**使用者故事 User story**

如果你問我: 由傳統開發方式改革至敏捷開發，應該從哪裡開始呢? 我會回答從改善品質，加重測試開始，但對於真正的開發工作，則我會選擇從用來描述需求用的「使用者故事」開始。長久以來我一直以為:

**敏捷開發送給世界最好的需求描述，就是用「*故事*」的方式來說明需求。**

它是1998年被 Kant Beck 首先運用在XP極致編程裡，供使用者用來描述專案範圍的方法。這種威力無窮的需求描述方式，因為看起來太簡單了些(註1)，常常會讓人們忽略了它的影響。老實說；敏捷專案開發的成敗可是完全靠它來作支撐(註2)，要知道如果沒有好的需求描述、問題探討，怎麼會有好的產品產生呢?!

[](https://ruddyblog.files.wordpress.com/2015/08/all-agree_0.jpg)

Kent Beck 的初始目的是為了避免大家沒有討論清楚，

就各自自以為是的點頭開始工作。

**註1:**  看起來簡單的東西，實際執行起來並不一定會簡單。因為規範少了反倒很難具體的來實踐它。後來Bill Wake想出了[INVEST的六個法則](https://en.wikipedia.org/wiki/INVEST_(mnemonic))，最近則以Jeff Patton的[使用者故事地圖](http://shop.oreilly.com/product/0636920033851.do)。都對產出好用的使用者故事有著莫大的貢獻。

**註2:**據 Don Reinertsen 所說 80~85%的專案之所以未達到要求，完全是不正確的需求所造成。

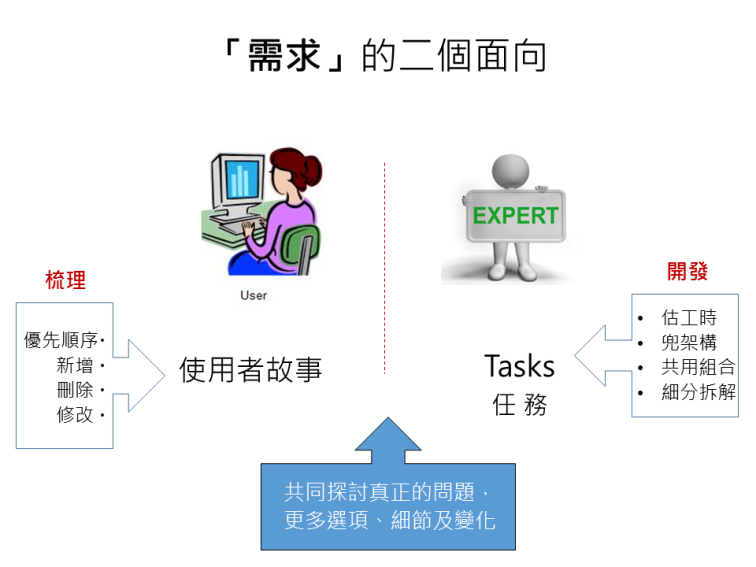
**Studies show that 80 to 85 percent of project failures are due to incorrect requirements.**

Don Reinertsen

author of : The Principles of Product Development Flow

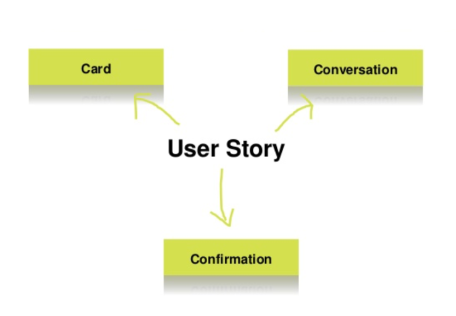
**使用者故事是最佳的溝通利器**

業務人員要依靠它來跟技術人員進行溝通，而工程師要依靠它來了解需求並拆解成為可以實作的任務。而它更是二者之間共同探討真正的問題所在、是否有更多選項、挖掘更多細節及接受更多變化的依據。下圖中顯示了需求的二個面向， 一個是對使用者的，它讓使用者足以用自己所熟悉的業務用詞來談需求，完全不用去描述到任何技術的東西。另一個是對程式設計人員的，它是以畫面、場景或是流程方式，來協助工程師快速學習業務知識用的。

[](https://ruddyblog.files.wordpress.com/2015/08/e68a95e5bdb1e789875_1.png)

善用溝通才能發掘真正的問題，也才能提煉出更好的設計。

Ron Jeffries在2001寫下了使用者故事著名的 3C理論，用來討論、確認故事的各部分組成。1C:卡片(*Card*): 極端抽象又簡潔的說明。2C:談話(*Conversation*): 無時無刻都可進行的溝通與討論。3C:確認(*Confirmation*): 為談話的內容及目標作成越正式越好的結論。明顯扼要地說明了，使用者故事是用來溝通用的。(很多使用者把它拿來做為規格文件使用，那就大錯特錯了。)

[](https://ruddyblog.files.wordpress.com/2015/08/3cc.png)

Ron Jeffries 的 3C

**※舉一個例子來描述一下它的使用步驟:**

**步驟一、**如以下的陳述，針對某一個目的寫下使用者故事  –> 查詢客戶資料。

**身為一個使用者，我希望可以運用姓氏或名字來搜尋我的客戶。**

**As a user, I want to search for my customers by their first and last names.**

**步驟二、**工程師由使用者故事推敲如何解題  –> 進行功能拆解。

※將使用者故事拆解成任務(Tasks):

**Task 1:  畫面製作。**至少需要輸入: 姓氏、名字的輸入欄位及搜尋的按鈕。

**Task 2:  訊息顯示。**顯示進行中、錯誤訊息。

**Task 3:  處理搜尋功能呼叫及取回資料。**

**Task 4:  結果顯示。**討論以何種元件(List、Grid、…) 作為結果顯示。

**步驟三、**雙方進行討論，定義何謂完成 –> 討論這樣做是否解決了真正的問題。

客戶與程式開發人員就可以針對使用者故事進行討論，確認描述的故事是否解決了真正的問題。

也許找到"使用者的 email “或 “電話號碼“才是查詢者真正的目的。為此；就應該把查到的資料用這兩個欄位作為顯示的重點。還有；一次顯示資料的筆數? 是否需要進行排序? 再加上必要標誌等功能，也需要視各種情況進行適當調整。

### ****經過與客戶討論之後的可能結果 — 範圍與限制都更明確了****

### 上面的範例並沒有將最後的利益說清楚，因此留下了讓客戶與程式設計人員有很大的討論空間(是優點也是缺點!)。下面則是二則運用不同的使用者(角色)將同一個使用者故事**利益設限**的範例:

### ****身為****一個電話傳呼員，****我希望****可以運用姓氏或名字來搜尋我的客戶電話****以便於****我可以直接按鈕呼叫出去。

### ****身為****一個推銷員，****我希望****可以運用姓氏或名字來搜尋我的客戶Email 地址****以便於****我可以直接按鈕送出我的廣告。

**》上面是一個單一且明確的範例，若是把它加以運用在一個團隊開發的專案上。則3C便成了:**

**[ 1C ]**  一份書面的故事描述，用來當作計畫或提示用。

**[ 2C ]** 在計畫會議時做為有關故事的對話、討論，用來具體化故事的各種細節。

**[ 3C ]** 測試。用來確認撰寫的故事是否完成解決了它的問題，最終成為了基本的文件。

**【小結】**

建構滿足使用者需求的軟體，最好的方法是從"使用者故事"開始。它即簡明扼要又清楚明確地描述對實際用戶有價值的功能。但它(使用者故事)不是規格文件，而是用來探討解答的抽象化方式，對於敏捷而言；真正的規格文件是誕生於持續的開發與維護當中。(敏捷文件；它跟程式的架構有著相同的產生方式，也就是衍生出來的，是一邊設計而同時又一邊撰寫出來的)

回答一下**:  為何「改革要從改善品質，加重測試開始」呢?**

**因為加重測試可以喚來紀律，而紀律是開始進行改革時最最需要的東西。**