为什么用户故事必须是一个完整的功能？

User Story，也就是用户故事，它从用户的角度描述用户期望得到的功能。
在描述用户故事的内容时，一般包含三个要素：用户的角色、需要的功能、功能的价值。一种常见的用户故事描述方式可能像这样：
作为一个【角色】，我想要【功能】，以便【功能的价值】。
在一个完整的用户故事里，除了描述之外，还包含优先级、工作量评估、验收标准。验收标准是用户接受的条件和业务规则。当开发人员开发完一个功能，做完单元测试之后，他找到产品负责人。产品负责人按照验收标准逐条验证后，他就可以决定是接受还是拒绝这个用户故事。如果接受，意味着这个用户故事已经完成，开发人员可以继续开发下一个用户故事。
以上提到的是验收标准的作用，那用户故事的描述和验收标准的区别是什么呢？用户故事的描述侧重于描述功能，而验收标准则侧重于要实现这些功能所需要的条件。无论是用户故事的描述，还是验收标准，都是需求功能相关的，

。

大多数敏捷相关的书籍都会提到以上内容，相信大多数团队也是这么做的，直到这几天看到某公司的开发团队拆分用户故事的方式…
打开当前迭代周期的所有用户故事列表，我发现有的把GUI页面作为一个用户故事，有的把支持页面的API作为用户故事，有的把前后端的集成作为一个用户故事……
于是，就有了我与对方Scrum Master之间的这段对话。

> 我：用户故事应该是一个完整的功能，你们拆分的用户故事不是完整的功能，更接近实现这个用户故事的任务（Task）。
> □□
>
> SM：是的，我知道。我们的目的是为了拆分得更细致，让每个用户故事的工作量（Estimate）保持在比较小的数字上。

□□

我：
如果希望每个用户故事的工作量比较小，这就需要我们把功能的颗粒度拆得更小，而不是把实现功能的步骤当成用户故事。
每个用户故事应该是一个独立、完整、可交付的功能，而不是要实现这个功能的一部分。
□□

SM：但是，敏捷里并没有强制要求每个用户故事是一个完整的功能吧？
□□

我：很多书没有直接明说。这样吧，让我们回顾一下用户故事的描述：
作为一个[用户]，我想要[功能]，以便[价值]
。用户想要的的是一个功能，而一个具体的功能代表它是可以交付的，用户可以使用的，对吗？一个没有业务逻辑的页面不是用户想要的功能，支持页面与服务端交互的API也不是用户想要的功能，页面与API的集成也不是。所以说，每个用户故事应该是可以交付的功能。
□□

SM：话虽如此，我们这样拆分也不会有什么影响吧？
□□

我：影响非常大……
□□

有哪些影响呢？
影响项目计划
如果每个用户故事不是完整的功能，写每个用户故事的描述似乎不是一件简单的事情。我能说“作为一个用户，我想要2个API，以便支持登录页面功能”吗？显然不妥。用户不关心到底是2个API还是3个API，那不是他需要的。API是实现这个功能的开发人员需要的，它应作为任务（Task）。
除此之外，用户故事之间存在一定的依赖关系，在做项目计划时，不得不根据依赖关系来调整它们的优先级，可能花不少时间。
影响团队开发和交付
到了开发阶段，页面相关的“用户故事”开发完了，或者API相关的页面开发完了，此时产品负责人还没发验证，只有当二者集成相关的“用户故事”（加上引号，因为严格地讲，这不是一个合格的用户故事）完成了，产品负责人才能验证，一旦发现不是自己想要的，三个“用户故事”都得返工。
我们知道，在敏捷里，交付周期是固定的，交付内容不固定。在上面的例子中，由于

各个“用户故事”之间的依赖性，到了迭代周期结束的时候，很有可能页面、API、集成这三个“用户故事”都无法交付。从燃尽图来看，当三个“用户故事”没有完成的时候，燃尽图的曲线在理想线的上方，好几天都没有下降，突然某一天（三个都完成了），下降了很多。这种情况，交付的风险太大。

上述是从燃尽图来看迭代周期的风险，分析具体原因呢？在整个迭代周期里，很多用户故事在完成之前都没法验证，等到迭代周期快结束的时候才能验证，当验证不通过时，又得返工，然而，这时已经没有多少时间了。

所以，**每个用户故事应该是一个独立、完整、可交付的功能。**