测试是团队过程中内在组成部分，并且是一项几乎连续的活动。因此，不需要特定的检查单或入口准则来确定测试是否可以开始。这也适用于敏捷团队中，组件的测试从一个测试级别（例如单元测试）到另一个测试级别（例如验收测试）。

测试出口准则（特殊实践 2.4）是所谓的完成定义（DoD）的一部分。非常重要的是，完成定义有测试相关的特定准则，例如测试覆盖率和产品质量（缺陷）。迭代应该导致执行商定的一组用户故事并符合测试完成定义的（测试）准出标准。通常情况下，不符合出口准则的故事将被放入待开发列表中，并可能在下一次迭代中处理。当然，在敏捷中，从一个测试级别到另一个测试级别没有退出准则。不仅完成定义存在于迭代级别，而且它也常存在于跨越多次迭代的版本发布级别上。

特殊实践 2.5 定义暂停和恢复准则很可能与敏捷生命周期不相关。由于测试是敏捷软件开发过程中不可或缺的一部分，因此它不会被视为与其他迭代活动分离且独立的活动。如果存在阻碍性问题，可能会被视为潜在或实际对测试进展的威胁，这些问题将在每日站立会议中讨论。在这个讨论中，团队将决定需要采取什么行动来解决问题。因此，正式的暂停和恢复标准不是必要的，也不需要被定义，其它相关的问题将作为正常敏捷惯例的一部分来处理。因此，敏捷惯例可以作为这种特定实践的一种替代。

2.2.3 SG3 建立测试估算

对于敏捷团队，在迭代计划期间将对测试进行详细估算。高阶（测试）估算是在发

布计划期间进行的，也可能是在待开发列表调整期间进行的。所有估算当然都是作为团队估算完成的，其中包括完成每个故事所需的全部工作。非常重要的是在评估时确保测试活动得到了考虑。这可以通过将测试活动确定为单独的任务并随后估算每个单独的测试任务来完成，或者通过估算每个用户故事来完成，从而明确和考虑到要需要进行的测试活动。作为敏捷团队成员的测试人员将参与评估工作。计划卡片或 shirt-size 模仿估算是在敏捷软件开发中使用的典型估算技术。

下一次迭代要完成的工作通常由用户故事定义。用户故事的规模要尽可能地小些，以便可以估算。可估算是 INVEST [Wake]定义的用户故事准则之一，可适用于作为迭代的一部分。在迭代计划期间，敏捷团队通常会识别和定义与测试相关的任务。测试任务将与其他开发任务一起在任务板上获得。这些任务是进行迭代估计的基础。为即将到来的迭代定义的一组任务可用作一种工作分解结构（用于顺序项目）。

在版本发布计划中，通常会对用户故事或故事分集进行更高阶的定义，但并不会将其详细描述为特定的任务。这当然会使估算更加困难和不准确。如前所述，敏捷项目不会建立工作分解结构作为估算的基础，但是可能在版本层面受益于一个简单的图表，该图表将待开发的产品可视化，从而适当地确定工作范围。

虽然敏捷团队会以相对非正式的方式进行估算，但估算的理念应该是很清楚的（即需要考虑哪些因素）。基于基本原理的讨论可以提高估算的准确性。通常在敏捷项目中，估算侧重于规模（使用故事点）或工作量（使用理想化人日作为估算单位）。成本通常不会作为敏捷项目评估过程的一部分。

因此，除 3.2 定义测试生命周期这一特殊实践外，TMMi 特殊目标及其特殊实践完全适用。迭代和敏捷软件开发的基础之一是以小块来开展工作。因此，已经确定的任务通常足够详细，可以作为（测试）估算的基础。因此，敏捷中不需要为测试活动定义生命周期以作为估算的额外基础。

2.2.4 SG4 制定测试计划

再次强调，测试计划是关于前期思考（“活动”）而不是定义相关测试计划（“文档”）。敏捷的大多数测试计划活动（由 TMMi 特定目标定义）将在版本和迭代计划期间执行。但是，这些测试计划活动的结果通常不会记录在测试计划中，特别是对于可以在任务板上反映的迭代计划。

由于测试是作为版本和迭代计划的组成部分来执行的，所以产生的“计划”还将包含测试活动。与使用顺序生命周期开发的详细计划不同，敏捷项目中的计划更像是用户故事（待开发列表）的排序以及反映版本和迭代优先级的任务列表，例如基于期望的业务价值交付。思维导图在这里经常用作辅助技术。任务板将反映迭代优先级。用户故事的明确版本发布和迭代优先级是已定义好的，并分别为每一用户故事定义要执行的任务（包括测试任务），因而不需要单独建立明确的（测试）时间表。

在项目层面，测试人员是构建团队的一部分；预先确定了对测试或多技能人力资源的需求。随着项目的变化或规模的增长，人们可能很容易忘记为特定团队/项目初步选择个人的理由，例如具体的技能需求或与团队/项目相关的特定经验。这为写下人们的技能需求提供了一个很好的理由，为团队/项目选择原因提供相关的备份信息。一旦定义了敏捷团队，测试人员或多或少就已经确定了。在迭代计划中，如果需要，讨论确定在迭代中执行测试所需要的（附加的）资源和技能，例如非功能性测试，以确保团队具有足够的测试资源，知识和技能来执行所需的测试。

Scrum 主管应该确保产品负责人为测试提供必须的输入，例如回答问题并就用户故事进行讨论。产品负责人应该到位，这也需要预先组织。

最初的项目风险评估应该是发布和迭代计划的一部分。迭代过程中进一步识别（和管理）项目风险是日常站立会议的一部分，通常通过障碍日志进行记录。在障碍日志中记录测试问题也很重要。障碍应该在日常的站立会议上讨论，直到问题解决。

尽管没有制定具体的详细测试计划并形成文档，但特殊实践 4.5 建立测试计划在敏捷上下文中仍然有用。然而，在测试计划过程中进行的讨论的结果可能会以轻量级的形式被捕获，例如一个思维导图。

2.2.5 SG5 获得对测试计划的承诺

在敏捷内部，开发和建立测试途径和测试计划的过程都是基于团队的操作，可能由（作为团队成员之一）测试专业人员领导。产品质量是团队的责任。因此，如果团队遵循正确的过程，版本发布计划和迭代计划的制订已经隐含了对（测试）途径和（测试）计划的承诺，因为这是团队合作的结果。这与在传统环境中工作的方式当然有巨大的差异，通常情况下，测试人员准备测试计划，然后需要获得明确的承诺。

在敏捷项目中，敏捷团队（包括产品负责人）必须理解并同意产品风险的优先级列表以及要执行的测试以缓解产品风险。理解和承诺可以在版本发布计划和迭代计划之后，通过简短的展示介绍给团队，包括解释产品风险，测试方法及其基本原理。

在评估过程中（参见 SG 3 建立测试估算），已经完成工作量的估算。团队的（测试）资源通过敏捷团队的设置来解决。估计和选择要开发的用户故事以可用资源为起点（约束）。因此，再次协调工作和资源级别并不是一项有意义的活动。特殊实践 5.2 协调工作和资源级别因此与敏捷上下文不相关的实践。每日站立会议将用于解决迭代期间的任

何资源问题，并立即（重新）分配适当的资源，或者减少可交付物并协调到未来的迭代中，或调整版本发布计划。

2.3 过程域 2.3 测试监督与控制

测试监督与控制的目的是提供对测试进度和产品质量的理解，以便在测试进度显著偏离计划和产品质量显著偏离预期时采取适当的纠正措施。

遵循敏捷宣言及其配套原则，与传统项目相比，敏捷项目中的一些监督和控制有着根本性的不同。虽然监督控制是敏捷项目的重要组成部分，但并不意味着坚持严格的计划是目标，事实上，相反的是，宣言和原则都提到欢迎改变。测试监控可以在敏捷的背景下解释为提供最佳实践，以不断调整计划以保持其实时最新，这正是敏捷方法所推荐的。

从测试监督和控制的角度来看，这意味着我们不是由计划驱动的，而是不断审视我们的测试进度和结果，并适当调整我们的计划和方法 - 新产品风险可能会变得明显。测试计划是一个有生命的实体，需要不断地审查和更新，因为有新的依据信息或得到反馈。敏捷项目还需要牢记“更大的图景”，不仅在迭代级别，而且在更高的（发布）级别进行监控。

需要注意的是，由于测试是一个完全集成到敏捷团队整个过程中的过程，因此测试监督与控制也是敏捷团队整体监督和控制机制的一个组成部分。因此，测试人员不像传统项目那样向测试经理报告，而是向团队报告。

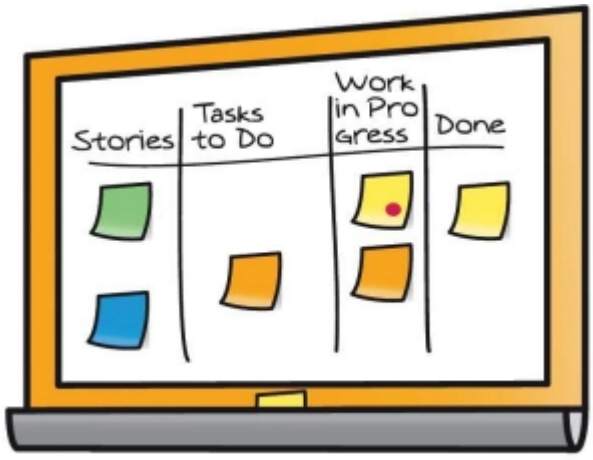
由于 TMMi2 级的测试计划过程域侧重于发布和迭代计划，因此同样适用于测试监督和控制。可以预计测试监控将在计划和发布两个级别执行。

2.3.1 SG1 根据计划监控测试进度

敏捷团队中的测试人员利用各种方法来监控和记录测试进度，例如，敏捷任务板上的测试任务和故事的进展，以及燃尽图。然后，可以使用 wiki 仪表板和仪表板式电子邮件等媒介，以及在站立式会议中以口头形式将这些信息传达给团队的其他成员。团队可以使用燃尽图来跟踪整个版本和每次迭代中的进度。燃尽图表通常会显示预期的速度和要实施的待实现功能列表（团队绩效）的进展情况。有时，当测试环境资源稀缺且至关重要时（例如，对于非功能测试），将使用特定的燃尽图来针对迭代计划期间约定的测试环境资源使用计划进行监控。另一种常见做法是通过任务板跟踪测试环境问题，例如使用在故事卡上标记为“环境原因测试被阻断”的标签，或者在板上创建一个单独的列，这个面板上的环境问题都等着直至被解决。这是关于创建一个阻断进程的测试环境的影响的可见方法。

为了及时提供整个团队在当前状态（包括测试状态）的详细的可视化展示，团队可以使用敏捷任务板。在任务板上记录故事卡片，开发任务，测试任务以及在迭代计划期

间创建的其他任务，通常使用颜色协调卡片来确定任务类型。在迭代期间，通过将看板上的这些任务移动到诸如待执行，正在执行和已完成的列中来管理进度。敏捷团队可以使用工具在 Agile 任务板上维护自己的故事卡片，这可以实现仪表板和自动化的状态概览。然而，有些团队不会为个别活动创建具体任务，但可能只是使用故事卡并在此卡中注释以表示测试活动，并参考敏捷工具或 wiki 文档中测试活动被文档化的地方。任务板上的测试任务通常与为用户故事定义的接受准则相关。当测试任务的自动化脚本测试，手工脚本测试和探索性测试达到通过状态，测试任务会移至任务板的已完成列。



已完成定义是衡量进度的退出标准。完成定义还应该与测试活动相关联，并展示所有的准则，这些准则在用户故事测试可被称为“已完成”之前需要被满足。请注意，与测试任务相关的测试准则仅构成团队同意完成的一部分，即已完成定义。完成准则的定义通常应用于多个级别，例如迭代级别和发布级别。整个团队定期检查任务板的状态，经常在每日站立会议期间确保任务以可接受的速度在整个板上移动。如果任何任务（包括测试任务）没有移动或移动速度过慢，则会触发基于团队的讨论，分析可能阻碍这些任务进度的问题。每日站立会议包括敏捷团队的所有成员，包括测试人员。在这个会议上，他们将目前的现状和实际进展传达给团队的其他成员。任何可能阻断测试进程的问题都会在日常的站立会议中进行沟通，因此都会意识到这些问题并可以采取相应的措施。这样风险管理也被整合到这些日常会议中。任何项目风险，包括测试的项目风险，例如测试环境缺乏可用性，都可以在日常站立会议期间进行沟通和解决。（请注意，监控产品风险是监控产品质量的一部分，因此将在下文中针对特定目标 SP 2 根据计划和期望监控产品质量进行讨论。）任务管理的日常会议以及与任务变更相关的敏捷团队实践都已被很好地证明，在 TMMi 测试监督与控制过程域中符合特殊实践的意图。

敏捷中的里程碑审查是在迭代完成之后进行的。测试成果（例如，针对完成定义的检查）将成为迭代评审的一部分。向利益干系人进行 DEMO 演示并一起组织讨论交付产品的商业价值和质量。

在制订迭代计划、迭代评审（DEMO 演示）和回顾中，产品负责人代表利益干系人。产品负责人通过讨论产品待开发列表参与其中，并将在整个迭代过程中为用户故事的阐述和测试设计提供反馈和输入。其他利益干系人在迭代结束的评审期间参与。因为产品

负责人代表利益干系人深入了敏捷开发的工作，因此不需要对利益干系人的参与进行具体监控。请注意，产品负责人积极参与敏捷项目是成功的一个关键因素，但在实践中有时很难实现。如果出现后者这种情况，那么将在日常站立会议中进行报告和讨论，并作为项目风险进行管理（见上文），因为这会影响整个团队的有效性和效率。

2.3.2 SG2 根据计划和预期监控产品质量

对于产品质量监控，大体上使用与进度监控相同的机制（见上面的 SG1）。在敏捷方面，对于产品风险监控，重点在于定期会议上审查产品风险清单，而不是审查任何详细的风险记录。会议上将会讨论新识别的产品风险或产品风险变化，例如作为探索式测试的结果，讨论协商并要求测试执行达成一致。各种产品风险的状态通常通过仪表板上的图表来显示。

最佳实践是，只有在与系统成功集成并测试之后，一个特性才会被视为已完成。在某些情况下，周期性地进行强化或稳定迭代，以解决任何滞留的缺陷和其他形式的技术债。敏捷团队使用基于缺陷的度量以监控和改进产品质量，类似于传统开发方法中捕获的测试通过/失败率，缺陷发现率，发现和修复的缺陷等。迭代期间发现和解决的缺陷数量以及可能成为下一次迭代待开发列表的一部分的未解决缺陷的数量应在每日站立会议期间进行监控。为了监控和改进整体产品质量，许多敏捷团队还使用客户满意度调查来获得有关产品是否符合客户期望的反馈。

测试出口准则，例如测试覆盖率和产品质量（缺陷），是已完成定义（DoD）的一部分。一般通过任务看板来监督对达成一致的准出标准的遵守情况，即只有符合其完成定义准则的用户故事才能将其认定为“已完成”。

每日站立会议是用于几乎连续执行产品质量评审的机制。敏捷中的里程碑产品质量评估是在每次迭代完成后进行的。向利益干系人进行 DEMO 演示，一起组织讨论交付产品的商业价值和质量。产品质量根据定义的完成准则进行验证和确认。

与过程域“测试计划”一致，监控入口，暂停和恢复准则的具体做法在敏捷上下文中很可能不适用。有关更多信息，请参阅建立测试计划过程域的特定目标 2 测试方法，其中解释了为什么这些准则通常不适用于敏捷环境。

2.3.3 SG3 管理纠正措施直至关闭

当出现例如燃尽图中的预期偏差和/或敏捷任务板上（测试）任务和故事的进展不足时，敏捷团队将很快就会注意到问题。这些问题和其他问题（例如，阻断测试进展的问题）经常在日常站立会议期间进行沟通（见上文），因此整个团队都意识到了这些问题。这会触发基于团队的讨论，来分析这些问题。结果可能是基线需要更新（例如，从迭代待开发列表中移除一个或多个用户故事），完成定义可能过于严格或者应该对工作方式进行调整。团队将决定纠正措施并与产品负责人一起执行。在迭代待开发列表改变的情况下，产品负责人向团队提交更新的迭代待开发列表，并且将本次迭代待开发列表

中移除的用户故事带到下一次迭代并在迭代计划期间讨论。

在敏捷项目中管理纠正措施主要是自组织团队的基本责任。团队可以定义和实施适当的纠正措施，或将任何问题升级为 Scrum 主管的关注的“障碍”。Scrum 主管通常有责任支持团队管理问题关闭。讨论和管理纠正措施的典型事件是日常站立会议和回顾会议。已经同意的纠正措施可以通过产品待办事项或（在迭代中）通过任务板管理为“任务”或“产品待开发列表”。

2.4 过程域 2.4 测试设计和执行

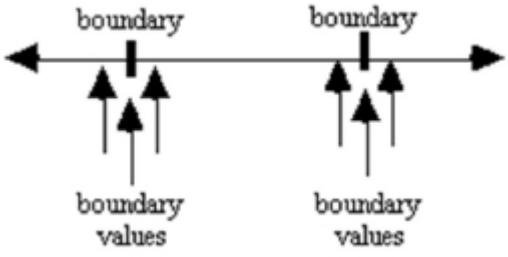
测试设计和执行的目的是，通过建立测试设计规格，使用测试设计技术，实施结构化测试执行过程，以及管理测试事件直至关闭，来提高在测试设计与执行期间测试过程的能力。

虽然敏捷方式和传统连续方式的项目的基本目标（“消解风险并测试软件”）是相同的，但测试方法通常是差异很大的。在敏捷方面，灵活性和能够对变化做出反应是重要的起点。此外，测试分析和设计，测试实施和测试执行不是顺序的测试阶段，而是并行执行，重叠和迭代执行。测试文档的详细程度是另一个关键区别。尽管基于规范的测试设计技术仍然可以适用并被使用，但通常在敏捷项目中使用更多的是基于经验和缺陷的技术。随着对单元和集成测试的重视程度的加强，诸如语句和分支（覆盖）测试等白盒技术也更广泛地被应用。最后一个主要区别是回归风险的水平，这要求在各种测试级别进行更多的回归测试。理想的回归测试是高度自动化的。尽管有很多不同之处，但最终在测试设计和执行过程域的上下文中，始终是创建测试，消解产品风险，运行测试和发现缺陷。

2.4.1 SG1 使用测试设计技术进行测试分析和设计

在敏捷方式中，测试分析和设计以及测试执行是相互支持的活动，通常在整个迭代过程中并行运行。在连续的生命周期项目中，测试分析通过测试人员检查测试依据（例如需求）并在测试依据被创建的评估可测试性来进行。在敏捷项目中，测试人员是协作创建和优化用户故事的团队成员之一。频繁的非正式评审是在开发需求的同时进行的，包括形成每个用户故事的验收准则。这些准则由业务代表，开发人员和测试人员共同定义的。通常，测试人员基于其独特视角，通过识别缺失的细节和保证可测性使用户故事更为完善。测试人员可以以开发式问题向业务代表询问用户故事及其接受的标准，并提出测试用户故事的方法，这是测试人员的价值。因此，测试分析不是一个显式的独立活动，而是测试人员在协作用户故事开发中扮演其角色的一部分。

根据对用户故事的分析，标识测试条件[1]。从测试的角度来看，通过分析测试依据来确定测试什么- 这些是测试条件.测试条



件（有时称为测试情况）基本上是对需要测试/ 覆盖的“ 事物”的标识[Black，Van Veenendaal]。只要规定的验收准则足够详细和清晰，就可以很好地承担传统测试条件的作用。测试条件随后被转化为测试，这些测试需要覆盖所定义和被认可的验收准则。除针对用户故事的测试分析外，在更高层次上执行测试分析通常也是有益的。例如，分析特征或史诗故事或故事集合以识别比用户故事级别更抽象的测试条件，并跨越多个用户故事。基于规范的测试设计技术通常有助于根据用户故事和验收准则推导出测试条件。但是，在敏捷中，这些测试设计技术的使用通常不那么明显，因为测试人员根据他们的经验掌握了技术并能够在上下文中灵活地使用它们。测试条件将以轻量级格式记录，而在传统的工作方式中，它们被记录为测试设计规范文档的一部分。有时候，尤其是在探索性测试被广泛使用的项目中，测试条件只是通过头脑风暴来识别，并被记录为测试想法（成为测试章程的一部分）。定义测试条件也是建立水平可追溯性（见下文）和管理（自动化）回归测试集的覆盖范围的基础，从而开发自动化测试以涵盖特定测试条件。

随着测试先行在敏捷中的应用，覆盖测试条件集合的测试将在代码开发之前或者至少与代码开发同步进行（可能是自动化的）。对于自动化单元测试，可以考虑像测试驱动开发这样的方法。对于更高的测试级别，行为驱动开发（BDD）和验收测试驱动开发（ATDD）是流行的敏捷开发方法，也支持测试。BDD 和 ATTD 都与自动化测试高度相关。

对于大多数手动测试而言，随着团队的测试执行工作进展，测试将被标识和细化。测试通常没有像传统项目那样详细的文档化记录，而是以探索性测试中测试想法的形式记录。对于复杂和关键领域，为了覆盖风险，使用传统的正式测试设计技术来进行测试设计和测试用例开发可能是最佳的方法。然而，与传统的顺序生命周期环境中的测试相比，采用这种正式方法的文档数量也会受到限制。测试的优先级依照其所覆盖的用户故事的优先级。用户故事的优先级取决于商业价值；最高商业价值代表用户故事的最高优先级。建立测试来覆盖功能性和非功能风险是很重要的，但在敏捷方式中，还要特别注意涵盖回归风险。

为支持测试条件和执行测试，所需的特定测试数据要被识别出来。但是，在敏捷中与传统环境相比，所需的测试数据通常不会首先被指定为测试规范文档的一部分。相反，如果必要的工具和/或功能可用，则立即创建测试数据以便立即开始执行手动测试。但是，对于自动化测试，通常需要预先指定数据。

需要建立和维护需求，测试条件和测试之间的可追溯性。团队需要明确他们已经在测试中涵盖了各种用户故事和验收准则。尽管可能不需要正式的需求管理工具，但团队确实需要为每个需求分配标识符来组织和管理需求，。通过这些标识符可确保测试是完备的，达到了约定的（测试完成和验收）准则。在许多敏捷组织中，用户故事是制定一套相关验收准则和对应测试的基础。一旦这些测试建立起来，测试本身通常会成为详细的已商定的需求。使用这种方法可能足以实现 TMMi 中与横向可追溯性相关的需求管理

的具体实践的意图。

2.4.2 SG2 执行测试实施

测试实施是将所有需要启动测试执行的工作事项到位。通常，支持测试执行的测试文档（例如测试规程）的开发被最小化。但是，自动化（回归）测试脚本应是优先考虑并被开发。回归测试准备也应尽快启动，并与其他测试活动并行进行。

在敏捷环境中，详细的测试规程并不常见。在一个知识完备的团队中工作时，测试很可能会被记录为更高层次的抽象形式。这对于那些执行测试的人来说已经足够了，因为他们具备所期望的执行测试所需的领域和技术知识。那些执行测试的人员在团队内部工作，这样他们就能更好地理解编码的内容和编码方式，以及功能如何适用于各种使用方式。由于迭代内和迭代之间的变化程度通常很高，开发详细的测试规程也会带来很多维护工作。通常，更多的测试是以探索性测试方式来执行。在探索性测试中，没有制定详细的测试规程，而是使用单页测试章程来描述测试思路和指导测试人员开展工作。当然，执行测试所需的特定测试数据需要事先创建。

许多敏捷团队使用自动化测试。当前典型的方法是测试驱动开发（TDD），行为驱动开发（BDD）和验收测试驱动开发（ATDD）。使用这些方法中的一种或多种，创建自动化测试脚本将作为测试实施活动的一部分。

持续集成是敏捷项目的关键实践。测试是团队中完全集成的活动且并行执行，而不是一个独立的活动或单独的阶段。因此特殊实践 2.3-指定预测试规程在敏捷团队中变得没有必要。事实上，特别规定和执行正式预测试是没有意义的。但是，如果某些测试（例如系统集成测试或硬件/软件集成测试）在敏捷团队的范围之外执行，他们可能会指定一个准入测试来评估敏捷团队的输出质量，并确定已交付的产品是否满足测试。

在敏捷环境中，不会创建具体的测试执行日程表，因此特殊实践 2.4 制定测试执行日程表很可能不适用。测试在迭代过程中执行的顺序具有高度的灵活性，尽管这将依据各种用户故事的优先级。要在迭代计划中确定要执行的各种测试，并将通过敏捷任务看板作为测试任务进行管理。

2.4.3 SG3 进行测试执行

在迭代开发中的软件项目当然需要测试。继敏捷宣言之后，与这一过程域的其他目标一样，测试执行所需要的文档比传统的顺序项目，密度会低很多。通常不会产生详细的测试规程和测试日志。测试，尤其是回归测试，应尽可能自动地执行，尽可能保证易于执行。

在一个敏捷项目中，软件至少每天交付，并且测试是敏捷开发过程的一个内在组成部分，通过相互合作工作以交付高质量的产品。没有专门的独立开发和测试团队。因此，特殊实践中定义的如 3.1 执行预测试这类入口测试可能不适用于纯粹的敏捷上下文。然而，当一些测试在敏捷团队之外执行时（经常发生的情况），通常由敏捷团队之外进行

测试的测试团队负责执行入口测试。请记住，在敏捷团队开始特定用户故事的验收测试之前，有些活动有望完成。这些活动通常是“已完成定义”的一部分，例如，商定的单元测试已经完成并通过。

测试实施按照迭代计划中定义的优先级进行。部分测试可以使用文档化的测试规程来执行，但通常情况下，更多的测试将使用探索性和基于会话的测试作为其框架来执行。在探索性测试中，以预先准备的测试章程为指导，同步进行测试设计和测试执行。测试章程提供测试目标和需要覆盖的测试条件，并指定测试时间框架。在探索性测试期间，最近的测试结果用于指导下一次测试。

随着迭代开发，对组织和结构化回归测试的需求也越来越高。尽管有时是手工完成的，但最好使用工具来完成自动回归测试。回归测试尤其是单元测试和集成测试通常是持续集成过程的一部分。持续集成是敏捷软件开发中的重要实践。它基本上是一个自动化的构建和测试过程，至少每天进行一次，以便尽早检查和尽早发现问题。持续集成允许定期运行自动回归测试，并将代码质量和达到的代码覆盖情况向团队快速提供反馈。

测试过程中发生的事件可能会被记录并报告。在敏捷项目中通常会讨论是否应该记录所有发生的事件。特别是在团队位于同一地点的情况下，测试人员需要与开发人员进行交流，事件/缺陷可以立即被修复且可以纳入下一次构建时，可能不需要记录，只需要修复即可。在敏捷中，通常不是所有发现的事件都会被记录并相应地进行管理。如前所述，很多只是由团队成员（可能是测试人员）非正式提出，并由引入缺陷的开发人员立即修复。一些数据可能丢失，但同时过程的开销减少。有些团队只记录会跨越迭代的事件，如果今天无法修复，有些团队会记录这些事件，有些团队只记录高级别事件。如果未记录所有发现的事件，则必须有准则来确定应记录哪些事件以及不记录哪些事件。记住，目的是为了纠正缺陷，而不是记录事件。此类准则的一个例子是：“如果在下次每日站立会议和每日构建之前能够解决缺陷，则不需要记录。任何无法在一天内关闭或影响团队外干系人的缺陷都应该记录下来。

有时，事件会记录在敏捷任务板上，既可以作为现有任务的标签，也可以作为单独的任务，其阻断用户故事及任务被完成情况进行可视化。在任务板上使用便利贴可以接受，以便管理您的缺陷，而这通常是同处一地的团队选择的方法。敏捷团队随后将处理这些事件并对其进行管理直至关闭（参见下文中该过程域的特殊目标 SG 4）。有些人选择在工具中记录和文档化发现的事件，例如缺陷管理和跟踪工具。应该尽可能地被精益化使用这样的工具，如果做为测试事件报告的要素，但不带来或只带来很少的附加价值，那就不要强制提交。

在测试执行过程中记录数据，以确定测试的项目是否符合已定义的接受准则，且确实可以标记为“已完成”，这也被认为是敏捷团队的一个良好实践。在应用基于经验的技术（如探索性测试）时也是如此。对文档可能有用的信息包括这些的例子，如测试覆

盖率（覆盖的程度，还有待测试的程度），测试期间的观察结果。例如被测系统和用户故事似乎已稳定，风险列表（ 哪些风险已被涵盖，哪些风险已被列入最重要的风险），发现的缺陷和其他情况以及仍处于打开状态的问题等，都是测试期间需要观察的。

记录的信息应该以一种方式捕获和/或总结成某种形式的状态管理工具（例如，测试管理工具，任务管理工具，任务板），以便团队和干系人更容易了解所有进行的测试的当前状态。当然，团队不仅仅需要测试任务的状态，而且需要知道用户故事的整体状态。关注所有相关活动是否都完整，而不仅仅是测试，否则从故事的角度来看，都是不完整的。

2.4.4 SG4 管理测试事件直至关闭

如上所述，在敏捷环境中，通常不会记录所有发生的事件。这个特定的目标仅适用于那些需要记录且需要管理直至关闭的事件。原则上，敏捷上下文中管理事件很简单。在事件记录在敏捷任务板上后，无论是作为现有任务的标签还是作为单独的任务，它都被视为阻止故事发生的缺陷，成为未完成而需要完成的任务。标准的敏捷开发方式将其与阻碍项目进展的任何其他障碍同样对待。它将在站立式会议中讨论，并由团队解决，并可能在任何时候从任务板中消失。

如果决定将发生的事件推迟到另一次迭代中进行处理，它们只会成为产品的另一个期望选项（或更改）。将它们添加到待办事项列表中并相应地设置优先级。当优先级设置得足够高时，他们将在下一次迭代中被团队接收。请注意，优先级通常根据业务定义，但有时需要优先处理与所谓技术债务相关的事件。一些团队有专门的迭代来清理技术债务相关的事故/缺陷。然而，这不是标准的敏捷建议做法，因为债务应定期偿还。

作为如何在敏捷环境中处理事件的示例，首先想象一下团队正处于一个迭代过程的中段，在处理其中一个用户故事时，团队确定无法将其“已完成”，因为其中一个验收准则未得到满足，可能是由于在开发过程中引入了缺陷。敏捷团队倾向于在当前迭代中尽快修复这些缺陷，否则他们将继续进行未完成的工作（相当于精益库存）。这些事件将通过任务板进行可视化并得到管理。

其次，迭代评审时，假设所有故事依据完成定义准则都是完成的。但是，如果在评审会议期间，在演示故事时，有些事情的行为与客户所希望的方式不同。考虑到，在迭代开始之前，这些非预期的行为并未与业务一起确实地讨论。在这种情况下，所生成的软件功能已得发挥作用，可能需要的工作是将事件转化为另一个故事，并将其添加到产品待办事项列表中。

在日常（站立）会议和回顾会议中，监控迭代过程中发现和解决的缺陷数量以及未解决的缺陷数量，以及监控可能成为下一次迭代的待办事项中的一部分的缺陷数量，这是一个很好的做法。

在传统的顺序生命周期环境中，通常会有一个特定的配置控制委员会（CCB）来处

理，审查和决定事件的处置。敏捷中的这一角色由授权团队接管。当然，业务代表（例如产品负责人）在决定事件的处置方面发挥着重要的作用。处理事件是一种开销，它正在减少在其他事情的关注和努力。对于事件，我们越试图以正式过程来列示、考查、和安排计划进行处理，开销越大。像这样的复杂事件管理过程就是通常意义上的浪费。最好由敏捷团队归纳出一些基本原则，然后采取相应的行动进行处理和管理。

2.5 过程域 2.5 测试环境

测试环境的目的是，建立和维护一个适当的环境，包括测试数据，使我们以已管理和已重复的方式执行测试成为可能。在测试环境过程域中，建立并维护一个适当的测试环境，其中包括通用测试数据，使我们以已管理和已重复的方式执行测试成为可能。当然，在一个使用敏捷生命周期的软件开发项目中，适当的测试环境是不可或缺的。因为迭代周期短，测试环境需要非常稳定并保持可用。测试环境中的问题总是会立即影响迭代的进度和结果。因此，正确管理测试环境和测试数据的配置和变更是至关重要的。

测试环境在很多方面取决于待测系统中的其他方面，但通常测试环境的规格、实施和管理与敏捷软件开发项目并行进行。测试环境的实施通常需要很多时间，并且可能非常复杂，使得在迭代的有限时间周期内[Van der Aalst 和 Davis]几乎不可能完成这项任务。它还涉及组成敏捷团队的不同干系人和工程师。敏捷方法已经被开发来支持软件开发，而不是为了测试环境或基础设施的开发和管理。虽然在实施敏捷时可能会有一些变化，但测试环境或基础架构开发和管理的执行方式仍然与传统环境中的方式基本相同。

2.5.1 SG1 开发测试环境需求

在项目早期执行测试环境需求规格。评审需求规格，以确保其正确性、适用性、可行性和现实操作环境的准确表现。提早定义需求规格的好处是有更多时间来获得和/或开发需要的测试环境和组件，例如模拟器、桩或驱动器等。

在一些敏捷开发项目中，执行所谓的初始迭代（迭代 0），其中就包含了导出和规格化测试环境需求。

2.5.2 SG2 执行测试环境的实施

有时，除了例如培训，高阶产品风险评估和建立已完成定义之外的其他活动，将在最初的迭代中进行（部分）实施。通过这种方式，我们的目标是实现第一次软件开发迭代就已经具有（部分）正常工作的测试环境。技术环境操作行为或问题可能是产品待开发列表的一部分。

与顺序生命周期中的测试环境过程域相同，定义测试环境（包括通用测试数据）可以从使用一个的完整计划顺序生命周期开始，但是通常更好的做法是尽快从实现开始，

并且在第一次迭代开始前有可用的初始版本的测试环境。

事实上，在测试环境中很难通用。在某些领域，桩和驱动程序正在迅速被服务虚拟化取代，通常可以执行更快的测试环境实施。在一些敏捷项目中，测试环境的配置高度自动化。测试环境可以在几分钟或几小时内完成，而不是几天。现在还有更多的使用虚拟机器、虚拟服务器和虚拟服务。此外，云可以协助提供所需的环境，并再次提升访问速度。

2.5.3 SG3 管理和控制测试环境

测试环境的管理和控制将贯穿整个敏捷项目。有时候敏捷团队需要自己建立和维护测试环境，但通常这是由敏捷团队之外的独立团队完成。当这些活动由敏捷团队自己完成时，这当然意味着团队有足够的技术知识能够执行这些任务。这也意味着这些任务成为敏捷过程的一部分，例如，他们需要在计划会议中处理，放到任务板上，并可能在日常站立会议中讨论，以防出现任何问题。

作为测试环境过程域的主要结论，特殊目标和特殊实践仍然适用，本质上不会改变，改变的是他们在生命周期中的时机。

3. TMMi 3 级已定义

3.1 过程域 3.1 测试组织

测试组织过程域的目的是确定和组建一组技能高超的人负责测试工作。此外，这个组织还根据他们对组织当前测试过程和测试过程资产的优缺点的深入了解对组织的测试过程改进以及测试过程资产进行管理。

3.1.1 SG1 建立测试组织

测试组织是一个经常被误解的过程域。许多人认为 TMMi 需要一个独立的测试组甚至独立的测试部门来进行独立测试。尽管这是一种可能性，但也有其他符合 TMMi 要求的组织模型。除了准备和执行测试的独立测试组织之外，测试能力中心也是一种选择。所谓的测试能力中心的典型任务和责任是建立、管理和改进测试过程，并为项目提供诸如测试经验、知识和技能的资源。测试能力中心通常拥有测试方法和过程的所有权。请注意，日常工作中，测试人员通常是敏捷团队的一员，但最重要的是测试人员属于测试组织，例如测试能力中心或测试协会。

测试组织需要在与测试和质量问题相关的领域具备领导力。这样团队的工作人员被称为测试专家。测试能力中心方式与敏捷方式很匹配，并确保测试作为一个特定技术领域得到保持，认真对待。当然还有敏捷的思想领袖不赞成保留特殊的技术领域。虽然某些人是这样的建议，但是 TMMi 要求保持测试作为一门技术领域。毋庸置疑，只有从商业的角度看有意义才能做到这一点，因此当没有商业价值时，TMMi 可能不是首选的改进

途径。

在敏捷中，测试组织可以采用测试协会的形式。测试协会通常是一个非正式的，测试人员自我管理的组织，他们也是不同的敏捷团队的成员。测试协会的重要活动可以是知识共享，小组学习，趋势观察等。协会的会员资格是自愿的。在实践中，一个测试协会经常以双周组织碰面活动。可以为测试协会指派更正式的任务，如建立测试职业发展路径，组织培训，确定、规划和实施测试过程改进。确保测试协会基本保持非正式和自我管理非常重要。制约测试协会成功的一个重要因素是其成员投入时间执行测试协会的各项活动。

独立的测试人员往往能更有效地发现缺陷，平衡独立与敏捷是一个挑战。一些敏捷团队中保留专门的独立测试团队，并在每个迭代冲刺的最后几天按需分配测试人员。这可以保持独立性，这些测试人员可以对软件提供客观，公正的评估。然而，由于时间压力，对产品新功能缺乏了解以及与业务干系人和开发人员之间的关系问题，这种方法常常导致出现问题。另一种选择是建立一个专门、独立的测试机构，在项目开始阶段将测试人员长期分配给敏捷团队，让他们保持独立性，并同时获得对产品的良好理解以及与其他团队成员有较强的联系。此外，独立测试机构可以让敏捷团队以外的专业测试人员开展长期和/或与迭代无关的活动，例如开发自动化测试工具，执行非功能测试，创建和支持测试环境以及数据，并执行可能不适合放进迭代冲刺中的测试级别（例如，系统集成测试）。请注意，在敏捷团队中也可以进行独立测试。只要资产或代码的创建者以外的其他人也测试产品，那么这仍然是独立的。因此，开发人员可以通过代码审查和/或单元测试来测试彼此的代码，或者使用例如不同的角色从用户角度对测试做出贡献的业务分析师。最后，但同样重要的是，独立性可以通过组织和责任结构进行组织，但首要的是有正确的批判性思维。

3.1.2 SG2 为测试专家建立测试职能

测试被认为是一种有价值的职业和技术领域。敏捷团队需要具备测试知识和技能。测试专家将提供这些,并在执行测试活动时指导其他团队成员。测试职能和测试职业发展路径被定义，并由测试培训计划提供支持。典型敏捷测试人员所需的知识和技能与传统测试组织不同。测试人员当然仍然需要“传统”测试知识和技能，此外还需要测试自动化的知识和技能，并能够在测试之外执行任务，例如脚本或需求工程。由于敏捷测试人员在团队中工作，提高软技能与技术技能同等重要。敏捷测试人员是一种所谓的 T 形测试人员，这应该反映在测试职能描述中。在敏捷测试环境中，测试组织由具备良好技能和渴望成为优秀测试专家的测试人员组成。他们被分配到一个特定的测试职能。在

敏捷环境下，由于整个团队都在进行测试活动，测试人员能在非测试职能中详细阐述需要考虑的测试活动，角色和所需的测试知识和技能，这是非常重要的。

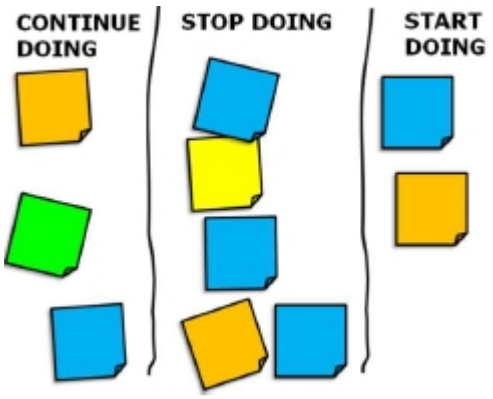
3.1.3 SG3 建立测试职业路径

测试职业发展路径的建立使测试专业人员能够提高他们的知识，技能，地位和回报，从而使他们能够提升自己的职业生涯。根据定义的测试职业发展路径，为测试组织的每个成员开发和维护个人测试职业发展计划。这个特殊目标及其特殊实践也完全适用于敏捷环境。敏捷强调人的方面，这显然是一个重要的特殊目标。敏捷团队需要（每位）测试人员了解他们的具有挑战性的工作，可以指导其他团队成员，并且富有激情。这其中很多都是通过测试职业路径解决的。

然而，与遵循顺序生命周期的组织相比，敏捷组织中的测试职业发展路径的实际定义通常会有所不同。在传统组织中，测试职业路径通常关注纵向增长（从测试人员到测试经理），敏捷组织中的测试职业路径倾向于关注横向增长（从初级测试人员到高级测试人员，或许还有测试教练）。通常不需要各种层级职能和角色，例如测试管理和更高阶的管理需求随着产品负责人或 Scrum 主管的角色发展。请注意，许多传统的测试管理任务仍然相关，但这些任务由其他人接管（例如 Scrum master，敏捷团队）。

3.1.4 SG4 确定，计划和实施测试过程改进

这个特殊目标所描述的改进过程及其具体实践主要由敏捷团队执行的回顾会议覆盖。迭代回顾对于敏捷团队来说是一个机会，可以检查自己并定义在下一次迭代期间要实施的改进措施。回顾一般发生在迭代评审之后和下一次迭代计划之前。



在回顾过程中，团队讨论迭代中哪些进展顺利，哪些可以改进，以及在下一次迭代中将致力于改进的内容。在每次回顾中，敏捷团队确定并计划提高产品质量的方法，例如通过改进工作过程或优化完成定义。回顾会议可以产生测试相关的改进决策，重点关注测试的有效性，测试效率和测试自身的质量。回顾会议还可以解决应用程序，用户故事，功能或系统界面的可测试性问题。缺陷的根本原因分析可以推动测试和开发的改进。测试人员在回顾会议中发挥重要作用。他们是团队的一部分，并带来独特的视角。所有团队成员，测试人员和非测试人员都可以提供有关测试和非测试活动的信息。

到回顾结束时，敏捷团队应该已经标识（测试）的改进，它将在下一次迭代中实施。在下一次迭代中实现这些改进是适应敏捷团队本身的进化。团队通过投票确定下一次迭代中最重要的改进，因此（测试）改进待处理时间可能会超过一次迭代，但最终会实现。团队只执行在下一次迭代中可实施的改进，这一点很重要，因此获得最优解可能需要两

到三次迭代。测试人员应该对测试改进负责，以确保他们在团队中得到实施。回顾是一个重要的机制，可以让团队在项目的整个生命周期中不断发展和改进。

由于范围往往局限于上一个周期或迭代，因此小型但频繁的改进主要集中在解决具体的项目问题上。这些改进的重点往往不在于跨项目学习和改进的组织级制度化。如果观察测试改进是如何组织和管理的，可能会发现关注点不是在组织级别的（测试）过程组，而是更聚焦在项目中团队的自我管理。这并不是一件坏事，创建一种机制是最重要的，使得本地测试成功的改进经验与组织的其他成员共享，这些改进经验可能超越了敏捷团队力量可以解决的问题。一个可能的解决方案是尽可能让测试组织的代表参加当地的回顾会议，以确保在整个组织内实现共享和在需要时进行制度化改进。测试过程的改进人员也打破组织壁垒，消除影响整个组织测试的更广泛或基本的问题。

3.1.5 SG5 部署组织测试过程并合并经验教训

对于敏捷项目，重要的是根据需要尽可能多地获取和重用组织标准测试过程和测试过程资产，并产生附加价值。测试组织将确保这些过程和过程资产部署在所有项目中。（关于组织标准测试过程和测试过程资产的更多信息以及他们与敏捷项目的关系，请参考测试生命周期和集成过程域。）

组织标准测试过程的实施以及敏捷项目中测试过程资产的使用可以由自组织团队自己进行监控，例如，通过在回顾中进行解决。通过参加这些会议的测试组织的代表（例如测试过程改进人员），可以跟踪实施情况并解决跨项目问题。此人还可以从敏捷团队那里获得宝贵的经验教训和测试改进回到测试组织，随后可将提供的经验教训和测试改进提交为测试过程改进建议，并进行相应管理。从敏捷团队的测试中汲取的经验教训因此被纳入组织的标准（敏捷）（测试）过程和测试过程资产。

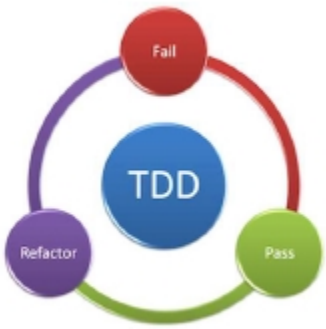
3.2 过程域 3.2 测试培训方案

测试培训方案过程域的目的是开发一个培训体系，该体系将致力于开发员工的知识与技能，从而使得测试任务及角色能够得到有效高效执行。

测试培训方案的主要目标是为测试人员和其他团队成员提供必要的（测试）培训。由于敏捷宣言声明“个人和交互 与过程和工具 相比更为重要”，敏捷环境中的过程在文档化时通过是轻量级的，而不是解决所有可能的情况。然而，人员培训则成为需要关注的关键成功因素，需要聚焦，而其过程定义几乎成为“轻量级”的对立面。高品质的培训方案确保参与测试的人员能不断提高测试知识和技能，并获得必要的领域知识和与测试相关的其他知识和技能。由于环境上下文中，质量和测试都是整个团队的责任，因此预计测试相关培训不仅仅提供给专业测试人员，而且提供给整个敏捷团队。

3.2.1 SG1 建立组织测试培训能力

该过程从确定组织的战略测试培训需求开始。除了测试人员的传统培训需求外，例如测试设计技术和覆盖率测量，现在还需要获取有关敏捷测试特性的知识和技能。例如敏捷宣言及其原理，Scrum，Kanban 和 XP 等框架，敏捷测试概念（如测试金字塔和测试象限），测试驱动开发（TDD），行为驱动开发（BDD），验收测试驱动开发（ATDD），用户故事开发，包括支持验收标准的定义并审查它们的可测性，测试自动化和探索性测试。此外，一些典型的测试管理知识和技能领域，例如计划，估算（例如计划卡片），监控和风险分析（如风险卡片）现在将成为大多数在敏捷团队中工作的测试人员所需知识和技能的一部分，而不仅仅是测试经理。战略测试培训需求的确定不仅限于测试人员，还要为敏捷团队的其他成员明确测试相关的培训需求。



就像任何其他组织（级过程域）一样，测试培训需求将被转化为（轻量级）培训方案，可能也会同时解决一些特定的项目测试培训需求。培训方案需要进行评审，并获得相应的承诺。请注意，尤其是在敏捷方面，通过非正式沟通方式（例如在岗培训，午餐培训课程，教练辅导和指导）可以有效地传授一些技能，而其他技能则需要正式培训。应根据知识和技能领域确定满足特定测试培训需求的适当方法，例如，通常工作中以在岗培训方式对探索性测试就是一个重要方式。

3.2.2 SG2 提供必要的测试培训

在很大程度上，在传统和敏捷组织中，SG2 的具体实践是相同的。根据组织培训方案，确定团队成员需要接受的必要培训，以便能够有效地执行测试任务。随后安排计划，获得所需资源，并进行培训。这似乎是一个明确的陈述，但一些敏捷组织很难找到时间进行培训。敏捷团队总是把工作安排得很满，努力在最后期限前完成迭代。当然，培训时间需要在迭代计划中加以安排，特别是参与者在下一次迭代中可参与培训的时间很少时。

如上所述，非正式培训方式通常在敏捷组织中使用。例如，除了正式培训之外，通常会不断鼓励和使用在职培训和指导。实际上，指导是每个人都有责任，并且在组织的各个层面上都是需要的。结对工作是敏捷团队的另一个常用工具，用于技能提升和知识转移。在使用 Scrum 时，Scrum 教练承担了在 Scrum 实践中指导团队的责任。

对非正式或正式的测试培训，都需要进行记录并评估测试培训的有效性。参加（测试）培训也可作为回顾会议的一部分进行评估，由此进行讨论，参与培训后，人员的知识和技能是否更有利于执行测试任务。

3.3 过程域 3.3 测试生命周期和集成

测试生命周期和集成的目的是建立并维护一个易用的组织测试过程资产库（例如，

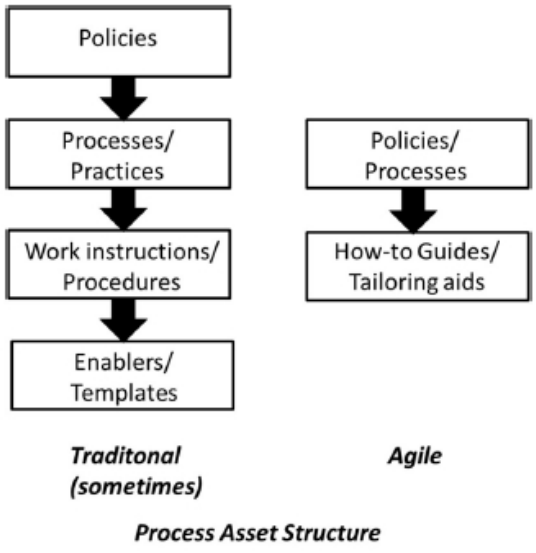
一个标准测试生命周期）和工作环境标准，并将测试生命周期与开发生命周期集成和同步。集成的生命周期确保了测试在项目中尽早参与。测试生命周期和集成的目的还在于根据已识别的风险和已定义的测试策略，跨多个测试级别定义一致的测试方式，并根据定义的测试生命周期建立一个总体测试计划。

3.3.1 SG1 建立组织测试过程资产

测试组织的一项重要职责是根据组织的测试方针和目标来定义、记录和维护标准测试过程。这其中，包含对要执行的各种测试类型和级别的描述。虽然 TMMi 没有要求必须完整和详细地定义和记录测试过程，但具备经常关注的模板，并能通过例子进行阐述，是建立一种通用工作方式的好方法。另请参阅 Bob Galen 的敏捷质量和测试三大支柱，他将标准列为检查表，模板和存储库，这是敏捷软件测试[Galen]的支柱之一。模板标识了收集信息和并通过其结构隐含记录获得结果所需的活动。过程不需要一定是一个严格的活动序列。当存在依赖关系时，它们可以通过模板中的注释或表单中的字段捕获，直到其他先决条件字段完成后才可操作。模板具有传达过程真实意图的实用价值。模板还避免了“大篇幅”过程文档中常见的歧义。作为一个敏捷组织，如果您拥有有效的过程，您可能只需要以轻量化的方式记录它们，并且可能会添加一些内容。敏捷组织在建立过程时经常使用的一些关键指导原则：

- 过程中“必须做的事情”与准则分开阐述
- 没有过程（描述）超过两页（目标，软规则）
- 过程不包含“如何操作”信息或工具信息，除非已决定跨所有项目强制执行此操作，而不考虑其大小和规模。
- 单独的指导方针包含剪裁/计划选项以及“如何操作”信息。请注意，通过“如何操作”信息，人们也可以决定参考现有的书籍或论文

在许多大型组织中，看到四个级别的过程资产（例如方针，过程/实践，工作指令/规程和启用者/模板）并不罕见。这不是因为 TMMi 需要，而是许多大型组织选择实施他们过程资产的方式。虽然过程资产的选择



取决于每个组织，但大多数敏捷组织发现，两级过程资产已足够。这是通过将方针声明与相关过程描述合并来实现的，该过程描述封装了执行过程的“必须完成的工作”。第二个层次包含“如何操作”指导过程实施并对其进行调整。这个级别可以被视为定制过程的辅助工具，给团队自主权以决定如何在定义的框架内工作，并且通常包括支持模板。在大多数敏捷组织中，一步一步的过程被工具指南和培训/指导所取代。如上所述，过程是模板（例如测试章程模板，测试会议工作表模板或测试计划模板）的使用，模板中隐含了过程中必需的活动。在开发有效的敏捷过程描述时，这是一种常见的技巧。[McMahon]

组织的标准测试过程集合可以由项目组裁剪，以创建其特定的已定义过程。在 SP 1.3 的背景下建立剪裁标准和指导方针，今天的大多数组织都使用了所谓“向下裁剪”的方法。这种方法要求人们花费精力解释为什么不需要。这为最简单的项目增加了大量工作，这不是一种敏捷友好的方法。当你量身定制时，你能走多远？有最低限定吗？但是，如果你从最低限度开始，每个人都必须做最低限度的设置，那么你就可以消除把“必须做的”给裁剪掉的风险。一个“向上裁剪的方法与敏捷方法更加一致，并且完全符合 TMMi。对于希望保持敏捷性的敏捷组织或寻求提高敏捷性的高成熟（测试）组织来说，定制是一个非常强大的机制。一套简单的准则有助于在向上裁剪时提出正确的问题。没有准则来帮助，有些人往往会“向上裁剪”太多。

为了支持标准测试过程的部署，需要建立了维护一个测试过程资产库，通常使用 wiki，测试人员可以在其中找到模板，最佳实践和清单等，这些将有助于他们日常工作。只要团队在测试活动中使用所提供的资产时获得附加价值，创建测试过程资产库在敏捷环境中也是有意义的。测试过程数据库的存在或多或少有相同的理由，其中测试数据（例如估计，覆盖率，质量度量）被收集并提供给团队。确保无论收集或提供什么信息，跨团队学习和分享经验都能为团队增加价值。

当然，在敏捷环境中，工作环境标准也用于指导项目工作环境的创建。

3.3.2 SG2 集成测试生命周期模型与开发模型

标准测试生命周期模型定义了各种测试级别的主要活动和可交付成果。标准测试生命周期模型与开发生命周期模型保持一致，从而将测试活动按阶段，里程碑，交付成果和活动进行整合。生命周期集成的完成是确保测试在早期就参与项目项目。在敏捷生命周期已被采纳的情况下，这听起来像是没有什么可做的；这一切都已经被考虑在内。在敏捷中，开发和测试应该在生命周期内完全整合，测试应该尽早进行。

在许多使用敏捷的组织中，情况确实如此，也有一些组织在这方面很纠结。他们的迭代类似于一个小型的 V 模型生命周期的开发，其中测试只在迭代结束时开始，通常在时间压力下执行，有时甚至还被缩短。当然，这不是一个完整的敏捷开发和测试生命周期集成的的应有实现。然而，它表明，对于一些“敏捷”的组织来说，这个特殊目标是

一个需要改进的领域。通过定义测试如何参与并在迭代早期执行，形成共识很重要。

这个特殊目标因此也适用于进行敏捷软件开发的组织。如果做得适当，开发和测试应该完全整合，并就如何完成这个问题达成共识。在这种情况下，没有其他需要做。如上所述，在其他情况下该特殊目标是一个非常重要的改进领域，并且应该在迭代中引导完成更好的开发和测试活动。

3.3.3 SG3 建立主测试计划

在 TMMi 3 级，测试涉及主测试计划，该测试计划针对所有测试级别的测试任务，责任和测试方法进行协调。这样可以避免不同测试级别之间不必要的冗余或遗漏测试，并且可以显着提高整体测试效率和质量。主测试计划描述了测试策略在特定项目中的应用，包括要执行的特定级别以及这些级别之间的关系。作为测试计划过程域的一部分，TMMi 2 级（测试）计划已经在发布和迭代级别都得到了解决。一个敏捷团队通常会执行多个测试级别，这些都应该作为发布和迭代计划的一部分进行适当处理。

但是，一些根据敏捷生命周期模型开展工作的项目采用了一种混合方法，因为除了在敏捷团队中进行测试之外，还有一些测试是在敏捷团队之外进行的。在这种情况下，主测试计划将定义和管理从敏捷团队到在其范围外执行测试级别的关系，例如特定的非功能测试（安全性，性能），硬件/软件集成测试，系统集成测试或 beta 测试。整体看这一特殊目标，所有相关的特殊实践都通常情况下更有适用。尤其是在大型项目中，随着组织的发展，以一致的方式记录主测试计划和相关决策开始具有更大的价值。当一个组织使用规模化的敏捷框架时尤其如此。

如果所有的测试都是在敏捷团队中进行的，那么特定的主测试计划没有附加价值，因此这种 TMMi 目标在这种情况下很可能是不相关的。这里的发布和迭代级别的计划预计将涵盖所有必要的测试方面。

3.4 过程域 3.4 非功能测试

非功能测试过程域的目的是为了提高测试过程能力，从而在测试计划、测试设计和执行的时候考虑到非功能测试。

非功能测试是否重要，和所使用的生命周期无关。正在开发的产品将决定进行哪些非功能测试。这对于顺序生命周期模型和敏捷开发模型都是如此。因此，非功能测试也适用于敏捷软件开发。

然而，根据非功能方面的特征和所使用的测试方法，一些在传统方式测试时执行的非功能测试（例如性能测试和可靠性测试）不能在短期迭代中实施。由于这些非功能测试可能会根据所使用

的方法和技术花费数周时间来设置组织和执行因此它们的整体时间开销不太适用于迭代的快速节奏。因此，我们需要不同的方法来进行非功能测试。这是敏捷带来的更好的根本性变化之一，因为质量属性在迭代过程中已经进行了早期测试，并且没有像传统的生命周期模型那样放在一边，直到项目最后期进行。并非所有的组件/功能都可以从第 1 次迭代中获得并进行完整的非功能测试，但是这种测试结果可以在早期指出质量问题。

用户故事应该定义功能和非功能属性，以确保为用户和客户开发正确的产品。ISO 25010 质量特性可以帮助构建质量要求，测试人员应该考虑这些非功能属性[ISO25010]。

3.4.1 SG1 非功能执行非功能产品风险评估

敏捷项目的产品风险会议，如特殊目标 SG1 所确定和描述的那样，执行风险评估作为测试计划过程域的一部分，现在将明确地扩展到非功能方面的相关风险。在发布级别，风险评估现在也可用于非功能测试，可以基于产品愿景进行。在迭代级别，这是以用户故事中定义的非功能需求定义为主要输入执行的。

最好，包括产品负责人在内的所有团队成员以及可能的其他干系人都应参与产品风险会议，在一些非功能领域，可能需要专家来协助。产品风险评估过程通常会形成一份记录了优先级的产品（非）功能性风险清单。正如在测试计划过程域所述，在敏捷项目中，与使用顺序生命周期模型的传统项目相比，所使用的过程和生成的文档将更加轻量化。

3.4.2 SG2 建立非功能测试途径

定义了相关非功能质量属性的测试途径，以消减已识别和优先考虑的非功能产品风险。对于特定的迭代，要在迭代计划期间识别要测试的非功能特征。要测试的非功能特性的优先级列表通常涉及要在此迭代中测试的用户故事。在迭代期间，新的非功能产品风险可能会变得更突出，需要进行额外的测试。通常在每日站立会议上讨论需要额外测试的新的产品非功能风险等问题。

在迭代级定义的非功能测试方式可以降低非功能风险，通常基于非功能风险级别和类型的覆盖，标识适当的非功能测试途径和测试技术。通常它还将解决支持工具的使用，以及非功能测试自动化的方法。在发布级别建立非功能测试的测试方式将处于更高级别，应基于程序或组织级别上定义的测试策略以及定义的主测试计划（如果存在）。（参见 SG3 在测试生命周期与集成过程域，在测试生命周期与集成中是否建立一个主测试计划）。发布和迭代级别的非功能测试方式都是整体测试方式的一部分，并将在团队/项目 wiki 上进行保持或展示。

非功能出口准则是所谓的已完成定义（DoD）的一部分。DoD 具有与非功能测试相关的特定准则是很重要的，例如平均时间间隔（MTBF）或“前端网页已针对 OWASP 前 10 级风险列表进行测试”。迭代应该导致执行一组商定的非功能用户故事（包括他们的验收准则），并满足 DoD 定义的非功能（测试）退出准则。请注意，非功能属性也可以成

为功能性用户故事验收准则的一部分，并且确实需要指定为单独的非功能用户故事。完成定义不仅存在于迭代级别，经常还有跨越多次迭代的发布级别的完成定义。当然，发布级别的 DoD 可能也有非功能相关的准则。

3.4.3 SG3 制定非功能测试分析与设计

这个特定的目标很大程度上遵循相同的实践，但现在从非功能角度来看，就像测试设计与执行过程域中的特殊目标 SG1 使用测试设计技术执行测试分析与设计一样。在测试分析和设计过程中，非功能测试的测试途径被转化为有形的测试条件和（具体的）测试。在敏捷中，测试分析与设计以及测试执行是相互支持的活动，通常在迭代过程中并行进行。对于大多数非功能测试也是如此。因此，非功能测试分析不是一个明确的单独活动，而是测试人员在协助用户故事开发中扮演其角色的一部分工作。

基于对非功能用户故事的分析，确定了非功能测试条件。测试条件基本上是需要测试/覆盖的“事物”的标识。验收准则如果规定的足够详细和清晰，就可以用来接管传统测试条件的角色。验收准则随后被转化为非功能测试。考虑到非功能测试，在更高级别执行测试分析通常是有益的，而不仅仅局限在用户故事层级。例如，分析一个特性或一个史诗故事或一个故事集合以识别比用户故事级别更抽象的非功能测试条件，并跨越多个用户故事。在敏捷使用测试优先原则的情况下，涵盖一系列非功能测试条件的非功能测试将在代码开发之前或至少与之并行进行识别（并可能是自动化的）。

对于大多数手工方式的非功能测试，随着团队进行非功能测试执行，测试将被识别/改进。测试通常没有像传统项目那样详细记录，而是采用探索性测试时的测试思路。但对于复杂和关键的非功能领域，使用正式的非功能测试设计技术来进行测试设计和测试用例开发可能是覆盖风险的最好方法。但是，与传统的顺序生命周期环境中的非功能测试相比，采用这种方法的文档量也会有限。非功能测试的优先级通常遵循他们所覆盖的用户故事的优先级。但是，优先顺序也可能由准备和执行某种非功能测试所需的时间所驱动。

识别支持执行非功能测试所需的特定测试数据。在敏捷中，所需的测试数据通常不是首先在测试规范文档中完全指定的。通常通过将其记录为用户故事的一部分来指定的。但是如果提供了必要的工具和/或相关功能可用，则通常会立即创建非功能测试所需的测试数据，以便立即开始执行非功能测试。与手工测试不同，对于自动化非功能测试，通常需要预先指定数据。

需要建立和维护非功能需求，测试条件和测试之间的可追溯性。团队需要明确表示他们已经涵盖了各种非功能用户故事和验收准则，作为他们测试的一部分。在许多敏捷组织中，用户故事是制定一套相关验收准则和随后测试的基础。使用这种方法可以实现从需求到测试的横向可追溯性。

3.4.4 SG4 执行非功能测试实施

执行测试实施需要将启动测试所需要的各项事务到位。通常，开发支持测试执行的测试文档（例如测试规程）的是最小的任务。优先考虑自动化（回归）测试脚本的开发。

非功能测试实施将遵循许多实践，这些实践在过程域测试设计与执行中的特殊目标 SG2 执行测试实施这一中被描述。

具体需要做什么以及如何实现非功能测试主要取决于定义的方式，使用的技术以及需要测试哪些非功能特性以及测试到什么程度。例如，需要较少准备工作的探索性测试可能适用于可用性测试，但通常不太适合完备的可靠性和性能测试。

对于某些非功能质量属性，测试数据的可用性是必不可少的，需要在测试实施期间创建。这与传统项目大体相同，可能以简短的方式记录下来，仍然允许回归测试中重用。

当然，测试的实施和准备将尽快开始，与其他（测试）活动并行进行。这不是一个单独的阶段，而是一组需要执行的活动（在任务板上列出），以实现高效且有效的测试执行。

3.4.5 SG5 执行非功能测试实施

与此过程域中的前一个目标一样，我们将在很大程度上回顾本文前面讨论的过程域测试设计与执行过程域的相关特殊目标 SG 3 执行测试实施部分。在一个敏捷环境中执行非功能测试，报告测试事件和编写测试日志的实践基本上与敏捷环境中的功能测试相同。通常情况下，文档密度要比传统项目少得多。通常不会产生详细的测试规程和测试日志。

非功能测试执行与迭代计划期间定义的优先级一致。一些测试可以基于文档化的测试规程来执行，但是通常还是会使用基于探索性和基于会话的测试作为其框架来执行许多非功能测试。随着迭代开发，组织和构建回归测试的需求增加。这可以手动完成的，但最好使用自动回归测试和支持工具完成。当然，回归测试也适用于已被确定为需要重点测试的系统的非功能方面。

测试期间发现的非功能事件可能会被记录并由团队报告。在敏捷项目中通常会进行讨论，是否所有发现的事件都应该被记录下来。有些团队只记录可以脱离迭代范围的事件，有些团队记录当天无法修复的事件，有些只记录高优先级事件。如果确定不记录所有的事件，则必须提供准则以确定应记录哪些事件，哪些事件不用记录。有时，事件会记录在敏捷任务板上，既可以作为任务的标签，也可以作为单独的任务，可视化展示它阻断用户故事以及任务完成需要的工作。有些人选择记录和记录使用工具找到的事件，例如缺陷管理或缺陷跟踪工具。这样的工具应该尽可能地被精益化使用，如果做为测试事件报告的要素，但不带来或只带来很少的附加价值，那就不要强制提交。非功能

敏捷团队中，在非功能测试执行过程中记录数据以确定测试的项目是否符合已定义的验收准则，是否可以确实标记为“已完成”，这也被认为是一种很好的做法。记录的信息应该以一种方式捕获和/或归总到某种形式的状态管理工具中（例如，测试管理工具，任务管理工具，任务板），以便团队和干系人更容易了解当前状态所有已执行的测

试。

3.5 过程域 3.5 同行评审

同行评审过程域的目的是验证工作产品是否满足指定的需求并且尽早而有效地移除选定的工作产品中的缺陷。其重要的必然性影响是开发更易理解的工作产品并且可能预防更多的缺陷。

3.5.1 SG1 建立同行评审途径

敏捷团队通常不会进行正式的同行评审，因为他们通常没有一个确定的时间将团队成员聚在一起来提供产品反馈。然而，他们确实通过在整个开发过程中持续进行的不太正式的同行评审来实现同行评审的意图。这在许多敏捷组织中是很常见的做法。但是在开展这些活动时仍然需要一定的纪律。该 TMMi 目标涵盖了在项目中建立同行评审方式的实践，评审方式定义评审活动应该如何，何地和何时进行，以及这些活动是正式还是非正式的。建立同行评审途径也适用于敏捷项目，然而，通常情况下如何组织评审和应用的是非常不同的。

通常在敏捷项目中执行的同行评审的例子：

- 在迭代过程中定期对团队和业务干系人关于规格说明（例如，用户故事）进行的细化/梳理；
- 与其他团队成员每日面议，公开讨论正在开发的工作产品，例如代码或测试，并提供反馈；
- 至少在迭代期间快结束时，尽早向客户进行产品展示并与客户沟通。

需求规格质量不佳通常是项目失败的主要原因。需求规格问题可能是由于用户缺乏对其真实需求的洞察力，系统缺乏全球视野，冗余或矛盾的特性以及其他不当的交流。在敏捷开发中，编写用户故事是为了从业务代表，开发人员和测试人员的角度捕捉需求。在顺序开发过程中，这种功能的共同愿景是在写完需求之后通过正式评审完成的；在敏捷开发中，这种共同愿景是在需求被建立的过程中，不断地非正式评审来完成的。这些非正式审查会议通常被称为待开发列表细化或待开发列表梳理会议。在细化会议期间，业务代表和开发团队（以及干系人，如果有的话）使用评审技术来找到所需的实施细节并澄清未解决的问题。

团队几乎不间断地评审正在开发的工作产品是整个团队工作方式的一部分。它们的执行旨在尽早发现缺陷并识别改进的机会。像 XP 或结对这样的敏捷方法和技术还包括同行评审，作为为团队创建反馈循环的核心实践。当然，如果高复杂度或高风险，团队可以选择应用半正式或正式的评审技术，例如审查。在这些情况下，有明确的理由来支持更正式的审查，并采用更加严谨的工作方式。评审方式准则的一部分就是定义何时使用更正式的评审技术。

虽然需求工程强调通过非正式评审、走查、审查、或分角色阅读等方法进行需求验证，但敏捷方法通过经常和早期的反馈努力验证需求，以快速实现有价值的产品增量。通过以集成产品增量的形式快速展示结果，减少了对早期正式验证的需求。如果增量不能完全满足干系人的要求，那么增量将以新需求的形式重新回到产品待办事项中，并与所有其他待办项目区分优先次序。在迭代过程中，实际构建的内容的演示是一种非常有效的方式，可以激发围绕具体对象的基于验证的对话。没有什么能够提供对话焦点，比如能够真正看到事物的真实运作。演示是在迭代评审期间执行的一项活动。向干系人展示特性并共同讨论，对产品待办事项或发布计划进行必要的调整，以反映讨论中得到的对于产品的新认识。

3.5.2 SG2 执行同行评审

当然，就像任何方式一样，它不应该仅作为一个定义的和商定的方式存在，它应该被遵守。团队需要在迭代期间花费大量时间在待办事项回顾提炼工作上，将非正式（或可能）正式评审作为其日常工作的一部分，并定期向干系人演示当前成果，至少在每次迭代结束时，向干系人/客户演示。

准入准则通常只适用于正式评审，因此在敏捷项目中很可能较少或不相关。但是，退出准则是适用的并具有增加价值。INVEST [WAKE]是在评审和更新用户故事时，敏捷项目中通常提到的一组退出准则的示例。INVEST 是一个首字母缩略词，它包含了构成一个好的用户故事的以下概念：

- 独立-Independent
- 可协商-Negotiable
- 有价值-Valuable
- 可预测-Estimable
- 小-Small
- 可测试-Testable.

当然，其他质量相关的退出准则也存在并且也可以被有益地使用。

作为敏捷团队成员的测试人员参与评审会议非常重要。对于那些讨论和评审工作产品的会议尤其如此，这些会议在整个迭代过程中都会用作测试的基础，例如回顾提炼会议。建议在回顾改进时至少有一名开发人员和一名测试人员在场，以确保存对于提供的系统的不同视角。通常，测试人员的独特视角将通过识别缺失的细节或识别非功能需求来改善用户故事。测试人员可以特别支持某个用户故事的验收准则的识别和定义。测试人员通过向业务代表询问有关用户故事的开放式的“如果？”问题，提出测试用户故事

的方法以及确认验收准则，这是测试人员的贡献。

特殊实践“2.3 分析同行评审数据”与正式评审尤其相关。通过正式评估，我们花费了大量的精力，为了确保这项工作的高效率和有效性，同行评审数据被收集并传达给团队，以便学习和调整评审过程。如前所述，在敏捷项目中，正式的评审（例如检查）较少见。鉴于收集数据是正式审查的重要组成部分，而非正式审查却少见。因此，有人可能会争辩说，在敏捷的背景下，详尽的同行评审数据收集、分析与交流是不太重要的。在同行评审中，决定进行收集数据的情形下，需考虑以下问题：

- 在收集这些数据之后，谁将使用这些数据？
- 这些数据与我们的业务目标有何关系？

如果没有人有意义地使用这些数据，那么不要浪费宝贵的资源来收集它们。作为结论，对于大多数敏捷项目特殊实践“2.3 分析同行评审数据”将很可能被视为不相关。请注意，一些基本的宝贵评审数据仍将在通用实践中收集和使用，例如，GP 2.8 监控与控制过程。

4. TMMi 4 级已测量

正如本文档前面所述，重要的是要记住 TMMi 4 级、5 级与 TMMi 3 级之间有自然的断层。关于将一个组织提升到 TMMi 4 级和 5 级的价值，仍然持续地存在激烈的争议。尤其是敏捷组织被推荐慎重选择 TMMi 4 级和 5 级有意义且有附加值的实践。尽管 TMMi 是全面的，为了取得成功，组织必须要识别出关键的测试实践以及需要关注的优化项。有趣的是，敏捷的联合创始人 Jeff Sutherland，发表了一篇论文，讨论高成熟度的实践与敏捷结合使用的价值[Sutherland et al]。下文将提供，如何以不太正式的形式将敏捷方式与更高的 TMMi4 级和 5 级以及它们的实践结合起来获得价值的参考信息。首先，与已有的说法有些不同，在尝试实现 TMMi 2 级、3 级测试实践的同时，也可以已经考虑有选择性地将 TMMi 4、5 级实践与敏捷方法结合使用，来达成关键业务目标。你不必也不应该等待。要灵活使用，不要拘泥于 TMMi 2 级、3 级测试实践。根据你的业务驱动使用模型中最具价值的过程域、目标和实践。如此一来，某种程度上将 TMMi 视为连续模型使用不必过于严格地遵守成熟度等级要求。

4.1 过程域 4.1 测试测量

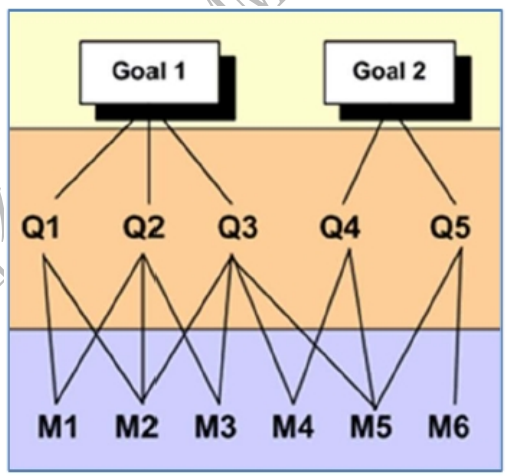
测试测量的目的是识别、收集、分析和应用度量，支持一个组织客观地评估测试过程的有效性和效率，它的测试人员的生产力，最终产品的质量以及测试优化的结果。这

样，测试组织将发展和保持一个测试测量的能力，用于支持信息管理的需求。

敏捷方法有助于获得在项目中，必须执行精准测量工作的人的承诺。但是当涉及到确保正确的测量贯穿于整个组织，和确保这些从组织的业务需要衍生出的度量与测量目标之间保持一致时，敏捷方法并不倾向于对它们提供更多的支持。这些是 TMMi 测试测量过程域能带给敏捷组织的好处。

4.1.1 SG1 匹配测试测量和分析活动

测量方案似乎常常要收集尽可能多的数据。一个测量方案应该高度专注，只有有意义的数据才应该被收集。首先，为测量方案定义明确的目标，并且定期进行回顾，举个例子，“它应该能让我们在项目中更成功地测试。”。接着，和干系人讨论哪些测试度量标准能真正支持这些目标。基于这次头脑风暴，可以定义一个有限的核心度量标准集。



目标-问题-度量 (GQM) [Van Solingen and Berghout]是一个支持该方式的非常实用的方法。。需要基于业务需要的特定上下文定义相关度量指标。公司标准度量指标通常难以满足实际（测试）过程改进。测量的目的是为了指导决策。使用小型授权团队得到有意义的度量指标，然后在短周期内进行审查和完善。

从始于增加（商业）值出发的专注的测试测量方案，TMMi 测试测量过程域完全符合敏捷的简单性原则。确保组织内有精益的测试测量以及数据收集策略。值得指出的是，测试测量方案的进化可以从几个集中的度量目标和几个关键度量出发，然后随着业务需要不断发展和变化对其进行扩展或修改。

需要注意的是，敏捷会更多专业在基于团队和系统型思考。这可能会导致某些度量

拓展到团队和整个系统，而不仅仅局限于测试本身。这很可能导致在测量分析阶段带来更多的挑战。

当然，基于测量目标，定义度量指标和将为谁收集度量数据，但是下面提供一些敏捷团队典型的测试度量的例子：

- 缺陷周期时间：敏捷团队应该努力尽快修复缺陷。实际上，协作敏捷方法最主要的目标之一是更快地修复缺陷，以便更快地发布软件。

- 缺陷溢出（缺陷的数量递延到未来的版本）敏捷团地旨在每轮迭代都可以生产出可工作的软件。通过简单计算每一轮迭代或冲刺结束时遗留的缺陷，缺陷溢出测量了给定迭代或迭代冲刺时间内无法被修复的缺陷。在一些敏捷组织内，这被称为技术债务。

- 每一轮迭代发现的缺陷数量

- 发布到生产/给顾客后发现的缺陷数量

- 团队外部人员发现的缺陷数量

- 客户支持请求的数量

- 自动测试覆盖率

- 代码覆盖率

4.1.2 SG2 提供测试测量结果

一旦根据业务需要定义了相关度量，在传统的组织内，该特定目标内的特定实践通常都能在组织内以大体相同的方式被使用。。唯一的不同是在实践中，记住要尽可能以精益的方式来落地。收集需要的测试测量数据，并检查其完备性与完整性，随后按计划对数据进行分析，并将结果传达给所有相关干系人。

4.2 过程域 4.2 产品质量评估

产品质量评估的目的是对产品质量建立一个定量理解，从而支持特定项目的产品质

量目标的实现。

4.2.1 SG1 建立产品质量和优先级的可测量项目目标

本质上，无论是在传统环境中或者敏捷背景下，定义和设置可测量优先目标没有真正的区别。总体目标是为满足顾客和最终用户对优质产品的需求而奋斗。在敏捷背景下，有很多选择可以做到这点。一些使用“完成定义”来定义产品质量的可测量目标，另一些则使用特征，史诗故事或用户故事来定义他们，借此，他们也可以使用验收标准来使它们可测量。这其中，选择（工作方式）取决于产品质量目标的性质。有关质量属性的概述，请参考[ISO25010]。

“完成定义”（DoD）是必要的团队活动和/或工作产品的全面清单，以此来确保不仅在功能方面还在产品质量方面，仅交付真正完成的用户故事。在 DoD 内说明产品质量目标的好处是，在开发增量时，它可以使整个团队看见它们，从而采取行动。显然，正如在 DoD 中提到的质量目标需要适用于整体的应用。如果质量目标不能整个适用，也许最好的替代工作方式是将他们定义成一个特性或者史诗。

有时，开始时只能在发布级别指定一个相关产品质量属性的高阶列表。这些将会在细化提炼会议中逐渐以特征、史诗故事或用户故事的形式详细阐述。随后，在一轮具体的迭代开始时，团队将会考虑是否将一个或者多个指定产品质量目标作为该特定迭代的目标的一部分。

4.2.2 SG2 项目产品质量目标的实际进度被量化和管理

一旦设定了定量的产品质量目标，这些目标将会受到监测和控制，在必要时可能会被调整。由于产品质量通过可测量的验收标准，将被定义为完成定义（DoD）和 / 或作为特定的特性、史诗或用户故事的一部分，同样的方法和技术也可用于实现产品质量目标进度的跟踪和管理，例如已经在 TMMi2 级的过程域 2.3 测试监督与控制中解释和描述的那样。进度每天都会被测量，并在日常的站会上讨论进度、质量和风险。此外，产品质量评估通常也由提供测量基础设施的测试测量过程域所支持。

4.3 过程域 4.3 高级评审

高级评审的目的是在 TMMi 3 级同行评审过程域的实施基础上,在产品生命周期的早期对产品质量进行评估，通过将同行评审（静态测试）与动态测试相结合的方式来加强测试策略和测试方法。

4.3.1 SG1 同行评审方法和动态测试方法相协同

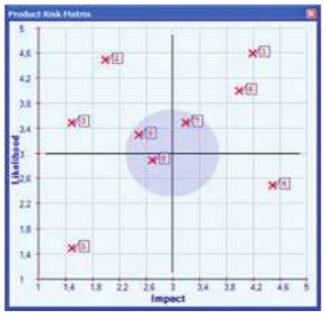
传统项目中的静态测试，如评审，通常与独立测试团队正在进行的动态测试没有联系，在敏捷环境中的预期是完全不同的。在敏捷环境中，确保产品质量的方法应该是将所有与测试相关的活动（静态和动态）组织为一个连贯和协调的方法。当然，敏捷团队的职责是集成静态和动态测试并找到最佳的组合。指望敏捷团队来领导建立连贯有效的集成方法。因此，静态测试方法（同行评审）应该与动态测试方法保持一致和协调。

4.3.2 SG2 在产品生命周期早期通过同行评审来评估产品质量

在 TMMi4 级中，组织为软件产品和相关工作产品设定了量化目标。同行评审在实现这些目标时发挥着必不可少的作用。在 TMMi3 级中，同行评审主要是为了发现缺陷，而现在的重点是评估产品/文档质量以便在生命周期的早期对产品质量进行控制。评审实践得通过应用，包括抽样、应用出口准则和预先规定的规则等实践得到了增强。事实上，这就是 Tom Gilb 所说的敏捷审查，它将重点从清理转移到抽样、测量、动机和缺陷预防[Gilb]。这显然是一种适合高成熟度敏捷组织的实践。另一个例子是将用户故事的 INVEST 规则 [Wake]当作就绪定义（DoR），并通过对照这些规则检查用户故事来测量用户故事的质量。一个就绪定义意味着故事必须立即进入可操作。

4.3.3 SG3 根据生命周期早期的评审结果调整测试方法

当同行评审结果和动态测试已经进行协同，早期评审结果和数据会影响产品风险和测试方法。正如在这个过程域的特殊目标 1 中所述，在敏捷中，质量是团队任务，基于验证和确认结果的活动将在团队会议上讨论。使用早期测试结果决定后续测试活动的原则，与敏捷团队和项目中的工作方式几乎完全一致。



5. TMMi 5 级优化

TMMi5 级的目标和实践源自于高成熟度的质量管理和过程改进方法。在这个高度成

熟的阶段，目标和实践几乎独立于应用中的软件生命周期。当然，一个基于精益敏捷的组织将以不同于传统的多等级组织的方式来处理事情，但是围绕质量控制、过程优化和缺陷预防的基本实践将在很大程度上保持不变。这样做的理由是，这些实践被视为质量管理实践，通常在高于操作产品开发的级别上执行。因此，这些实践与项目或产品交付流中软件开发和/或测试的主要活动没有直接关系。

5.1 过程域 5.1 缺陷预防

缺陷预防的目的是识别和分析整个开发生命周期中缺陷的常见原因，并对未来可能出现类似缺陷的预防措施进行定义。

敏捷组织通常已经对缺陷预防有所关注，这是建立该过程域的具体目标和具体实践的重要基础。高效的敏捷组织更关注于缺陷预防而不仅仅是缺陷发现。缺陷预防的定义包括将“过程中工作”的缺陷，与在开发用户故事的迭代之外逃逸的缺陷分离开来。

敏捷回顾是团队反思、学习并不断改进其工作的一种实践。虽然回顾最常用于探索当前的工作方式，但它们也可以用来调查质量问题，或者是同意可以提高软件交付质量的行动。

缺陷预防这个过程域包括了一种在跨团队、产品和价值流中识别和分析常见缺陷原因的实践，并定义了一种在今后消除这种类型缺陷的常见原因的具体措施。所有缺陷，无论是在开发、测试或上线后发现的，都在这个过程域的范围内。由于此级别的缺陷预防需要测量数据，因此缺陷预防是建立在 TMMi 4 级测量实践和有关开发、测试和产品质量的可用测量数据的基础上。

敏捷组织中的缺陷预防活动基本上有两种不同的类型：团队级和跨团队级。与团队相关的缺陷预防活动是团队的责任，并已经在 TMMi 3 级中得到了解决，跨团队的缺陷预防活动由测试过程小组或测试协会执行，这些活动由该过程域专门处理。

5.1.1 SG1 确定缺陷的常见原因

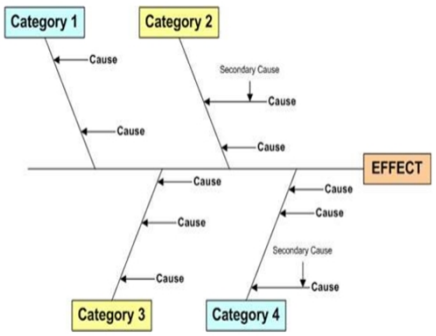
敏捷改进包括测试过程改进，它通常是基于频繁的反馈循环进行的。由于范围常常局限于前一次迭代，因此在集中解决特定的局部问题上，会做出一些微小而频繁的改进。这些改进的焦点往往不是跨团队学习或改进的制度化。只在团队层面上解决问题也很容易导致次优化或与全局失去联系。针对此问题，TMMi 过程域在现有的敏捷实践上增加了价值，现在缺陷预防也应用于组织层面上（跨团队、产品和价值流），并且通常由（测

试）过程小组或测试协会管理（见第 3.1.1 节）。

选择何种缺陷进行分析应该基于包括风险和业务价值等各种因素。应该将重点放在具有最大附加价值（通常是降低成本或风险）的缺陷预防和/或缺陷最为严重的领域。虽然缺陷预防过程域通常用于缺陷，但它也可以用于处理与速度相关的问题等其他问题。

尽管测试过程小组可以协调缺陷预防活动，但这应该由跨领域团队完成，例如，与来自需求工程、系统工程和/或软件开发的代表一起，因为改进行动通常会影响其他领域。当然，这在敏捷交付方法和团队中已经很常见了。但是，也需要根据改进措施，确保所有涉及的领域被囊括在内。

用于评估缺陷根本原因的分析方法通常用于缺陷预防和常见的敏捷实践，这些方法包括因果图、鱼骨图或 5 个为什么方法。



TMMi 模型为这一特殊目标定义的三种特殊实践（1.1 定义缺陷选择参数，1.2 选择缺陷进行分析，1.3 分析选定缺陷的根本原因和常见原因）也适用于敏捷组织。

5.1.2 SG2 确定优先级同时确定系统地消除缺陷的常见原因的措施

无论是在敏捷方式还是连续方式的生命周期中，在这个特定的目标下，执行特定的实践并没有真正的差异。当然，被提出的实际解决方案和后续的改进建议是不同的，因为它需要适应敏捷的工作方式，例如，通过创建额外的待办事项，而非批准形式化的行动建议。采取适当的行动来尽量减少问题再次发生的可能性。这通常意味着将过程改进与额外指导和/或培训团队成员及其他人员相结合。人们需要通过一种综合的方式来处理人员和过程问题，而非人为地将其分开。

此外，敏捷组织是建立在自我管理的团队之上的。这意味着团队可以选择实现和/或拒绝哪些改进。因此，在确定优先级和确定措施时，必须有足够的团队代表参与讨论这些改进是否确实为他们提供了附加价值。

此外，改进的部署方式也将有所不同，它不再是自上而下的，而更像是在自愿的基

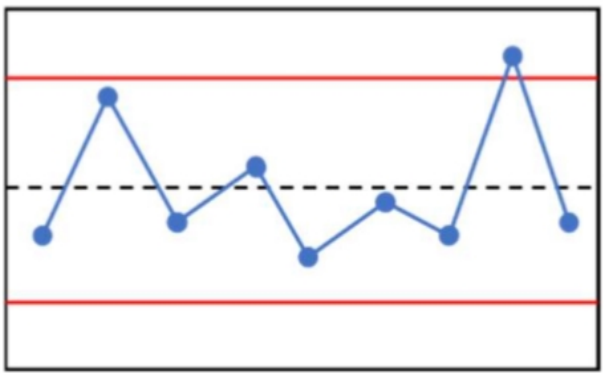
础上给自我管理团队的建议。关于测试改进部署的更多信息可以在后面的过程域测试过程优化中找到。

5.2 过程域 5.2 质量管控

质量管控的目的在于统计管理和控制测试过程。在这个等级上，测试过程的性能是完全可预测并稳定在可接受范围内的。为了预测产品的质量、提高测试效率，在项目层级上使用基于代表性样本的统计方法进行测试。

5.2.1 SG1 建立一个统计控制的测试过程

无论生命周期方法如何被应用，质量管控的原则与要点都是相同的。过程质量管控包括为标准测试过程的性能设立目标。基于来自团队的测试过程性能的测量，进行分析与调整，使测试过程性能保持在可接受的范围内。当测试过程性能稳定在可接受范围内时，所定义的测试过程、相关的测量值以及可接受的测量值则可作为基线，用于统计管控测试过程性能。组织标准测试过程的测试过程能力，即：一个团队可期望达到的测试过程性能，现在已经被完全理解及知晓了。因此，对于与这些期望之间存在的偏差，团队可以尽早采取行动并持续地保证它们在可接受范围内执行。



为统计管控选择子过程非常重要，要选择那些对业务至关重要，而且过去就曾出现过问题的子过程。经验表明，通常最能满足这些标准的子过程并非孤立于单独的团队，而是跨多团队、项目和产品交付流的。当使用敏捷方法和技术时，工作被划分成短迭代，在每个迭代中完成的工作包括执行多个开发和测试活动。因此敏捷开发比传统开发更有效地支持对统计管控最有价值的子过程的监测，因为更短的周期使测试过程性能可以几乎立即得到反馈。快速的反馈对于团队而言具有很高的价值，这也是一个重要的敏捷原则。

5.2.2 SG2 以统计学方法执行测试

产品质量管控建立于产品在其预期环境中的运行概况[Musa]和使用模型的基础之上，以便在统计上做出有效的推论，从而产生具有代表性的测试样本。这种方法尤其适

用于系统级，它使用统计测试方法基于这个代表性样本来预测产品质量。换句话来说，在测试由使用或操作配置文件所表示的所有可能使用的子集时，测试结果可以作为获得产品综合性能结论的基础。

这基本上是一种非常适合敏捷的方法，因为它本质上是精益的，并由操作使用和产品评审会议的反馈驱动。但是，敏捷组织需要具备相应的成熟度才能应用这种方法。而且该方法还紧密地建立及依赖于测量和所收集到的可用数据。

一个在测试中使用统计学方法的组织能够量化质量信心级别和可信度（相关术语的更多信息，请参阅 TMMi 框架）。当使用统计测试时，通常将自信级别和可信度作为“完成定义”的判断标准。

5.3 过程域 5.3 测试过程优化

测试过程优化的目的是不断地改进组织中已有的测试过程，识别可能适合的新的测试技术（如，测试工具或测试方法），并以有序的方式将它们转换到组织中。测试过程优化还支持跨组织再利用测试资产。这些改进维护着组织的产品质量和测试过程性能目标，因为这些目标来源于组织的业务目标。

5.3.1 SG1 选择测试过程改进

收集并分析测试改进提案。测试改进提案来自内部来源，通常是跨团队、跨项目、跨产品流的。内部来源包括来自缺陷预防活动、团队评审、回顾、章程和协会的问题和想法。在对成本和收益进行分析后，有意向的测试改进建议将先被试行，以评估新的和未经证实的重大变更，然后再将它们恰当地部署到整个团队中。最后，对应用于整个团队的测试改进做筛选。

5.3.2 SG2 对新测试技术进行评估，以确定其对测试过程的影响

作为测试过程优化的一部分，集中组织的测试过程小组（测试协会）积极主动地在市场上寻找新技术，例如工具、方法、技术或技术创新，以提高敏捷团队的测试能力。通过保持对测试相关技术创新的意识，同时对其进行系统地评估和试验，组织与团队代表协商，选择适当的测试技术来提高其产品质量和测试活动的生产力。先在团队中对新的和未经证实的测试技术进行试验、评估，然后再将其纳入标准实践。

5.3.3 SG3 部署测试改进

测试改进和合适的新测试技术被部署在整个敏捷团队以改进测试。它们的有利之处被仔细斟酌，且关于新创新的信息在组织中传播。当然，在传统组织和敏捷组织中的部署是有所不同的。传统组织中的部署最经常带来的结果是所有相关人员都采用新的实践。而在敏捷组织中，团队拥有更多的自主权，是自我管理的。这也再次阐明两种组织模式的部署过程是不同的。测试改进和适当的新测试技术被酌情合并到组织的测试过程资产中，组织及提供培训和支持。然而，敏捷团队最终应该自行决定这些测试改进能否成为其工作方式的一部分。

5.3.4 SG4 建立高质量测试资产的再利用

在 TMMi 级别 3 中，一些跨团队、项目和产品流的测试件复用可能已经存在了，但在 TMMi 级别 5 中，测试资产的复用被视为一个主要目标。团队不该再闭门造车，而要利用现有的专业知识和资产，这样既省时又省力。可能被复用于整个组织中的高质量测试资产被识别，例如，在回顾会议上、课程学习期间、测试工会的讨论中，或测试评估中。随后，可复用的测试资产被筛选，并以一种可修改的格式添加到中心测试资产库中，以便在未来的产品交付过程和项目中得到复用。

6. 概述适用性 TMMi 特殊目标和实践

6.1 TMMi 评估

为了帮助主任评估师和评估师进行评估，本章对敏捷环境中过程域、特殊目标和特殊实践的适用性进行了概述。在前几章中，极少数特殊实践已经被识别出来很可能有很小的相关性，甚至可能不相关。下文列出了这些特殊实践。在 TMMi 评估期间，主任评估师和评估师需要明确地评估这些项在特定敏捷上下文中的相关性。如果明确某一项确实很少或甚至不相关，则应在本组织的过程中和在评估报告中以合理的理由将其清楚地记录下来。

6.2 TMMi 2 级管理

过程域 2.2 测试计划

SG2 建立测试途径	适用性
- SP2.3 定义入口准则	*很可能少适用或不相关*
- SP2.5 定义暂停和恢复准则	*很可能少适用或不相关*

SG3 建立测试估算	适用性
- S3.2 定义测试生命周期	*很可能少适用或不相关*
SG5 获得对测试计划的承诺	适用性
- SP5.2 协调工作和资源级别	*很可能少适用或不相关*

过程域 2.3 测试监督与控制

SG2 根据计划和预期监控产品质量	适用性
- SP2.3 监督产品风险	*很可能少适用或不相关*
- SP2.5 监督暂停和恢复标	*很可能少适用或不相关*

过程域 2.4 测试设计和执行

SG2 执行测试实施	适用性
- SP2.3 指定预测试规程	*很可能少适用或不相*
- SP2.4 制定测试执行日程表	*很可能少适用或不相关*
SG3 执行测试实施	适用性
- SP3.1 执行预测试	*很可能少适用或不相关*

6.3 TMMi 3 级定义

过程域 3.5 同行评审

SG2 执行同行评审	适用性
- SP2.3 分析同行评审数据	*很可能少适用或不相关*

6.4 TMMi 4 级已测量

所有 TMMi4 级的过程域、特殊目标和特殊实践都适用。没有特殊实践被标识为很可能少适用或不相关。

6.5 TMMi 5 级优化

所有 TMMi5 级的过程域、特殊目标和特殊实践都适用。没有特殊实践被标识为很可能少适用或不相关。

参考文献

[Black, Van Veenendaal] Rex Black, Erik van Veenendaal and Dorothy Graham (2012), Foundations of Software Testing ISTQB Certification (3rd edition), Cengage

[Cohn] M. Cohn (2009), Succeeding with Agile: Software Development using Scrum, Addison-Wesley

[Galen] R. Galen (2015), Three Pillars of Agile Quality and Testing: Achieving Balanced Results in your Journey Towards Agile Quality, RGCG, LLC

[Gilb] T. Gilb (2004), Agile Specification Quality Control, International Council on Systems Engineering (INCOSE)

[ISO25010] ISO/IEC 25010 (2011), Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) System and Software Quality Models, International Organization for Standardization

[Sutherland et al] C.R. Jakobsen, K. Johnson and J. Sutherland (2007), Scrum and CMMI level 5: The Magic Portion for Code Warriors in: IEEE Agile 2007 Conference, Washington, D.C.

[McMahon] Paul. E. McMahon (2011), Integrating CMMI with Agile Development – Case Studies and Proven Techniques for Faster Performance Improvement, Addison Wesley

[Musa] J. Musa (1998), Software Reliability Engineering Testing, McGraw-Hill Education

[Van der Aalst and Davis] L. van der Aalst and Cecile Davis (2013), TMap NEXT in scrum – Effective testing in agile projects, Sogeti Nederland B.V.

[Van Solingen and Berghout] R. van Solingen and E. Berghout (1999), The Goal/Question/Metric method, McGrawHill

[van Veenendaal] E. van Veenendaal (2014), PRISMA: Product Risk Assessment for Agile Projects, in: Testing Experience, Issue 04/12, December 2012

[Wake] B. Wake (2003), INVEST in Good Stories, and SMART Tasks, xp123.com/articles/invest-in-good-stories-and-smart-tasks