敏捷读书之用户故事：《用户故事与敏捷方法》解读

本文分三个层次解读用户故事。

1. 企业基因：目标，人和流程的3P模型。
2. 用户故事细探。
3. 客户参与，产品探索和产品交付的全流程。

以下分别展开。

企业基因：目标，人和流程的3P模型

一个企业生存发展的内在支撑要素是它的基因。企业基因体现在目标，人和流程三个方面。敏捷和用户故事倾向于与之匹配的企业基因。这种基因通过敏捷工作方式的采纳而得到强化，从而成为企业的核心竞争力。

企业的目标是什么

这是一个很大的话题。在本文的范畴中，把它收窄到一个问题，即关注点在输出的软件产品本身。人和流程都是围绕着软件产品本身这个目的。通过用户故事这一简单的物件，结合敏捷方法，消除浪费，减少管理开销，减少与软件产品无关的中间物件。用户故事帮助到以目标为中心这一点。

复杂性与人

软件的复杂，导致需要多种角色。而多种角色的存在，又加剧了做事的复杂。

不同角色有不同的角度和不同的诉求。项目经理想跟踪进度。开发人员想实现系统。产品经理想要灵活性。测试人员想要度量。用户想要一个可用的系统。

经验表明，一旦任何一方在沟通中把持绝对地位，项目就会遭损失。我们需要一种协同工作的方法。在这种方法中，以目标驱动，同时通过简化的流程，让各方都能充分表达，发声，影响和贡献。

在这种协同工作的方法中，用户也是重要的参与者。在迭代与增量开发模式之下，当用户看到软件的早期版本，他们会想出新的点子。理想状态是，客户要在产品开发的整个

过程中参与。

我们可以把团队分为三个虚拟团队。

1. 客户团队：产品负责人，销售人员，客户。
2. 产品探索团队：产品负责人，架构师，用户体验设计师，开发人员。
3. 产品交付团队：产品负责人，开发团队，Scrum Master。

三个虚拟团队的参与方式，在本文第三部分全流程的介绍中详解。

不能直接与用户接触的团队，需要与用户代理合作。用户代理包括：

用户的经理：

对于内部系统软件，有时候用户的经理会是用户代理。在层层汇报层层夸大的组织当中，用户的经理有可能是错误信息的来源。

开发经理：

开发经理的诉求有可能是想提早给人介绍令人兴奋的新技术，或想提前完成项目以获得奖励。

销售人员：

销售人员可能会急功近利，关注哪些如果没有的话可能导致丢单的故事。有可能的话，让他们把你（产品负责人）介绍给客户。

领域专家：

领域专家的意见，可能导致软件只适用于跟领域专家有类似水平的用户使用。

市场营销团队：

他们可能会关注数量，轻视质量。他们的贡献，可以是指导优先级。

以前的用户：

需要了解他们的目标和动机是否和现在的实际用户一致。

客户：

客户是购买产品的人。用户是使用产品的人。两者可能不一致。

培训师和技术支持人员：

他们的意见，可能导致一个容易培训和容易支持的系统。

业务分析师或系统分析师：

有可能太过自信，喜欢空想而不是现场调查。倾向于喜欢花太多时间在前期工作如角色建模和故事编写工作坊。

开发人员。

由开发人员冒充用户，可能会开发出炫而无用的产品。

让各方都有全局观，需要组织文化的变革。而用户故事这个小小的物件及围绕它的基于合作的工作方式，可以平衡各方的声音，让产品探索和产品开发的效果更好。

最好的方法是让团队直接接触用户。一个捷径是：尽早发布产品，用产品与用户对话，从而打开与用户沟通的途径。

用户故事与敏捷方法，强化了全流程中各方的合作。

复杂的事和复杂的人导致了复杂的流程，而世界的本质是简单的

在传统的方法中，解决复杂性问题的方法是用复杂的流程，而复杂的流程又加强了系统的复杂。

在用户故事和敏捷方法中，有两个主要的方式化繁为简：

- 通过用户故事这个简单的物件统摄始终，贯穿需求和解决方案空间，创造价值拉动和流动。
- 打造和利用一个快速获取信息和反馈的过程，越早越好，越频繁越好。

用户故事细探

从两个问题开始

做产品始于两个根本问题：

- 用户想要达到的目标是什么？
- 达到这个目标的成本是多少？

为了回答这两个问题，我们需要把大目标分解为小目标，排优先级和估算。

用户故事是这个过程中使用的基本单元。比用户故事大或者模糊的可以叫做史诗，主题或者只是一个产品点子。

用户故事的一些重要特征

- 一目了然格式一致的表达。用户故事有两个部分：描述部分和验收标准。描述部分的常用格式是：As <用户>, I want <功能>, so that <目的>。验收标准可以有多条，常用的格式是：Given <前提条件>，When <动作>，Then<结果>。
- 鼓励推迟细节，只需有足够的信息以使项目前行。我们不需要一次性把项目的所有需求搞清楚，我们只需要在迭代之前把一个迭代要做的事搞清楚即可。而以用户故事作为基本单元，可以支持这种开发模式。
- 用户故事以合适的颗粒度，方便理解，方便排优先级，保证重要的事先做。随着时间的推移，不重要的事也许就不需要了。
- 用户故事鼓励通过交谈了解细节。强调对话而不是书面沟通。
- 用户故事的验收标准保证成果可以被审核验证。
- 以用户故事为载体，促进结构化沟通，使谈话有落地点。
- 从业务角度描述，可以同时被业务人员和开发人员理解。
- 用户故事的大小适合估算和做计划。颗粒度适合迭代开发。
- 支持随机应变的开发，检视和适应。
- 鼓励各方参与交流，传播隐性知识。

用户故事的3C，鼓励通过交谈了解细节

- Card卡片：用户故事通常写在卡片上，描述用户想要达到的目标，突出对用户有价值。
- Conversation对话：书面文档是一种低带宽沟通，表达信息的能力受限，接受信息的体验受限，信息传递能力受限。面对面交流是高宽带沟通，利用检验，适应和反馈促进沟通的有效性。用户故事模式鼓励对话。无形的对话为有形的卡片补充细节。
- Confirmation验收标准：无形的对话再次落实到有形的验收标准，并成为故事的一部分。

排优先级要考虑的要素

- 大部分用户对该功能的渴望程度。
- 少部分重要用户对该功能的渴望程度。
- 故事之间的关系，是否有开发的先后顺序。
- 技术实现的难度。

- 成本高低。

关于优先级的两个模型：

MoSCoW莫斯科定律：Must have必须有，Should have 要有，Could have 可以有，Woudn't have不会有。

Kano模型：没有不行的功能，线性功能，尖叫功能。

用户故事的INVEST准则。

- Independent: 用户故事之间是独立的。如果不独立，可能是划分的方法有问题，可重新划分。
- Negotiable: 用户故事是可讨论的。使用卡片是为了提醒对话。讨论的细节会变成测试。
- Valuable: 用户故事对用户有价值。有条件的话，让用户写或确认用户故事。
- Estimatable: 用户故事是可估算的。不可估算的原因可能是：缺乏领域知识（多与客户讨论，学习对方的语言，了解对方的想法，事实往往比想象的简单），缺乏技术知识（可以把故事分为两部分：探针实验spike，和实际开发），故事太大(分拆)。
- Size appropriately: 大小适中。故事的合适大小取决于团队，容量和使用的技术。

  需要分割的故事，可能是复合故事（按CRUD分拆），或复杂故事（可以分探针实验spike和实际开发两个故事，与其他故事一起排排优先级，然后在不同的迭代完成探针实验spike和实际开发）。
  需要合并的故事，包括若干小bug，或小改动。
- 用户故事是可测试的。故事的描述需使用具体的量化的描述，而不是绝对的含糊的描述。

用户角色建模，以用户为中心

用户角色建模的BSCR步骤：

1. Brainstorm: 产品探索团队头脑风暴出用户角色。
2. Sort: 对头脑风暴出的用户角色快速分组和识别层级。
3. Consolidate: 对每一角色逐一讨论，整合角色。
4. Refine：提炼角色。识别用户角色的：
   - 使用频次。

- 领域知识。
- IT知识。
- 使用目标。

用户建模的其他技术：

1. 典型用户画像。针对一种角色，虚构出一个人物，描述他的行为特征，使用场景，核心诉求。还可以给出画像。打印出来放在团队可见的地方，让团队在开发产品时对用户的需求能感同身受。
2. 极端用户。想象一个极端用户会有什么诉求，以产生新的灵感。这方面不用太用力。

搜集故事的方式

1. 用户访谈：不设定立场，问开放式问题和背景无关问题。
2. 问卷调查：问卷可简单直接，比如，了解用户使用软件功能的频率。
3. 直接观察用户如何使用产品。
4. 产品探索团队使用故事编写工作坊，头脑风暴，输出简单原型和用户故事。

验收测试

验收测试贯穿在产品开发的整个过程中，有两步流程，保证所花的时间和精力刚刚好：

- 记录测试要点。时机包括：和客户讨论故事并记录细节。有条件的话，让客户定义测试。
- 将测试要点变成测试。时机包括：迭代开始时，把写测试作为写代码之前的一项专门任务，以

保证开发的功能完备。在开发中后任何时候发现新的测试。测试人员要按照需求去测试，由需求和价值驱动，而不是按照程序员的描述去测试。产品负责人的业务知识和测试人员的怀疑精神是为测试做贡献的要素。有条件的话，让用户参与验收测试，但可以设计合适的测试表格以方便用户参与。

测试类型包括：功能测试，组件交互测试，易用性测试，性能测试，压力测试。

测试追求的是满足需求，捕获故障，提升质量，而不是覆盖率。测试覆盖率可依据知觉，知识和经验来判断。

用户故事的一些其他原则

- 从目标开始，大目标分解成小目标。
- 切蛋糕式划分故事，竖切而不是横切。
- 故事的描述要封闭，有完成感。故事针对一个场景，一个画面，一次完整的动作。
- 约束也可以作为一个故事。
- 故事的冰山法则：近期要开发的要划分的细而清晰，远期开发的可粗而模糊。
- 文档也可以作为故事。
- 故事中要包含用户角色。
- 一个故事只包含一种用户。
- 为了表述简单，故事使用主动语态。
- 有可能的话，让用户编写故事。
- 故事不用编号。可提炼出作为标识的关键词以方便讨论时引用。
- 一定不要忘记故事的意图。
- 避免镀金，开发不需要的功能。团队公开透明的工作方式可以帮助这一点。
- 非功能需求可以作为故事。
- 缺陷可以作为故事。

其他需求表达方式可能存在，而用户故事力求避免的问题

1. 忽略目标。
2. 关注范围而不是目标和价值，从而产生了范围变更这样的术语。
3. 冗长的文档，上下游交接和估算。
4. 需求太大。

用户故事可能的不足

1. 大型项目故事之间的关系错综复杂。
2. 大规模团队：需要在低宽带文档交流与高宽带面对面交流之间取得平衡。

解决规模化问题的思路是去规模化。从业务，架构，流程和人四个层面分别解耦，降低复杂度。

客户参与，产品探索和产品交付的全流程

客户参与，产品探索和产品开发的全流程步骤如下：

第一，产品负责人收集需求。利用前文讲到的客户团队和需求收集方法。

第二，产品探索团队利用故事写作研讨会对用户建模，编写用户故事，放入待办列表，排序，估算，制定发布计划。

好的故事估算方法：

- 允许团队随着对产品了解的加深随时改变想法。
- 要能适用于史诗和小故事。 估算精度随故事变大而变小。
- 估算快速。
- 能帮助提供进度和剩余工作的有用信息。
- 要能做到不精确也没事。
- 可以用来制定发布计划。
- 团队集体估算。

故事估算的单位可采用故事点或者理想天。

具体的估算方法：

- 产品负责人拿起一个故事，进行讲解，团队成员提问澄清。
- 团队成员用斐波那契序列的纸牌出牌估算，出到点数最大和最小的人说明，其他人可参与讨论。
- 大多数故事的估算会在一到两轮出牌后收敛。最多也不要超过三轮出牌。

还可用冒泡分桶法进行快速估算。把故事按从小到大从左到友排成一排，从冒泡法排到大家都认可为止。然后分组分到不同的桶里。桶的大小排列也是斐波那契序列。

每个迭代能完成的故事点数是速率。速率是一个靠谱的指标的依据是：故事的估算值与实际大小的偏差呈正态分布。每个迭代选取的故事的正负偏移可以彼此抵消。因此故事估算的精度不会影响速率指标的可靠性。速率指标可靠的前提是：在项目过程中始终采用一致化的估算方法，迭代之间的故事没有交叉重叠，项目过程中没有发生影响速率的大的异常。

速率和估算的几个注意事项：

- 不要拿速率来比团队。

- 分解后小故事估算的总合不一定等于大故事的估算。
- 任务工时的总和不需要跟故事点成正比。
- 结对编程不影响对故事的估算方法。
- 做到一半的故事不能计入速率。没有完全完成的事情是不靠谱的。鼓励一件流。迭代完成后不要修订速率。

发布计划的制定：

- 选定迭代长度。
- 预测速率。
- 产品的总故事点除以速率，即得出需要多少个迭代完成。

速率预测可采用：

- 历史值。
- 执行一轮看看速率是多少。
- 猜测。

可视化发布计划的方法：

- 墙。
- 电子表格。
- 甘特图。

第三，与客户确认发布计划。

第四，与交付团队梳理故事，达到准备好的状态（DoR - Definition of Ready）。故事要有清晰的验收标准，优先级和估算。团队成员有充分的理解。

第五，迭代计划，分解任务，生成迭代待办列表。

在迭代计划会上：

- 选取本迭代要完成的故事。
- 把业务故事分解为技术任务。分解也是做设计。不能技能的人员可以在同一个故事中合作。
- 团队做出承诺。

第六，每日工作，按故事和任务的优先级执行。

测量和监控速率的方法：

- 迭代故事点燃尽图。
- 迭代工时燃机图。
- 每迭代计划速率和实际速率趋势图。
- 每迭代计划和实际速率累计图。

这些图表只是团队监测趋势的工具。团队要在每天的工作中检视和适应以保证能达到完成迭代目标的趋势。

第七，迭代验收和演示。最好能有客户参与，以获得反馈。

结语

把用户故事放在目标，人和流程构成的体系，及从客户参与，到产品探索和交付构成的全流程中来看待。用户故事作为合适的颗粒度，促进人的合作，促进价值的拉动与流程的流动，帮助达到产品的目标。

用户故事促进双重改善。一方面是产品的改善。通过理解用户价值需求，快速实现和交付价值，获得反馈，获得学习，改进和提高产品。

另一方面是流程和工作方法的改善。通过速率和质量监测，发现工作流程中不流畅的地方，进行改善和提高。这种双重改善就是双环学习，使团队知道如何解决问题，和解决造成问题的原因，打造学习型组织，把工作方式变成组织的核心竞争力。