**Câu chuyện về cuộc chiến agile**

**Scrum và XP**

**từ những Chiến hào Chúng tôi đã Scrum như thế nào?**

** Henrik Kniberg** Viết lời tựa Jeff Sutherland, Mike Cohn

*Loạt sách “Enterprise Software Development”*

1 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**BẢN TRỰC TUYẾN MIỄN PHÍ**

**(không được phép in ấn từ bản này)**

Nếu bạn thích cuốn sách này, xin hãy hỗ trợ

tác giả và InfoQ bằng cách

**đặt mua bản in của cuốn sách tại**

http://www.lulu.com/content/899349 **(giá bán 22,95$)**

Cuốn sách được cung cấp tới bạn với sự trợ giúp của



Bản miễn phí được đăng tải trên trang web InfoQ.com, nếu bạn nhận được cuốn sách này từ bất cứ nguồn nào xin hãy hỗ trợ tác giả và nhà xuất bản bằng cách đăng ký với trang web InfoQ.com.

**Cuốn sách được cung cấp tại:**

http://infoq.com/minibooks/scrum-xp-from-the-trenches

2 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

***Scrum và XP từ những Chiến hào Chúng tôi đã Scrum như thế nào?***

**Tác giả: Henrik Kniberg**

**Dịch: Nguyễn Việt Khoa (TN), Bùi Thanh Hải, Vũ Thị Hà Thảo, Phạm Anh Đới, Nguyễn Ngọc Tú, Dương Lê Vinh, Nguyễn Bá Trường, Nguyễn Duy Hoàng**

**Hiệu đính và chú giải: Dương Trọng Tấn**

**Bản Trực tuyến Miễn phí**

Hỗ trợ bằng cách mua bản in tại địa chỉ:

http://infoq.com/minibooks/scrum-xp-from-the-trenches

Bản tiếng Việt được cung cấp bởi **HanoiScrum** tại:

http://hanoiscrum.net/hnscrum/resource

Mọi đóp góp, phản hồi về bản dịch, xin liên hệ hanoiscrum@gmail.com

3 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

© 2007 C4Media Inc

Bản quyền thuộc về C4Media Inc.

Trang web của nhà xuất bản C4Media: InfoQ.com

Cuốn sách này thuộc loạt sách về “Enterprise Software Development” trên InfoQ.

Để có thêm thông tin hoặc đặt mua cuốn sách này hoặc các cuốn khác trên InfoQ, mời bạn liên hệ với

books@c4media.com

Không được phép sử dụng, lưu trữ hoặc chuyển đổi nội dung của cuốn sách này dưới bất cứ hình thức nào như điện tử, bản sao, v.v. trừ những điều được quy định trong Phần 107 hoặc 108 thuộc Đạo luật về Bản quyền của Hoa Kỳ năm 1976, mà không có sự cho phép trước bằng văn bản của Nhà xuất bản.

Chỉ định này được sử dụng bởi các công ty để nhận ra các sản phẩm của họ được tuyên bố như là các thương hiệu. Trong tất cả các trường hợp tên sản phẩm xuất hiện bắt đầu bằng Chữ viết hoa hoặc VIẾT HOA TOÀN BỘ CÁC CHỮ, khi đó C4Media Inc. nhận thấy đó là dấu hiệu vi phạm quyền lợi. Tuy nhiên, độc giả nên liên hệ với các cơ quan chức năng để có đầy đủ các thông tin cần thiết liên quan đến thương hiệu và đăng ký thương hiệu.

Người biên tập: Diana Plesa

Thiết kế bìa: Dixie Press

Dàn trang: Dixie Press

Thư viện Congress Cataloguing-in-Publication Data:

ISBN: 978-1-4303-2264-1

Được in tại Hoa Kỳ

4 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Lời cảm ơn**

Tôi chỉ có một ngày cuối tuần để soạn bản thảo đầu tiên cho cuốn sách này nhưng chắc chắn đó là một ngày làm việc với cường độ cao! 150% tập trung cho công việc :o)

Xin cảm ơn vợ tôi là Sophia và các con tôi Dave và Jenny đã dành riêng cho tôi ngày cuối tuần này, và xin cảm ơn bố mẹ vợ tôi là Eva và Jorgen đã tới giúp tôi chăm sóc gia đình.

Cũng xin gửi lời cảm ơn tới các bạn đồng nghiệp của tôi tại Crisp ở Stockholm và những người trong nhóm Yahoo Scrumdevelopment đã xem xét và giúp tôi trong việc hoàn thành tài liệu này.

Và cuối cùng, xin cảm ơn tất cả các bạn đọc đã cung cấp cho tôi những thông tin phản hồi hữu ích. Đặc biệt, tôi rất vui mừng khi biết được bài viết của tôi đã tác động nhiều đến các bạn trong việc sử dụng Phương pháp Phát triển Phần mềm Linh hoạt (Agile)!

5 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Mục lục**

Lời cảm ơn......................................................................................................................................................4 Mục lục ............................................................................................................................................................5 Lời tựa của Jeff Sutherland........................................................................................................................10 Lời tựa của Mike Cohn................................................................................................................................12 Lời nói đầu ....................................................................................................................................................14 Giới thiệu.......................................................................................................................................................15

Khuyến cáo...............................................................................................................................................16 Lý do tôi viết cuốn sách này...................................................................................................................16 Vậy Scrum là gì?......................................................................................................................................16

Chúng tôi làm Product Backlog như thế nào...........................................................................................18 Các trường khác ......................................................................................................................................20 Làm thế nào chúng ta giữ Product Backlog ở mức nghiệp vụ ........................................................21

Chúng tôi chuẩn bị Họp Kế hoạch Sprint như thế nào ..........................................................................22 Chúng tôi họp Kế hoạch Sprint ra sao......................................................................................................24 Tại sao Product Owner phải có mặt......................................................................................................24 Tại sao chất lượng là thứ không thể thỏa hiệp ...................................................................................26 Các cuộc họp lập kế hoạch Sprint kéo dài … .....................................................................................27 Lịch trình Họp Kế hoạch Sprint..............................................................................................................28 Xác định thời gian cho Sprint.................................................................................................................29 Xác định mục tiêu Sprint.........................................................................................................................29 Nhóm quyết đưa story nào vào Sprint như thế nào?.........................................................................30 Product Owner tác động thế nào để story nào đó được đưa vào Sprint?......................................31 Nhóm quyết định đưa story vào Sprint như thế nào ..........................................................................34 *Ướ ới kỹ* .......................................................................................................34 *Ướ ớ ỹ* ....................................................................................................35 Chúng ta s s dụng k thuật ước t nh nào?..................................................................................39

6 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

ại sao chúng tôi s dụng các th ch mục nde Card ..................................................................40 ịnh ngh a oàn thành ........................................................................................................................43 ớc t nh thời gian s dụng Bộ bài Lập kế hoạch...............................................................................44 Ph n loại các story ..................................................................................................................................46 Phân tách các story thành những story nhỏ hơn................................................................................47 Phân tách story thành các tác vụ ..........................................................................................................47 Xác định thời gian và địa điểm cho buổi Họp Scrum Hằng ngày.....................................................49 Chúng ta cần dừng ở đ u ......................................................................................................................49 Các story k thuật (tech stories).............................................................................................................50 Hệ thống Theo dõi lỗi và Product Backlog...........................................................................................53 Họp Kế hoạch Sprint đã đến hồi kết!....................................................................................................54

Chúng tôi truyền thông trong các Sprint như thế nào ............................................................................55 Chúng tôi xây dựng Sprint Backlog như thế nào....................................................................................58 Khuôn mẫu cho Sprint Backlog .............................................................................................................58 S dụng bảng công việc taskboard như thế nào.............................................................................60 Cách làm việc với Biểu đồ Burndown...................................................................................................63 Những dấu hiệu cảnh báo trên bảng công việc ..................................................................................64 Này, thế còn về khả năng theo dõi thì sao?! .......................................................................................67 ớc tính ngày và giờ ..............................................................................................................................68 Chúng tôi bố trí phòng làm việc như thế nào ..........................................................................................69 Góc thiết kế...............................................................................................................................................69 ặt chỗ cho Nhóm ngồi cùng nhau!......................................................................................................70 Bố trí cho Product Owner một khu vực ................................................................................................72 Bố trí cho những người quản lý và người huấn luyện ở một khu vực ............................................72 Chúng tôi họp Scrum Hằng ngày như thế nào........................................................................................73 Chúng tôi cập nhật bảng công việc như thế nào................................................................................73 Ứng x với những người đến muộn.....................................................................................................74 Ứng x với những câu trả lời ôi không biết làm gì ngày hôm nay ..............................................74

7 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Chúng tôi Sơ kết Sprint ra sao ..................................................................................................................77 Lý do chúng tôi nhấn mạnh đến việc các Sprint cần kết thúc với một buổi demo.........................77 Những mục cần kiểm tra cho buổi Sơ kết Sprint................................................................................78 Giải quyết với những thứ chưa thể demo .........................................................................................78

Chúng tôi Cải tiến Sprint như thế nào ......................................................................................................80 Tại sao chúng tôi nhấn mạnh rằng tất cả các nhóm phải tiến hành cải tiến ..................................80 Chúng tôi tổ chức các buổi Họp Cải tiến như thế nào.......................................................................80 Lan truyền những bài học tới các nhóm khác.....................................................................................82 hay đổi hay không thay đổi ..................................................................................................................83 Ví dụ về những vấn đề có thể xảy ra trong quá trình cải tiến...........................................................83

"L ra chúng ta đã phải dành nhiều thời gian hơn để chia nhỏ story thành những hạng mục nhỏ hơn và các tác vụ ........................................................................................................................84

"Quá nhiều phiền nhiễu bên ngoài"...................................................................................................84 "Chúng tôi không đạt được cam kết và ch hoàn thành được một n a công việc ...................84 "Văn phòng của chúng tôi là quá ồn ào và lộn xộn".......................................................................84

Quãng ngh giữa các Sprint........................................................................................................................86 Chúng tôi lập Kế hoạch Phát hành và định giá các hợp đồng như thế nào.......................................89 Xác định ngưỡng chấp nhận của bạn ..................................................................................................89 ớc tính thời gian cho những hạng mục quan trọng nhất................................................................90 Tốc độ ớc tính.......................................................................................................................................92 ặt vào cùng trong một kế hoạch phát hành.......................................................................................93 Hiệu ch nh kế hoạch phát hành để thích nghi .....................................................................................94 Chúng tôi kết hợp Scrum với XP ra sao...................................................................................................95 Lập trình cặp (Pair programming) .........................................................................................................95 Phát triển ướng-Kiểm-th (Test-driven development)....................................................................96 TDD với mã nguồn mới.......................................................................................................................98 TDD với mã nguồn cũ .........................................................................................................................98 Thiết kế tiến hóa (Incremental Design) ................................................................................................99

8 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Tích hợp liên tục (Continuous Integration)...........................................................................................99 Sở hữu mã nguồn tập thể ....................................................................................................................100 Không gian làm việc giàu thông tin .....................................................................................................100 Tiêu chuẩn mã nguồn ...........................................................................................................................100 Làm việc với tốc độ\năng lượng bền vững........................................................................................101

Chúng tôi kiểm th như thế nào..............................................................................................................102 Bạn có l s không thể bỏ được giai đoạn kiểm th chấp nhận....................................................102 Giảm thiểu giai đoạn kiểm th chấp nhận .........................................................................................103 ăng cường chất lượng bằng cách đưa các nhà kiểm th vào Nhóm Scrum.............................104

Nhân viên kiểm th là người phát tín hiệu dừng .......................................................................104 Kiểm th viên s làm gì khi mà không có gì để kiểm th ? .........................................................105 ăng chất lượng bằng cách giảm thiểu công việc trong Sprint......................................................106 Kiểm th chấp nhận có nên là một phần của Sprint không?..........................................................106 Chu trình Sprint với chu trình kiểm th chấp nhận...........................................................................107

ướng tiếp cận 1: Không tiến hành xây dựng các thứ mới cho tới khi những thứ đã làm vẫn còn chưa hoàn thành .......................................................................................................................109

ướng tiếp cận tồi – tập trung vào việc xây dựng t nh năng mới ...........................................111 ừng chạy nhanh hơn mắt xích chậm nhất trong dây chuyền của bạn .......................................111 Quay lại với thực tế ...............................................................................................................................112

Chúng tôi quản lí nhiều nhóm Scrum như thế nào...............................................................................113 Bao nhiêu Nhóm? ..................................................................................................................................113 Nhóm ảo..............................................................................................................................................114 Kích cỡ nhóm tối ưu..........................................................................................................................115 ồng bộ hay không đồng bộ các Sprint?...........................................................................................116 Tại sao chúng tôi đưa thêm vai trò lãnh đạo nhóm .......................................................................117 Bố trí những thành viên vào các nhóm như thế nào........................................................................119 Nhóm chuyên biệt – có hay không ? ..................................................................................................120 ướng 1: các nhóm chuyên biệt về từng thành phần .................................................................120

9 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

ướng 2: Các nhóm liên thành phần. ............................................................................................122 Có tái sắp xếp lại các nhóm giữa Sprint hay không?.......................................................................123 Các nhân viên bán thời gian ................................................................................................................123 Chúng tôi thực hiện họp Scrum-của-các-Scrum như thế nào ........................................................124

Họp Scrum-của-các-Scrum mức sản phẩm ..................................................................................125 Họp Scrum-của-các-Scrum mức công ty.......................................................................................125 an en ọp Scrum Hằng ngày..........................................................................................................126 Các Nhóm chữa cháy ...........................................................................................................................127 Chia nhỏ Product Backlog nên hay không nên?...............................................................................128 Chiến lược 1: Một Product Owner, một Product Backlog............................................................128 Chiến lược 2: Một Product Owner, nhiều Product Backlog ........................................................132 Chiến lược 3: Nhiều Product Owner, mỗi người sở hữu một Product Backlog.......................133 Phân nhánh mã nguồn..........................................................................................................................134 Họp Cải tiến nhiều nhóm ......................................................................................................................135 Chúng tôi quản lí các nhóm ph n tán như thế nào ..............................................................................136 Làm việc ở nước ngoài.........................................................................................................................137 Các thành viên làm việc ở nhà ............................................................................................................138 Danh mục kiểm tra của Scrum Master...................................................................................................140 ở đầu Sprint.........................................................................................................................................140 oạt động hàng ngày............................................................................................................................140 Kết thúc Sprint........................................................................................................................................141 Lời chào tạm biệt .......................................................................................................................................142 Tài liệu nên đọc..........................................................................................................................................143 Về tác giả ....................................................................................................................................................144

10 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Lời tựa của Jeff Sutherland**

Các Nhóm cần có hiểu biết cơ bản về Scrum. Bạn làm thế nào để tạo và ước tính một Product Backlog? Bạn làm thế nào để triển khai chúng trong Sprint Backlog? Bạn quản lý biểu đồ Burndown và tính toán tốc độ của Nhóm như thế nào? Cuốn sách của Henrik là công cụ khởi nguồn cho các thực hành cơ bản, nó giúp các Nhóm vượt qua trở ngại để thực hành Scrum tốt hơn.

Thực hành tốt Scrum ngày càng trở lên quan trọng với các nhóm muốn được đầu tư dài hạn. Là nhà huấn luyện Agile cho một nhóm đầu tư mạo hiểm, tôi trợ giúp với mục tiêu của họ là chỉ đầu tư vào các công ty sử dụng Agile và thực hành tốt phương pháp này. Senior Partner của nhóm sẽ mời tất cả các công ty trong danh mục đầu tư nếu họ biết được tốc độ của các Nhóm của công ty đó. Họ sẽ khó mà trả lời đúng được câu hỏi đó vào lúc này. Các cơ hội đầu tư trong tương lai sẽ đòi hỏi những nhóm phát triển phần mềm phải nắm được tốc độ sản xuất của mình.

Tạo sao điều này lại quan trọng như vậy? Nếu các Nhóm không biết được tốc độ của mình, Product Owner không thể tạo ra được lộ trình đáng tin cậy cho việc phát hành sản phẩm. Nếu không xác định được thời điểm phát hành tin cậy, công ty có thể sẽ thất bại và các nhà đầu tư sẽ mất trắng khoản tiền của họ.

Các công ty nào dù lớn hay nhỏ, mới hay cũ, được tài trợ hay không được tài trợ đều có thể phải đối mặt với vấn đề này. Tại một cuộc thảo luận gần đây về việc triển khai Scrum của Google ở một hội thảo tổ chức ở London, tôi đã hỏi một trong số 135 người về việc có bao nhiêu người đang triển khai Scrum và đã nhận được 30 phản hồi tích cực. Sau đó tôi hỏi tiếp nếu họ đang triển khai phương pháp phát triển lặp (iterative) theo chuẩn của Nokia. Phát triển lặp là nền tảng trong Tuyên ngôn Phát triển Phần mềm Linh hoạt (Agile Manifesto) – cho phép cung cấp phần mềm chạy tốt nhanh hơn và đều đặn hơn. Sau nhiều năm cải tiến với hàng trăm Nhóm Scrum, Nokia đã đưa ra một số yêu cầu cơ bản đối với phát triển lặp:

● Các phân đoạn phải được đóng khung về thời gian dưới sáu tuần.

11 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● Mã nguồn phải được kiểm thử bởi QA1cuối mỗi phân đoạn và phải chạy đúng yêu cầu.

Trong số 30 người nói rằng họ đang triển khai Scrum, chỉ có một nửa trong số đó nói họ đáp ứng được nguyên tắc đầu tiên của Tuyên ngôn Agile với các tiêu chuẩn của Nokia. Tôi hỏi tiếp nếu họ đáp ứng được các chuẩn của Nokia đối với Scrum:

● Một Nhóm Scrum phải có một Product Owner và biết rõ về người đó.

● Product Owner phải tạo ra một Product Backlog với các ước tính do Nhóm đưa ra. ● Nhóm phải có một Biểu đồ Burndown và biết được tốc độ của mình.

● Không ai ngoài Nhóm được phép can thiệp vào công việc của Nhóm trong suốt thời gian triển khai một Sprint.

Trong số 30 người đang triển khai Scrum, chỉ có 3 người qua được bài sát hạch của Nokia đối với một Nhóm Scrum. Chỉ những nhóm này mới nhận được các đầu tư trong tương lai từ các đối tác đầu tư mạo hiểm của tôi.

Giá trị mà cuốn sách của Henrik chính là ở chỗ nếu bạn tuân theo các phương pháp thực hành mà tác giả đưa ra, bạn sẽ có một Product Backlog, các ước tính cho Product Backlog, một Biểu đồ Burndown, và biết được tốc độ của Nhóm cùng với nhiều phương pháp thực hành quan trọng khác dành cho việc triển khai Scrum hiệu quả hơn.

Bạn sẽ qua được các sát hạch của Nokia về Scrum và đáng được cân nhắc để đầu tư. Nếu bạn bắt đầu khởi nghiệp, bạn có thể được rót vốn từ các nhóm đầu tư mạo hiểm. Chính bạn có thể là tương lai của phát triển phần mềm và là người kiến tạo một thế hệ các sản phẩm phần mềm hàng đầu.

**Ts. Jeff Sutherland, Đồng tác giả của Scrum**

1 QA = Quality Assurance : bộ phận đảm bảo chất lượng. Công việc cụ thể của QA của từng công ty có sự khác nhau tương đối nhiều. Một số nơi QA thiên về đảm bảo quy trình, một số nơi QA thực hiện cả công việc của QC (quality control – kiểm soát chất lượng), có nơi thậm chí QA làm công việc của kiểm thử viên (tester).

12 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Lời tựa của Mike Cohn**

Cả Scrum và XP (Extreme Programming) đều đòi hỏi các Nhóm cần phải hoàn thành một phần công việc có thể chuyển giao được cuối mỗi phân đoạn. Các phân đoạn đó được thiết kế ngắn và đóng khung về thời gian. Việc tập trung vào cung cấp các mã nguồn chạy tốt trong một khoảng thời gian ngắn có nghĩa là Scrum và XP không có thời gian dành cho các lý thuyết. Các phương pháp này không theo đuổi việc vẽ ra các mô hình UML hoàn hảo bằng một công cụ nào đó, viết tài liệu đầy đủ về các yêu cầu của sản phẩm, hoặc viết mã nguồn có thể tương thích với tất cả các thay đổi được tưởng tượng ra trong tương lai. Thay vào đó, các Nhóm Scrum và XP tập trung vào hoàn thành các công việc. Những Nhóm đó chấp nhận rằng họ có thể gặp vấn đề trong quá trình triển khai, nhưng họ cũng nhận thấy rằng cách tốt nhất để tìm ra các vấn đề là dừng việc tư duy về phần mềm đó ở cấp độ lý thuyết (phân tích, thiết kế) và chuyển ngay sang việc bắt tay vào xây dựng sản phẩm.

Điều này tương tự với việc tập trung vào thực hiện hơn là chú trọng vào các lý thuyết được đề cập trong cuốn sách này. Rõ ràng, Henrik Kniberg hiểu được điều này ngay từ khi bắt đầu. Anh ấy không đưa ra các mô tả dài dòng về Scrum là cái gì; anh ấy chỉ ra cho chúng ta một số trang web đơn giản về các nội dung đó. Thay vào đó, Henrik bắt đầu ngay lập tức với việc mô tả cách mà Nhóm của anh ấy đã quản lý và làm việc với Product Backlog của họ ra sao. Từ đó anh ấy chuyển qua tất cả các thành phần và thực hành khác để triển khai tốt một dự án với Agile. Không hề lý thuyết. Không tài liệu tham khảo. Không có các chú thích. Không cần các thứ đó. Cuốn sách của Henrik không triết lý về việc tại sao Scrum thực thi được hoặc tại sao bạn muốn thử cái này hoặc cái kia. Cuốn sách chỉ mô tả cách để một nhóm Agile làm việc hiệu quả.

Đó là lý do mà cuốn sách có tiêu đề phụ là “Chúng tôi đã Scrum như thế nào?”. Đây có thể không phải là cách mà bạn thực hành Scrum mà chỉ là cách Nhóm của Henrik đã làm. Có thể bạn sẽ hỏi tại sao lại cần quan tâm đến cách triển khai Scrum của một Nhóm nào đó. Câu trả lời là bạn nên quan tâm bởi vì tất cả chúng ta có thể học được cách triển khai Scrum tốt hơn thông qua các câu chuyện của các nhóm khác, đặc biệt là những nhóm đang làm tốt với Scrum. Không có và sẽ không bao giờ có danh sách “Các thực hành tốt nhất trong Scrum” bởi vì Nhóm và bối cảnh của dự án mới là những điều cốt lõi so với các vấn đề khác. Thay vì các cách thực hành tốt nhất, những gì chúng ta cần là biết được những thực hành hiệu quả và bối cảnh để triển khai chúng thành công. Đọc đầy đủ về các nhóm thành công và cách thức mà họ đã làm sẽ giúp bạn có những chuẩn bị tốt khi gặp phải các trở ngại trong quá trình triển khai Scrum và XP.

13 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Henrik cung cấp nhiều các cách thực hành hiệu quả cùng với những bối cảnh cần thiết để trợ giúp chúng ta hiểu tốt hơn về cách thực hành Scrum và XP trong các dự án của mình.

**Mike Cohn,**

**Tác giả cuốn sách “*Agile Estimating and Planning*” (Ước tính và Lập kế hoạch linh hoạt) và “*User Stories Applied for Agile Software Development”* (Ứng dụng User Story trong Phát triển Phần mềm Linh hoạt).**

14 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Lời nói đầu**

**Chào các bạn, Scrum đem lại hiệu quả trong công việc!**

Scrum hiệu quả trong công việc! Ít nhất là với chúng tôi (bao gồm khách hàng hiện tại của tôi ở Stockholm, tên của họ tôi không tiện nêu ra ở đây). Hy vọng nó cũng sẽ hiệu quả với công việc của các bạn! Có thể cuốn sách này sẽ trợ giúp bạn trong quá trình triển khai.

Đây là lần đầu tiên tôi thấy một phương pháp phát triển phần mềm (xin lỗi Ken2, một *framework* mới đúng) thực sự hiệu quả chứ không chỉ là sách vở. Đưa vào là dùng được ngay. Tất cả chúng tôi từ nhà phát triển, kiểm thử đến quản lý đều vui mừng vì điều này. Scrum đã trợ giúp chúng tôi vượt qua những tình huống khó khăn và cho phép chúng tôi duy trì sự tập trung và động lực mặc dù thị trường bất ổn và suy giảm về nhân sự.

Tôi không nên nói rằng tôi đã rất ngạc nhiên nhưng thực sự là như vậy. Sau khi đọc một số cuốn sách có vẻ thú vị về Scrum, nhưng phải là quá hay mới đúng (và tất cả chúng ta đều biết câu nói “khi một điều gì đó có vẻ quá tốt là đúng…”). Vì vậy cũng là hợp lý khi có một chút hoài nghi. Nhưng sau một năm triển khai Scrum tôi đã ấn tượng và sẽ tiếp tục sử dụng Scrum mặc định cho các dự án mới của mình cho tới khi nào có một lý do đủ mạnh để không dùng đến nó nữa.

2 Ken Schwaber, đồng tác giả của Scrum, cùng với Jeff Sutherland

15 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**1**

**Giới thiệu**

Bạn mới bắt đầu áp dụng Scrum trong tổ chức của mình, bạn đã áp dụng Scrum được một vài tháng, bạn mới có những kiến thức cơ bản về Scrum, bạn đã đọc một số quyển sách về nó, hay bạn đã có chứng chỉ Scrum Master. Xin chúc mừng bạn!

Nhưng bạn vẫn có thể gặp bối rối trong một số trường hợp.

Theo Ken Schwaber, Scrum không hẳn là một phương pháp luận, mà là một *khung làm việc* (framework). Điều đó có nghĩa là lý thuyết Scrum không hẳn sẽ nói cho bạn chính xác, cụ thể những việc cần phải làm.

Một tin tốt là tôi sẽ nói cho các bạn chính xác và rất chi tiết về cách mà tôi đã triển khai Scrum. Và, một tin xấu là những mô tả đó *chỉ là cách của tôi* khi áp dụng Scrum. Nó không có nghĩa là bạn phải áp dụng đúng theo những gì tôi làm. Thực thế, tôi cũng sẽ áp dụng theo cách khác trong những hoàn cảnh khác.

Sức mạnh và cốt lõi của Scrum là bắt buộc chúng ta phải thích ứng với từng hoàn cảnh cụ thể.

Phương pháp tiếp cận của chúng tôi là kết quả kinh nghiệm của một năm áp dụng Scrum cho nhóm phát triển khoảng 40 người. Công ty chúng tôi đã từng gặp các tình huống khó khăn trong công việc như phải làm việc ngoài giờ, có vấn đề nghiêm trọng về chất lượng, chữa cháy liên tục, trễ hạn, v.v. Chúng tôi đã quyết định sử dụng Scrum, nhưng không thực sự áp dụng cho hoàn toàn công ty, mà chỉ áp dụng cho cá nhân tôi. Với hầu hết mọi người trong nhóm phát triển lúc bấy giờ, "Scrum" là một thuật ngữ lạ lẫm như thể họ chỉ mới nghe thấy đâu đó ngoài hành lang, nó không có ý nghĩa cho công việc hàng ngày của họ.

Qua một năm áp dụng Scrum với tất cả các cấp trong công ty, chúng tôi đã thử nghiệm với các nhóm lớn nhỏ khác nhau (3 - 12 người), với độ dài Sprint khác nhau (2 - 6 tuần), với những cách định nghĩa “hoàn thành” khác nhau, với các định dạng khác nhau về Product Backlog và Sprint Backlog (Excel, Jira, thẻ chỉ mục), với các chiến thuật kiểm thử khác nhau, các phương thức demo

khác nhau cũng như các cách đồng bộ hóa các nhóm Scrum khác nhau, v.v. Chúng tôi cũng đã thử

16 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

nghiệm với phương pháp XP - theo các cách khác nhau như: xây dựng liên tục, lập trình theo cặp, phát triển hướng kiểm thử (Test Driven Development - TDD), hay kết hợp chúng với Scrum.

Đó là một quá trình học hỏi liên tục và chắc chắn câu chuyện sẽ không kết thúc ở đây. Tôi tin rằng chúng tôi vẫn sẽ tiếp tục cải tiến hơn nữa (nếu chúng tôi vẫn duy trì được các cuộc Họp Cải tiến Sprint - Sprint Retrospectives) và sẽ đạt được những hiểu biết mới để áp dụng tốt nhất Scrum trong hoàn cảnh của mình.

**Khuyến cáo**

Tài liệu này không đưa ra một phương pháp chuẩn mực trong việc triển khai Scrum! Chúng tôi chỉ mô tả một cách áp dụng Scrum, và kết quả sẽ được sàng lọc liên tục theo thời gian. Thậm chí, bạn có thể cho rằng mọi điều chúng tôi viết là sai.

Tất cả mọi điều trong tài liệu này phản ánh quan điểm của cá nhân tôi và nó không phải là một tuyên bố chính thức từ Crisp hay từ những khách hàng hiện tại của tôi. Với lý do này, tôi đã tránh đề cập đến bất kỳ sản phẩm hay con người cụ thể.

**Lý do tôi viết cuốn sách này**

Khi tìm hiểu về Scrum, tôi đã đọc một vài cuốn sách về Scrum và Agile có trên các website và các diễn đàn về Scrum, tôi đã nhận chứng chỉ từ Ken Schwaber, chất vấn ông ấy và đã dành nhiều thời gian để thảo luận với các đồng nghiệp của tôi. Nhưng có lẽ, một trong những nguồn thông tin giá trị nhất lại chính là những va vấp thực tế. Những va vấp thực tế đã chuyển thành những Lý thuyết và Thực hành v.v. và v.v. Bạn thực sự làm gì trong hoàn cảnh đó? Chúng đã giúp tôi xác định (hoặc tránh được) những sai lầm điển hình mà các thành viên mới tiếp cận Scrum đã gặp phải.

Do đó, đây là cơ hội để tôi gửi lại các bạn, câu chuyện về những va vấp của chúng tôi.

Tôi hi vọng rằng cuốn sách này sẽ đem lại cho các bạn những giải pháp hữu ích trong những hoàn cảnh tương tự. Xin hãy khai sáng cho tôi.

**Vậy Scrum là gì?**

Ồ, xin lỗi. Với trường hợp bạn mới tiếp xúc với Scrum và XP, xin hãy xem qua một số trang web sau3:

● http://agilemanifesto.org/

3 Bạn đọc có thể tiếp cận các tài liệu cơ bản về Agile bằng tiếng Việt tại http://hanoiscrum.net

17 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● http://www.mountaingoatsoftware.com/scrum

● http://www.xprogramming.com/xpmag/whatisxp.htm

Nếu bạn không đủ kiên nhẫn để làm điều đó, xin hãy cố xem qua chúng. Các thuật ngữ tối thiểu của Scrum sẽ được giải thích ở đó và bạn cần phải biết về chúng để đọc những dòng thú vị tiếp theo trong cuốn sách này.

18 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**2**

**Chúng tôi làm Product Backlog như thế nào**

Product Backlog là trái tim của Scrum. Tất cả sẽ được bắt đầu từ đây. Về cơ bản, Product Backlog là một danh sách ưu tiên hóa gồm các yêu cầu, các story, hoặc các tính năng của sản phẩm, hay thứ gì đó tương tự như vậy. Đó là những điều mà khách hàng mong muốn và được mô tả bằng các thuật ngữ của khách hàng.

Chúng ta gọi đó những *story* (hay đầy đủ là user story), hoặc đôi khi là các *hạng mục backlog* (backlog items).

Những story đó bao gồm các trường sau:

● **ID** (Số định danh) **–** một định danh duy nhất, là một con số tự tăng. Điều này nhằm tránh việc nhầm lẫn các story khi chúng ta đổi tên chúng.

● **Name** (Tên) – một cái tên ngắn, có tính mô tả cho story. Ví dụ “Xem lịch sử giao dịch của bạn". Phải đủ rõ ràng để Nhóm Phát Triển và Product Owner hiểu chính xác chúng ta đang nói về vấn đề gì, và cũng đủ rõ ràng để chúng ta phân biệt với các story khác. Thường từ 2 - 10 từ.

● **Importance** (Độ quan trọng) – Mức độ quan trọng của story mà Product Owner đánh giá. Ví dụ 10. Hoặc 150. Cao = quan trọng hơn.

o Tôi muốn tránh dùng thuật ngữ “độ ưu tiên” (priority) bởi vì độ ưu tiên 1 thường được xem là độ ưu tiên “cao nhất”, điều này sẽ trở nên rất tồi tệ nếu như bạn có ý định đặt một story khác có độ ưu tiên cao hơn. Bạn sẽ đặt độ ưu tiên đó như thế nào? Độ ưu tiên 0? Hay độ ưu tiên -1?

● **Initial estimate** (Ước tính ban đầu) – một ước tính ban đầu của đội dự án về khối lượng làm việc cần thiết để thực hiện story này so với các story khác. Đơn vị là điểm (story point) và nó tương ứng với “ý tưởng ngày công” (man-days).

o Hãy hỏi đội dự án “nếu bạn có thể lấy một số lượng người tùy ý đối với story này (không quá ít và không quá nhiều, thường là 2 người) sau đó vào phòng làm việc với đầy đủ thực phẩm, các bạn làm việc một cách hết sức tập trung, sau bao nhiêu ngày các bạn sẽ có thể bước ra khỏi phòng làm việc với một sản phẩm được coi là

19 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

hoàn thành, có thể demo được, đã được test, có thể phát hành được?”. Nếu câu trả lời là “với 3 người ở trong phòng làm việc tập trung sẽ mất khoảng 4 ngày” thì ước tính cho story đó là 12 điểm.

o Điều quan trọng không phải để bạn xác định được ước tính chính xác tuyệt đối (ví dụ một story 2 điểm sẽ mất 2 ngày để hoàn thành), điều quan trọng là cần ước tính với độ chính xác tương đối (ví dụ một story 2 điểm sẽ chiếm một nửa khối lượng công việc so với story 4 điểm)

● **How to demo** (Cách để demo) - một bản mô tả ở mức độ cao về cách demo tính năng này trong buổi demo4cuối Sprint. Về cơ bản thường là một đặc tả kiểm thử đơn giản. “Làm thế này, sau đó làm thế này, thì cái này sẽ xảy ra".

o Nếu bạn triển khai TDD 5(Phát triển Hướng Kiểm Thử), mô tả này có thể được sử dụng như một mã giả (pseudo-code) cho mã kiểm thử chấp nhận (acceptance test).

● **Notes** (Các lưu ý) – bất kỳ thông tin nào khác, dùng để phân loại, tham chiếu tới các nguồn tài nguyên khác, v.v. Thường là rất ngắn gọn.

| **PRODUCT BACKLOG (Ví dụ)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | **ID Tên** | **Độ quan trọng** | **Ước tính** | **Cách để demo** | **Lưu ý** |
| 1 | Gửi tiền vào | 30 | 5 | Đăng nhập, mở trang gửi tiền, gửi vào 10€, chuyển sang trang tài khoản của tôi và kiểm tra tài khoản nó đã được tăng thêm 10€. | Cần một biểu đồ tuần tự UML. Chưa cần quan tâm  tới việc mã hóa ở thời điểm  này. |
| 2 | Xem lịch sử giao dịch của | 10 | 8 | Đăng nhập, bấm vào “Các giao dịch”. Thực hiện việc gửi tiền. Quay lại trang | Sử dụng kỹ thuật phân trang để tránh việc phải truy vấn CSDL lớn. Thiết kế tương |

4 Trong tài liệu này, tác giả dùng từ Demo để nói về buổi Họp Sơ kết (Sprint Review) theo Scrum Guide.

5 TDD = Test-Driven Development: một chiến thuật phát triển phần mềm được điều hướng bởi kiểm thử. Còn gọi là Test-First, phương pháp này đòi hỏi lập trình viên phải viết ra các test case trước khi lập trình. Đây là một phương pháp thực hành quan trọng trong XP. Xem thêm chương “Chúng tôi kết hợp Scrum và XP như thế nào”.

20 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

|  | bạn |  |  | “Các giao dịch” và kiểm tra thông tin. | tự trang xem danh sách người dùng. |
| --- | --- | --- | --- | --- | --- |

Chúng tôi đã thử nghiệm với rất nhiều trường khác, nhưng cuối cùng, sáu trường trên là những trường mà chúng ta thường sử dụng nhất qua các Sprint.

Chúng ta có thể làm Product Backlog với một tài liệu Excel đã mở chức năng chia sẻ (nhiều người dùng có thể đồng thời chỉnh sửa tập tin). Chính thức thì Product Owner sở hữu tài liệu này, nhưng chúng ta muốn những người khác trong Nhóm Phát Triển cũng biết về nó. Có nhiều khi một nhà phát triển muốn mở tài liệu này ra để làm rõ hơn một điều gì đó hoặc thay đổi một ước tính nào đó.

Cũng vì thế, chúng ta không đặt tài liệu này trong một kho kiểm soát phiên bản (version control repository); thay vào đó chúng ta đặt nó vào một ổ đĩa chia sẻ. Đây là cách đơn giản nhất để cho phép nhiều người sửa đồng thời mà tài liệu không bị khóa hay bị xung đột.

Hầu hết những tài liệu khác có thể để ở kho kiểm soát phiên bản.

**Các trường khác**

Đôi khi chúng ta sử dụng thêm các trường khác trong Product Backlog, mục đích là làm cho Product Owner cảm thấy thuận tiện hơn khi sắp xếp độ ưu tiên.

● **Track** (Nhánh) – một sự phân loại cho story, ví dụ “back office” hoặc “optimization”. Theo cách đó, Product Owner có thể dễ dàng lọc ra tất cả các hạng mục “optimization” và đặt mức ưu tiên thấp, v.v..

● **Components** (thành phần) – Thường là các ô kiểm (checkbox) trong tài liệu Excel, ví dụ “database, server, client”. Đây là nơi mà Nhóm Phát Triển hoặc Product Owner có thể xác định các thành phần kỹ thuật nào sẽ tham gia vào quá trình triển khai story. Điều này hữu ích khi bạn có nhiều nhóm Scrum, ví dụ một nhóm “back office” và một nhóm client, và bạn muốn các nhóm dễ dàng chọn các story để thực hiện.

● **Requestor** (người yêu cầu) – có thể Product Owner muốn biết khách hàng hoặc người liên quan nào yêu cầu hạng mục này, để có thể gửi các phản hồi trong quá trình thực hiện.

● **Bug tracking ID** (mã bug) – nếu bạn có một hệ thống theo dõi lỗi riêng, như chúng tôi sử dụng Jira, sẽ rất hữu ích để theo dõi các lỗi liên quan tới một story.

21 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Làm thế nào chúng ta giữ Product Backlog ở mức nghiệp vụ 6**

Nếu Product Ownder biết về kỹ thuật, anh ta có thể thêm các story như “Đặt chỉ mục cho bảng Event”. Tại sao anh ta làm việc này? Mục tiêu thực sự có thể là một điều gì đó chẳng hạn như “tăng tốc độ tìm kiếm các event trên form dành cho back-office”.

Điều đó có nghĩa rằng các chỉ mục không phải là nút cổ chai dẫn tới form tìm kiếm bị chậm. Nó có thể là một thứ gì đó hoàn toàn khác. Nhóm Phát Triển sẽ là người phải tìm cách giải quyết vấn đề, vì thế Product Owner nên tập trung vào các mục tiêu nghiệp vụ hơn là kỹ thuật.

Khi tôi nhìn thấy các story mang quá nhiều tính chất kỹ thuật, tôi thường hỏi Product Owner một chuỗi các câu hỏi “tại sao” cho tới khi chúng tôi tìm ra mục tiêu ẩn chứa trong đó. Sau đó chúng tôi phân tích lại story theo hướng của các mục tiêu ẩn bên dưới nó (đó là “tăng tốc độ tìm kiếm các event trong biểu mẫu tìm kiếm của back office”). Mô tả kỹ thuật ban đầu kết thúc như một ghi chú (“Đặt chỉ mục cho bảng Event có thể giải quyết vấn đề này”).

6 Không giống như đặc tả hệ thống (System Requirement Specification), các user story là cách thể hiện yêu cầu người dùng ở dạng cô đọng và đơn giản nhất, dùng ngôn ngữ người dùng để thể hiện. Việc này có tác dụng lớn cho việc thảo luận và tìm hiểu nhu cầu của người dùng theo cách chân thực nhất: thực sự người dùng muốn gì? chức năng này để giải quyết nghiệp vụ nào? Do vậy, việc giữ Product Backlog ở mức nghiệp vụ là rất quan trọng.

22 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**3**

**Chúng tôi chuẩn bị Họp Kế hoạch Sprint như thế nào**

OK, chúng ta sẽ nắm bắt được ngày lên kế hoạch cho Sprint một cách nhanh chóng thôi. Một bài học chúng tôi học đi học lại đó là:

**Bài học:** Hãy chắc chắn Product Backlog đã định hình trước khi tiến hành Họp Kế hoạch Sprint.

Điều đó có nghĩa là gì ? Đó là tất cả các story được xác định một cách đầy đủ hay chưa? Đó là việc ước lượng (estimate) đã chính xác chưa ? Đó là thứ tự ưu tiên đã được xác định chưa? Nếu câu trả lời là: chưa, chưa và chưa! Có nghĩa là:

● Product Backlog chưa được coi là xong và Product Backlog cần phải được thực hiện cho xong đã! (chẳng lẽ bạn phải tưởng tượng ra nó sao?)

● Có một Product Backlog và một Product Owner (cho từng sản phẩm).

● Tất cả các công việc sẽ có đánh giá về tầm quan trọng của chúng, các đánh giá này là khác nhau.

o Thực tế, sẽ không có vấn để gì nếu các công việc có độ quan trọng thấp hơn có cùng giá trị, do vậy dù thế nào các công việc này sẽ không được nâng mức độ lên trong suốt thời gian lên kế hoạch Sprint.

o Bất kỳ một story nào mà Product Owner tin có một khả năng nào đó được đưa vào trong Sprint tiếp theo thì nên có duy nhất một mức độ quan trọng.

o Đánh giá độ quan trọng chỉ dùng để bố trí các công việc cho phù hợp. Do đó nếu công việc A có độ quan trọng là 20 và công việc B có độ quan trọng là 100, có nghĩa là B quan trọng hơn A. Nó không có nghĩa là B quan trọng hơn A gấp 5 lần. Nếu B có mức đánh giá quan trọng là 21 thì nó sẽ vẫn có nghĩa như khi nó có độ quan trọng là 100!

o Sẽ rất có ích nếu ta để một khoảng giãn cách khi đánh giá mức độ quan trọng giữa các công việc khác nhau, chẳng hạn trong trường hợp công việc C đưa thêm vào có độ quan trọng hơn A nhưng nhỏ hơn B. Tất nhiên bạn sẽ có thể sử dụng độ quan trọng cho C là 20.5, nhưng điều đó không hay lắm, thay vì thế chúng ta nên để một khoảng giãn cách giữa các công việc.

23 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● Product Owner phải *hiểu rõ* từng story (thông thường anh ta chính là tác giả, nhưng trong một số trường hợp người khác bổ sung yêu cầu mà Product Owner có thể xác định độ ưu tiên). Anh ta không cần biết chính xác chúng được triển khai ra sao, nhưng anh ta sẽ phải hiểu tại sao có story đó.

**Lưu ý:** Những người không phải là Product Owner có thể bổ sung story vào Product Backlog. Nhưng họ không được gán độ quan trọng cho công việc đó, quyền này hoàn toàn chỉ dành cho Product Owner. Product Owner không được bổ sung thêm các ước lượng (estimate), quyền này là thuộc hoàn toàn của Nhóm Phát Triển.

Một số hướng khác chúng tôi đã thử và đánh giá:

● Sử dụng Jira 7(hệ thống giám sát lỗi chúng tôi dùng) để quản lý Product Backlog. Tuy nhiên, hầu hết Product Owner đều cảm thấy khó thích ứng với hệ thống này. Excel là một cách hay và dễ dàng để thao tác trực tiếp. Bạn có thể dễ dàng đặt màu, sắp xếp lại các công việc, bổ sung thêm cột mới, các lưu ý, có thể xuất và nhập dữ liệu, v.v..

● Sử dụng các công cụ hỗ trợ quy trình agile như VersionOne, ScrumWorks, XPlanner, v.v. Chúng tôi đã không quan tâm nhiều tới việc đánh giá các công cụ này nhưng có thể chúng tôi sẽ thực hiện việc đó trong tương lai.

7Jira (www.atlassian.com/software/jira) là một ứng dụng web dùng cho các nhóm phát triển phần mềm với công dụng chính là quản lí và theo dõi lỗi.

24 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**4**

**Chúng tôi họp Kế hoạch Sprint ra sao**

Lập kế hoạch cho Sprint là buổi họp rất quan trọng, đây là sự kiện quan trọng nhất trong Scrum (theo quan điểm của tôi về vấn đề này). Buổi họp lập kế hoạch cho Sprint được thực hiện không tốt có thể làm ảnh hưởng rất lớn tới toàn bộ Sprint.

Mục đích của buổi họp kế hoạch là cung cấp thông tin đầy đủ cho Nhóm 8giúp họ có thể có đầy đủ thông tin làm việc trong vài tuần, để Product Owner tin tưởng nhóm có thể hoàn thành các công việc đó..

OK, điều này quả thực còn mơ hồ. Kết quả cụ thể của buổi họp lập kế hoạch Sprint là: ● Xác định mục tiêu của Sprint.

● Danh sách thành viên của Nhóm (mức độ cam kết của họ, nếu không phải là 100%). ● Sprint Backlog (= danh sách các công việc thực hiện trong Sprint).

● Xác định được ngày demo Sprint (Sơ kết Sprint).

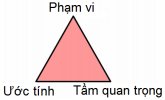
● Xác định thời gian và địa điểm cho buổi Họp Scrum hằng ngày (Daily Scrum). **Tại sao Product Owner phải có mặt**

Đôi khi Product Owner không sẵn sàng bỏ thời gian để họp lập kế hoạch Sprint với Nhóm. “Các anh em, tôi đã đưa ra đây những gì tôi muốn. Tôi không có thời gian để tham dự buổi họp lập kế hoạch của các bạn”. Đó là một vấn đề tương đối nghiêm trọng.

Lý do tại sao toàn bộ Nhóm và Product Owner phải có mặt tại buổi họp lập kế hoạch Sprint, bởi vì mỗi tính năng của sản phẩm chứa ba tham số mà có mối quan hệ phụ thuộc lẫn nhau.

8 Trong tài liệu này, tác giả dùng Nhóm (The Team) để chỉ Nhóm Phát triển. Đây là cách dùng từ cũ. Từ tháng 10- 2011, Ken Schwaber và Jeff Sutherland thống nhất cách dùng từ khác trong bản cập nhật của Scrum Guide (http://www.scrum.org/scrumguides/). Theo đó, có ba vai trò trong một Nhóm Scrum, bao gồm Product Owner, Scrum Master và Nhóm Phát triển (Development Team).

25 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



Phạm vi và tầm quan trọng được xác định bởi Product Owner. Ước tính (estimate) do Nhóm thực hiện. Trong buổi họp lập kế hoạch Sprint, ba tham số này liên tục được điều chỉnh bằng đối thoại trực tiếp giữa Nhóm và Product Owner.

Thường các Product Owner bắt đầu buổi họp bằng cách tóm tắt mục đích mong muốn của anh ta đối với Sprint và các tính năng quan trọng nhất của sản phẩm. Tiếp theo, Nhóm sẽ thực hiện ước lượng thời gian cho mỗi story, bắt đầu với story quan trọng nhất. Khi thực hiện việc này, đội sẽ đưa ra các câu hỏi quan trọng về phạm vi – “Tính năng ‘delete user’ có bao gồm việc hủy bỏ các giao dịch đang chờ đối với user đó hay không ?” Trong một vài trường hợp câu trả lời sẽ làm cả Nhóm ngạc nhiên, đòi hỏi Nhóm thay đổi ước tính của mình.

Trong một vài trường hợp, ước tính thời gian cho story sẽ không theo mong muốn của Product Owner. Điều này có thể cần yêu cầu anh ta thay đổi mức độ quan trọng của story. Hoặc thay đổi phạm vi của story, điều này sẽ dẫn tới Nhóm phải ước tính lại, v.v và v.v.

Hình thức cộng tác trực tiếp là nền tảng đối với Scrum, và thực tế là nền tảng của tất cả phương pháp phát triển phần mềm linh hoạt.

Bạn sẽ làm gì nếu Product Owner vẫn nhất định không dành thời gian tham gia buổi họp lập kế hoạch Sprint? Tôi thường thử một trong những chiến lược sau, lần lượt là:

● Cố gắng thuyết phục Product Owner hiểu tại sao sự tham gia trực tiếp của anh ấy là không thể thiếu và hy vọng anh ta thay đổi suy nghĩ.

● Thử để một ai đó trong Nhóm tình nguyện đóng vai như là người đại diện Product Owner trong suốt buổi họp. Nói với Product Owner rằng “Vì anh không thể tham dự buổi họp, chúng tôi sẽ để Jeff đại diện cho anh ở đây. Anh sẽ ủy nhiệm hoàn toàn cho Jeff việc thay đổi thứ tự ưu tiên, phạm vi của story trong suốt cuộc họp. Tôi mong anh hãy truyền đạt mong muốn của anh càng nhiều càng tốt trước khi cuộc họp diễn ra. Nếu anh không thích Jeff đại diện, anh có thể gợi ý cho tôi một ai khác, miễn là người này sẽ có thể tham gia cuộc họp với chúng tôi từ đầu tới cuối.”

● Cố gắng thuyết phục nhà quản lý bổ nhiệm một Product Owner mới.

26 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● Hoãn thực hiện Sprint cho tới khi Product Owner sắp xếp được thời gian để tham dự buổi họp. Trong khi đó, từ chối cam kết về bất cứ việc bàn giao sản phẩm nào. Để cho Nhóm làm bất cứ thứ gì mà họ cảm thấy quan trọng nhất trong mỗi ngày trì hoãn đó.

**Tại sao chất lượng là thứ không thể thỏa hiệp**

Trong hình tam giác phía trên tôi đã cố ý tránh tham số thứ 4 là *chất lượng*. Tôi gắng phân biệt giữa *chất lượng bên trong* và *chất lượng bên ngoài*.

● Chất lượng bên ngoài là những gì người sử dụng hệ thống nhận được. Một giao diện người dùng chậm và không trực quan là một ví dụ về chất lượng bên ngoài kém.

● Chất lượng bên trong nói đến những vấn đề mà thường không nhìn thấy từ phía người sử dụng, nhưng nó có ảnh hưởng sâu sắc đến khả năng bảo trì hệ thống. Những thứ như tính nhất quán trong các thiết kế hệ thống, các trường hợp cần kiểm thử, mã nguồn dễ đọc, cải tiến mã nguồn, v.v..

Nhìn chung mà nói, một hệ thống với chất lượng bên trong cao có thể vẫn có chất lượng bên ngoài thấp. Nhưng một hệ thống với chất lượng bên trong thấp sẽ hiếm khi có được chất lượng bên ngoài cao. Rất khó để xây dựng một cái gì tốt đẹp dựa trên một nền tảng yếu kém.

Tôi coi chất lượng bên ngoài như một phần của mục tiêu chất lượng. Trong một vài trường hợp, khi cảm thấy nghiệp vụ đã hoàn hảo chúng ta có thể cho phát hành (release) một phiên bản của hệ thống mà giao diện người dùng vừa chậm, vừa xấu, sau đó phát hành một phiên bản tốt hơn sau này. Tôi đưa sự đánh giá đó cho Product Owner, vì anh ta là người chịu trách nhiệm xác định phạm vi.

Tuy nhiên, chất lượng bên trong không đưa ra để thảo luận. Đó là trách nhiệm của Nhóm để đảm bảo chất lượng hệ thống trong mọi tình huống và điều này không thể thỏa hiệp một cách đơn giản được. Không bao giờ.

(Mà, OK, hầu như là không bao giờ)

Vậy chúng ta nói về sự khác biệt giữa các vấn đề về chất lượng bên trong và bên ngoài như thế nào?

Trong trường hợp Product Owner nói rằng “OK các chàng trai, tôi tôn trọng ước định thời gian của các bạn là 6 điểm story (story point), nhưng tôi chắc chắn các bạn có một vài thay đổi nhanh để thực hiện việc này với một nửa thời gian nếu các bạn đặt toàn tâm toàn ý vào nó.”

27 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

A ha! Anh ta đang thử sử dụng chất lượng bên trong như một tham số. Làm sao bạn biết? Bởi vì anh ta muốn chúng tôi giảm ước tính story point mà không “trả chi phí” cho phần giảm bớt đó. Từ “thay đổi nhanh” kích hoạt một cảnh báo trong đầu bạn …

Và tại sao chúng tôi không cho phép điều này?

Với kinh nghiệm của mình tôi thấy việc hy sinh chất lượng bên trong luôn luôn là một ý tưởng rất tệ hại. Việc tiết kiệm thời gian sẽ trả giá đắt về chi phí trong cả ngắn hạn và dài hạn. Khi nền tảng mã nguồn được phép bắt đầu với chất lượng tồi thì rất khó để làm nó chất lượng trở lại sau này.

Thay vào đó tôi cố hướng cuộc thảo luận tập trung vào mục tiêu cần thực hiện. “Nếu với Product Owner việc cần cho ra đời tính năng sớm là quan trọng, chúng ta có thể giảm bớt mục tiêu đi để thực hiện nó nhanh chóng hơn? Có lẽ chúng ta có thể đơn giản hóa việc quản lý lỗi và thực hiện tính năng ‘quản lý lỗi nâng cao’ trong một story khác trong tương lai ? Hoặc chúng ta giảm độ ưu tiên của một số tính năng khác trong sản phẩm để có thể tập trung vào tính năng quan trọng hiện tại?”

Khi Product Owner hiểu được rằng chất lượng bên trong không thể thương lượng được, thay vào đó anh ta thường nhanh chóng hướng tới thảo luận về các tham số khác.

**Các cuộc họp lập kế hoạch Sprint kéo dài …**

Điều khó nhất trong cuộc họp lập kế hoạch Sprint là:

● Mọi người không nghĩ họ sẽ mất quá nhiều thời gian cho cuộc họp.

● … thực tế lại như vậy !

Mọi thứ trong Scrum chỉ được phép nằm trong một khung thời gian (time-box). Tôi thích buổi họp tuân theo quy tắc nhất quán và đơn giản như thế. Chúng tôi cố gắng tuân thủ quy tắc đó.

Vì chúng tôi sẽ làm gì khi một buổi họp lập kế hoạch cho Sprint đã được đặt trong mộ khung thời gian sắp tiến gần tới thời điểm kết thúc và không có dấu hiệu cho thấy đã xác định được mục tiêu Sprint hay Sprint Backlog? Chúng ta có cắt ngắn nó không??? Hay chúng ta kéo dài cuộc họp thêm một giờ? Hoặc chúng ta dừng cuộc họp và tiếp tục vào ngày tiếp theo?

Điều này thường xuyên xảy ra, đặc biệt đối với những Nhóm mới. Vậy bạn phải xử lý thế nào? Tôi không biết. Nhưng chúng tôi đã làm gì ? Ồ, được thôi, tôi thường quyết định một cách dứt khoát ngắn gọn. Chấm dứt cuộc họp. Để Sprint đấy. Cá biệt hơn, tôi nói với Nhóm và Product Owner “vậy thôi, cuộc họp này kết thúc trong vòng 10 phút nữa. Chúng ta không thực sự có nhiều kế hoạch cho Sprint này. Chúng ta sẽ thực hiện những gì hiện chúng ta có chứ, hay chúng ta sẽ tổ

28 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

chức họp thêm 4 tiếng vào 8 giờ sáng ngày mai nhé?”. Các bạn có thể đoán xem họ sẽ trả lời những gì … : :o)

Tôi đã thử kéo dài thêm cuộc họp. Điều đó thường không đạt được điều gì, bởi vì mọi người mệt mỏi. Nếu họ không đưa ra được một kế hoạch tử tế trong 2-8 giờ (hoặc theo khung thời gian bạn đề ra), họ có thể cũng sẽ không làm thêm được gì trong thời gian thêm đó. Lựa chọn tiếp theo thực tế là OK hơn, đó là hẹn một cuộc họp mới vào ngày tiếp theo. Ngoại trừ trường hợp mọi người thiếu kiên nhẫn và muốn tiếp tục Sprint, và không muốn mất thêm thời gian để lập kế hoạch.

Vì vậy tôi cắt bớt thời gian đi. Và tất nhiên, Sprint sẽ vẫn chưa ra đâu vào đâu cả. Tuy nhiên, bù vào đó, Nhóm đã có được một bài học rất có giá trị và buổi họp lập kế hoạch tiếp theo sẽ hiệu quả hơn nhiều. Hơn nữa, mọi người sẽ ít phản kháng hơn khi bạn đề xuất thời gian họp mà trước đó họ nghĩ là quá dài.

Hãy học để tuân thủ khung thời gian của bạn, học để thiết lập độ lớn của khung thời gian sát với thực tế hơn. Điều này áp dụng cho cả việc xấc định độ dài cuộc họp cũng như Sprint.

**Lịch trình Họp Kế hoạch Sprint**

Có một số kiểu lên lịch họp cơ bản sẽ giúp bạn giảm bớt nguy cơ trong việc chia điểm dừng trong phạm vi khung thời gian.

Đây là ví dụ về một kiểu lên lịch chúng tôi sử dụng.

**Họp Kế hoạch Sprint** 13:00 – 17:00 (nghỉ giải lao 10 phút sau mỗi giờ họp)

● **13:00 – 13:30**. Product Owner trình bày mục tiêu của Sprint và tóm tắt Product Backlog. Địa điểm triển khai, thời gian và ngày triển khai Sprint được thiết lập.

● **13:30 – 15:00**. Nhóm Phát triển ước tính thời gian, và chia thành các tác vụ cần thiết. Product Owner cập nhật đánh giá mức độ quan trọng cho các hạng mục. Các tác vụ được làm rõ. “Cách thức tiến hành” được điền vào cho tất cả các hạng mục có tầm quan trọng cao.

● **15:00 – 16:00**. Nhóm chọn story để đưa vào Sprint. Thực hiện tính toán tốc độ để đảm bảo tính khả thi.

● **16:00 – 17:00**. Chọn thời gian và địa điểm để họp Scrum hàng ngày (nếu có khác biệt với Sprint trước đó). Chia nhỏ hơn các story vào các đầu việc phải làm (tasks).

29 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Lịch tiến độ không nhất thiết bắt buộc y nguyên như vậy. Scrum Master có thể đưa ra các khung thời gian ngắn hay dài hơn nếu thấy cần thiết trong quá trình họp.

**Xác định thời gian cho Sprint**

Một trong những kết quả cần đạt được trong cuộc Họp Kế hoạch Sprint là xác định được ngày triển khai Sprint. Điều đó có nghĩa bạn phải quyết định khung thời gian cho Sprint.

Vậy một Sprint kéo dài bao lâu là tốt ?

Sprint ngắn là tốt. Chúng cho phép công ty "linh hoạt”, ví dụ, thay đổi định hướng thường xuyên. Sprint ngắn = chu kỳ phản hồi ngắn = chuyển giao thường xuyên hơn = phản hồi khách hàng thường xuyên hơn = tốn kém thời gian đi sai hướng ít hơn = học và cải tiến nhanh hơn, v.v..

Tuy thế, Sprint dài cũng tốt. Nhóm có nhiều thời gian hơn để tạo đà, họ có nhiều thời gian hơn để khắc phục các vấn đề mà vẫn đảm bảo mục tiêu của Sprint, bạn ít bù đầu hơn cho các buổi họp lập kế hoạch Sprint, tổ chức demo, v.v..

Nhìn chung mà nói Product Owner thích các Sprint ngắn còn nhà phát triển thì thích các Sprint dài. Do đó độ dài của Sprint cần phải dung hòa. Chúng tôi đã thử nghiệm rất nhiều về vấn đề này và rút ra độ dài thích hợp cho chúng tôi là: 3 tuần. Hầu hết các Nhóm của chúng tôi (không phải tất cả) làm việc với các Sprint 3 tuần. Nó đủ ngắn để đảm bảo cho chúng tôi sự cộng tác linh hoạt, đủ dài cho Nhóm đạt được mạch làm việc và khắc phục các vấn đề nảy sinh trong Sprint.

Chúng tôi kết luận một điều là: *Thử nghiệm* độ dài khác nhau cho Sprint ban đầu. Đừng lãng phí quá nhiều thời gian cho việc *phân tích*, chỉ chọn độ dài vừa phải và ghi nhận kết quả đối với một hoặc hai Sprint, sau đó thay đổi độ dài.

*Tuy nhiên,* khi bạn quyết định độ dài mà bạn thấy tốt nhất, hãy *duy trì* nó trong một khoảng thời gian nhất định. Sau vài tháng thử nghiệm, chúng tôi thấy 3 tuần là hợp lý. Do đó, chúng tôi thực hiện các Sprint với thời hạn 3 tuần. Đôi khi cảm giác nó cũng hơi dài, đôi khi lại cảm thấy quá ngắn. Nhưng bằng cách duy trì cùng một độ dài, nó sẽ giúp tạo ra cho Nhóm một nhịp đập (heartbeat) mà mọi người cảm thấy thoải mái tham gia. Không có tranh luận nào về ngày phát hành bởi vì mọi người biết rằng cứ 3 tuần thì có một lần phát hành (release).

**Xác định mục tiêu Sprint**

Điều này cần làm trong toàn bộ thời gian họp. Tại một số thời điểm trong buổi họp lập kế hoạch tôi hỏi “vậy mục tiêu của Sprint này là gì?” và mọi người ngây lại nhìn chằm chằm vào tôi và Product Owner.

30 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Vì một vài lý do rất khó để đạt được mục tiêu của Sprint. Nhưng tôi đã thấy điều đó thực đáng giá để theo sát mục tiêu đề ra. Thà nhắc tới mục đích tuy phá đám giữa chừng còn hơn là không nhắc lần nào. Mục tiêu có thể là “làm ra nhiều tiền hơn” hoặc “hoàn thành 3 story có mức ưu tiên cao nhất” hoặc “gây chú ý cho CEO” hoặc “làm cho hệ thống đủ tốt để đưa vào các phiên bản beta đang chạy” hoặc “bổ sung các hỗ trợ cơ bản cho văn phòng” hoặc là bất cứ thứ gì. Điều quan trọng là nó sẽ thuộc vấn đề về nghiệp vụ, chứ không phải là khía cạnh kỹ thuật. Điều này tức là nói về những thứ mà những người bên ngoài nhóm có thể hiểu.

Mục tiêu của Sprint sẽ trả lời câu hỏi cơ bản “*Tại sao* chúng ta thực hiện Sprint này? Tại sao tất cả chúng ta vẫn tiếp tục công việc thay vì đi nghỉ?” Thực tế, một cách để giúp Product Owner làm rõ mục đích của Sprint là đặt ra các câu hỏi như vậy.

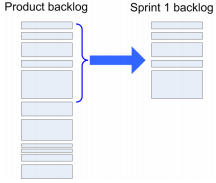
Mục tiêu không phải những thứ mà dễ dàng đạt được ngay. “Gây ấn tượng cho CEO” có thể là một mục tiêu tốt, nếu anh ta chưa ấn tượng bởi hệ thống hiện có. Trong trường hợp như vậy mọi người có thể về nhà mà vẫn đạt được mục tiêu của Sprint.

Mục tiêu của Sprint có thể có vẻ rất ngớ ngẩn và được nghĩ ra trong suốt thời gian lập kế hoạch Sprint, nhưng nó thường xuất hiện ở giữa Sprint khi mọi người bắt đầu lẫn lộn về những gì họ sẽ phải làm. Nếu bạn có một vài Nhóm Scrum (như tôi có) làm việc với nhiều sản phẩm khác nhau, sẽ rất hữu ích để liệt kê ra các mục tiêu của Sprint của tất cả các đội trong một trang wiki (hay bất cứ thứ gì) và đặt chúng tại nơi nổi bật để mọi người trong công ty (không chỉ quản lý cấp cao) biết những gì công ty đang thực hiện - và lý do thực hiện chúng!

**Nhóm quyết đưa story nào vào Sprint như thế nào?**

Một trong những hoạt động chính của buổi Họp kế hoạch Sprint là quyết định story nào sẽ được đưa vào Sprint. Cụ thể hơn, các story gì sẽ được chuyển từ Product Backlog vào Sprint Backlog.

31 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



Hãy nhìn vào hình ở phía trên. Mỗi một hình chữ nhật thể hiện 1 story, được sắp xếp theo độ quan trọng. Tính năng quan trọng nhất nằm ở đỉnh danh sách. Kích thước của mỗi hình chữ nhật thể hiện kích thước của tính năng đó (ví dụ: các story được ước tính bằng số điểm). Chiều cao của hình ngoặc nhọn màu xanh thể hiện *tốc độ ước tính* của nhóm, ví dụ, bao nhiêu điểm mà Nhóm tin rằng có thể hoàn thành trong Sprint tiếp theo.

Sprint Backlog ở bên phải là hình ảnh của các story lấy từ Product Backlog. Nó thể hiện danh sách các tính năng mà Nhóm sẽ phải thực hiện trong Sprint này.

*Nhóm* quyết định có bao nhiêu story đưa vào Sprint. Không có Product Owner hay bất kỳ ai khác can thiệp vào quyết định này.

Điều này nảy sinh 2 câu hỏi :

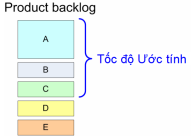
1. Làm thế nào Nhóm quyết định đưa story nào vào Sprint?

2. Product Owner làm thế nào để tác động đến các quyết định của họ?

Tôi sẽ bắt đầu với câu hỏi thứ hai.

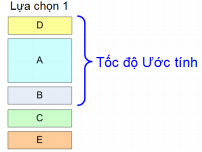
**Product Owner tác động thế nào để story nào đó được đưa vào Sprint?** Tôi sẽ kể về các tình huống dưới đây mà tôi gặp phải trong suốt buổi họp lập kế hoạch

32 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



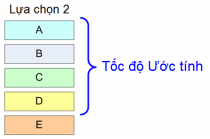
Product Owner không hài lòng D khi thấy story không được đưa vào Sprint. Anh ta sẽ có những lựa chọn gì trong suốt buổi họp lập kế hoạch?

Lựa chọn thứ nhất đó là ưu tiên hóa lại Product Backlog. Nếu anh ta nâng tầm quan trọng của story D lên cao hơn, Nhóm sẽ bắt buộc phải bổ sung nó vào Sprint (trong trường hợp này story C sẽ bị đẩy ra khỏi Sprint Backlog).

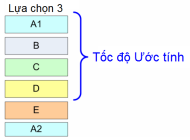


Lựa chọn thứ hai là thay đổi phạm vi – giảm phạm vi của story A cho tới khi Nhóm tin là story D sẽ phù hợp với Sprint này hơn.

33 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



Lựa chọn thứ ba là chia nhỏ story. Product Owner có thể quyết định rằng có một vài yếu tố của story A không thực sự quan trọng, do đó anh ta chia A thành hai story A1 và A2 với mức độ quan trọng khác nhau.



Như bạn thấy, mặc dù Product Owner thường không thể điều khiển được tốc độ ước tính, nhưng có nhiều cách anh ta có thể tác động để story nào đó được đưa vào Sprint.

34 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Nhóm quyết định đưa story vào Sprint như thế nào**

Chúng tôi sử dụng 2 kỹ thuật để làm việc này:

● Trực giác (Gut Feeling)

● Tính toán tốc độ (Velocity)

● **Scrum Master:** "Này các bạn, chúng ta có thể kết thúc story A trong Sprint này được không?" (ám chỉ tính năng quan trọng nhất trong các hạng mục của Product Backlog)

● **Lisa:** "À. Tất nhiên là được. Chúng ta có 3 tuần, và tính năng này cũng không khó lắm"

● **Scrum Master:** "OK, vậy với tính năng B thì thế nào?" (ám chỉ tính năng quan trọng thứ 2)

● **Tom và Lisa cùng trả lời:** "Vẫn không vấn đề gì"

● **Scrum Master:** "OK, vậy sau tính năng A, B thì làm tiếp C được không?" ● **Sam (hướng tới Product Owner):** "Tính năng C bao gồm việc xử lý Advanced Error?" ● **Product Owner:** "Không, cậu có thể bỏ qua việc đó, chỉ triển khai xử lý lỗi cơ bản." ● **Sam:** "nếu vậy thì C cũng ổn."

● **Scrum Master:** "OK, thế còn tính năng D?"

● **Lisa:** "ừm… "

● **Tom:** "Tôi nghĩ là chúng ta có thể."

● **Scrum Master:** "Chắc chắn 90%? hay chỉ 50%?"

● **Lisa và Tom:** "Hơn 90%".

● **Scrum Master:** "OK, vậy ta sẽ làm thêm D. Thế còn tính năng E?"

● **Sam:** "Có lẽ được."

35 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● **Scrum Master:** "90% hay 50%?"

● **Sam:** "Tôi nghĩ khoảng 50%".

● **Lisa:** "Tôi cũng thế"

● **Scrum master:** "OK, vậy ta để lại sau. Chúng ta sẽ hoàn thành các tính năng A, B, C, D. Chúng ta có thể hoàn thành E nếu được, nhưng không ai chắc chắn nên ta sẽ không đưa vào Sprint lần này. Mọi người thấy thế nào?"

● **Mọi người cùng trả lời:** "OK"

Kỹ thuật Gut Feel áp dụng tốt cho những nhóm nhỏ hay những Sprint ngắn.

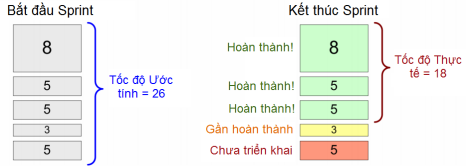
Kỹ thuật này bao gồm 2 bước:

1. Xác định *Tốc độ ước tính*

2. Tính toán xem bao nhiêu story có thể làm mà không vượt quá tốc độ ước tính.

Tốc độ là thước đo “lượng công việc hoàn thành”, ở đó, mỗi hạng mục được đặt một trọng số trong ước lượng ban đầu.

Hình ảnh dưới đây là một ví dụ về *tốc độ ước tính stimat v locit* ban đầu của một Sprint và *tốc độ th c t actual v locit* ở cuối Sprint đó. Mỗi hình chữ nhật là một story, con số bên trong ô chữ nhật đó là ước tính ban đầu.



36 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Lưu ý rằng tốc độ thực tế được dựa trên bản ước tính *ban đ u* của mỗi story. Bất kỳ thay đổi nào về thời gian ước tính hoàn thành tính năng trong suốt Sprint sẽ bị lờ đi.

Tôi đã nghe thấy phản hồi rằng: “Điều này hữu ích như thế nào? Tốc độ cao hay thấp có thể phụ thuộc vào rất nhiều yếu tố! Các lập trình viên yếu kém, sự đúng đắn của ước tính ban đầu, sự mở rộng phạm vi, không lập kế hoạch cho những xáo trộn trong suốt Sprint, v.v.!”

Tôi đồng ý rằng các con số này rất khô khan. Nhưng nó vẫn hữu ích, đặc biệt khi so sánh với việc không có gì để xem xét. Nó cung cấp cho bạn một số dữ kiện cứng. “Không kể những lý do trên, ở đây ta chỉ tính đến sự khác nhau tương đối giữa việc chúng ta đã dự kiến bao nhiêu và đã hoàn thành được bao nhiêu trong thực tế”.

Điều gì xảy đến với những story “gần hoàn thành” trong Sprint? Tại sao chúng ta không lấy một phần điểm của story đó để đưa vào tốc độ thực tế? Vâng, điều này nhấn mạnh đến thực tế của Scrum (và cả trong Phát triển phần mềm linh hoạt (Agile) lẫn phương pháp sản xuất tinh gọn nói chung (Lean Manufacturing9), đó là chỉ nói về những việc đã được thực sự hoàn thành, có thể chuyển giao, đã kết thúc. Giá trị của những thứ mới hoàn thành một nửa là số không (thực tế là bị bác bỏ). Xin hãy đọc cuốn sách “Managing the Design Factory” của Donald Reinertsen hay một trong những cuốn sách của Poppendieck10 để biết rõ thêm về điều này.

Vậy thì chúng ta cần thông qua thủ thuật phức tạp gì để có tốc độ ước tính?

Một cách rất đơn giản để ước tính tốc độ là xem lại những gì mà nhóm đã triển khai trước đây. Tốc độ của họ trong một vài Sprint gần đây là gì? Sau đó giả định tốc độ bám theo khoảng đó cho Sprint tới.

Kỹ thuật này được biết với cái tên *Th i ti t của ngà hôm ua* Nó chỉ phù hợp với những nhóm đã thực hiện được một số Sprint (khi đã có được số liệu thông kê) và những Sprint tiếp theo được hoàn thành với phương thức gần giống với những Sprint cũ, với cùng kích thước nhóm trong cùng điều kiện làm việc, v.v.. Điều này tất nhiên không phải lúc nào cũng được như vậy.

9 Lean Manufacturing (hay Lean Production): sản xuất tinh gọn, có xuất xứ trực tiếp từ Hệ thống sản xuất Toyota (Toyota Production System), là một phương pháp được xây dựng dựa trên ý tưởng cơ bản về loại bỏ các dư thừa không cần thiết trong chu trình phát triển sản phẩm, tối ưu hóa luồng sản xuất để tạo ra các giá trị cao nhất cho khách hàng. Lean có ảnh hưởng rộng lớn trong các ngành công nghiệp, rất gần gũi với Agile và có ảnh hưởng lớn trong các hoạt động thực tiễn của những người thực hành Agile. Một trong các phương pháp Agile được chuyển đổi gần như trực tiếp từ Lean Manufacturing sang thế giới phần mềm, là Lean Software Develpoment.

10 Tom Poppendieck và Mary Poppendieck, các tác giả của loạt sách về Phát triển Phần mềm Tinh gọn (Lean Software Development).

37 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Có một vấn đề phức tạp hơn là phải tính toán được nguồn lực cơ bản. Giả sử chúng tôi đang phải lên kế hoạch cho một Sprint kéo dài 3 tuần (15 ngày làm việc) với một nhóm 4 người. Lisa sẽ có 2 ngày nghỉ. Dave thì chỉ có thể làm việc với 50% thời gian và sẽ có 1 ngày nghỉ. Hãy liệt kê tất cả những điều đó…



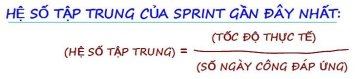
… 50 ngày công sẵn sàng cho Sprint này.

Giá trị trên có phải là tốc độ ước tính? Không! Do đơn vị ước tính của chúng tôi là điểm (story point), nên trong trường hợp này, đó mới chỉ là giá trị “số ngày công lý tưởng”. Một ngày công lý tưởng là một ngày làm việc tập trung, thực sự hiệu quả, không bị làm phiền, điều này rất hiếm. Ngoài ra, chúng tôi còn phải tính đến những việc không nằm trong dự kiến nhưng vẫn phát sinh trong Sprint, ví dụ như nhân viên nghỉ việc vì ốm, v.v.

Do vậy, *tốc độ ước tính* của chúng tôi chắc chắn phải nhỏ hơn con số 50. Nhưng nhỏ hơn bao nhiêu? Chúng tôi sẽ dùng đến khái niệm “hệ số tập trung” (focus factor) để xác định điều đó.



“Hệ số tập trung” là một số ước lượng độ tập trung công việc của nhóm. Một hệ số tập trung thấp có nghĩa là nhóm có thể bị phân tâm bởi nhiều việc khác hoặc thời gian ước tính của họ là quá lạc quan.

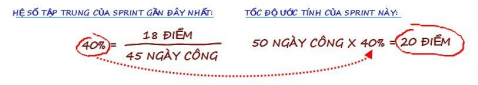


38 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Cách tốt nhất để xác định hệ số tập trung phù hợp là hãy dựa vào Sprint gần nhất (hay tốt hơn là dựa trên mức bình quân của một số Sprint gần nhất).

*Tốc độ th c t* là tổng các giá trị ước tính của tất cả các story được hoàn thành trong Sprint gần nhất.

Giả sử Sprint gần nhất có 18 điểm được hoàn thành với nhóm 3 thành viên bao gồm Tom, Lisa và Sam, làm việc trong 3 tuần với tổng số 45 ngày công. Bây giờ chúng tôi cần tính ra tốc độ ước tính cho Sprint tiếp theo. Để giải quyết những thứ phức tạp, nhóm có thêm thành viên mới gia nhập là Dave. Tính cả các kỳ nghỉ và một số thứ khác, chúng tôi có 50 ngày công cho Sprint tiếp theo.



Vì vậy, tốc độ ước tính của chúng tôi với Sprint tới là 20 điểm. Điều đó có nghĩa là nhóm nên thêm các story vào Sprint cho tới khi việc bổ sung đó xấp xỉ 20 điểm.



Trong trường hợp này, nhóm có thể chọn 4 story đứng đầu có tổng số điểm là 19, hoặc chọn 5 story với tổng số điểm là 24. Hãy bảo họ chọn giải pháp 4 story, do nó cho giá trị gần nhất với 20 điểm. Nếu không chắc chắn, hãy chọn ít story hơn.

Do 4 story có tổng số điểm là 19 nên tốc độ ước tính cuối cùng của họ cho Sprint này là 19.

39 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Kỹ thuật “thời tiết ngày hôm qua” khá là dễ áp dụng, nhưng hãy vận dụng nó với chút ít điều chỉnh cho phù hợp. Với trường hợp Sprint gần nhất có kết quả kém vì hầu hết các thành viên trong nhóm phải nghỉ ốm một tuần, khi đó có thể an toàn để bạn giả định rằng tình trạng không may như thể sẽ không lặp lại và bạn có thể ước tính với hệ số tập trung cao hơn cho Sprint tiếp theo. Với trường hợp nhóm vừa cài đặt được hệ thống build liên tục (continuous build system11), bạn có thể tăng hệ số tập trung tới một giá trị tốt hơn. Trong trường hợp nhóm có thêm một thành viên mới, chúng ta cần giảm hệ số tập trung vì phải tốn thêm chi phí đào tạo anh ta.v.v..

Mỗi khi có thể, hãy xem xét một vài Sprint gần đây và lấy trung bình cộng để có được những ước lượng thực tế hơn.

Điều gì sẽ đến nếu như nhóm vừa mới được thành lập và bạn không có chút dữ liệu thống kê nào? Hãy xem lại “hệ số tập trung” của các nhóm khác có cùng hoàn cảnh.Làm gì nếu bạn không tìm ra được nhóm nào tương tự? Hãy ước đoán “hệ số tập trung”. Một tin tốt là giá trị ước đoán đó chỉ áp dụng ở Sprint đầu tiên. Sau đó, bạn đã có được dữ liệu thống kê và bạn có thể tiếp tục tính toán và cải thiện giá trị các hệ số và tốc độ ước tính.

Thường thì “mặc định” hệ số tập trung sử dụng cho nhóm mới có giá trị vào khoảng 70%, đó là con số mà các nhóm của chúng tôi đạt được qua thời gian.

**Chúng ta s s dụng k thuật ước t nh nào?**

Tôi đã nói ở trên một số kỹ thuât: Trực giác, tính tốc độ dựa trên “tình hình ngày hôm qua” và tính tốc độ dựa trên số lượng ngày công khả dụng và hệ số tập trung ước tính.

Vậy chúng tôi sử dụng kỹ thuật nào?

Chúng tôi thường phối hợp tất cả các kỹ thuật để tìm được con số tương đối. Chúng không tốn nhiều thời gian.

Chúng tôi xem xét hệ số tập trung và tốc độ thực tế của Sprint gần nhất. Chúng tôi xem xét tổng nguồn lực sẵn sàng cho Sprint này và ước tính hệ số tập trung. Chúng tôi thảo luận về bất kỳ sự khác nhau giữa hai hệ số tập trung này và có những điều chỉnh cần thiết.

Một khi chúng tôi có được danh sách sơ bộ các story được đưa vào Sprint, tôi sử dụng kỹ thuật “Trực giác” để kiểm tra. Khi đó, tôi cũng yêu cầu nhóm tạm thời bỏ qua các con số trên và chỉ

11 Các hệ tự động build liên tục, còn gọi là các hệ tích hợp liên tục (Continuous Integration), như Maven, Hudson, CruiseControl v.v., là các phần mềm tự động tải về mã nguồn và các tài nguyên cần thiết từ kho lưu trữ mã nguồn, biên dịch và xây dựng các gói thực thi (executables). Nhờ vào quá trình tự động hóa này, Nhóm phát triển không phải biên dịch và build thủ công, tiết kiệm được công sức trong phát triển phần mềm.

40 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

nghĩ về việc “cảm thấy” có đưa vào Sprint này được không. Nếu cảm thấy quá nhiều, chúng tôi sẽ loại bỏ một hoặc hai story, và ngược lại.

Cuối ngày làm việc hôm đó, mục tiêu của chúng tôi chỉ đơn giản là xác định những story nào sẽ được đưa vào Sprint. Hệ số tập trung, nguồn lực khả dụng và Tốc độ ước tính chỉ có ý nghĩa để đạt được các mục tiêu đó khi kết thúc Sprint.

**Tại sao chúng tôi s dụng các th ch mục nde Card)**

Tại hầu hết cuộc họp lập kế hoạch Sprint, được sử dụng để xử lý các story trong Product Backlog. Uớc tính, đặt lại độ ưu tiên, phân loại và chia nhỏ các story, v.v.

Chúng tôi thực hiện điều này như thế nào?

Vâng, mặc định, các nhóm sẽ sử dụng máy chiếu, hiển thị backlog trên Excel, và một thành viên (thường là Product Owner hoặc Scrum Master) sẽ sử dụng bàn phím, duyệt lướt qua mỗi story và mời mọi người thảo luận. Cả nhóm và Product Owner sẽ thảo luận mức độ ưu tiên, thông tin chi tiết và tiến hành cập nhật trực tiếp trên Excel.

Nghe có vẻ rất tốt? Thực ra cũng không hẳn. Thường là rất tệ. Và điều tồi tệ hơn, nhóm thường không nhận ra cho đến khi gần kết thúc cuộc họp, khi đó, nhóm nhận ra rằng họ vẫn chưa thông qua được danh sách các story!

Một giải pháp tốt hơn nhiều là tạo ra các *thẻ chỉ mục* và *án ch ng lên tư ng* (hoặc một bảng lớn) Điều này cho ta cái nhìn rất trực quan nếu so với máy tính và máy chiếu, bởi vì:

● Mọi người có thể đứng gần và đi lại xung quanh đó = họ sẽ nhìn thấy thường xuyên và lưu ý lâu hơn.

41 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● Mọi người đều cảm bản thân mình được tham gia (không chỉ mỗi một anh chàng đang dùng bàn phím).

● Nhiều story có thể được chỉnh sửa đồng thời.

● Việc đánh giá lại mức độ ưu tiên rất đơn giản - chỉ việc di chuyển các thẻ chỉ mục.

● Sau cuộc họp, các chỉ mục có thể được giữ lại trong phòng làm việc của nhóm và sẽ được sử dụng như là bảng công việc trên tường (xem trang 59, "Chúng tôi xây dựng Sprint Backlog như thế nào").

Bạn có thể hoặc là tự tay viết *thẻ chỉ mục* hoặc (như chúng tôi thường làm) sử dụng một chương trình đơn giản để có thể in các thẻ trực tiếp từ Product Backlog.

*TB. hư ng tr nh scri t nà c trên bsit của tôi http://blog.crisp.se/henrikkniberg.*

**Lưu ý:** sau cuộc họp lập kế hoạch Sprint, Scrum Master của chúng tôi thường cập nhật lại Product Backlog trong Excel, với bất cứ thay đổi nào đã được thực hiện trên các thẻ chỉ mục. Vâng, đây chỉ là một chút rắc rối về mặt thủ tục, nhưng chúng tôi thấy rằng hoàn toàn chấp nhận được nếu nhìn lại tính hiệu quả của các *thẻ chỉ mục* trong các cuộc họp lập kế hoạch Sprint.

Một lưu ý về trường "Độ quan trọng". Điều quan trọng này được mô tả trong Product Backlog trên Excel khi in ra. Có nó, các *thẻ chỉ mục* sẽ dễ dàng được sắp xếp theo mức ưu tiên của công việc (thường thì chúng tôi đặt các hạng mục quan trọng hơn ở bên trái, thấp hơn ở bên phải). Tuy nhiên, khi đã dán xong trên bảng, ta có thể không cần để ý đến mức quan trọng của nó nữa và thay vào

42 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

đó, nhìn vào thứ tự sắp xếp trên bảng để biết được độ quan trọng. Nếu Product Owner đã hoán đổi vị trí các thẻ đó, đừng lãng phí thời gian để cập nhật lại thông tin về độ quan trọng trên các tờ giấy.

Chỉ cần cập nhật lại *mức* độ quan trọng của trong Product Backlog trên Excel sau cuộc họp.

Việc ước tính thời gian sẽ dễ hơn (và chính xác hơn) nếu ta chia các story thành các tác vụ. Trên thực tế, chúng tôi sử dụng thuật ngữ "hoạt động (activity)" vì ở Thụy Điển, từ “tác vụ (task)” có nghĩa là một cái gì đó *hoàn toàn* khác :o)

Điều này cũng cũng tốt và dễ dàng hơn để tiến hành với các *thẻ chỉ mục*. Bạn có thể chia nhóm làm việc thành từng cặp và cùng nhau chia nhỏ từng story một.

Cụ thể, chúng tôi thực hiện điều này bằng cách thêm vào các ghi chú nhỏ đính k m dưới mỗi story, mỗi ghi chú là một công việc cần thực hiện.



43 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



Chúng tôi không cập nhật lại Product Backlog trên Excel để thêm vào những công việc được phân tách ở trên, có hai nguyên nhân:

● Các công việc được tác nhỏ thường không cố định, tức là chúng thường bị thay đổi hoặc được định nghĩa lại trong suốt Sprint. Do đó, sẽ rất khó để tổng hợp vào Product Backlog trên Excel.

● Product Owner không cần phải tham gia vào những công việc ở mức độ chi tiết như vậy.

Chỉ với các thẻ chỉ mục viết các story, những mẫu giấy viết các công việc được tách ra có thể được tái sử dụng trong Sprint Backlog (xem thêm trang 45 "Chúng tôi xây dựng Sprint Backlog như thế nào").

**ịnh ngh a Hoàn thành**

Điều quan trọng là Product Owner và nhóm đồng thuận cho một định nghĩa rõ ràng về "hoàn thành". Một story được cho là hoàn thành khi tất cả mã nguồn đã được kiểm thử? Hoặc là chỉ khi được triển khai vào môi trường kiểm thử và được xác nhận bởi một nhóm kiểm thử tích hợp? Mỗi khi có thể, chúng tôi đều sử dụng định nghĩa hoàn thành "sẵn sàng để triển khai sản phẩm", nhưng đôi khi chúng tôi phải sử dụng định nghĩa hoàn thành "đã triển khai trên dịch vụ kiểm thử và sẵn sàng để kiểm thử chấp nhận".

Thời gian đầu, chúng tôi đã sử dụng danh sách kiểm tra chi tiết cho việc này. Nhưng bây giờ, chúng tôi chỉ cần nói "một story được hoàn thành chỉ khi kiểm thử viên trong nhóm Scrum chấp nhận". Điều đó có nghĩa là kiểm thử viên đảm bảo rằng Nhóm Phát triển đã hiểu được yêu cầu của Product Owner, và story “hoàn thành” đủ để vượt qua định nghĩa hoàn thành đã được chấp nhận.

44 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Chúng tôi đã nhận ra rằng các story cần được xử lý theo cách khác nhau. Một story có tên là "Biểu mẫu truy vấn người dùng " cần được xử lý rất khác so với story có tên "Hướng dẫn vận hành". Trong tính năng thứ hai, định nghĩa "hoàn thành" chỉ đơn giản là "được chấp nhận bởi nhóm vận hành". Đó là lý do tại sao cảm giác chung thường tốt hơn so với danh sách kiểm tra mô phạm.

Nếu bạn thường nhầm lẫn về định nghĩa hoàn thành (như chúng tôi đã vướng phải lúc đầu), bạn nên có hẳn một trường "định nghĩa của hoàn thành" trên mỗi story.

**ớc t nh thời gian s dụng Bộ bài Lập kế hoạch12**

Việc ước tính là hoạt động của cả nhóm - mọi thành viên cần tham dự vào việc ước tính tất cả các story. Tại sao vậy?

● Trong lúc lập kế hoạch, chúng tôi thường không biết chính xác ai sẽ làm phần việc nào trong story.

● Các story thường đòi hỏi vài người với một vài kỹ năng khác nhau (như thiết kế giao diện, lập trình, kiểm thử, v.v.).

● Để có một ước tính, mỗi thành viên trong nhóm cần hiểu biết story đó làm về cái gì. Với việc yêu cầu mọi người tham gia ước tính mỗi hạng mục, chúng tôi đã đảm bảo mọi thành viên đều hiểu rõ về hạng mục đó. Việc này tăng sự thông suốt giữa các thành viên để giúp đ nhau trong lúc thực hiện Sprint. Nó cũng tăng sự hiểu biết về những câu hỏi quan trọng cho những story sẽ làm trong giai đoạn sau.

● Khi yêu cầu mọi người tham gia ước tính một story nào đó, chúng tôi thường nhận thấy những sự đánh giá khác của các thành viên về cùng một story. Điều này tốt cho việc tìm hiểu sâu hơn và thảo luận sớm về nó.

Nếu bạn yêu cầu nhóm cung cấp một ước tính, thường thì thành viên hiểu rõ nhất về story đó sẽ là người đầu tiên nêu ý kiến. Thật không may, điều này sẽ ảnh hưởng rất nhiều đến ước tính của những người khác.

12 Planning Poker, còn gọi là Scrum Poker, là một sáng tạo của Mike Cohn – một lý thuyết gia và là nhà thực hành Agile nổi tiếng. Đó là một bộ bài với các lá bài được đánh số theo dãy Fibonacci (0,1, 2,3,5,8 ..) được dùng trong ước tính tương đối và lập kế hoạch. Xem thêm trên http://hanoiscrum.net về cách dùng, cũng như tải về một bộ bài tiêu chuẩn để in ra và dùng cho nhóm của bạn.

45 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Có một kỹ thuật tuyệt vời để tránh gặp phải vấn đề này - nó được gọi là Planning Poker (theo tôi biết thì được đề xuất bởi Mike Cohn)



Mỗi thành viên trong nhóm sẽ nhận được một bộ bài 13 lá như hình trên. Mỗi khi phải ước tính một story nào đó, mỗi thành viên sẽ chọn một lá bài có ghi giá trị mà mình ước tính (tính theo điểm) và đặt úp xuống trên bàn. Khi tất cả các thành viên trong nhóm đã chọn xong lá bài của mình, các lá bài sẽ được lật lên cùng một lúc. Theo cách này, mỗi thành viên buộc phải đánh giá theo suy nghĩ của chính mình thay vì bị ảnh hưởng bởi ước tính của người khác.

Nếu có sự khác biệt lớn giữa hai giá trị ước tính, nhóm sẽ thảo luận về những khác biệt đó và cố gắng tạo ra một cái nhìn chung về những việc phải làm với story đang xem xét. Họ có thể phải chia ra các công việc nhỏ. Sau đó, nhóm thực hiện ước tính lại. Vòng lặp như vậy lặp đi lặp lại cho đến khi các ước tính tiến gần tới giá trị chung, tức là khi mà tất cả đánh giá cho story đó *xấ xỉ* như nhau.

Một điều quan trọng cần nhắc các thành viên là họ đang ước tính cho tổng khối lượng công việc cần làm cho story đó, chứ không phải là đánh giá chỉ "phần việc của họ". Kiểm thử viên không nên chỉ ước tính cho riêng phần kiểm thử.

Lưu ý rằng các con số có thể không liên tục. Ví dụ như không có giá trị nào nằm giữa 40 và 100. Tại sao vậy?

Điều này tránh cho những cảm giác sai lầm về tính chính xác khi đưa ra những ước tính lớn về thời gian. Nếu một story được ước tính vào khoảng 20 điểm, không quá cần thiết để thảo luận rằng nó lên là 20, 18 hay 21. Tất cả chúng ta đều hiểu rằng đó là một story phức tạp, mất nhiều công sức và khó để đưa ra ước tính chính xác hơn. Do đó, 20 là con số có thể “chơi” được.

Chúng ta có cần ước tính chi tiết hơn không? Chia nhỏ các story thành những story nhỏ hơn và ước tính cho những story nhỏ đó!

46 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Không, bạn không thể làm theo kiểu cộng hai con số 5 và 2 để thành 7. Bạn phải chọn hoặc là 5, hoặc là 8, không có số 7 ở đây.

Một số lá bài đặc biệt cần lưu ý:

● 0 = "story này đã hoàn thành" hay "story này quá đơn giản, chỉ mất vài phút để hoàn thành". ● ? = "Tôi thực sự không hiểu gì về nó. Không có gì cả."

● Biểu tượng cốc cà phê = "Tôi nghĩ là mình quá mệt rồi. Nghỉ ngơi một chút nhé." **Ph n loại các story**

Điều tồi tệ nhất xảy ra khi nhóm đang thuyết trình đầy tự hào về một tính năng mới tại buổi demo Sprint, và Product Owner cau mày nói "vâng, tốt đấy, nhưng đó *không h i là thứ tôi êu c u*!"

Làm sao để đảm bảo những gì Product Owner hiểu cũng là những gì nhóm hiểu về mỗi story? Hoặc mỗi thành viên trong nhóm có cùng sự hiểu biết giống nhau về mỗi story? Vâng, bạn không thể. Tuy nhiên, có một số kỹ thuật đơn giản để xác định những hiểu lầm thô thiển nhất. Kỹ thuật đơn giản nhất là đảm bảo rằng tất cả các trường thông tin cần thiết của mỗi story được viết ra đầy đủ (cụ thể hơn, là với mỗi story có mức độ ưu tiên cao được quan tâm trong Sprint này).

**Ví dụ :**

Nhóm phát triển và Product Owner rất hài lòng với kế hoạch Sprint và chuẩn bị kết thúc cuộc họp. Scrum Master nói: “Đợi một chút, story ‘Thêm mới người dùng’ vẫn chưa có ước tính. Hãy ước tính nó đã!” Sau một vài vòng chơi Poker, nhóm đạt được đồng thuận 20 điểm cho story đó, nhưng Product Owner đứng dậy ngay “Cái gì cơ?!”. Sau vài phút thảo luận nóng, vấn đề được phát hiện ra, nhóm đã hiểu sai phạm vi của story ‘Thêm mới người dùng’, họ nghĩ rằng nó sẽ là ‘một giao diện web đẹp với các chức năng thêm mới, di chuyển, xóa và tìm kiếm người dùng’, trong khi Product Owner lại hiểu ‘thêm mới người dùng đơn giản thực hiện bằng tay thông qua câu lệnh SQL trong cơ sở dữ liệu’. Họ đã phải ước tính lại với 5 điểm cho story này.

**Ví dụ :**

Nhóm Phát triển và Product Owner rất hài lòng với kế hoạch Sprint và chuẩn bị kết thúc cuộc họp. Scrum Master nói: “Đợi một chút, story ‘Thêm mới người dùng’, sẽ được demo như thế nào?”. Một số tiếng xì xào được phát ra và sau một phút, một số thành viên đứng dậy phát biểu “Vâng, đầu tiên chúng ta cần đăng nhập vào website, và sau đó...”, và Product Owner liền ngắt lời “đăng nhập vào website?! Không, không, không, chức năng này không cần thiết chút nào, chỉ cần người quản trị thực hiện vài lệnh SQL đơn giản là được”.

47 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Mô tả "cách thức demo" cho story có thể (và nên) *th t ng n*. Nếu không, bạn sẽ không thể kết thúc cuộc họp lập kế hoạch Sprint đúng giờ. Đó là mô tả mức đơn giản trong tiếng Anh về việc làm thế nào để thực hiện các kịch bản kiểm thử đặc trưng nhất. "Làm điều này, rồi làm việc kia, sau đó xác minh điều này".

Tôi nhận thấy rằng mô tả đơn giản này thường phát sinh những hiểu lầm lớn về phạm vi của một story. Phát hiện vấn đề sớm cũng tốt, đúng không?

**Phân tách các story thành những story nhỏ hơn**

Trong ước tính, các story không nên quá nhỏ hoặc quá lớn. Nếu bạn có một loạt các story với 0,5 điểm, bạn có thể đang là nạn nhân của quản lý vi mô. Mặt khác, một story có 40 điểm có nguy cơ cao trở thành “*hoàn thành một h n*”, cái đó không đem lại giá trị cho công ty và chỉ gia tăng chi phí quản lý. Hơn nữa, nếu *tốc độ ước tính* của là 70 và có hai tính năng ưu tiên cao nhất đều là 40 điểm thì việc lập kế hoạch sẽ rất khó khăn. Bạn phải lựa chọn hoặc dưới mức chỉ tiêu (chỉ chọn một story) hoặc vượt mức chỉ tiêu (chọn cả hai story).

Tôi nhận thấy rằng luôn có thể phân tách một story lớn thành những story nhỏ hơn. Chỉ cần bảo đảm những story nhỏ đó vẫn có giá trị nghiệp vụ khi chuyển giao.

Thường thì chúng tôi cố gắng đưa những story về khoảng 2 - 8 ngày công. Một nhóm điển hình thường có tốc độ vào khoảng 40-60 điểm, điều này cho phép chúng tôi xử lý khoảng 10 story với mỗi Sprint. Đôi khi ít hơn với 5 story và đôi khi nhiều hơn với 15 story. Đó cũng chính là số lượng *thẻ chỉ mục* mà chúng tôi có thể quản lý được.

**Phân tách story thành các tác vụ**

Chờ một chút, điều khác biệt giữa “task” (hạng mục công việc, tác vụ, công việc) và “story” là gì? Câu hỏi này rất quan trọng.

Khá dễ để phân biệt hai khái niệm này. Story là những thành phần của sản phẩm chuyển giao được mà Product Owner quan tâm. Tác vụ không phải là thành phần chuyển giao, hoặc là những thành phần mà Product Owner không quan tâm.

Ví dụ về việc tách một story thành các story nhỏ hơn:

48 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Ví dụ về việc tách một story thành các tác vụ:



Dưới đây là một số ghi nhận thú vị:

● Các Nhóm Scrum mới thường không muốn dành thời gian để phân tách story thành các tác vụ như ví dụ minh họa ở trên. Một số còn cảm thấy hướng tiếp cận này giống như phương pháp thác nước (waterfall).

● Khi Nhóm hiểu rõ về các story sẽ giúp việc phân tách chúng thành các tác vụ dễ dàng hơn.

● Việc phân tách thường làm gia tăng các công việc do đó dẫn đến ước tính thời gian có thể tăng lên, và cũng giúp việc lập kế hoạch Sprint sát với thực tế hơn.

49 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● Phân tách công việc còn giúp cho các buổi Họp Scrum Hằng ngày hiệu quả hơn (Xem nội dung “Chúng tôi tiến hành Họp Scrum Hằng ngày như thế nào” ở trang 48).

● Thậm chí, nếu việc phân tách công việc không chính xác và có thể làm thay đổi công việc khi triển khai, thì những lợi ích ở trên vẫn được chấp nhận.

Vì vậy, chúng tôi cố gắng để lập khung thời gian phù hợp cho Sprint, nhưng nếu vượt quá thời gian cho phép chúng tôi sẽ bỏ chúng ra khỏi Sprint (Xem phần “Chúng ta cần dừng ở đâu” bên dưới).

*Lưu ý* – chúng tôi triển khai TDD (Phát triển Hướng Kiểm thử) nghĩa là công việc đầu tiên với hầu hết các story là “viết một kiểm thử cho kết quả sai” và công việc cuối cùng là tái cấu trúc (refactor = tăng tính dễ đọc của mã nguồn và loại bỏ mã nguồn lặp).

**Xác định thời gian và địa điểm cho buổi Họp Scrum Hằng ngày**

Một công việc thường bị bỏ quên trong các buổi Họp Kế hoạch Sprint đó là “xác định thời gian và địa điểm Họp Scrum Hằng ngày”. Nếu không thực hiện việc này Sprint của bạn sẽ dễ có một khởi đầu tồi tệ. Buổi Họp Scrum Hằng ngày đầu tiên là buổi khởi động cần thiết, ở đó mọi thành viên sẽ quyết định nơi khởi đầu công việc của mình.

Tôi thường thích họp vào buổi sáng hơn. Nhưng, tôi phải thừa nhận rằng chúng tôi không thực sự cố gắng để thử tiến hành các cuộc Họp Scrum Hằng ngày vào buổi trưa hoặc buổi chiều.

**Những nhược điểm của việc Họp Scrum Hằng ngày vào buổi chiều:** khi bạn tiến hành công việc vào buổi sáng, bạn phải cố gắng nhớ những gì bạn nói với mọi người vào chiều ngày hôm qua về những công việc mà bạn sẽ làm hôm nay.

**Những nhược điểm của việc Họp Scrum Hằng ngày vào buổi sáng:** khi bạn bắt đầu công việc vào buổi sáng, bạn phải cố gắng nhớ những gì bạn đã làm hôm qua để bạn còn báo cáo nó.

Theo tôi thì nhược điểm đầu là tồi hơn, vì hầu hết những điều quan trọng lại là những thứ mà bạn *sẽ h i làm* chứ không phải những thứ mà bạn *đã làm*.

Phương thức mặc định của chúng tôi là chọn thời gian sớm nhất được chấp nhận bởi tất cả các thành viên trong Nhóm. Thường là 9:00, 9:30, hoặc 10:00. Điều quan trọng nhất đó là thời điểm mà mọi thành viên trong nhóm đều thực sự chấp nhận.

**Chúng ta cần dừng ở đ u**

OK, vậy là thời gian đang trôi đi. Tất cả những thứ mà chúng ta muốn làm trong suốt thời gian họp kế hoạch Sprint, chúng ta cần cắt bỏ những gì khi mà thời gian họp đã hết?

50 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

À, tôi sử dụng danh sách ưu tiên dưới đây:

**Ưu tiên :** Mục tiêu của Sprint và ngày demo. Đây là điều tối thiểu bạn cần có để bắt đầu một Sprint. Nhóm phải có mục tiêu, ngày kết thúc, và họ cần làm việc đúng với Product Backlog. Vậy thì thật là tệ, vâng, và bạn nên xem xét để tổ chức một cuộc họp lập kế hoạch mới vào ngày mai, nhưng nếu bạn thực sự cần phải bắt đầu Sprint này thì bạn có thể vẫn tiến hành được. Mặc dù vậy, thành thật mà nói, tôi chưa bao giờ thực sự phải bắt đầu Sprint với thông tin ít ỏi như thế.

**Ưu tiên :** Liệt kê ra các story mà Nhóm phải thực hiện trong Sprint này.

**Ưu tiên 3:** Ước tính cho tất cả các story trong Sprint.

**Ưu tiên 4:** “Cách thức demo” được điền cho mỗi story trong Sprint.

**Ưu tiên 5:** Tính toán tốc độ và tài nguyên cho Sprint của bạn thực sự đã được kiểm tra. Bổ sung danh sách các thành viên của Nhóm và những cam kết của họ (nếu không bạn không thể tính toán được tốc độ).

**Ưu tiên 6:** Xác định thời gian và địa điểm Họp Scrum Hằng ngày. Điều này chỉ mấy một chút thời gian để quyết định, nhưng nếu các bạn đã hết thời gian cho việc đó thì Scrum Master đơn giản có thể quyết định điều này sau buổi họp và gửi email cho mọi người.

**Ưu tiên 7:** Tách các story thành các tác vụ. Việc phân tách này có thể được thực hiện hằng ngày kết hợp với các buổi Họp Scrum Hằng ngày, nhưng nó sẽ có thể làm gián đoạn tiến độ của Sprint.

**Các story k thuật (tech stories)**

Đây là một vấn đề phức tạp: các story k thuật. Hoặc các hạng mục phi chức năng hay bất kỳ thứ gì bạn muốn gọi tên.

Tôi đang đề cập đến những thứ cần phải được thực hiện nhưng không có khả năng đem lại, không có liên quan trực tiếp tới bất kỳ một story cụ thể nào, và không có giá trị trực tiếp với Product Owner.

Chúng tôi gọi chúng là “story k thuật” (tech story).

Ví dụ:

● **Cài đặt máy chủ build liên tục**

Tại sao lại cần phải làm việc này: bởi vì nó tiết kiệm khối lượng lớn thời gian cho các nhà phát triển phần mềm và giảm thiểu phát sinh những vấn đề lớn khi tích hợp ứng dụng ở cuối phân đoạn.

51 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● **Viết mô tả tổng quan thiết kế hệ thống**

Tại sao lại cần phải làm việc này: Bởi vì các nhà phát triển có thể quên thiết kế tổng thể và dẫn đến viết những mã nguồn không phù hợp. Cần có một tài liệu mô tả “bức tranh lớn” về hệ thống để giúp mọi người tuân thủ cùng một thiết kế.

● **Cải tiến tầng DAO** (Data Access Object)

Tại sao lại cần phải làm việc này: Bởi vì tầng DAO thực sự lộn xộn làm mất nhiều thời gian của mọi người do nhầm lẫn và gây ra những lỗi không cần thiết. Làm sạch mã nguồn sẽ tiết kiệm thời gian cho nhà phát triển và tăng cường sức mạnh cho hệ thống.

● **Nâng cấp Jira** (phần mềm kiểm soát lỗi)

Tại sao lại cần phải làm việc này: Phiên bản hiện hành có nhiều lỗi và chạy chậm, nâng cấp sẽ giúp các bạn tiết kiệm thời gian.

Có những story mang ý nghĩa thông thường không? Hoặc chúng là những tác vụ không có kết nối với bất cứ một story cụ thể nào? Ai là người đặt độ ưu tiên cho chúng? Product Owner có nên tham gia vào công việc này?

Chúng tôi đã tiến hành rất nhiều các cách thức khác nhau để quản lý những story kĩ thuật. Chúng tôi đã thử xử lý chúng như những story thuộc phân lớp thứ nhất, giống như bất kỳ những story khác. Điều này là không tốt, khi Product Owner thiết lập ưu tiên trong Product Backlog, nó giống như việc so sánh các quả táo với các quả cam. Trong thực tế, vì mục đích minh bạch, các story “công nghệ” thường nhận được độ ưu tiên thấp với mô-típ kiểu như “vâng thưa các vị, tôi chắc chắn một dịch vụ build liên tục là rất quan trọng, nhưng hãy cứ xây dựng cho chúng tôi các tính năng về quản lý doanh thu trước đã? Sau đó các vị có thể bổ sung những viên kẹo công nghệ của mình cũng vẫn được mà?”

Trong một số trường hợp, Product Owner đúng, nhưng thường thì không. Chúng tôi đã kết luận rằng Product Owner không phải lúc nào cũng đủ điều kiện để tạo ra sự cân bằng. Vì vậy sau đây là những gì chúng tôi làm:

● Cố gắng tránh những story “công nghệ”. Cố tìm cách để đưa một story “công nghệ” thành một story thông thường với giá trị nghiệp vụ vừa phải. Với cách đó Product Owner có cơ hội tốt hơn để tạo những sự cân bằng một cách chính xác.

● Nếu chúng tôi không thể chuyển một story “công nghệ” thành một story thông thường, thì xem xét đến việc thực hiện chúng như một hạng mục công việc trong một story khác. Ví dụ

52 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

“cải tiến tầng DAO” có thể được xem là một task nằm trong story “chỉnh sửa user”, vì nó có liên quan đến tầng DAO.

● Nếu cả hai cách trên đều không thực hiện được, hãy định nghĩa chúng là một story “công nghệ”, và đặt riêng chúng trong một danh sách. Hãy để Product Owner trông thấy chúng nhưng không được phép thay đổi. Sử dụng các *“hệ số t trung”* và *“tốc độ ước tính”* để đàm phán với Product Owner và khéo léo đưa thêm Sprint thời gian để triển khai các story

“công nghệ” đó.

Xem xét ví dụ sau (một cuộc đối thoại tương tự điều này đã xảy ra trong một cuộc Họp Kế hoạch Sprint của chúng tôi).

● **Nhóm:** “Chúng tôi có một số công nghệ bên trong cần phải được triển khai. Chúng tôi muốn dùng 10% thời gian để thực hiện chúng, nghĩa là giảm các thành phần trọng tâm từ 75% xuống 65%. Điều đó có được không?”

● **Product Owner:** “Này, không được đâu! Chúng ta không có thời gian cho việc đó!”

● **Nhóm:** “Thôi được, anh hãy xem lại Sprint trước (tất cả mọi người đều quay nhìn vào tốc độ được vẽ nguệch ngoạc trên bảng). Chúng ta đã ước tính tốc độ là 80, và tốc độ thực sự là 30, đúng không?”

● **Product Owner:** “Chính xác! Vì vậy chúng ta không có thời gian cho việc triển khai mấy thứ công nghệ đó! Điều chúng ta cần là các tính năng mới!”

● **Nhóm:** “Được rồi, nguyên nhân dẫn tới tốc độ của chúng ta tồi tệ như vậy là bởi vì chúng ta tốn quá nhiều thời gian để cố gắng tạo ra các bản phát hành thống nhất cho vieecj kiểm thử”.

● **Product Owner:** “Đúng rồi, sao nữa?”

● **Nhóm:** “Tốt, tốc độ của chúng ta sẽ tiếp tục tồi tệ như vậy nếu chúng ta không làm gì để cải thiện nó.”

● **Product Owner:** “Đồng ý, sao nữa?”

● **Nhóm:** “Vì vậy chúng tôi đề nghị rằng chúng ta sẽ dành khoảng 10% thời gian của Sprint này để cài đặt một máy chủ build liên tục và một số thứ khác trợ giúp công việc tích hợp. Điều này có thể sẽ làm tăng tốc độ lên ít nhất là 20% cho Sprint này và các Sprint sau đó!”

● **Product Owner:** “Ồ thật chứ? Tại sao chúng ta không làm điều này từ Sprint trước?!”

53 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● **Nhóm:** “À… bởi vì anh không muốn chúng tôi làm…”

● **Product Owner:** “Ồ, ừ, tốt thôi, đây là một ý tưởng tốt, triển khai luôn đi!”

Dĩ nhiên, một lựa chọn khác đó là để Product Owner đứng ngoài hoặc đưa cho anh ta một hệ số tập trung không có khả năng thương lượng. Nhưng chẳng có lý do gì để không cố gắng đạt được sự đồng thuận trước tiên.

Nếu Product Owner là một thành viên có thẩm quyền (và chúng tôi may mắn có được điều này) tôi khuyên hãy để anh ấy có cơ hội thể hiện khả năng của mình và hãy để anh ấy thiết lập các ưu tiên tổng thể. Minh bạch là một trong những giá trị cốt lõi của Scrum, đúng vậy không?

**Hệ thống Theo dõi lỗi và Product Backlog**

Đây là một vấn đề đòi hỏi sự khôn khéo. Excel là một công cụ tốt dành cho Product Backlog. Nhưng bạn vẫn cần một hệ thống theo dõi lỗi (bug tracking system), và Excel không có khả năng thực hiện được điều này. Chúng tôi sử dụng Jira.

Vậy chúng tôi làm thế nào để đưa vấn đề sử dụng Jira vào cuộc họp lập kế hoạch Sprint? Theo tôi chẳng cần làm gì cả, cứ lờ điều đó đi và tập trung vào các story.

Chúng tôi đã thử một vài chiến lược sau:

● Product Owner in ra những hạng mục có độ ưu tiên cao nhất trong Jira, mang chúng tới cuộc Họp Kế hoạch Sprint, và dán lên tường cùng với các story (do đó mặc thể hiện việc so sánh độ ưu tiên của các hạng mục đó với các story khác).

● Product Owner tạo ra các story tham chiếu tới các hạng mục trong Jira. Ví dụ “Chỉnh sửa những lỗi nghiêm trọng xảy ra khi gửi báo cáo tới bộ phận hành chính, Jira-124, Jira-126, và Jira-180”.

● Sửa lỗi được coi là việc nằm ngoài Sprint, tức là một Nhóm giữ hệ số tập trung ở mức đủ thấp (ví dụ 50%) để đảm bảo rằng họ có thời gian dành cho việc sửa các lỗi. Sau đó chỉ cần giả định rằng Nhóm sẽ mất một khoảng thời gian nhất định trong mỗi Sprint dành cho việc sửa các lỗi được ghi nhận trong Jira.

● Đưa Product Backlog vào Jira (thay cho Excel). Coi các lỗi như bất kỳ một story nào đó.

Thực sự chúng tôi không đánh giá chiến lược nào là tốt nhất; trong thực tế nó tùy thuộc vào các Nhóm khác nhau và mỗi Sprint khác nhau. Mặc dù vậy tôi thường có thiên hướng đi theo chiến lược thứ nhất. Chiến lược đó vừa hay vừa đơn giản.

54 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Họp Kế hoạch Sprint đã đến hồi kết!**

Chà, tôi không bao giờ nghĩ rằng chương này là về các cuộc họp lập kế hoạch cho Sprint lại dài như thế! Tôi đoán rằng nó phản ánh quan điểm của tôi về việc Họp Kế hoạch Sprint là sự kiện quan trọng nhất mà bạn cần làm khi triển khai Scrum. Sử dụng nhiều sự nỗ lực để đạt được những thứ đúng đắn, và sẽ dễ dàng hơn cho các công việc sau này.

Cuộc họp lập kế hoạch Sprint sẽ thành công nếu mọi người (tất cả các thành viên của Nhóm và Product Owner) vui vẻ khi rời khỏi cuộc họp, và mỉm cười khi tỉnh dậy vào mỗi buổi sáng, và hoan hỉ trong buổi Họp Scrum Hằng ngày.

Khi đó, tất nhiên tất cả mọi thứ vẫn có thể đi xuống một cách tồi tệ, nhưng ít nhất các bạn không thể đổ lỗi cho kế hoạch Sprint :o)

55 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**5**

**Chúng tôi truyền thông trong các Sprint như thế nào**

Giữ cho toàn bộ công ty biết về những gì đang được làm là rất quan trọng. Nếu không mọi người có thể sẽ phàn nàn, thậm chí tệ hơn là có những nhận định sai lầm về những gì đang diễn ra. Chúng tôi sử dụng một “trang thông tin về Sprint” cho việc này.

**Nhóm Jackass, Sprint 15**

**Mục tiêu Sprint**

o Bản Beta sẵn sàng phát hành!

**Sprint Backlog** (ước tính trong ngoặc tròn)

o Gửi tiền (3)

o Công cụ chuyển đổi (8)

o Đăng nhập dành cho hành chính(5)

o Quản trị người dùng (5)

Tốc độ ước tính: 21

**Lịch làm việc**

o Thời gian của Sprint: từ 06/11/2006 đến 24/11/2006

o Họp Scrum hàng ngày: 9:30 - 9:45, tại phòng làm việc của Nhóm

o Demo Sprint: 24/11/2006, 13:00, tại quán ăn tự phục vụ

**Nhóm**

o Jim

o Erica (Scrum Master)

o Tom (75%)

o Eva

o John

56 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Đôi khi chúng tôi cũng bổ sung thêm thông tin về cách thức tốt nhất dành cho việc demo một story.

Sau buổi họp kế hoạch Sprint, Scrum Master tạo trang này, sau đó đưa thông tin này lên wiki, và gửi lại cho toàn bộ công ty sớm nhất có thể.

**Tiêu đề: Sprint 15 của Nhóm Jackass đã bắt đầu**

Chào mọi người! Nhóm Jackass hiện tại đã bắt đầu làm Sprint 15. Mục tiêu của chúng tôi là demo một bản beta sẵn sàng để phát hành vào ngày 24/11.

Xem chi tiết thông tin của sprint tại:

http://wiki.mycompany.com/jackass/sprint15

Chúng tôi cũng có một trang “bảng điều khiển” trên wiki, nó sẽ liên kết với tất cả các Sprint đang được thực hiện.

**Bảng điều khiển của doanh nghiệp**

Các Sprint đang được thực hiện

o Nhóm X sprint 15

o Nhóm Y sprint 12

o Nhóm Z sprint 1

Thêm vào đó, Scrum Master sẽ in ra những trang thông tin của Sprint và dán chúng lên tường bên ngoài phòng làm việc của nhóm. Vì vậy bất kỳ ai đi qua đó cũng có thể nhìn thấy thông tin về Sprint đó và biết được nhóm nào đang làm việc gì. Nếu có thêm thông tin về thời gian và địa điểm Họp Scrum Hằng ngày và demo Sprint, anh ta sẽ biết nơi cần đến để biết thêm những thông tin khác.

Khi Sprint gần kết thúc, Scrum Master nhắc nhở mọi người về việc sắp tới ngày demo:

57 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Tiêu đề: Demo Sprint của nhóm Jackass vào ngày mai lúc 13:00 tại quán ăn.**

Chào mọi người! Các bạn được mời đến tham dự buổi demo Sprint lúc 13:00 tại quán ăn vào ngày mai (Thứ Sáu). Chúng tôi sẽ chạy thử bản beta sắp được phát hành.

Xem chi tiết thông tin của Sprint tại:

http://wiki.mycompany.com/jackass/sprint15

Làm như vậy, chắc chắn sẽ không có ai nói rằng mình không biết việc gì đang diễn ra.

58 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**7**

**Chúng tôi y dựng Sprint Backlog như thế nào** Bạn đã làm công việc này khá lâu rồi? Úi chà, tốt đấy.

Bây giờ ta đã tiến hành xong buổi họp kế hoạch Sprint và đã kết luận rõ ràng công việc trong Sprint mới, đây là thời điểm để Scrum Master tạo một Sprint Backlog. Việc này cần được hoàn thành sau buổi họp kế hoạch Sprint, nhưng phải xong trước buổi họp Scrum Hằng ngày đầu tiên.

**Khuôn mẫu cho Sprint Backlog**

Chúng tôi đã thử nghiệm với nhiều định dạng khác nhau cho Sprint Backlog, bao gồm cả Jira, Excel, và bảng công việc treo trên tường. Hầu hết khi mới sử dụng Excel, có rất nhiều mẫu (template) Excel sẵn có cho Sprint Backlog, bao gồm cả biểu đồ Burndown được tự động sinh ra và một số thứ giống như vậy. Tôi có thể nói rất nhiều về cách chúng tôi cải thiện Sprint Backlog trên Excel. Nhưng tôi sẽ không làm như vậy ở đây, thậm chí là chỉ mộtví dụ.

Thay vì việc đó, tôi sẽ mô tả chi tiết những gì chúng tôi đã tìm thấy để có được một định dạng hiệu quả nhất cho Sprint Backlog – đó là một bảng công việc trên tường!

Tìm kiếm một bức tường lớn đang không sử dụng hoặc chứa những thứ chẳng mấy hữu ích như biểu tượng của công ty, biểu đồ cũ hoặc những bức vẽ xấu. Làm sạch bức tường (hãy xin phép để làm điều này nếu bạn không có thẩm quyền). Dán lên đó một tờ giấy có khổ lớn (ít nhất là 2x2 mét, hoặc 3x2 mét cho một Nhóm với nhiều thành viên). Sau đó làm tiếp như sau:

59 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



Tất nhiên là bạn có thể sử dụng bảng trắng. Nhưng việc đó hơi lãng phí. Nếu có thể, hãy dành bảng trắng cho thiết kế thô và sử dụng những bức tường màu cho việc treo bảng công việc.

*Chú ý* - nếu bạn có nhiều công việc, đừng quên dán chúng bằng các mẩu băng dính, vì nếu không bạn có thể sẽ thấy chúng nằm la liệt trên sàn nhà vào một ngày nào đó.

60 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**S dụng bảng công việc taskboard) như thế nào**

****

Tất nhiên, bạn có thể thêm tất cả các kiểu cột khác nhau. Ví dụ cột *“ h tích hợ kiểm thử”* hoặc *“Đã được hủ bỏ”*. Tuy nhiên trước khi bạn gây thêm rắc rối, hãy suy nghĩ kỹ. Việc thêm những cột này có *th c s* cần thiết hay không?

Tôi đã phát hiện ra những thứ đơn giản và vô cùng giá trị cho những điều này, vì vậy tôi chỉ thêm vào những cột khi chi phí cho việc *không* làm như vậy là rất lớn.

**Ví dụ – sau buổi Họp Scrum Hàng ngày**

Sau buổi Họp Scrum Hằng ngày đầu tiên, bảng công việc có thể trông giống như sau:

61 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Như bạn có thể thấy, có ba công việc đã được “triển khai” (checked out), tức là Nhóm làm việc sẽ bắt đầu những mục công việc này trong ngày hôm nay.

Đôi khi, đối với những Nhóm lớn, một công việc có thể bị mắc kẹt ở tình trạng “triển khai” vì không ai trong nhóm nhớ rằng ai đang làm việc đó. Nếu điều này thường xảy ra trong một Nhóm thì cần đưa ra những chính sách như thêm vào tên của người thực hiện công việc vào mỗi công việc đã được chọn để triển khai.

**Ví dụ – Sau một vài ngày**

Một vài ngày sau bảng công việc có thể trông như thế này:

62 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



Như bạn có thể thấy chúng tôi đã hoàn thành story “gửi tiền” (tức là mã nguồn đã được đưa vào kho quản lý, đã được kiểm thử, đã được tái cấu trúc v.v.). Công cụ chuyển đổi đã được hoàn thiện một phần, chức năng đăng nhập cho bộ phần hành chính đã bắt đầu được triển khai, và chức năng quản trị người dùng vẫn chưa bắt đầu.

Chúng ta có 3 hạng mục chưa được lên kế hoạch, như bạn thấy ở phía dưới bên phải. Điều này cần phải nhớ khi bạn tiến hành Họp Cải tiến Sprint.

Dưới đây là một ví dụ thực về một Sprint Backlog khi sắp kết thúc Sprint. Nó trông lộn xộn hơn các quy trình Sprint, nhưng điều đó chấp nhận được vì thời gian tồn tại của nó không lâu. Khi một Sprint mới được triển khai chúng ta dọn dẹp và làm mới lại chúng để chuẩn bị cho Sprint Backlog mới.

63 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



**Cách làm việc với Biểu đồ Burndown**

Hãy phóng to Biểu đồ Burndown:



Biểu đồ này cho thấy:

64 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● Trong ngày đầu tiên của Sprint, ngày 1/8, Nhóm đã ước tính có khoảng 70 điểm (story point) cẩn phải thực hiện. Điều này có ảnh hưởng đến tốc độ ước tính của toàn bộ Sprint.

● Vào ngày 16/8 Nhóm ước tính có khoảng 15 điểm còn lại cần phải làm. Đường thẳng với nét đứt nằm ở trên cho thấy họ đang trong tầm kiểm soát, tức là với tốc độ như vậy họ sẽ hoàn thành tất cả các công việc khi kết thúc Sprint.

Chúng ta bỏ qua ngày nghỉ cuối tuần trên trục X vì công việc hiếm khi được làm vào cuối tuần. Chúng ta có thể thêm vào các ngày cuối tuần nhưng điều này sẽ dễ gây hiểu lầm khi xem biểu đồ, do đó phần ngày nghỉ cuối tuần sẽ được thêm vào các “gạch thẳng” trông giống như một ký hiệu cảnh báo (warning sign).

**Những dấu hiệu cảnh báo trên bảng công việc**

Nhìn lướt nhanh bảng công việc sẽ giúp bất kỳ ai có thể thấy được Sprint đang diễn ra như thế nào. Scrum Master có trách nhiệm đảm bảo rằng Nhóm hành động dựa trên những dấu hiệu cảnh báo giống như sau:



65 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



66 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



67 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO



**Này, thế còn về khả năng theo dõi thì sao?!**

Việc theo dõi (traceable13) tốt nhất tôi có thể gợi ý trong phần này là chụp một kiểu ảnh số của bảng công việc hàng ngày. Nếu đó là việc mà bạn phải làm. Tôi thỉnh thoảng làm như vậy nhưng không bao giờ cần phải đào bới lại những bức ảnh đó.

Nếu việc theo dõi là quan trọng với bạn, thì có thể giải pháp sử dụng bảng công việc không dành cho bạn.

13 Traceability là khả năng theo dõi và lưu vết sự tiến triển, thay đổi trạng thái của công việc hoặc bản thân bảng công việc. Một bảng công việc bằng Excel hoặc các phần mềm quản lí dự án thường có khả năng lưu vết tốt hơn là bảng công việc. Việc này thường có ích khi chúng ta thanh tra lại cách làm, quy trình, đặc biệt là khi có sai hỏng; các vết lưu được sẽ cung cấp bằng chứng để ra quyết định sau đó.

68 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Nhưng tôi thực sự đề nghị bạn hãy thử ước tính giá trị thực tế của việc lưu vết chi tiết mỗi Sprint. Một khi Sprint được hoàn thành và mã nguồn chạy tốt đã được gửi đi và tài liệu đã được triển khai, có ai đó thực sự quan tâm bao nhiêu story đã được hoàn thành trong ngày thứ 5 của Sprint hay không? Có ai thực sự quan tâm tới thời gian được ước tính cho việc “viết một kiểm thử sai cho story ‘Gửi tiền’” không?

**ớc t nh ngày và giờ**

Hầu hết trong các cuốn sách và bài viết về Scrum bạn sẽ tìm thấy công việc (task) được ước tính theo giờ, chứ không phải theo ngày. Chúng tôi đã từng làm như vậy. Công thức chung của chúng tôi là: 1 ngày làm việc (man-day) hiệu quả = 6 giờ làm việc (man-hours) hiệu quả.

Bây giờ chúng tôi không làm như vậy, ít nhất là với tất cả các Nhóm của chúng tôi, do một số nguyên nhân dưới đây:

● Ước tính giờ làm việc là quá nhỏ, việc này có xu hướng tạo nên rất nhiều công việc 1-2 giờ và do đó sẽ trở thành quản lý vi mô (micro-manage).

● Từ đó suy ra rằng tất cả mọi người đang nghĩ đến ngày làm việc theo bất kỳ cách nào, nó sẽ được nhân lên 6 trước khi viết thành giờ làm việc. “Hừm, công việc này cần một ngày để làm. Ô tôi phải viết ra số giờ, tôi sẽ viết 6 giờ và sau đó..”.

● Hai đơn vị tính khác nhau gây ra nhầm lẫn. “Nên ước tính bằng ngày làm việc hay giờ làm việc đây?”

Vì vậy bây giờ chúng tôi dùng ngày làm việc làm đơn vị thời gian cơ bản để ước tính (dù chúng tôi gọi nó là điểm (story points)). Giá trị thấp nhất của chúng tôi là 0.5, tức là bất kỳ công việc nào nhỏ hơn 0.5 đều được hủy bỏ hoặc kết hợp lại với một công việc khác, hoặc để ước tính là 0.5 (không có hại lớn cho gì cho việc ước tính đó). Hay và đơn giản.

69 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**7**

**Chúng tôi bố tr phòng làm việc như thế nào**

**Góc thiết kế**

Tôi đã nhận thấy những cuộc thảo luận về thiết kế thường thú vị và giá trị nhất khi diễn ra một cách tự nhiên phía trước bảng công việc (taskboard).

Vì lý do này, chúng tôi cố gắng sắp xếp không gian này như “góc thiết kế” hoàn hảo 

Điều này thực sự rất có ích. Không có cách nào tốt hơn để quan sát tổng quan hệ thống hơn là đứng tại góc thiết kế và có thể nhìn sang cả hai phía tường, sau đó nhìn vào máy tính và thử bản

70 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

build mới nhất của hệ thống (nếu bạn may mắn có các bản build liên tục, xem trang 95, *“Chúng tôi k t hợ Scrum với XP ra sao”*).

“Bức tường thiết kế” chỉ là một bảng trắng lớn chứa các mảnh giấy và các bản in của các tài liệu thiết kế quan trọng nhất (sơ đồ tuần tự, các bản mẫu thiết kế giao diện, mô hình miền, v.v.).

Hình trên: một cuộc Họp Scrum Hằng ngày đang diễn ra ở một góc như vậy.

Hừm... biểu đồ Burndown trông khá đẹp và đi xuống thẳng một cách đáng ngờ đúng không nhỉ. Nhưng Nhóm khẳng định rằng đó là sự thật :o)

**ặt chỗ cho Nhóm ngồi cùng nhau!**

Khi nói đến cách bố trí bàn và chỗ ngồi có một điều không cần tranh cãi.

Đặt chỗ cho Nhóm ngồi cùng nhau!

Để làm rõ một chút những gì tôi nói là

**Đặt chỗ cho Nhóm ngồi cùng nhau!**

71 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

Mọi người không thích phải di chuyển. Ít nhất ở nơi tôi làm việc là như vậy. Họ không muốn phải nhấc tất cả các đồ dùng của họ, g máy tính, di chuyển tất cả giấy tờ sang một chiếc bàn mới và sắp xếp mọi thứ lại từ đầu. Khoảng cách càng ngắn thì sự ngại ngần càng giảm. “Thôi nào ông chủ, sao lại phải di chuyển thêm 5 mét nữa làm gì?”

Tuy nhiên, khi xây dựng các nhóm Scrum hiệu quả, không có giải pháp nào thay thế cả. Chỉ có cách sắp xếp cho họ ngồi cùng nhau! Thậm chí bạn phải xâm phạm vào sự riêng tư của mỗi cá nhân, chuyển tất cả các thiết bị của họ ra và tẩy các vết bẩn cà phê cũ của họ. Nếu không có không gian cho Nhóm thì hãy tạo ra không gian. Ở đâu đó. Thậm chí nếu bạn phải đặt Nhóm ở tầng hầm. Di chuyển các bàn xung quanh, hối lộ quản lý tòa nhà, làm bất cứ điều gì có thể. Chỉ cần Nhóm được ngồi cùng với nhau!

Khi Nhóm của bạn làm việc cùng nhau thì thành quả sẽ có ngay lập tức. Chỉ sau một Sprint Nhóm sẽ nhất trí rằng ý tưởng để cả Nhóm cùng nhau là ý tưởng tốt (tôi rút ra từ kinh nghiệm cá nhân về điều này, không có gì để nói nếu Nhóm của bạn không quá bảo thủ để thừa nhận).

Bây giờ cần hiểu “cùng nhau” nghĩa là gì? Các bàn làm việc được sắp xếp ra sao? Vâng, tôi không có bất kỳ một ý kiến kiên quyết nào về việc bố trí bàn làm việc tối ưu. Và cho dù tôi có làm, tôi cho rằng hầu hết các Nhóm không vui vẻ về khả năng quyết định chính xác cách để bố trí bàn làm việc của họ. Thường thì phải tính tới các ràng buộc khác nữa – các nhóm lân cận, cửa nhà vệ sinh, máy móc lớn giữa phòng, v.v.

“Cùng nhau” nghĩa là :

● **Có thể nghe rõ:** Bất cứ ai trong Nhóm có thể nói với người khác mà không cần phải nói to hoặc rời khỏi bàn của anh ta.

● **Khả năng quan sát:** Bất cứ ai trong Nhóm có đều có thể nhìn thấy những người khác. Mọi người đều có thể nhìn thấy bảng công việc (taskboard). Không nhất thiết là phải đủ gần để có thể *đọc* được những thứ viết trên bảng mà chỉ c *n nh n th*ấy là được.

● **Riêng biệt:** Nếu toàn bộ Nhóm của bạn đột nhiên đứng lên và tham gia vào một cuộc thảo luận tự phát và trực tiếp, không ai ngoài nhóm của bạn ở gần đó bị quấy rầy. Và ngược lại.

“Riêng biệt” không có nghĩa là Nhóm phải hoàn toàn tách biệt. Trong căn phòng có thể phải đủ để Nhóm của bạn có được không gian riêng và tường bao xung quanh phòng đủ lớn để lọc các tạp âm từ các nhân tố ngoài Nhóm.

Và chuyện gì sẽ xảy ra nếu bạn có một Nhóm phân tán? Vâng, vậy không may cho bạn rồi. Sử dụng càng nhiều các kỹ thuật hỗ trợ có thể để giảm thiểu ảnh hưởng – hội thảo truyền hình, webcam, các công cụ chia sẻ màn hình, v.v.

72 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Bố tr cho Product Owner một khu vực**

Product Owner sẽ đủ gần để Nhóm có thể đi tới và hỏi anh ta vài điều, và anh ta có thể đi tới bảng công việc. Nhưng anh ta sẽ không được ngồi cùng Nhóm. Tại sao? Bởi vì có thể anh ta sẽ không kiềm chế được bản thân và can thiệp vào chi tiết, và Nhóm sẽ không “kết dính” một cách đúng đắn (tức là đạt được sự gắn kết, tự quản, trạng thái siêu năng suất).

Thành thật mà nói, đây là việc mà tôi suy diễn. Tôi chưa thực sự nhìn thấy trường hợp nào mà Product Owner ngồi cùng với Nhóm, do đó tôi không có kinh nghiệm thực tế để nói rằng đó là ý kiến tồi. Đó chỉ là trực cảm và được nghe nói từ các Scrum Master khác.

**Bố tr cho những người quản lý và người huấn luyện ở một khu vực** Với tôi điều này hơi khó viết do tôi vừa là quản lý vừa là người huấn luyện …

Công việc của tôi làm việc càng gần với Nhóm càng tốt. Tôi thiết lập các Nhóm, di chuyển giữa các Nhóm, lập trình cặp với mọi người, huấn luyện các Scrum Master, tổ chức các buổi Họp Kế hoạch Sprint, v.v. Nhìn lại, mọi người nghĩ đây là **điều tốt**, do tôi có kinh nghiệm với phương pháp phát triển phần mềm linh hoạt.

Nhưng, sau này, tôi cũng là giám đốc phát triển, vai trò là giám đốc quản lý chức năng (gia nhập Darth Vader music). Điều này có nghĩa là với việc tham gia một Nhóm sẽ tự động làm cho Nhóm trở nên ít tự quản lý hơn. “Quỷ tha ma bắt, ông chủ ở đây, ông ý có thể có rất nhiều ý kiến về những điều chúng ta sẽ thực hiện và ai sẽ làm gì. Tôi sẽ để ông ấy nói về vấn đề này.”

Quan điểm của tôi ở đây là, nếu bạn là một nhà huấn luyện Scrum (và có lẽ cũng là người quản lý), hãy tham gia càng chặt chẽ càng tốt. Nhưng chỉ trong thời gian có hạn thôi, sau đó ra ngoài và để nhóm gắn kết và tự quản lý. Kiểm tra Nhóm trong chốc lát (không quá thường xuyên) bằng cách tham gia các buổi sơ kết Sprint và nhìn vào bảng công việc, lắng nghe vào các buổi họp hằng ngày. Nếu bạn thấy có lĩnh vực nào cần cải tiến, gặp Scrum Master và hướng dẫn cho anh ta. *Không* phải là trước mặt Nhóm. Một ý hay khác là tham gia các buổi Họp Cải tiến Sprint (xem trang 81, *“ h ng tôi i ti n S rint như th nào”*), nếu Nhóm của bạn tin rằng sự có mặt của bạn không đủ để làm họ im lặng.

Đối với các Nhóm Scrum thực hiện tốt chức năng, đảm bảo rằng họ có mọi thứ họ cần, sau đó hãy để họ nằm ngoài tầm kiểm soát (ngoại trừ trong các buổi Sơ kết Sprint).

73 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**8**

**Chúng tôi họp Scrum Hằng ngày như thế nào**

Các buổi Họp Scrum Hằng ngày được chúng tôi nói đến khá nhiều trong cuốn sách. Các buổi họp bắt đầu đúng giờ, hàng ngày tại cùng một địa điểm. Thời kỳ bắt đầu chúng tôi vào một căn phòng riêng biệt để thực hiện lập kế hoạch Sprint (những ngày đó chúng tôi sử dụng Sprint Backlog điện tử), tuy nhiên hiện tại chúng tôi thực hiện họp Scrum Hằng ngày phía trước bảng công việc trong phòng làm việc của Nhóm. Không có gì phải bàn cãi về điều này cả.

Chúng tôi thông thường đứng họp, nhờ thế sẽ giảm được nguy cơ cuộc họp kéo dài quá 15 phút. **Chúng tôi cập nhật bảng công việc như thế nào**

Chúng tôi thường cập nhật bảng công việc trong buổi họp Scrum Hằng ngày. Khi mỗi thành viên mô tả về những gì anh ta đã làm hôm trước và sẽ làm hôm nay, anh ta sẽ di chuyển miếng giấy ghi công việc tới các vị trí tương ứng trên bảng công việc. Nếu anh ta đề cập tới một công việc không nằm trong kế hoạch, anh ta đưa thêm một mẩu giấy mô tả công việc đó. Anh ta cập nhật ước tính thời gian của anh ta để hoàn thành, anh ta viết ước tính thời gian mới lên trên miếng giấy ghi công việc đó và gạch bỏ ước tính cũ. Đôi khi Scrum Master thực hiện động tác thay đổi vị trí các miếng giấy ghi công việc trong khi mọi người nói.



Một số Nhóm có quy ước mỗi người sẽ cập nhật bảng công việc *trước* mỗi buổi họp. Công việc đó cũng tốt. Chỉ cần quyết định quy ước đó và thực hiện theo nó là được.

Cho dù Sprint Backlog của bạn định dạng kiểu gì thì hãy cố gắng để toàn Nhóm cùng tham gia vào việc cập nhật nó hằng ngày. Chúng tôi đã thực hiện các Sprint ở đó Scrum Master là người nắm

74 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

giữ chính Sprint Backlog, phải để mắt đến nó hằng ngày, và đề nghị mọi người cập nhật các ước tính thời gian còn lại của họ lên đó. Nhược điểm của nó là:

● Scrum Master mất khá nhiều thời gian vào việc quản lý Sprint Backlog, thay vì việc hỗ trợ Nhóm và loại bỏ những trở ngại trong công việc.

● Các thành viên Nhóm không chú ý tới trạng thái của Sprint, vì vậy dẫn tới Sprin Backlog không phải là thứ họ quan tâm. Việc thiếu vắng sự phản hồi này làm giảm tính linh hoạt và sự tập trung của Nhóm.

Nếu Sprint Backlog được thiết kế tốt sẽ giúp mỗi thành viên cập nhật công việc của mình một cách dễ dàng.

Ngay sau buổi họp Scrum Hằng ngày, một người nào đó sẽ tổng kết lại tất cả các ước tính thời gian (bỏ qua các công việc ở cột “hoàn thành”) và cập nhật một điểm mới trên biểu đồ Sprint Burndown.

**Ứng với những người đến muộn**

Một vài Nhóm có một thùng đựng tiền xu và phiếu. Khi bạn đến muộn, thậm chí chỉ muộn một phút, bạn phải đặt vào thùng một lượng theo quy định. Không có gì để thắc mắc cả. Nếu bạn có gọi trước cuộc họp và nói bạn sẽ muộn thì bạn vẫn sẽ phải nộp như thế.

Bạn chỉ không bị phạt nếu bạn có lý do chính đáng chẳng hạn bạn có lịch hẹn khám bác sĩ hoặc bạn sẽ tổ chức đám cưới của mình hay đại loại cái gì đó tương tự.

Tiền trong thùng được sử dụng cho các sự kiện chung. Chẳng hạn mua bánh hamburger khi chúng tôi chơi game vào các buổi tối :o)

Việc này cũng thú vị lắm. Nhưng nó chỉ cần thiết đối với những Nhóm mà thành viên thường hay đến muộn. Một số Nhóm không cần áp dụng hình thức như vậy.

**Ứng với những c u trả lời Tôi không biết làm gì ngày hôm nay**

Không phải là không có người nói “Hôm qua tôi đã làm bla bla bla, nhưng hôm nay tôi chưa biết phải làm gì” (này, đó là một mẩu thông tin đưa vào để cho nó có vấn đề). Bây giờ làm gì?

Tôi sẽ kể về Joe và Lisa là những người không biết hôm nay làm gì.

Nếu tôi là Scrum Master, tôi chỉ tiến đến và để cậu bên cạnh nói, nhưng tạo sự chú ý cho những người không có gì để làm. Sau khi mọi người đã nói các xong công việc của họ, tôi duyệt bảng công việc với toàn Nhóm, từ trên xuống dưới, và kiểm tra xem mọi thứ đã đồng bộ chưa để mọi

75 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

người biết mỗi công việc có ý nghĩa gì, v.v.. Tôi mời mọi người bổ sung thêm các miếng giấy ghi các công việc mới. Sau đó, tôi hướng trở lại những người không biết phải là gì nói “bây giờ chúng ta đã duyệt qua bảng công việc, bạn có ý tưởng gì về những thứ bạn có thể làm hôm nay chưa?” Hy vọng họ sẽ thấy.

Nếu không, tôi xem có cơ hội nào để tiến hàng lập trình cặp ở thời điểm này không. Sau đó với Niklas tiến hành làm phần giao diện (GUI ) cho chức năng quản trị người dùng vào hôm nay. Trong trường hợp này tôi gợi ý một cách lịch sự rằng Joe hoặc Lisa có thể lập trình cặp với Niklas công việc đó. Đấy là việc tôi thường làm.

Và nếu các thức đó không giải quyết được, thì đây là chút mánh khóe:

**Scrum Master:** “OK, ai muốn demo bản phát hành betacho chúng ta ?” (giả sử đó là mục tiêu của Sprint)

**Nhóm:** Im lặng trong bối rối

**Scrum Master:** “Chúng ta không thực hiện việc này?”

**Nhóm:** “ừm .. không”

**Scrum Master:** “Khỉ thật” Tạo sao không ? Thế còn gì để làm đây?”

**Nhóm:** “Vâng chúng tôi thậm chí chưa có server kiểm thử để chạy và kịch bản build đang bị lỗi.”

**Scrum Master:** “Aha.” (bổ sung thêm hai đầu việc lên bảng công việc). “Joe và Lisa, các bạn có cách nào để trợ giúp chúng ta ngày hôm nay không?”

**Joe:** “Ừm… Tôi sẽ thử tìm một vài server kiểm thử ở đâu đó”.

**Lisa:** “… và tôi sẽ cố gắng sửa lỗi kịch bản build”.

Nếu bạn may mắn, một ai đó sẽ trình bày bản phát hành beta mà bạn yêu cầu. Tuyệt vời! Bạn đã đạt được mục tiêu Sprint của bạn. Nhưng bạn sẽ phải làm gì nếu bạn đang ở giữa Sprint? Dễ thôi. Chúc mừng Nhóm vì công việc đang hoàn thành tốt, lấy một hoặc hai tính năng từ khu vực “Những phần kế tiếp” nằm ở bên phải phía dưới bảng công việc của bạn, đặt chúng vào cột “chưa kiểm tra” nằm ở phía bên trái. Sau đó thực hiện lại buổi Họp Scrum Hằng ngày. Thông báo cho Product Owner là bạn đã bổ sung một vài công việc vào Sprint.

Nhưng sẽ ra sao nếu Nhóm chưa đạt được mục tiêu Sprint, Joe và Lisa vẫn từ chối để chọn ra một việc nào đó có ích để làm. Tôi thường xem xét sử dụng một trong những chiến lược dưới đây (đây không phải dùng là chiến lược hay, nhưng đây là kế sách cuối cùng):

76 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

● **Làm cho hổ thẹn:** “Vâng, nếu các bạn không có ý tưởng gì để hỗ trợ Nhóm, tôi khuyên bạn về nhà, hay đọc sách hoặc làm gì đó. Hoặc ngồi quanh đây cho đến khi có người cần sự trợ giúp của bạn.”

● **Sử dụng cách truyền thống:** Đơn giản giao cho họ một nhiệm vụ.

● **Áp lực ngang hàng:** Nói rằng “Joe và Lisa, các bạn hãy thoải mái sử dụng thời gian rảnh của mình, tất cả chúng tôi sẽ chỉ đứng ở đây cho đến khi các bạn chọn được việc gì đó để hỗ trợ chúng ta hoàn thành công việc”.

● **Thân phận nô lệ:** Nói rằng “Tốt thôi, hôm nay các bạn có thể giúp Nhóm gián tiếp bằng việc trở thành quản gia. Lấy cà phê, mát-xa cho mọi người, đi đổ rác, nấu bữa trưa cho chúng tôi, và bất cứ cái gì mà chúng tôi đề nghị trong cả ngày hôm nay nhé”. Bạn có thể ngạc nhiên bởi Joe và Lisa sẽ nhanh chóng chọn một việc gì đó về kỹ thuật để làm :o)

Nếu một ai đó thường bắt bạn phải đi quá xa để giải quyết, thì có lẽ bạn nên gạt người đó sang bên cạnh và tiến hành một vài huấn luyện nghiêm khắc. Nếu vấn đề không giải quyết được, bạn cần xem xét xem người này có quan trọng với Nhóm của bạn hay không.

Nếu anh ta *không uá uan trọng*, hãy thử loại anh ta ra khỏi Nhóm của bạn.

Nếu anh ta *uan trọng*, cố tạo cặp cho anh ta với một người nào đó có thể đóng vai trò như “người chăm sóc” anh ta. Joe có thể là một lập trình viên và nhà kiến trúc lão luyện, chỉ có điều anh ta thực sự không thích người khác nói anh ta phải làm gì. Tốt thôi. Giao cho Niklas nhiệm vụ là người chăm sóc thường trực của Joe. Hoặc chính bạn làm nhiệm vụ đó. Nếu Joe quan trọng đối với Nhóm của bạn thì điều này rất đáng để làm. Chúng tôi đã gặp các trường hợp như vậy và những gợi ý ở trên ít nhiều có tác dụng.

77 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**9**

**Chúng tôi Sơ kết Sprint ra sao**

Sprint Demo (hoặc Sơ kết Sprint như mọi người thường gọi) là một sự kiện quan trọng của Scrum nhưng thường có xu hướng bị mọi người đánh giá thấp.

“Ồ, thực sự là chúng tôi phải tiến hành demo hay sao? Thực sự có chẳng có nhiều điều vui vẻ để mà trình diễn cả!”

“Chúng tôi không có thời gian để chuẩn bị cho một buổi demo gì gì... đó!” “Tôi không có thời gian tham dự các buổi demo của nhóm khác!”

**Lý do chúng tôi nhấn mạnh đến việc các Sprint cần kết thúc với một buổi demo**

Một buổi Sơ kết Sprint thành công, mặc dù có thể không gây ấn tượng mạnh, nhưng lại có ảnh hưởng sâu sắc.

● Nhóm có được niềm tin vào thành quả của họ. Họ *c m thấ những điều tốt đẹ* . ● Những người khác biết được những gì mà nhóm đang thực hiện.

● Buổi demo còn thu nhận được các phản hồi quan trọng từ những người liên quan.

● Các buổi demo là (hoặc nên là) một sự kiện mang tính cộng đồng để các nhóm khác nhau có thể trao đổi và thảo luận về công việc của họ. Đây chính là giá trị mang lại.

● Tiến hành demo sẽ tạo áp lực để nhóm tiến tới *hoàn chỉnh th c s một số thứ* và phát hành nó (ngay cả khi nó chỉ là môi trường kiểm thử). Không có buổi demo, chỉ chúng ta mới biết rằng 99% công việc đã hoàn thành. Với các buổi demo chúng ta có thể chỉ có một số hạng mục được hoàn thành, nhưng chúng là những hạng mục đã *th c s hoàn thành*, điều này (trong trường hợp của chúng tôi) còn tốt hơn có cả đống những thứ mới chỉ *s hoàn thành* và sẽ làm ô nhiễm Sprint tiếp theo.

Nếu một nhóm ít nhiều bị áp lực để tiến hành demo Sprint, thậm chí khi họ không có nhiều thứ thực sự hoàn chỉnh, buổi demo sẽ gặp phải khó khăn. Nhóm sẽ lúng túng và dễ sơ sẩy trong khi

78 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

demo và những tràng pháo tay chỉ là nửa vời. Mọi người sẽ cảm thấy một chút tiếc nuối với nhóm, một số có thể sẽ tức tối vì họ đã lãng phí thời gian tham dự một buổi demo tệ hại.

Quả là một sự đau đớn. Nhưng nó có hiệu quả như một liều “thuốc đắng dã tật”. *S rint tới,* Nhóm sẽ thực sự cố gắng để có được những công việc được gọi là *hoàn thành*! Họ sẽ cảm thấy rằng “tốt thôi, có thể chúng ta sẽ chỉ trình diễn 2 thứ vào Sprint tới thay vì 5, nhưng những thứ chết tiệt này sẽ thực sự CHẠY NGON!”. Nhóm biết rằng họ sẽ phải làm một buổi demo mà không xảy ra vấn đề gì, đây là dấu hiệu làm tăng cơ hội có một điều gì đó hữu ích để demo. Tôi đã từng thấy điều này xảy ra một vài lần.

**Những mục cần kiểm tra cho buổi Sơ kết Sprint**

● Hãy chắc chắn rằng bạn trình bày rõ được mục tiêu của Sprint. Nếu có ai đó tham dự buổi demo mà không hiểu về bất cứ điều gì liên quan đến sản phẩm của bạn, hãy dành một chút thời gian để mô tả về nó.

● Đừng tốn quá nhiều thời gian chuẩn bị cho buổi demo, đặc biệt là để có một bản thuyết trình hào nhoáng. Cắt bỏ những nội dung vớ vẩn và chỉ tập trung trình bày các mã nguồn thực sự chạy tốt.

● Giữ nhịp độ cao, tức là tập trung những nội dung chuẩn bị của bạn vào việc tạo ra bản demo có tốc độ cao chứ không phải đẹp.

● Giữ cho nội dung demo ở mức hướng vào các nghiệp vụ, bỏ qua các chi tiết kỹ thuật. Tập trung vào “những gì chúng tôi đã làm” hơn là “cách mà chúng tôi đã làm”.

● Nếu có thể, hãy để cho khách tham dự sử dụng thử sản phẩm.

● Đừng trình bày việc sửa một loạt các lỗi nhỏ và các tính năng tầm thường. Đề cập đến chúng nhưng không demo, vì điều đó thường tốn thời gian và giảm trọng tâm đối với những thứ quan trọng hơn.

**Giải quyết với những thứ chưa thể demo**

**Thành viên Nhóm:** “Tôi sẽ không trình bày tính năng này, bởi vì nó không thể demo được. Chức năng đó yêu cầu ‘Tăng khả năng đáp ứng của hệ thống để có thể phục vụ đồng thời 10,000 người dùng’. Tôi có thể không bỏ công sức để có được 10,000 người dùng đồng thời cho việc demo có được không?”

**Scrum Master:** “Anh đã hoàn thành tính năng này chưa?”

79 | SCRUM VÀ XP TỪ NHỮNG CHIẾN HÀO

**Thành viên Nhóm:** “Vâng, tất nhiên rồi”.

**Scrum Master:** “Làm sao anh biết được điều này?”

**Thành viên Nhóm:** “Tôi đã cài đặt hệ thống vào trong một môi trường kiểm thử hiệu suất, khởi động với 8 máy chủ và tạo những yêu cầu đồng thời để thử nghiệm hệ thống”.

**Scrum Master:** “Nhưng anh có cái gì để xác định rằng hệ thống sẽ đáp ứng được 10,000 người dùng”.

**Thành viên Nhóm:** “Có. Các máy kiểm thử hơi tồi, chúng không thể đáp ứng được 50,000 yêu cầu đồng thời trong suốt quá trình kiểm thử”.

**Scrum Master:** “Làm sao anh biết được điều này?”

**Thành viên Nhóm (**đã bắt đầu nản): “Vâng, tôi có báo cáo đó! Anh có thể tự xem, nó cho thấy cách thiết lập việc kiểm thử và cách gửi đi nhiều yêu cầu!”

**Scrum Master:** “Ồ, tuyệt vời! Vậy thì khi nào demo anh chỉ cần trình bày báo cáo này và thảo luận nó với người tham dự. Điều này tốt hơn là chẳng làm gì đúng không?”.

**Thành viên Nhóm:** “Ồ, như vậy là đủ sao? Nhưng nó hơi xấu, cần phải chỉnh sửa cho đẹp hơn”.

**Scrum Master:** “OK, nhưng đừng tốn quá nhiều thời gian cho việc đó. Nó không cần phải đẹp, chỉ cần đầy đủ thông tin”.