ScrumMaster 的檢查清單

作者

Michael James (mj4scrum@gmail.com)

2007 年 9 月 14 日

最後修改於 2016 年 2 月 2 日

譯者

陳超 Chen, Chao (chao.wings@gmail.com)

陳仕傑 Chen, Joey (jay.hatelove@gmail.com)

范啓明 Fan, Kevin (kevincmfan@gmail.com)

李境展 Li, Tomas (tomasli@gmail.com)

林昱承 Lin, Yves (yveslin@gmail.com)

2016 年 3 月 22 日

編輯

方彥雯 Fang, Yenwen (yenwenf@gmail.com)

全職的引導者？

一個稱職的 ScrumMaster 可以同時協助兩個或三個團隊。如果你把自己的角色只侷限在舉辦會議、落實時間盒 (Timeboxes) 以及只處理團隊成員所提出來的阻礙，那麼扮演一個兼職的 ScrumMaster 是沒問題的。這樣的情況下，團隊的表現可能比原本還好，甚至超過組織在導入 Scrum 前對團隊的期待，也許也不會發生什麼大災難。

但如果你期待團隊在組織的轉型過程中，能達到前所未有、出乎所有人意料的成就，那你需要考慮成為一個**優秀**的 ScrumMaster。一個優秀的 ScrumMaster 一次只專注於協助**一個團隊**。所以我們建議，在 Scrum 團隊剛成立的時候，由一個專職的 ScrumMaster 搭配一個約七人的團隊。

如果你還沒發現 ScrumMaster 需要做哪些事情，關注一下你的 Product Owner、你的團隊、團隊的技術實踐，以及團隊所屬的組織。雖然沒有一個藥方能對所有的團隊都有效，但我還是嘗試列出一份檢查清單，供 ScrumMaster 檢視一些他們常忽略的狀況。請依最後一頁說明的方式，在每個格子上標注 √、Δ、?、N/A。

第一部分：我的 Product Owner 情況如何？

ScrumMaster 能協助 PO 找到更好的方式來維護產品待辦清單（Product Backlog）和發佈計畫（Release Plan），藉此提昇 PO 的工作效能。（注意，產品待辦清單的優先順序是 PO 的責任。）

☐ PO 是否按照最新的想法為產品待辦清單上的事項排序？

☐ 所有利害關係人的需求和願望，是否都已經包含在產品待辦清單上了？
提醒：待辦清單是**逐漸湧現**的。

☐ 產品待辦清單的規模是否維持在容易管理的大小？為了讓待辦事項（Item）的數量方便管理，清單上高優先級的待辦事項應該要具體、細膩一點（粒度較小），而低優先級的待辦事項則涵蓋面大一點（粒度較大）。過度分析低優先級的待辦事項是沒有效率的，因為需求會隨著開發團隊、利害關係人、和客戶之間的持續交流而有所變化。

☐ 待辦清單上的需求（尤其是高優先級的待辦事項），是否可以用符合INVEST原則的使用者故事，來做出更好的表達？INVEST原則是用來驗證使用者故事是否：夠獨立（Independent）、有討論空間（Negotiable）、有價值（Valuable）、可估算（Estimable）、適當大小（Small/Size），還有可測試（Testable）[1]。

☐ 你是否有讓 PO 瞭解什麼是**技術債**（Technical Debt），以及如何避免技術債嗎？其中一種方式是把撰寫自動化測試和軟體重構列入「完成的定義」（DoD）裡。

☐ 你的待辦清單是否是個**資訊暖爐**（散播資訊），是否擺在一個所有利害關係人都會注意的地方，而且他們可以一目了然地瞭解目前產品的開發狀況？

☐ 如果你透過自動化工具來管理待辦清單，是否每個人都可以輕鬆方便地使用這個工具？如果 ScrumMaster 沒有保持資訊得以持續散播的話，那自動化工具可能反而會變成**資訊冰箱**（冷凍資訊）。

[1] http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/

☐ 如果把資料直接印出來給大家看，可以幫助散播資訊嗎？

☐ 如果製作大張的視覺化圖表，可以幫助散播資訊嗎？

☐ 你有幫助 PO 把 Item 整理到合適的發佈時程或不同優先群組中嗎？例如更改 Item 的優先順序，把原本同時發佈的 Items 放到不同的發佈時程，或反過來。

☐ 所有人都清楚知道發佈計畫是否還符合現實情況嗎？也許在Sprint Review中讓每個人都看到產品發佈燃盡圖 [2]的情況。圖表能顯示出團隊待辦事項的消化以及新增速率，以便更早發現時程與範圍的變化。

☐ 你的 PO 是否有在 Sprint Review 後，依據現實情況來調整發佈計畫？例如有些能按時發佈充分測試過的產品的 PO，他們會依情況在每個 Sprint 重新規劃 Item 的優先順序與產品的發佈計畫。因為他們可能發現了更重要的事情需要做，得把某一些需求往後延遲到更後面的發佈。

第二部分：我的團隊情況如何？

在你以身作則跟團隊合作時，有可能太著迷在技術性的工作中。請記得你的主要職責是引導團隊。

☐ 你的團隊是否進入**神馳**狀態 [3]？比如說：

- ✓ 為何而戰：明確的目標（清晰的期待和規則，可達成的目標，跟個人的技術和能力相匹配）。
- ✓ 全神貫注：集中和專注，注意力高度集中在所關注的少數事情上。
- ✓ 如臂使指：行動和想法高度協調。
- ✓ 直言不諱：給予直接且即時的回饋（不論成功或失敗都能明確揭露，讓行為可以依此調整）。
- ✓ 量力而為：平衡能力和挑戰（不會太難也不會太簡單）。
- ✓ 運籌帷幄：自認為對情況和活動有足夠的掌控能力。
- ✓ 渾然忘我：心理得到滿足，很直覺的就進行活動。

[2] Mike Cohn, *Agile Estimation and Planning* (2005).
[3] Mihaly Csikszentmihalyi, *Flow: The Psychology of Optimal Experience* (1990).

☐ 團隊成員是否看起來互相喜歡，可以互相開玩笑，也會慶祝彼此的成功？

☐ 團隊成員是否彼此都覺得對高標準有責任，而且互相砥礪成長進步？

☐ 是否有些問題或機會因為團隊不安或不自在，所以不願拿出來討論？[4]

☐ 是否有嘗試過各種不同的形式和地點來舉行Retrospective？[5]

☐ 團隊有把注意力放在 Sprint Goals 上嗎？其中一種方法是，也許可以在 Sprint 進行中，回顧一下這次 Sprint 的待辦事項的 Acceptance Criteria，確保大家對 Sprint Goal 的認知是一樣的。

☐ Sprint 工作看板是否有反應出團隊真正在做的事情？小心所謂的「黑暗物質」，即沒有被揭露的工作，還有大過於一天的工作事項。跟 Sprint Commitments 無關的工作，往往是完成 Commitment 的阻礙。

☐ 你的團隊是否有 3-9 個成員，是否擁有足夠的技術能力分佈，足以產出潛在可交付產品增量？

☐ 團隊的工作看板是否是最新狀態？
*譯註：作者特別強調是實體的工作看版，而非在某個自動化工具上、只有一些人關注的 Sprint Backlog。

☐ 團隊用來自我管理的事物是否能被團隊看到，並且團隊很容易就能使用？
*譯註：例如工作看板、燃盡圖、團隊公約（Working Agreements）等等。

☐ 這些事物是否能保持不被愛管閒事的人干擾？團隊外的人對日常活動過度審查，也許會阻礙團隊內部的透明度和自我管理。

☐ 團隊成員是否自願領取工作事項？

---

[4] Marshall Rosenberg, *Nonviolent Communication: A Language of Life: Life-Changing Tools for Healthy Relationships* (2003). 或是請有能力減輕不自在感的引導者加入對話。

[5] Derby/Larson *Agile Retrospectives: Making Good Teams Great* (2006).

☐ 償還技術債是否有明確的列入 **DoD**，持續讓程式碼變好，讓團隊能輕鬆自在地在上面工作？

☐ 團隊成員是否把職稱放在一邊，並願意對所有已同意的工作一起負責？如測試，製作使用者文件等等。

第三部分：我們團隊技術實踐的情況如何？

☐ 在你們開發系統時，能不能只需要「按下一個測試按鈕」，就可以讓每個人（不論同一個團隊或其他團隊）很方便得知目前的系統是否如同預期般的正常運作（沒把之前正常運作的功能弄壞了）？通常可以透過 xUnit 等單元測試框架（如 JUnit、NUnit）來實現。

☐ 針對自動化端對端系統測試與自動化單元測試的比例，是否取得適當平衡？

☐ 團隊是否用同樣的程式語言撰寫產品的系統測試和單元測試？避免使用只有部份成員才懂得怎麼維護的程式語言或工具，這會造成合作上的困擾。

☐ 你的團隊是否有發現在系統測試和單元測試之間存在有用的灰色地帶？[6]
*譯註：作者想表達的是，大部分的測試是分佈在單純的系統測試和單純的單元測試中間，有效的測試比測試的名稱重要。

☐ 當迴歸測試失敗時，CI Server[7]的警報是否會自動響起？團隊是否能在幾小時，甚至幾分鐘內處理完畢？（就像Kent Beck說過，只做到每日構建的是膽小鬼）。

☐ 是否**所有**的測試結果都有匯總到 CI Server 上呈現？

☐ 相對於預先大量設計，團隊成員是否已經發現持續演進的設計與經常重構所帶來的成功與樂趣？[8]重構有一個嚴謹的定義：在不影響外部使用行為的情況下改變內部結構。一般來說，應該每個小時都會重構好幾次，不管是針對重複的程式碼、過於

[6] http://blogs.collab.net/agile/junit-is-not-just-for-unit-testing-anymore
[7] http://www.martinfowler.com/articles/continuousIntegration.html
[8] Martin Fowler, *Refactoring: Improving the Design of Existing Code* (1999).

複雜的邏輯（過深的程式碼區塊或太長的方法）、糟糕或無法識別的命名、物件之間不必要的耦合性等等。重構前必須有自動化測試保護，才能確保結果的正確性。忽視重構的重要性將使得產品越來越難修改，尤其是很難找到優秀的開發人員願意在糟糕程式碼上工作。

☐ 你們的 DoD 是否有包含完整的自動化測試涵蓋率和重構？使用 TDD 可以更容易達成這個目標。

☐ 團隊成員是否大部分時間都在結對編程（Pair Programming）？結對編程可大幅提升程式的可維護性和減少錯誤。雖然對開發人員舒適圈的界線來說，是種挑戰，而且有時候看起來花費更長的時間（如果你在乎的是程式碼的行數，而不是可交付功能的品質）。以身作則，跟團隊成員一起進行結對編程，也許有人會開始喜歡這樣的開發方式。

第四部分：我們的組織情況如何？

☐ 跨團隊的溝通是否適量且持續發生？「Scrum of Scrum」只是其中一種方法，而且通常不是最好的方法。[9]

☐ 即使跨越了架構上的界限，各團隊是否仍可獨立產出可運作的產品功能？[10]

☐ 你和組織內的其他 ScrumMaster 是否經常碰面，協助處理組織層面的阻礙清單？

☐ 如果可以，團隊所碰到的阻礙，是否有貼在開發人員主管辦公室的牆上？成本、進入市場的時間延遲、失去的客戶機會是否能被量化成錢？（但請記取Ken Schwaber的教訓：「一個陣亡的ScrumMaster對團隊沒有幫助。」[11]）
*譯註：Ken Schwaber 的意思是：如果 ScrumMaster 離開了團隊，那對團隊一點幫助都沒有。所以要小心選擇戰場，不但讓自己「存活」，還要證明自己的存在對產品帶來價值。

---

[9] 其他方法請見 http://less.works/less/framework/coordination-and-integration.html
[10] http://featureteamprimer.org
[11] Ken Schwaber, *Agile Project Management with Scrum* (2004).

☐ 你們是否屬於少數的那些能兼容個人職涯發展與團隊共同目標的組織？如果你們在升遷[12]上只重視開發或架構設計等工作，不重視測試、測試自動化或使用者文件的工作價值，那就代表你們「不是」這樣的組織。

☐ 你的組織是否被獨立第三方認可為最好的工作環境，或是產業中的領先者？

☐ 你是否正在創造一個**學習型組織**？

結論

如果你做到了大部分的事項而且還有多餘的時間，我希望有機會能和你交流。

沒有一個固定的公式可以讓人更有創造力。這份表格也許可以，也許不可以，對你的情況有所幫助。

當你察覺到你能帶來一些改變時，也許會發現自己害怕了起來。別擔心，這代表你正走在對的路上。

[12] Alfie Kohn, *Punished By Rewards: The Trouble with Gold Stars, Incentive Plans, A's, Praise, and Other Bribes* (1999).

*譯註：

ScrumMaster對四個面向的關注程度可以用以下圖片表示 [13]

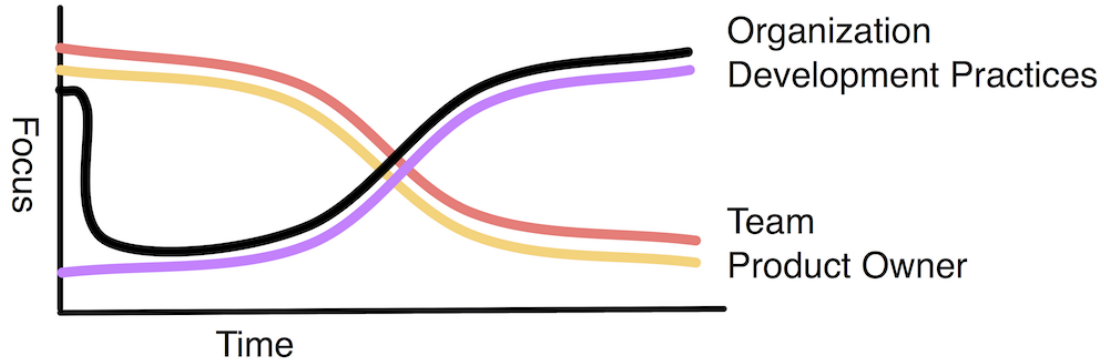

Figure 1 ScrumMaster focus over time

在與作者 Michael James 溝通時，他強調雖然這上面寫的是 ScrumMaster 要做的事，但他的本意是 ScrumMaster 要讓這些事情發生，也許一開始教導或自己示範，但最終目標是由自組織的團隊把這些責任扛起來。

[13] http://less.works/less/structure/scrummaster.html#ScrumMasterfocus

組織阻礙表

表層問題

根本原因（用 5 個為什麼）

商業面上的影響

情感面上的影響

明確且可執行的需求

組織阻礙表

表層問題

根本原因（用 5 個為什麼）

商業面上的影響

情感面上的影響

明確且可執行的需求

組織阻礙表

表層問題

根本原因（用 5 個為什麼）

商業面上的影響

情感面上的影響

明確且可執行的需求

組織阻礙表

表層問題

根本原因（用 5 個為什麼）

商業面上的影響

情感面上的影響

明確且可執行的需求

組織阻礙表

表層問題

根本原因（用 5 個為什麼）

商業面上的影響

情感面上的影響

明確且可執行的需求

組織阻礙表

表層問題

根本原因（用 5 個為什麼）

商業面上的影響

情感面上的影響

明確且可執行的需求

說明

如果你是在訓練課程中得到這份檢查清單，同時你現在(或最近)的雇主曾試著跑 Scrum，或任何類似 Scrum 的方法，請依據你的觀察回答每個問題。回答時請用下面符號：

√ （做得很好）
Δ （可以改進而且知道如何著手）
? （可以改進但不知道如何改進）
N/A（不適用或無益處）

如果你現在 (或最近) 的雇主不曾試過跑 Scrum 或任何類似 Scrum 的方法，請使用下列符號回答每個問題：

√ （做得很好，或很容易就可以做到很好）
Δ （是個挑戰而且知道如何著手）
? （是個挑戰但不知道如何著手）
N/A（不適用或無益處）

回答完所有問題後，在附錄的組織阻礙表上列出 2 到 6 個阻礙，無所謂是不是在填寫清單時發現的。只列下你至少有 1%的把握可以使上力的阻礙。