Active Listening 积极倾听

在沟通过程中，接收方需要收到、理解发送方所说的内容并给与反馈。

Affinity Estimating 亲和估计

快速估计大规模需求未完项的一种技术，利用衬衫尺寸、咖啡杯尺寸或斐波那契序列中的数字，将用户故事快速置于规模类似的群组中。

Agile 敏捷

基于敏捷宣言的一系列项目管理原则。敏捷强调自我组织型团队、客户合作、快速发布版本、响应变化、提升价值。

Agile Manifesto 敏捷宣言

2001 年签署的一份文档，给出了敏捷项目和敏捷方法论的指导原则。敏捷宣言由 4 项声明组成，阐述了各种敏捷方法论共同的价值观。

Agile Modeling 敏捷规模

过程或系统工作流的一种表示法，代码化前便于团队回顾。相对于代码，干系人及其他非程序开发人员更容易理解模型并用模型来工作。

Agile Space 敏捷协作

鼓励集中办公、紧急协作、面对面沟通、公开透明的团队协作方式。

Agile Theme 敏捷主题（参见主题）

Analysis 分析：通过研究问题及潜在需求找出可能的解决方案。

Artifact 组件：过程输出物或者工作产出物，典型形式为文档、图纸、模型或代码。

Backlog 未完项（又称需求待办列表，参考产品未完项或迭代未完项）

Iteration Backlog 迭代未完项：一次特定迭代所要完成的工作。迭代未完项预期在整个迭代过程中“燃尽”（不要和产品未完项混淆）

Product Backlog 产品未完项

所有已知的、即将通过项目实现的需求，无论是规划的迭代还是版本发布中包含的需求都算在内。

Brainstorming 头脑风暴

从群组中收集想法的一种方式，目标是在短时间内引发大量想法并激发出创新见解。采用头脑风暴时，参与者脑力震荡，快速抛出想法，在每个人讲完所有想法之前禁止进行评论或讨论建议。

Burn-Down Chart 燃尽图

迭代过程中及迭代结束时用户沟通开发进展的一种图表，图中显示出已完成和剩余的需求数。燃尽图的设计理念是工作未完成项会随着项目的推进而不断减少或是“燃尽殆尽”。

Burn Rate 燃烧率

燃烧率是敏捷团队消耗的成本，或是团队的资源消耗率，最常见的计算方式是把各项团队成本简单叠加在一起，一般用每次迭代成本、每周成本、每月成本或其他对执行组织有意义的方式来表示。如果团队每周“燃烧”的成本是 15000，那么可用 15000/周来表示该团队的燃烧率。

Burn-up Chart 燃起图

与燃尽图相反，燃起图展现了时间与已完成功能之间的关系。随着需求不断完成、价值不断累积，图中的工作进展走势也不断上升。燃起图并未显示过程中工作，因此它无法用来精确预测项目的结束时间。

Capability 能力（参见史诗故事）

Epic story 史诗故事

非常大型的故事，可以横跨多个迭代周期。史诗故事在战术层面上使用前必须分解为一个个相关的用户故事。

Cause and Effect Diagram 因果图（参见根本原因图）

Root Cause Diagram 根本原因图

别名石川图、鱼骨图和因果图的一种图表。根本原因用于说明不同因素如何相互关联，同时识别出根源问题。

Ceremony 仪式

敏捷项目召开的例行会议，例如迭代计划会议、每日站立会议、迭代评审会议，以及迭代回顾会议。

Change 变更

变更在敏捷项目中通常指需求变化。敏捷拥抱需求变化，即使出现在项目后期，也将其视为团队可以提供给用户的一种竞争优势

Charter 章程

标志项目正式开始的文档。项目章程制定于项目启动阶段，通常包含批准项目的原因、总体预算、主要里程碑。关键成功因素、约束条件、前提条件、以及允许团队开始工作的授权

Chicken 鸡

参与敏捷项目的人员，但非全身投入。鸡组成员不应成为核心团队成员，但可以提供意见和信息。参考“猪”

Pig 猪

全身心投入敏捷项目并受项目结果影响的人员。

Coach 教练

在极限编程（XP）方法论中，教练是保持团队聚焦于学习及 XP 过程的人员。教练是 XP 价值观的体现，帮助团队不断提升并交付价值。

Collaboration 合作

为共同目标一起工作。

Collection Code Ownership 代码集体所有

整个团队所有人对全部代码负责的一种环境。这意味着团队的每位成员都可以维护、修改其他成员的代码。代码集体所有不鼓励专业分工和特立独行。

Colocation 集中办公

整个团队集中在一个房间一起工作。

Command and Control 命令和控制

非敏捷的一项原则，由组织结构图中的高层人员做出决策并逐级下达给团队。

Communication 沟通

信息共享。在敏捷团队中，信息沟通应该是透明和自由流动的，整个团队应该清楚的意识到项目各方面所发生的事。

Compliance 合规

符合规定。合规是项目批准启动的理由之一。

Cone of Silence 静锥区

为一位或多位团队成员营造的不易分心和被打扰的环境。

Cone of Uncertainty 不确定性锥区

项目早期由于很多信息未知导致估算困难，不确定性锥区是描述这项困难及如何逐渐改善的术语，它表明如果接近工作启动在进行估算，能得到更为精准的估算结果。

Conflict 冲突

团队意见分歧的领域。适当的冲突是良性的，且为敏捷项目所鼓励的，因为这些冲突会引发流程改进并开发出更高质量的产品。

Conflict Resolution 解决问题

当冲突发生时，协商出一个各方面都能接受的解决方案。

Continuous Integration 持续集成

定期核查每位团队成员工作进展并进行整个系统编译和测试的开发实践。最严格的做法是每天以迅速找出可能引入的系统错误为目标进行操作。

Coordination 协作

团队成员为了达到生产力更高及团队合作的目标而一起协同工作。

Cumulative Flow Diagram 累积流量图

展示功能未完项、过程中工作及完成功能与时间关系的一种图表，是信息发射源的组成部分。

Customer 客户

真正的客户或是对商业价值进行定义和排优先级的客户代表。客户是敏捷团队的组成部分。

Cycle Time 循环时间

开发完一项需求或是一个用户故事所需花费的时长。

Daily Stand-up Meeting 每日站立会议

通常是每天工作开始时召开的简短例会，整个团队成员均需参加并简要回答 3 个问题：你昨天做了什么，你今天计划做什么，以及，你是否遇到什么障碍。每日站立会议对沟通交流和尽早发现问题很重要，绝大多数敏捷方法论都将会议时长严格限制在 15 分钟。

Decide As Late As Possible 尽可能晚决策

敏捷的实践方法，尤其体现在精益中，意思是只要有责任保留所有可能的途径就尽量推迟决策、同时基于尽可能多的已知情况做决策。

DEEP

描述产品未完成项理想属性的缩略语，含义是：详略适宜的（Detailed Appropriately）、可估计的（Estimatable）、涌现式的（Emergent）、排好优先级的（Prioritized）。

Disaggregation 解聚

将史诗故事或大型故事分解成小型用户故事。解聚类似于传统项目的分解。

Documentation 文档

敏捷宣言认为，文档的价值低于工作软件。敏捷从业者通常认为文档“刚好够”正合适。

Done 完成

需要明确定义并被整个团队一致认同的术语。定义什么叫完成很重要，这样当某位团队成员说他“完成”了一件工作时，每位团队成员的理解才能完全相同。

DRY

“不要重复你自己（Don t Repeat Yourself）”的缩写。DRY 强调了要最大限度去做尚未完成的工作。例如，“这个模块不太 DRY”就表示代码与已完成部分有重复，需要重写，不能直接利用该模块。

Earned Value Management（EVM）挣值分析

在当前点衡量和沟通项目进展及变化趋势的一种方法。如要使用，挣值管理最适用于敏捷项目的迭代级别。

Emergent 涌现式的

是缩略语 DEEP 的组成部分，表示用户故事完成后产品未完项条目也会随着项目的进行而逐渐增长和变化。

Emotional Intelligence 情商

与他人交往和影响他人的能力。情商与传统的智商测量并无直接关联。团队负责人与团队打交道时，情商是一种重要的领导技能。

Empowerment 授权

敏捷团队的必要属性。在敏捷项目中，授权的概念是指团队能够做出必要的决定来增加交付价值。传统项目则相反，典型做法是必须征得上级同意才能做决定，或者多数情况下上报，由上级来做决定。

Engineering Task 工程任务（参见任务）

Task 任务

也成为工程任务 ,是对工作一种较小的划分 ,由单人承接 ,而且通常能迅速完成。任务是比用户故事更小的一种划分，而且不像用户故事，任务可能会交付价值给用户，也可能不会。

Escaped Defects 漏网缺陷

产品由团队交付给客户后由客户报告发现的缺陷。漏网缺陷数量的激增很可能表明团队存在流程方面的问题。

Extreme Persona 极端人物

一个团队制造出的及其夸张的人物，目的是引发一般人可能考虑不到的需求。

Persona 人物

团队创造的一个虚拟人物或身份 ,用来模拟与系统的交互 ,以便收集需求。例如，如果团队创建了一个销售系统点 ,那他们可以虚构一个叫“小明”的人物来代表如下行为的用户，及买下一大堆物品，然后再当天稍晚全部退货并获记账存款。

eXtreme Programmaing（XP）极限编程

一种高度自律的敏捷方法论，以一周迭代周期及程序员结对编程的方法运行。XP 定义的角色有教练，客户、程序员、跟踪员和测试员。

Feature 特性

在敏捷社区中意思多变的一个单词，在 PMI-ACP 考试准备过程中可以当成故事的同义词。

Fibonacci Sequence 斐波那契序列

常用于敏捷估算中的一种数列，以 0，1 开头，将前面两位数字相加得到下一位数字，从而得到下述序列：0,1,2,3,5,8,13,21,34,55,89,144……斐波那契序列用于计划扑克或类似估算方法时，通常会简化为 0,1,2,3,4,5,8,13,20,40,100……

Fishbone Diagram 鱼骨图（参见根本原因图）

Ishikawa Diagram 石川图（参见根本原因图）

Root Cause Diagram 根本原因图

别名石川图、鱼骨图和因果图的一种图表。根本原因图用于说明不同因素如何相互联系，同时识别出根源问题。

Five Whys Technique 五问法

丰田公司流行的一种根本原因分析方式，连续不断的问五次“为什么”，抓住深层问题一次比一次更深入的发问。

Focus 专注

极为重要的团队纪律，推动通过诸如每日站立会议、敬业的团队协作、信息发射源和闲置过程中的工作等敏捷方法来实现。多数敏捷从业者认为推动和保护团队专注工作是教练、敏捷教练或 ScrumMaster 的工作。

Force Field Analysis 立场分析

对推动和阻碍潜在/真实变革的力量及其力度进行分析的一种技术。

Functionality 功能

在敏捷语境中，系统为客户或用户增加交付价值所执行的一个操作。如果用户无法看到或感受到什么，那就不能算功能。

Grooming 梳理

通过不同的活动来清理产品未完项，如删除条目、分解或进行估算。

Ground Rules 基本规则

适用于全体团队成员的不成文规定，应该与团队中每位成员进行沟通。举例：无需逐个询问，可以预期团队中的每位成员早上 8 点会集合召开每日站立会议就是一项基本规则。

High-Bandwidth Communication 高带宽沟通

面对面的沟通，称之为高带宽是因为很多信息通过肢体语言、身体语言、面部表情和语音语调来传送。高贷款沟通是首选的项目沟通方式。

Ideal Time 理想时长

指如果没有什么事干扰或引起分心的话，完成一项任务所需耗费的时长。一些敏捷项目使用理想时长而非实际时长来进行估算。

Incremental Delivery 增量交付

敏捷的理念，指产品应该分成一个个小小功能陆续交付给用户，而不是等到所有功能全部开发完毕整理交付给客户，这样客户在产品演进过程中就能及时施加影响，且随着产品开发而不断学习，并最终收益。

Information Radiator 信息发射源

团队与其他干系人之间用来沟通项目状态的一组组件。信息发射源对保持团队工作进展的透明度及能见度很重要。

Information Refrigerator 信息冷冻机

与信息发射源意思相反的术语，表示必须开发和探索才能理解的不可见图表。

Innovation Game 创新游戏

从产品负责人、用户及干系人引出需求的一组练习。创新游戏采用一种更新颖的方式来收集需求，有助于框定提出需求的过程。

Interaction 交互

在敏捷语境中，通常指团队成员、客户以干系人之间面对面的交谈。敏捷宣言阐明了“个人和交互”比“流程和工具”更受推崇。

Internal Rate of Return（IRR）内部收益率

一种以赚取的利率来表达盈利的方法，有助于组织衡量各种可选投资的收益回报。IRR 指标越大越好。

INVEST

描述一个好的用户故事理想属性的缩略语，代表了独立的（Independent）、可协商的（Negotiable）、对用户或客户有价值的（Valuable）、可估计的（Estimatable）、短小的（small）、以及可测试的（Testable）。

Iteration 迭代

敏捷项目中重复的工作循环。迭代通常包含一个简短的计划会议，然后是一个时期的工作，最后是回顾会议，评估流程和结果并作相应调整。

几个迭代可以组合形成一个版本发布，同时迭代可以在项目过程中重复多次。敏捷原则指出迭代周期持续时间从 2 周到 2 个月不等，而极限编程方法论则把迭代周期压缩到 1 周。

Iteration Backlog 迭代未完项

一次特定迭代所要完成的工作。迭代未完项预期在整个迭代过程中“燃尽”（不要和“产品未完项”混淆）。

Iteration Retrospection 迭代回顾

一次迭代结尾召开的会议，团队成员在会上讨论已经完成的工作及如何在下次迭代中进行改进。迭代回顾会议的焦点是流程改进。

Kaizen 改善

一种日式管理哲学，指持续改进。项目启动后，通常完成一部分就交付，改善倡导由团队发起小规模频繁的变革。

Kanban 看板

一种日式管理哲学，字面意思是“信号”。看板主要推动过程中工作（WIP）的可视化及对团队允许的过程中工作的数量进行限制。

在一个典型的看板环境中，团队每位成员正在进行的工作都会显示在看板面板上，看板面展示了工作及其完整的流程阶段。团队成员只有先完成手头上的工作才能承接下一项任务，此谓限制过程中工作（Limiting WIP）。

Kanban Board 看板面板

显示过程中工作（WIP）的一个组件。看板面站展现了各阶段工作流（如启动、设计、编码、测试、测试和完成）及每个工作流中所含任务。团队成员及感兴趣的干系人只要扫一眼就能了解团队正在进行的工作及目前处于哪个阶段。看板面板鼓励可视化和对过程中工作进行限制。

Last Responsible Moment 最后负责时刻

指团队应该尽可能晚决策，并保留所有选择尽可能时间长的术语。

Lean 精益

基于以下 7 项原则的一种敏捷方法论：消除浪费，增强学习，尽量延迟决策，尽快交付，授权团队，构建完整性和全盘检视。精益寻求最大化过程中工作（WIP）的可视性，是一种高度自律的方法论。

Mataphor 比喻

采用比喻式意味着用真实世界中的事物来描述系统组件，从而帮助没有技术背景的干系人来理解问题和解决方案。极限编程（XP）方法论尤其倡导使用比喻方式来沟通。

Methodology 方法论

规定项目应该如何计划、执行及控制的一组特定的做法、流程和组件。

Minimal Marketable Feature（MMF）最小可售功能

能为用户增加价值的最小单位的交付物。典型的单个最小可售功能包含一组用户故事。通过将项目拆分为若干个 MMF，团队可以专注于开发一组有价值的小功能，并迅速的交付给客户。

Negotiable 可协商的

是缩写 INVEST 的一部分，INVEST 描述了一个好的用户故事应有的各种属性。可协商表示用户故事不是一成不变的，可是具有可塑性的，再从记事卡片转变为工作软件的过程中是可以被质疑和修改的。

Net Present Value（NPV）净现值

计算项目价值时把货币的时间价值作为因素计入的一种方法。净现值计算公式中还会考虑项目成本。

Osmotic Communication 渗透式沟通

以聆听方式发生的沟通。一位团队成员无意中听到作战室中另外两位团队成员交谈并知晓相关信息就是渗透式沟通的一个实例。

Pair Programming 结对编程

极限编程方法论所倡导的一种实践方式，开发人员两人一组一起工作，一个程序员“敲键盘”写代码，另一个旁观其操作，以不同的视角参与编程。理想情况下，结对编程比单兵作战效率更高。

Parking Lot Chart 停车场图

主要在需求收集活动中使用的一种图表，用来叫停任何可能游离当前目标的会话（或用来叫停任何与当前议题无关的会话）。某位干系人提出的一个问题可能很重要，但不一定与当前议题相关，可以把这个问题“停放”一旁暂停搁置（即记录在停车场图上），稍后再做商议。活动结束时应该重温一遍停车场图上的所有事项。

Plan-Do-Cheak-Act（PDCA）PDCA 循环（计划-执行-检查-调整）

由戴明提出并普及的工作循环，提出工作应该按照计划、执行、检查、调整的顺序循环进行。相比传统项目，敏捷项目以规模更小、速度更快的迭代方式来实现 PDCA 循环。

Planning Game 计划游戏

创建版本发布计划的一种手段。团队会估算开发每个需求所要花费的工夫，利用上述信息，客户综合考虑商业价值和所花工夫/成本后排出需求的优先级。

游戏与“博弈”在英文中同词，“博弈”是经济意义上的一个术语，指玩家需要基于其他玩家的决定来做出的决定和进行取舍的一种游戏。

Planning Poker 计划扑克

遍布于敏捷项目的形式多样的一种训练。传统方式是团队成员先各自估计开发某项用户需求所需花费的功夫，并在索引卡上写下估算结果。然后所有团队成员同时翻开各自的索引卡，接下来的讨论将集中估计结果中的高值和贬值，目的是为了弄清为何团队成员认为这项需求开发特别困难或特别容易。接着团队开展讨论，需要的话重复上述过程，直到所有成员的估计结果达到相对一致。估计结果可以是小时数、天数和理想天数，或诸如 XS，S，M，L 或 XL 这样的近似估计，或是斐波那契序列中的点数。

PMI-ACP 美国项目管理协会—敏捷管理专业人士资格认证

美国项目管理协会推出的敏捷管理专业人士资格认证，旨在被认证者提升敏捷实践水平，展示敏捷专业知识。

Present Value（PV）现值

计算项目价值时把货币的时间价值作为因素计入的一种方法。将一个潜在投资项目或投资机会机会和另一个进行比较时，现值计算是非常有用的方法。

Process Tailoring 过程裁剪

对敏捷过程进行提炼，以适应项目或环境需求。过程定制基于下述原则：过程应该服务于项目而不是相反。在项目的整个生命周期中，敏捷过程定制会重复进行多次。

Product Backlog 产品未完项又称产品需求列表

所有已知的、即将通过项目实现的需求，无论是规划的迭代还是版本发布中包含的需求都算在内。

Product Owner 产品负责人

敏捷项目中的角色，代表客户、用户和干系人，理解并倡导潜在的需求的总体商业价值。产品负责人负责维护产品未完项，并在每次迭代计划会议上牵头第一部分的讨论。

Product Roadmap 产品路线图

显示当前产品功能及/或规划产品功能概况的一个组件。产品路线不如发布计划那么详细。

Programmer（XP）程序员（XP）

极限编程方法论中定义的一个角色，以结论的方式进行工作，一人写代码，另一人旁观并给出意见建议。

Progressive Elaboration 渐进明细

计划工作并非堆积在前期，而是周期性发生的一种迭代方法。采用渐进明细方式 ed 项目的典型做法是，计划一部分、执行一部分、监控一部分，然后重复上述周期。敏捷项目是高度的渐进明细型项目。

Project 项目

为创造独特的产品、服务或成果而进行的临时性工作。

Qualitative 定性的

无法测量但仍会影响产品质量的描述性因素。

Quality 质量

符合规范或要求。

Quantitative 定量的

影响产品质量的可（量化）测量因素。

Refactoring 重构

对工作代码进行重组，使得代码结构与相应功能保持一致，同时更易于维护。重构不应对产品功能或性能产生明显影响。

Relative Sizing 相对规模估计

基于另一个功能或者用户故事的规模来估计当前的规模。不进行实际的时长估计，而是将一组用户故事从难度最高到最低进行排序，是相对规模估计的一个实例。

Release 版本

一组封装的迭代结果，旨在交付给最终用户和客户。每个版本交付的工作软件被期望是高度稳定的。

Return on Investment（ROL）投资回报率

指组织得到的回报与投资总额的百分比。

Risk 风险

任何会给项目带来正面或负面影响的不确定性因素。

Risk Burn-Down Chart 风险燃尽图

一种图表，上面罗列了影响项目成功且与每项需求相关的各种风险。随着项目不断推进和功能完成开发，项目风险将会不断减少或“燃烧殆尽”。

Role 角色

个人与敏捷项目打交道的方式。敏捷项目的角色包括客户、产品负责人、干系人、团队成员、测试员、跟踪员、敏捷教练或 ScrumMaster，以及教练。

Root Cause Analysis 根本原因分析

一种分析问题的技术，通过透过问题表象来了解引起这些现象的根本问题。根本原因分析有时需要借助根本原因图和五问法来进行。

Root Cause Analysis 根本原因图

别名石川图、鱼骨图和因果图的一种图表。根本原因图用于说明不同因素如何相互关联，同时识别出根源问题。

Scrum

由 Jeff Sutherland 和 Ken Schwaber 开发的一种流行的敏捷方法论，主要角色包括敏捷教练、ScrumMaster、产品负责人和开发团队，同时规定了团队召开每日站立会议的实践方式。Scrum 迭代称之为冲刺。Scrum 由可视性、检验和适应性三大支柱所支撑。Scrum 冲刺之前先要召开计划会议，冲刺完成后还要召开回顾会议

Scrum Of Scrums Scrum 合作会议

多个 Scrum 开发团队参加的会议 ,参会人员通常是 ScrumMaster 或特派代表。Scrum 合作会议讨论各团队的开发进展，并协调多个团队之间的工作。该技术经常用于超大规模 Scrum 项目 ,对于这种项目其开发团队必须细分为较小团队。

SDLC

系统开发生命周期（System Development Life Cycle）的缩写。SDLC 是一种非敏捷的、瀑布式开发方式，规定项目应该以对前期计划浓墨重彩的长周期方式来进行。

Self-Organization 自我组织

敏捷理念之一，指团队不应该被管头管脚或指手画脚，而应该以更有机的方式来构成。

Servant Leadership 服务型领导

敏捷领导力原则之一，指出领导人角色（如教练或 ScrumMaster）采用服务团队的方式进行领导效果最好。服务式领导奉行己所不，欲勿施于人。

Silo 孤军奋战

单人或小团体以一种孤立的，与外界极少交互的方式进行工作的状况。相比孤军奋战，敏捷项目青睐更为开放的和透明的沟通方式。

Smells 气味

通常会影响敏捷团队和敏捷项目的问题。这些问题被形象的归位气味，表明面临的状况似乎有些不太对劲。

Spike 探针

Scrum 项目持续一周到一个月的一次迭代。

Stakeholder 干系人

任何与项目利益相关的人，无论影响是积极的还是消极的。

Stakeholder Management 干系人管理

让干系人知晓项目最新进展。与干系人进行沟通并满足他们要求的过程。干系人管理始自正确识别干系人。敏捷项目重视干系人的参与，同时利用干系人的专长和精力为项目服务。

Standardized Test 标准化测试

同一位参试人员每次考试结果都相差无几的一种测试。标准化测试的目标是形成这样的一条结果曲线，即只允许预定百分比的参试人员通过测试。标准化测试的目的是衡量知识掌握程度和理解程度。

Story Card 故事卡

写有用户故事的一种索引卡。采用索引卡的形状是为了限制细节数量和提前计划团队如何应对。

Story Map 故事地图

一组未交付、积压的故事，可以根据用户功能来拆分和组织，目标是设定正确的开发优先级。

Story Point 故事点

表示用户故事难度估计（所费功夫）的测量单位。故事点可以用小时数、天数、衬衫尺寸（XL，S，M，L，XL，XXL）或斐波那契序列中的数字来表达。

Sustainability 可持续性

可无限期维持的一种团队节奏或速度。敏捷团队要避免再一次迭代或版本发布末尾出现时间紧迫的情况，相反要尝试保持一种紧张但平稳的节奏。

Swarming 蜂拥式

整个团队专注于单个用户故事的一种敏捷合作技术。蜂拥式可用于整个产品未完项或仅用于单个有难度的故事。

Task 任务

也称为工程任务 ,是对工作一种较小的划分 ,由单人承担 ,而且通常能迅速完成。任务是比用户故事更小的一种划分，而且不像用户故事，任务可能会交付价值给客户，也可能不会。

Team 团队

经过授权的个体组合，共同为项目向客户交付价值而负责。

Technical Debt 技术债务

团队选择目前暂时不实施的技术决策，但如果一直不做最终会成为阻碍。例如，团队开发代码时选择走捷径，即使这条捷径并不符合组织规定的编码标准，这样做就会产品技术债务。

Test-Driven Development 测试驱动开发（测试先行开发）

一种有利于敏捷的开发实践，先对模块的验收测试进行定义，在编写模块代码。这样代码将围绕着通过这些测试而构造，只要写对代码就必然应该通过测试。

Tester 测试员

极限编程方法论定义的角色，帮助客户定义验收测试，然后周期性检查产品是否通过验收测试，并就测试结果和相关人员进行沟通。

Theme 主题

一组故事、一次迭代或一个版本背后的主要目的。由产品负责人或客户来确定主题 ,同时得到团队的一致同意。例如 ,某位产品负责人确定某次迭代的主题为 “报

告”或“连贯性”，这种情况下为这次迭代所选择的的用户故事都要围绕该主题展开。

Timeboxing 时间盒

设定一个固定的交付日期来约束项目或版本发布然后在进度允许的情况下实现尽可能多的交付价值和功能开发。

Tracker 跟踪员

极限编程方法论中定义的角色，在项目中衡量团队的工作进展并据此进行沟通，团队的跟踪员会跟踪迭代计划和发布计划及测试工作进展，通常会将跟踪结果公布于信息发射源。

Traditional Management 传统管理模式

敏捷语境中的一种参照，指自上而下或瀑布式的项目管理模式，强调详细规划、执行和控制，开发周期较长，通常客户较少介入团队工作。传统项目中，团队是被动管理型，而非自我组织型。

Transparency 透明度

敏捷项目的价值观之一，体现方式是展示团队每位成员手头的工作，同时每人的工作进展让团队其他成员都能看见。

User Story 用户故事

一个或多个用户增加价值的商业需求。撰写用户故事所需的工作量相对较小。典型做法是将用户故事记录在故事卡。用户故事根据商业功能而非使用专业术语来描述。例如，团队需要优化数据库不会定性为一个用户故事，除非用户要求提升技能，而这是数据库优化可能导致的结果。用户故事即使写下后仍然是易变的（可协商的）。

Validation 确认

确保产品会被客户所接受的过程。

Value 价值

寻求如何为客户增加价值是多数团队决策的驱动力。

Value Stream Mapping 价值流程图

以消除浪费为目标的分析一连串流程的方法，目的是通过映射使分析师更深入的了解系统。价值流映射法是精益的一种技术，用来分析系统中的材料和信息流转，并识别和消除其中的浪费。

Value-Based Prioritization 基于价值的优先级

让产品负责人或客户基于功能的交付价值来决定哪项功能先开发的一种做法，同样也应用于项目论证概念中。

Velocity 速度

一次固定迭代中团队可以交付的需求或用户故事数。

Verification 核实

确保产品符合规范的过程。

Virtual Team 虚拟团队

地理上分散在各处的团队。虽然敏捷项目偏好团队集中办公，但很多时候虚拟团队的方式更切合实际，虚拟团队使得项目沟通及其很多方面变得更加困难。

Visibility 可视性

每位团队成员的工作及进展应该对所有利益相关的干系人保持透明的理念。工作及进展应该公开展示，让每个人都能看见。

War Room 作战室

整个团队聚焦于专属空间一起工作的一个地方。作战室有助于促进沟通，形成团队意识，以及避免出现孤岛。

Waterfall 瀑布式方式

一种传统的、非敏捷式的管理哲学，偏好项目前期进行详细计划，紧接着是很长一段执行和控制期。瀑布式开发方法论以抗拒需求变化而知名。

Wideband Delphi Estimating 宽带德尔菲估计法

团队成员聚在一起演示用户故事，讨论面临的挑战，然后私下进行估算的一种估计技术。每个故事的估算结果都会被匿名标注在图表上，然后团队就故事点范围进行讨论，并尝试达成普遍共识。

WIP Limits 过程工作限制（半成品限制）

对过程中工作（WIP）进行限制，这样团队就能集中精力完成开发、保质保量和交付价值。

Wireframe 线框图

一种轻量级的、非功能性的用户界面设计，展示了主要的界面元素及其如何交互。团队无须编写代码，通过线框图就可以传递给用户有关系统如何工作的概念。

Work-In-Progress（WIP）过程中工作（在制半成品）

已着手进行开发的需求或任务。WIP 通常会在信息发射上公开展示，其进展也会随着工作流的进行而同步显示。

Workflow 工作流

开发过程中一系列一致同意且被团队遵循的工作阶段。例如，如果某团队决定每项任务都要经过下述阶段：启动，设计，编码，测试，集成，和完成，那么这一系列阶段就代表了该团队的工作流。