Agile Pitfalls

หลุมพรางอไจล์ : คู่มือโค้ชและทีมมือใหม่

Varokas Panusuwan

Agile Pitfalls

หลุมพรางอไจล์ : คู่มือโค้ชและทีมมือใหม่

Varokas Panusuwan

This book is for sale at http://leanpub.com/agilepitfalls

This version was published on 2014-11-29

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Tweet This Book!

Please help Varokas Panusuwan by spreading the word about this book on Twitter!

The suggested hashtag for this book is #agilepitfalls.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

https://twitter.com/search?q=#agilepitfalls

สารบัญ

Acknowledgement

ขอบคุณ คุณพ่อ และ คุณแม่ ที่ให้ความสนับสนุนทุกอย่างตั้งแต่เล็กจนโต

ขอบคุณเพื่อนๆ พี่ๆ หัวหน้า และลูกน้อง ที่คอยแนะนำสั่งสอนแลกเปลี่ยนความรู้เรื่องอไจล์อย่างไม่เคยหยุด

ขอบคุณประสบการณ์ โอกาส ความสำเร็จ ข้อผิดพลาด ของตัวเองและของคนอื่นที่ได้โชคดีไปพบเจอ จนเรียบเรียงเป็นตัวหนังสือถ่ายทอดให้รุ่นต่อไปได้อ่าน

Introduction

แนะนำหนังสือ

หนังสือเล่มนี้รวบรวมมาเป็นข้อผิดพลาดหลุมพรางและกับดักที่พบเป็นประจำในทีมอไจล์ที่เพิ่งเริ่มหัด หรือ ทีมที่ขาดการแนะนำจากอไจล์โค้ชที่มีประสบการณ์ เขียนจากทดลองใช้ และจากการโค้ชทีมอไจล์ทั้งในประเทศไทยและในต่างประเทศ ทั้งทีมที่ทำงานอยู่ที่เดียวกัน กระจายกันอยู่ตามประเทศต่างๆ และทีมออนไลน์ ทั้งในทีมสตาร์ทอัพขนาดเล็ก จนถึงองค์กรขนาดใหญ่ที่มีทีมอไจล์หลายทีมประสานงานร่วมกัน

ตีแผ่สิ่งที่ทีมอไจล์ไม่ว่าจะเป็นแบบไหน หรือมีขนาดเท่าไหร่ มักจะก้าวพลาดเหมือนๆกันโดยไม่รู้ตัว สัญญาณอันตรายต่างๆ ที่โค้ชควรจะสังเกตุและตัวอย่างทางเลือกในการแก้ไข รวมถึงเจาะลึกแนวคิดและอุปนิสัยของทีมอไจล์ในประเทศไทย เหตุผลเบื้องหลัง ทำไมทุกทีมเริ่มใหม่จึงตกหลุมคล้ายๆ กัน

หนังสือ เล่ม นี้ เขียน มา จาก ความ รัก และ หลง ไหล ใน วิศวกรรม ซอฟต์แวร์ ระบบ กระบวนการ ผลิต ซอฟต์แวร์ และ การ ค้นหา ที่ ไม่ รู้จบ ของ วิธี การ ทำงาน เพื่อ ให้ ได้ มา ซึ่ง ซอฟต์แวร์ ที่ เกิด ประโยชน์ สูง สุด กับ ผู้ ใช้ งาน

ติชม แนะนำ และสั่งซื้อในรูปแบบ e-book ได้ที่ www.agilepitfalls.com[1]

[1] http://www.agilepitfalls.com

แนะนำผู้เขียน

วิศวกรคอมพิวเตอร์ผู้หลงไหลในการค้นหากระบวนการผลิตซอฟต์แวร์ที่มีคุณภาพ และสร้างความสุขให้กับผู้ใช้มากพอๆ กับผู้ที่พัฒนามันขึ้นมา และเชื่อว่าภาษาที่ใช้สื่อสารได้ดีที่สุดระหว่างโปรแกรมเมอร์คือตัวโปรแกรมที่เขียนอย่างสะอาดด้วยความตั้งใจ

ปัจจุบันทำงานอยู่ที่เมือง Seattle ประเทศสหรัฐอเมริกา ประสบการณ์เริ่มจากทีมที่เพิ่งหัดอไจล์ได้ไม่กี่เดือนจนขยายเป็นทีมอไจล์นานาชาติ ในบริษัทที่ได้รับการยกย่องจากในวงการว่าสามารถใช้อไจล์เป็นข้อได้เปรียบทางธุรกิจ โดยการตอบสนองความต้องการของลูกค้าได้รวดเร็วกว่าคู่แข่ง และได้รับการยกย่องจาก Forbes Magazine อยู่ใน World's 100 Most Innovative Companies ในปี 2013

หนึ่งในผู้ริเริ่มการจัดงานอไจล์ครั้งแรกของประเทศไทย Agile Thailand และ Agile Tour Bangkok ร่วมกับกลุ่ม Agile66 ในปี 2012 และในปีต่อๆ มา

เป็นผู้ริเริ่มโครงการ HuskyCode เปิดเวทีให้ผู้ที่ไม่มีโอกาสทดลองอไจล์ในงานของตนเอง มาทดลองทำจริงผ่านทีมออนไลน์ และนำเสนอผลลัพธ์ที่ได้ในงานสัมนาอไจล์ในประเทศ

ในเวลาว่าง เป็นผู้บรรยายเรื่องอไจล์ทั้งในประเทศไทยและต่างประเทศ ในบริษัทและมหาวิทยาลัยต่างๆ

การศึกษา ปริญญาตรี วิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพมหานคร และ ปริญญาโท Software Engineering, Carnegie Mellon Univerisity, Pittsburgh, USA

บทนำ

ยุคนี้สมัยนี้แล้วใครๆ ก็อยากอไจล์เป็นเหมือนกับเค้าบ้าง อยากมีทีมอไจล์เป็นของตัวเองบ้าง อยากทำให้องค์กรเราสามารถผลิตของออกมาให้ลูกค้าได้ใช้เร็วๆ มากกว่าเอาเอกสารเป็นเล่มๆไปส่ง ก็เป็นเรื่องโชคดีนะที่การที่จะพาทีมของเราไปสู่ความเป็นอไจล์ไม่ได้มีคำตอบเดียว แต่ก็โชคร้ายอยู่เหมือนกันที่ทางต่างๆ เหล่านี้มักจะเต็มไปด้วยหลุมพรางและกับดักมากมาย อาจจะทำให้เราตกลงไปได้ง่ายๆ ถ้าเราไม่ระวังตัวกันให้ดีๆ

ตลอดเวลาหลายปีที่ผ่านมาที่ผมได้สัมผัสทดลองความเป็นอไจล์ด้วยตัวเอง จากคำแนะนำของผู้รู้จริง จากที่เคยเจ็บมาก่อน จากคนที่ประสบการณ์โชคโชน แม้แต่จากผู้ที่เซ็น Agile Manifesto เองหลายคน และจากการที่ไปช่วยเขาทำทีมอไจล์ให้เกิดขึ้น ผมเห็นตัวอย่างมากมายทั้งที่ประสบความสำเร็จยิ้มแย้ม และที่ติดขัดติดหลุมอยู่ หลุมใหญ่บ้าง หลุมเล็กบ้าง

ผมเป็นคนที่รักในเรื่องของกระบวนการผลิตซอฟต์แวร์มาก (เรียกง่ายๆ ว่าบ้าก็ได้) ทุกๆ สิ่งที่ได้เรียนรู้เอง ทดลองแนะนำ เรื่องดีและไม่ดี ก็ติดตามถามดูว่าใช้แล้วได้ผลไหมยังไง ทดลองปรับเปลี่ยนไปเรื่อยๆ นานๆ เข้า พอช่วยมาหลายทีม ดูมาหลายคน ผมเริ่มเห็นสิ่งที่จดไว้มันมีแต่เรื่องเดิมๆ ซ้ำๆ กันทั้งนั้นเลย ผมเริ่มมองเห็นเป็นอาการของสิ่งที่ทำกันผิด หรือไม่เข้าใจกันบ่อย หลุมพรางที่เรามักจะมองไม่เห็นแล้วตกลงไป มาย้อนๆ กลับไปมองดูมันก็เริ่มเห็นเป็นรูปเป็นร่างขึ้นมา พอที่จะเล่าเป็นฉากเป็นตอนขึ้นมาได้

ผมเขียนหนังสือเล่มนี้ขึ้นมา เอาโน้ตต่างๆ ที่จดไว้มาจัดกลุ่มกัน มาเรียงกัน เพื่อจะเล่าย้อนถึงเหตุการณ์เหล่านั้น เรื่องราวต่างๆ ที่ทีมทั้งหลายจะต้องเผชิญ ณ จุดหนึ่ง

ของการเดินทางมุ่งหน้าสู่การเป็นอไจล์ ผมอยากให้ทีมที่เริ่มใหม่ๆ จะได้ไม่ต้องไปผจญหาคำตอบกับสิ่งเหล่านี้มากมายกันอีก

เราจะเล่าถึงส่งต่างๆ ที่จะเกิดขึ้น หลุมพรางและกับดัก เราจะเล่าถึงอาการและข้อสังเกตของเวลาที่เราไม่ได้ เอาหลักการของอไจล์มาใช้ เอามาใช้ไม่ถูกต้อง หรือบางทีก็ไม่เข้าใจอไจล์ไปเลยก็มี

ผมเชื่อว่าทุกปัญหาของอไจล์ ขึ้นอยู่กับบริบท บริษัท บริวารลูกน้อง เพื่อนฝูงและอีกมากมายที่แวดล้อมสภาพการทำงานของเราอยู่ เราจะลองมองปัญหาทุกอย่างแบบลงลึกใกล้ชิด เราจะส่องดูดีๆว่า อะไรนะ ที่เป็นปัญหาที่ทุกคนเจอกันบ่อยๆ ถามคนไหนก็เจอเหมือนๆ กัน มาลองขุดคุ้ยกันดูว่า จริงๆ แล้วต้นตอมันเป็นมาอย่างไร อะไรเป็นจุดกำเนิด เป็นสาเหตุที่พาเราไปสู่หลุมพรางเหล่านั้น เพื่อที่เราจะได้มาดูกันว่าจะรับมือกับมันได้อย่างไร อย่างมีระบบ

ทุกๆ เรื่องที่คุณเผชิญมันก็จะเป็นการผจญภัยของคุณเอง ทุกปัญหาที่พิชิตได้ก็จะเป็นรางวัลและความสำเร็จของคุณเอง หนังสือเล่มนี้เพียงแค่จะชี้ให้เห็นถึงบางจุดที่คุณอาจจะเป็นปัญหาอยู่ สิ่งเล็กสิ่งน้อยที่มันยังอาจจะขาดไป ซึ่งถ้ายังเป็นมือใหม่ก็ยังอาจจะมองไม่เห็นจุดนี้ได้ง่ายนัก

เมื่อไหร่ที่ทีมติดปัญหา ออกตัวไม่คล่อง ผมหวังว่าทุกปัญหาที่เราพูดกันในหนังสือเล่มนี้ ก็คงจะมีซักเรื่องหนึ่งแหละ ที่จะช่วยได้และทำอย่างไรที่เราจะก้าวเดินไปในทางที่จะทำให้เราส่งมอบงานที่มีคุณภาพให้ลูกค้าเราใช้ได้อย่างสม่ำเสมอ อย่าลืมว่าอไจล์เน้นที่การมองตัวเองและปรับปรุงตัวเอง (Inspect and Adapt) อยู่เป็นประจำ เราจะมองเห็นสิ่งที่เราไม่เคยมองเห็นมาก่อนในการปรับปรุงโพรเสส (process) ของเราเอง ปัญหาจะไม่หายไปเฉยๆ เพียงเพราะเราทำอไจล์ สิ่งที่เราทำและการย้อนกลับมามองตัวเองจะเป็นการทำให้ทีมเราดีขึ้นเรื่อยๆ

รูปแบบ สำนวน ภาษา

เล่มนี้ตั้งใจเลยว่าจะไม่พยายามเขียนให้มันเป็นทางการ ไม่พยายามทำตัวเหมือนตำราเรียนเน้นทุกอย่างเป๊ะๆ เพราะปัญหาหลายๆอย่าง ไม่ได้แก้ได้จากทฤษฎีของวิศวกรรมซอฟต์แวร์อย่างเดียว แต่เป็นการเข้าใจถึงธรรมชาติของคน นิสัยจากการทำงานแบบเดิมๆ รวมถึงวัฒนธรรมไทยที่ส่งผลถึงการทำงานร่วมกันเป็นทีมด้วย การเขียน เป็นภาษาพูดน่าจะถ่ายทอดอารมณ์ความรู้สึกของเหตุการณ์นั้นๆ ได้ตรงกว่าและเข้าใจง่ายกว่า เหมือนกำลังนั่ง แบ่งปันเรื่องราวให้กันตอนกินข้าวบนโต๊ะอาหาร

การเขียนเป็นภาษาพูดผมตั้งใจให้เป็นเหมือนการเล่าเรื่องราวประสบการณ์ต่างๆให้ฟัง เอาไว้อ่านก่อนนอนเล่นๆ เอาไว้อ่านคลายเครียดเวลาขี้เกียจเขียนโค้ดแล้วอ่านไปซักพักหนึ่งอาจจะไปเจอกับหลุมที่เราตกอยู่ก็ได้นะ

เวลาที่ผมจะอธิบายเรื่องอะไร ก็จะพยายามแปลเป็นภาษาไทยให้ตรงที่สุด และจะวงเล็บคำภาษาอังกฤษให้ครับ

ก้าวแรกสู่อไจล์

> “We are uncovering better ways of developing software by doing it and helping others do it”
>
> - บทนำ Agile Manifesto -

ก้าวแรกของอไจล์จะเป็นก้าวที่ยากที่สุด ไม่ใช่เพราะความซับซ้อนของมันหรอก แต่เป็นเพราะว่าอไจล์คือแนวทางกว้างๆ แทนที่จะบอกให้ทำอะไรตามคำสั่งที่มีใครเบื้องบนเขียนไว้ให้ทุกขั้นตอน

มันไม่ใช่เรื่องง่ายเลย ที่อยู่มาวันหนึ่งตัวเรา ทีม และองค์กรจะทิ้งสิ่งต่างๆ ที่เคยเชื่อและยึดกับมันมานานที่ส่งต่อกันมาหลายรุ่นแล้วว่ามันคือสิ่งที่ “คนแถวนี้เขาทำกันเป็นประจำ” มาลองสิ่งที่ดีกว่าในการทำงานได้มีประสิทธิภาพมากขึ้น ตอบสนองกับการเปลี่ยนแปลงได้ดีขึ้น และสนุกกับทุกวันในการสร้างซอฟต์แวร์มากขึ้น

ปัญหาของทุกทีมในการเริ่มต้นอไจล์ จากการทำงานแบบเดิม เกิดความเคยชินและทัศนคติในเรื่องของกระบวนการผลิตซอฟต์แวร์ทั้งนั้น คิดเอากันไปเองเลยว่ามันเป็นเรื่องไกลตัว ต้องมีคนอื่นมาคอยบอกทีมว่าต้องทำตัวยังไง ต้องเป็นผู้เชี่ยวชาญภายนอกที่มาบงการสั่งการให้เราปรับปรุงอะไรบ้าง

เรื่องเหล่านี้แก้ได้ด้วยความเข้าใจ สังเกตุ ทดลองและปรับปรุงระบบงานที่ทำอย่างสม่ำเสมอ

และเรื่องที่ทุกทีมทำพลาดเป็นประจำตอนเริ่มต้นก็คือ...

ทดลองอไจล์ใช้โปรแกรมไหนดี?

ถ้าจะมีคำถามยอดฮิตติดดาวอันดับหนึ่งตลอดกาลของคนที่ตื่นมาวันนี้แล้วเกิดนึกอยากเริ่มทดลองใช้อไจล์เลย ก็คือ คำถามว่า ใช้เครื่องมืออะไรดี? มีโปรแกรมอะไรให้ใช้ได้บ้าง?

คืออยากเป็นอไจล์แล้วล่ะวันนี้เห็นคนอื่นๆเค้าทำกันเราก็เลยอยากทำเป็นบ้าง เอาแบบตูมเดียวจบ นัดเดียวจอด มาให้ไวเลย

ก็ไม่รู้ว่าโชคดีหรือโชคร้ายของเรานะที่โลกยุคนี้แล้ว จะทำอะไรก็มีแอพให้ใช้ทั้งนั้น ถ้าลองไปหาในกูเกิ้ลดู จะเห็นโปรแกรมช่วยทำอไจล์อย่างน้อยเป็นสิบๆ มีทั้งที่ต้องเอามาลงในเครื่องตัวเอง หรือพวกที่ต้องเข้าไปใช้ในเวปเค้าก็มี ที่ให้ใช้ฟรีก็มี แต่ส่วนมากจะเสียเงินเพราะโปรแกรมเมอร์ทุกคนก็ต้องกินข้าวเหมือนๆกันทุกคน(จริงไหม?)

ทีนี้ก็เลยไม่รู้เลยว่าจะเอาไหนดี ผมตอบสั้นๆ ได้เลยว่า เริ่มแบบนี้ใช้โปรแกรมอะไรมันก็ผิดทั้งนั้นแหละครับ...

โปรแกรมไหนก็ผิด

ปัญหาไม่ได้อยู่ที่ว่าโปรแกรมไหน ดีหรือไม่ดี เหมาะหรือไม่เหมาะ ปัญหาอยู่ที่เราเข้าใจตัวเอง ทีมของตัวเอง และอไจล์ดีแค่ไหนตอนเริ่มต้น?

เริ่มวันแรกของการพยายามจะถีบตัวเองไปหาอไจล์ ถึงจะมีโค้ชหรือไม่มีโค้ชมาคอยบ่นให้ฟังข้างๆ ทุกคนไม่มีใครแน่ใจเลยว่าอไจล์คืออะไร เอาเข้าจริงๆ ตัว 4 บรรทัดกับอีก 12 หลักการใน Agile Manifesto[2] ตอนเริ่มก็ดูเป็นสิ่งที่มันกว้างมากๆ ทำตัวต่อไปไม่ถูก

[2] http://agilemanifesto.org/

พอทุกคนอ่านจบไปห้ารอบกว่าๆแล้ว ก็ยังนึกไม่ออกว่าถ้าทำไปแล้วจะทำให้ชีวิตเราดีขึ้นยังไง? ที่สำคัญคือ ยังมองไม่เห็นทางเลยว่า ข้อๆ ที่เขียนใน Manifesto จะทำให้เราเริ่มอไจล์ง่ายขึ้นตรงไหนเนี่ย? ใครโชคดีหาโค้ชมาได้ ถึงจะเก่งแค่ไหนก็ต้องใช้เวลาหลอกล่อ เกลี้ยกล่อม เล่นแบบฝึกหัดจัดเกมกันอีกนาน ของที่เขาสะสมมาเป็นหลายปี จะให้ถ่ายทอดกันชั่วโมงเดียวมันได้ไม่หมดหรอก ก็จะเริ่มมองเห็นเข้าใจกันทีละด้านเล็กๆ มุมเล็กๆ ซอกเล็กๆ

นอกจากนั้น ทีมหรือองค์ที่เริ่มจะลองเป็นอไจล์ก็มักจะเป็นทีมที่ทำงานแบบเดิมมาแล้วซักพักนึงแล้วด้วย (ที่มักจะเรียกกันว่า Waterfall เขียนเอกสารเป็นตั้งๆ นั่นแหละ) พอเริ่มเปลี่ยนมาเป็นอไจล์ก็ติดภาพเหมากันเอาไปเองว่า Software Engineering หรือ วิศวกรรมซอฟต์แวร์คือการทำงานตามกรอบแบบและวิธีการ process ที่มีใครฉลาดๆ□ ซักคนสร้างไว้ เพราะปกติในบริษัทจะมีทีมรับผิดชอบเรื่องนี้ไปเลยสั่งการลงมาโดยตรงเลยว่า ต้องเขียนเอกสาร กรอกฟอร์มอะไรกันบ้าง

โดนสะกดจิตอย่างนี้นานๆเข้าก็จินตนาการฝันหวานไปเองว่า โลกนี้คงมี process อะไรซักแบบที่เป็นเนื้อคู่ที่สร้างมาสำหรับแก้ปัญหาทุกอย่างสำหรับทีมของเรา หวังว่าวันที่หาเจอก็หยิบใช้ปุ๊บ ก็ติดกันปั๊บ เหมาะกันดีทุกอย่าง

จากนั้นก็สรุปกันต่อเองว่า อไจล์ก็เป็นแค่ขั้นตอนแก้ปัญหาให้เราได้ทุกอย่างแค่ทำตามสั่งไปเรื่อยๆ ก็เลยพยายามมองหาวิธีง่ายๆเร็วๆ นับ 1-2-3 เพื่อจะเป็นอไจล์ทีเดียว

ด้วยเหตุผลนี้ทุกคนก็เลยเริ่มอไจล์จาการมองหาโปรแกรมเป็นเรื่องแรกไงครับ เพราะหวังว่าโปรแกรมจะมาทำหน้าที่บังคับทำให้ทำงานไปตามขั้นตอน “ที่ถูกต้อง” แบบเดียวที่มีคนคิดมาไว้ดีแล้ว

พอเริ่มแบบนี้ไปซักพักก็จะงงกับโปรแกรม รู้สึกว่ามันไม่ได้ช่วยอะไรชีวิตเราเท่าไหร่ มีทางเลือกปุ่มกดอะไรก็ไม่รู้ตั้งเยอะแยะ มีช่องมีฟอร์มอะไรให้กรอก ที่เราก็พอจะอ่านเข้าใจว่าคืออะไรเหมือนกันแหละ แต่ไม่รู้ว่าช่วยให้ทำงานดีขึ้นได้ยังไง?

สุดท้ายก็หลายทีมก็ตัดสินใจทิ้งโปรแกรม ทิ้งอไจล์ไป แล้วก็สรุปเอาว่า “สิ่งนี้ไม่

เหมาะกับเรา”

Software กับ Process เป็นของเราเท่าๆ กัน

คนทุกคนที่ยึดอาชีพหลักเป็นการเข็นซอฟต์แวร์ออกมาให้โลกนี้ใช้งานจะรู้ว่ากว่าจะออกมาแต่ละ อย่างไม่ใช่เรื่องง่ายๆ เลย ต้องใช้เวลา ใช้พลังงาน ใช้ความคิดสร้างสรรค์ทุ่มเทให้กับโปรเจคอย่างเต็มที่ พอได้สิ่งที่มีประโยชน์ใช้งานได้ออกมาทุกคนก็จะมีความภูมิใจในซอฟต์แวร์ที่นั่งปั้นมาเองกับมือ จะมีความรู้สึกว่าเป็นเจ้าของ อยากทำให้ออกมาได้ดี ให้ลูกค้าได้ใช้แล้วเกิดประโยชน์มีความสุข

แต่หัวใจหลักของการทำงานที่จะประสบความสำเร็จไปได้เรื่อยๆ คือ ทีมและองค์กรต้องถือว่าตัวเองเป็นเจ้าของกระบวนการทำงาน□ วิธีการทำงาน รวมถึงวัฒนธรรมองค์กรของเราเองด้วย เพราะมันเป็นสิ่งที่เราต้องเจออยู่ทุกวัน ทำอยู่ทุกวัน

คิดได้อย่างนี้จะเห็นว่า ถ้าวิธีการทำงานเป็นของเราเอง ก็ไม่เห็นจะต้องถูกบังคับด้วยเครื่องมือหรือ framework การทำงานแบบใดแบบหนึ่งเลย อาศัยการศึกษาเข้าใจตัวเองดีกว่าว่า เราต้องการให้วิธีการทำงานของเรามีอะไรดีขึ้น เวลาอ่านข่าวแล้วเห็นคำพูดฮิตๆกันในวงการเนี่ย มันจะมาทำให้ชีวิตเราดีขึ้นได้แบบไหนยังไงบ้าง

อไจล์เป็นหลักการทำงานที่เชื่อเรื่องการสร้างคุณค่าให้กับลูกค้า ผ่านทางการส่งมอบของไปให้ดูและมาคุยกันปรับแก้ไขบ่อยๆ ไม่ใช้เป็นกระบวนการข้อกำหนดแบบใดแบบหนึ่งให้ทำเป๊ะๆ ซึ่งวิธีการจะเอาไปใช้กับทีมตัวเอง ต้องผ่านการสังเกตทดลอง และปรับใช้อยู่ตั้งนานกว่าจะรู้สิ่งที่เหมาะกับแต่ละที่ อไจล์คือการค้นหาและการทำความเข้าใจในระบบงานเรา มากกว่าการที่จะเสาะหาโปรแกรมที่ตรงล็อคกับเราพอดีเป๊ะ

คนทำโปรแกรมอไจล์เวลาเขาทำก็คงไม่ได้ออกแบบให้เอามาใช้ได้กับทีมเดียวจบต้องทำมาให้ใช้ได้หลายๆ แบบ เผื่อให้ทีมต่างๆกันเอาไปใช้ได้ทุกทีม ก็เลยมีทาง

เลือกเผื่อไว้มากมาย ถ้าทีมรู้แล้วว่าตัวเองอยากทำอะไรนำส่วนต่างๆไปใช้ตามที่ตัวเองต้องการ ทีมที่ไม่เป็นตั้งแต่แรกกระโดดลงมาลองใช้โปรแกรมเลยก็งงแน่นอนว่าจะไปต่อยังไง ทางเลือกมันเยอะเหลือเกิน

เครื่องมือค่อยตามมาหลังความเข้าใจ

ทีมถ้าไม่ได้ถูกบังคับจากเบื้องบน เช่นมีเจ้านายบอกให้ทำอไจล์วันนี้เลยพร้อมทั้งซื้อโปรแกรมมาให้ด้วยเสร็จสรรพ ตามปกติผมจะแนะนำให้ทีมเริ่มจากสองมือที่มีกับหนึ่งกระดานไวท์บอร์ดครับ นอกจากง่ายเริ่มได้โดยไม่ต้องลงทุนอะไรมากแล้ว ยังทำให้เราไม่ต้องไปเสียเวลากับการเถียงกันว่าจะใช้โปรแกรมอะไร? มีข้อดีข้อเสียยังไง? แพงไหม? ลงยังไง?

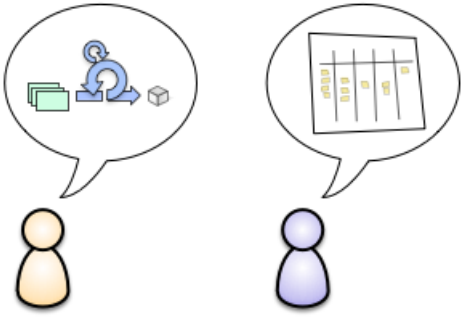
อยากให้กลับมาเถียงกันมากกว่าว่า เราทำอะไรได้บ้างเพื่อให้ทั้งทีมเห็นสถาณะตรงกัน มองเห็นปัญหาที่เกิดขึ้นในระบบได้ตรงกัน ลากมาถึงว่าเราจะทำให้วิธีทำงานเราดีขึ้นยังไง ผ่านทางกระบวนการที่เห็นตรงกันทุกคน

ลองไปดูว่าเค้าทำ Scrum หรือ Kanban กันยังไงแล้วลองวาดๆ เขียนๆ ดูว่าทำให้ทีมเราเข้าใจสถาณะกันดีขึ้นไหม? ถ้ามีอะไรไม่เข้าใจ จะเติมเส้นอะไร หรือเพิ่มช่องอะไรเข้าไปไหม ข้อดีของการใช้กระดานคือ เราจะวาดเส้น กำหนดกฎอะไรก็ได้ เพิ่มเข้ามาเมื่อไหร่ก็ได้ ลองจนกว่าจะเริ่มเข้าที่แล้วค่อยคิดว่าจะหาโปรแกรมอะไรมาใช้

สุดท้ายสิ่งที่เราขีดๆ เขียนๆ ก็เป็นสเปคอย่างดีเวลาจะไปเลือกซื้อโปรแกรม เพราะเรารู้แล้วว่าอยากได้อะไร ฟีเจอร์ไหนของโปรแกรมเป็นประโยชน์ สำคัญหรือไม่สำคัญยังไง ก็จะได้ภาพชัดกว่าเดาๆเอาตั้งแต่แรกตั้งเยอะ

หลายองค์กรติดใจไม่ยอมเลิกใช้กระดานกับกระดาษก็มี

Scrum หรือ Kanban ดี?



บางทีมไม่เสียเวลากับการเลือกเครื่องมือ แต่ก็เสียเวลาไปกับการมาเถียงกันว่าจะเริ่มอไจล์กันท่าไหนดี

ส่วนมากก็จะเริ่มกันที่ Scrum หรือไม่ก็ Kanban เนี่ยแหละ จริงๆแล้วมันมีอย่างอื่นอีกตั้งเยอะ เช่น Crystal, XP, OpenUP แต่ตอนคนดูจะให้ความสำคัญกับสองอย่างนี้มากที่สุด

มุมแดง: Scrum

Scrum เป็นวิธีการทำงานที่มีหลักการแน่นอนแบ่งเป็นหัวข้อชัดเจน ว่าจะต้องเอาทีมที่มีความสามารถหลากหลายมารวมอยู่ด้วยกัน แล้วต้องมีการกำหนดการออก Release อย่างสม่ำเสมอ ในเอกสาร Scrum ตั้งต้นจะบอกว่าทุกเดือน แต่ช่วงหลังๆ ทีมมักจะออกทุกสองสัปดาห์มากกว่า บางทีมลุยกว่านั้นก็ออกทุกสัปดาห์ยังได้เลย

นอกจากนั้นก็ยังมีข้อกำหนดเล็กๆน้อยๆ เกี่ยวกับการเข้ามาคุยกันทุกวันและการ

วางแผนในช่วงต่างๆ อีกไม่เยอะมาก รายละเอียดที่เหลือลองอ่านดูใน Scrum Guide[3] เองได้

แน่นอนว่าการกำหนดแนวทางคร่าวๆไปเลยว่าต้องมีกิจกรรมอะไรบ้าง พร้อมคำแนะนำว่าให้จัดออกมาใช้เวลาไหน ก็ทำให้เข้าใจง่ายเห็นภาพชัด แต่ก็มีผลทำให้จูงใจคนอื่นลองเปลี่ยนมาทำตามข้อเสนอของ Scrum ยากหน่อย เพราะอยู่ดีๆ เหมือนโดนล้มโต๊ะพลิกกระดานหลายอย่าง

โดยเฉพาะกับบริษัทที่มีแรงต้านเรื่องของการเปลี่ยนแปลงสูง หรือบริษัทที่เจ้านายเข้ามาจัดการกับการทำงานของทีมเราเยอะหน่อย ก็จะไม่ค่อยปลื้มกับการให้ทีมเปลี่ยนมาจัดการตัวเองพลิกกระดานแบบ Scrum เท่าไหร่ เพราะอยู่ดีๆเหมือนโดนลุกขึ้นมาต่อต้านการทำงานแบบเดิมซะอย่างนั้น เรื่องนี้ไม่ค่อยมีปัญหาถ้าจัดการค่อยๆใส่เข้ามาทีละน้อยๆ ค่อยๆคุยกับหัวหน้าเรื่องผลดีผลเสีย ถ้ามีโค้ชก็สามารถให้ช่วยคุยได้อีกแรงหนึ่ง

ยิ่งถ้าหัวหน้าเป็นคนเสนอมาเอง ก็ตัดปัญหานี้ไปได้เลยทีเดียว

มุมน้ำเงิน: Kanban

Kanban เริ่มจากอีกมุมนึง คือ การไม่พยายามไปเปลี่ยนอะไรให้คนเค้าแตกตื่น แค่เริ่มจากความเข้าใจของระบบงานที่ทีมเราเองทำอยู่ ว่าการจะดันของจากต้นจนจบใช้งานได้ จะต้องผ่านกระบวนการอะไรบ้าง งานชิ้นไหนอยู่ที่ไหน ให้ทุกคนเห็นตรงกันเหมือนกัน จากนั้นค่อยๆ ทดลองเปลี่ยนทีละนิดๆ แล้ววัดผลไปเรื่อยๆ

หลักการของการไปสู่ความเป็นอไจล์ของ Kanban รวมๆ คือ ถ้าทำให้ทุกคนเห็นชัดตรงกันว่าเราทำงานกันยังไง ก็สามารถทำให้ทุกคนเห็นสิ่งที่ทำไปแล้วเสียเวลาไม่เกิดคุณค่าระบบได้ตรงกัน พอมองออกเหมือนกันแล้ว ก็พยายามลดงานในระบบที่ไม่จำเป็นต้องทำ โดยใช้อาศัยเครื่องมือหลักคือ การจำกัดงานที่ค้างอยู่

[3] https://www.scrum.org/Scrum-Guide

ในระบบ(work-in-progress) ใครสนใจเพิ่มเติมลองอ่านได้จากหนังสือของ David J Anderson เรื่อง Kanban[4] ได้

เลือกยังไงดี

แล้วแบบไหนจะเหมาะกับองค์กรเราเนี่ย? ตอบได้เลยว่าถ้าขึ้นอยู่กับเราคนเดียวไม่ต้องสนใจชาวบ้านรอบข้าง เริ่มทำ Scrum ง่ายกว่าแน่ๆ เพราะมีเขียนบอกชัดเลยว่ามีอะไรต้องเปลี่ยนแปลงบ้าง มาเป็นเซตเหมือนชุดอาหารญี่ปุ่น นึกถึงยาชุดก็ได้ที่จัดมาแล้ว ว่าทีมมักจะต้องเสริมตรงไหนถึงจะทำให้ไปสู่อไจล์ง่ายขึ้น เช่น ทีมอไจล์มือใหม่มีปัญหาเป็นประจำในเรื่องของการตัดงานให้มีขนาดเล็กลง ก็เลยมีกรอบให้ส่งของที่ใช้งานได้ทุกๆ Sprint หรือปัญหาการไม่ได้คุยกันในทีมก็โดนกำหนดให้มาเจอกันอย่างน้อยชักวันละครั้ง สรุปคืออะไรที่ช่วยพาให้ทีมเป็นอไจล์ไปได้เร็วขึ้น ก็มีแนวทางเป็นกรอบกว้างๆมาให้แล้วลงมือเริ่มทำได้เลย ไม่ต้องเข้าใจทุกอย่าง 100% ก็เริ่มลงมือได้

ปัญหาของการพลิกกระดานเปลี่ยนมาใช้ Scrum รวดเดียวเลยมีสองอย่าง

อย่างแรกถ้าองค์กรมีมานานแล้ว วัฒนธรรมองค์กรแบบเดิมฝังรากลึกเหนียวแน่นและมีแรงควบคุมจากอำนาจลี้ลับด้านบนๆสูง ก็จะทำเปลี่ยนแปลงอะไรใหญ่ๆทีเดียวลำบาก โดยเฉพาะหลักการของทีมที่มีทุกอย่างครบในตัวเอง ทำให้ต้องไปดึงเอาคนทุกแผนกเกี่ยวข้องกับการทำซอฟต์แวร์มาอยู่ใกล้ๆกันตั้งแต่ต้นจนจบ ก็จะต้องเจอกับแรงต้านทานสูงมากถ้าคนที่ไปดึงมามีพลังภายในไม่สูงพอ

อย่างที่สองคือ การใช้ Scrum เปรียบเสมือนกินยาชุด มีตัวยาหลายๆอย่างแก้อาการหลายๆแบบในทีเดียวกัน ในความเป็นจริงแล้ว ทีมแต่ละทีมก็จะมีปัญหาเรื่องใหญ่เล็กไม่เท่ากัน สิ่งที่ทีมต้องการแก้ไขปรับปรุงมากที่สุดตอนนี้มันอาจจะไม่ตรงกับของหยิบมาจาก Scrum มาเริ่มลองทำก็ได้ ถ้ามองออกว่าคืออะไร

[4] http://books.google.com/books/about/Kanban.html?id=RJ0VUkfUWZkC

จะเริ่มจาก Kanban ก็ไม่ใช่ว่าจะง่ายกว่า (แค่ดูเหมือนจะง่ายกว่าเฉยๆ) เพราะตอนเริ่มก็แค่ตั้งกระดาน ตีเส้น แล้วก็แสดงสถาณะของงานที่อยู่ในทีมเท่านั้น ก็ดูเหมือนง่ายดี แต่พอสร้างกระดานเสร็จแล้วก็นั่งมองตาปริบๆ แล้วไงต่อดีล่ะ? การที่เริ่มจากหลักการกว้างๆถ้าไม่มีคนที่ประสบการณ์มาช่วยแนะนำจะติดอยู่ตรงนี้นานมาก เพราะดูไม่ออกว่าระบบการทำงานของตัวเองเป็นยังไง แล้วจะแก้ไขปัญหาได้ด้วยวิธีไหน? โดยเฉพาะถ้ามีคอขวดที่เกิดขึ้นในแต่ขั้นมีวิธีลองแก้ยังไง? ไม่มีตัวอย่างให้ไปหาอ่าน หาดูได้ที่ไหนซะด้วย ทีมมือใหม่ลองทำ Kanban เองเลยจะติดอยู่ตรงนี้นานมาก ส่วนมากก็จะไปยืมๆ เอาพวกกิจกรรมใน Scrum เข้ามาลองทำแก้ปัญหาไป แก้ตรงบ้างไม่ตรงบ้าง

สรุปแล้ว Scrum เอาไว้ลองทำได้เลย มีกรอบง่ายๆมาให้ลองทำอยู่แล้วเป็นชุด เริ่มแบบโหดๆยากๆ ไปเลยแล้วค่อยมาดูว่าเรายังต้องเสริมอะไรเพิ่ม หรือ มีอะไรที่ไม่จำเป็นเข้ามา ส่วน Kanban อาจจะทำได้ง่ายหน่อยในตอนเริ่ม แต่พอจะก้าวต่อไปไม่มีขั้นตอนแน่นอน ต้องอาศัยคนที่มีประสบการณ์ในการดูมาช่วยเยอะเหมือนกัน

สุดท้ายจะทำแบบไหน สิ่งที่สำคัญที่สุดคือ อย่ามัวแต่คิด อย่ามัวแต่วิเคราะห์นานๆ แล้วไม่ลงมือทำอะไรซักที ถึงจะเริ่มผิดก็ดีกว่าไม่ได้เริ่ม อย่างน้อยลองทำให้เห็นพิสูจน์ชัดๆไปเลยว่ามันไม่เหมาะ ดีกว่าเสียเวลาคนในทีมมาเถียงกันไปเถียงกันมา คนในทีมเวลาคุยกันเรื่องนี้ก็จะพยายามยกตัวอย่างสมมติจากอากาศลอยๆกันเองว่าถ้าทำแบบโน้นแบบนี้จะเป็นยังไง ซั่งตวงวัดวิเคราะห์ข้อดีข้อไม่ดีอยู่นานมาก ขอให้ลองเลือกมาซักอย่างเลยครับ แล้วจับเวลาไว้ซักสัปดาห์นึง แล้วลองสังเกตดีๆว่าเราได้เรียนรู้อะไรจากการทดลองใช้ของตัวเอง

ตกลงอไจล์คืออะไร?

ปัญหาที่ดูธรรมดาๆแบบนี้มักจะไม่ใช่ปัญหาแรกที่เจอในทีมในวันแรกที่เริ่มลองเป็นอไจล์ แต่รับรองว่าผ่านไปแค่ไม่กี่สัปดาห์ ก็ต้องมาจะเถียงกันเรื่องนี้แน่ๆ ซึ่งก็

น่าแปลกใจเหมือนกันนะว่า ทำไมทีมถึงมาสงสัยเรื่องนี้หลังจากเรื่องของการหยิบโมเดลสูตรไหนไปใช้ คือมักจะเถียงกันก่อนเลยว่าจะใช้ Scrum หรือ Kanban แต่ไม่ค่อยมีทีมไหนมาเริ่มคำถามแรกว่า อไจล์คืออะไร? ทำแล้วดียังไง?

เหมือนกับส่วนมากจะคิดว่าหยิบเลือกของที่ถูกต้องมาใช้แล้วก็จะทำให้ปัญหามันจบทุกอย่าง...

พอลองทำไปไม่กี่วัน ก็เริ่มมาสำนึกได้ว่าโลกของอไจล์ไม่ได้จบแค่ว่าเลือกโมเดลมาถูกต้องหรือเปล่า แต่ต้องหาอะไรมาเพิ่มต่อจากที่เขียนไว้ในหนังสืออีกเยอะ

คนในทีม จุดนี้ก็จะเริ่มพยายามหาต่อละว่าอะไรขาดไปบ้าง อ่านหนังสือเล่มอื่นเยอะขึ้น เอาไอเดียจากภายนอกมาเติมมาดัดแปลงของเดิมที่มีอยู่ รวมถึงเอาความคิดของตัวเองมาเสนอให้ทีมปรับปรุงระบบการทำงานเดิมที่มีอยู่ ซึ่งจริงๆเป็นเรื่องดีมากที่มาคุยกันเพื่อปรับปรุงระบบการทำงานของตัวเอง

แต่ทุกทีมจะมาติดหล่มเหมือนกันหมดก็คือ พอเริ่มมีหลายไอเดียเข้า ก็จะเริ่มเถียงกันนาน(มากๆ)ว่าไอเดียไหนเหมาะหรือไม่เหมาะ ลากเลยไปคุยกันเรื่องว่า ถ้าทำตามสิ่งที่เสนอมา ทีมยังจะเป็นอไจล์อยู่ไหม หรือจะทำให้ความเป็นอไจล์เสียไปไหม ซึ่งพอเถียงกันแบบถึงตรงนี้ ทุกคนก็พยายามจะมา “ปรับความเข้าใจ” กันว่าอะไรคือนิยามของอไจล์ของแต่ละคน

แค่อ่านคำบรรยายเฉยๆ ก็ฟังดูเหนื่อยแล้ว ทีมที่เถียงกันยาวจนจบกระบวนการนี้ก็ต้องเหนื่อยมากแน่ๆ สุดท้ายเหมือนทุกคนจะลืมไปแล้วว่ามาคุยกันว่าจะทำยังไงให้ทีมทำงานได้ดีขึ้น แล้วก็จะวนๆกันอยู่เรื่องแถวๆ นิยามของอไจล์ จนกว่าจะเบื่อกันเอง พอสลายวงไปก็ไม่ได้ปรับปรุงอะไรเพิ่มซักที มาติดล็อกกับการรอคำตอบจากสวรรค์ว่าจริงๆ แล้วอไจล์คืออะไร

เปิดหนังสือตอบ

คำตอบมาตรฐานขวานผ่าซากกางตำรามาให้ ก็คือโยนไปให้ต้นแหล่งเลย ว่าถ้ายึดเอาตามหลักการของ Agile Manifesto[5] ที่มีอยู่ 4 หัวข้อ 12 หลักการ ทำไปเรื่อยๆ ก็เป็นอไจล์แล้วล่ะ

ง่ายไหม? แต่ทำไมถึงกลายเป็นเรื่องยากไปได้?

สิ่งที่ยากในการเป็นอไจล์ ไม่ใช่การค้นหาว่าอไจล์คืออะไร แต่คือการเชื่อมโยงประยุกต์ให้หลักการกว้างๆ มาเกี่ยวข้องกับสิ่งที่ทำกันในทีมทุกๆวันได้ยังไงต่างหาก

ทีมอไจล์ใหม่มักคิดว่า ถ้าเข้าใจนิยามความหมายมากขึ้นก็ทำให้ “ตีแตก” ทุกอย่างในอไจล์ออกได้ทันที ทำให้เข้าใจกิจกรรมที่ทำใน Scrum หรือ Kanban โดยไม่ผิดเลย ก้าวไปข้างหน้าหรือใส่ของเข้ามาเพิ่มเติมเองได้มั่นใจไม่มีพลาดเอง

ทั้งหมดนี้ ลึกๆแล้วคือ อาการของทีมยังกลัวที่จะค้นหาสิ่งที่เหมาะสมให้กับตนเองและผ่านการทดลอง(ผิดและถูก)แล้วกลับมามองแล้วเรียนรู้แก้ไขนั่นเอง

เรียนรู้และปรับปรุง

อไจล์และโพรเสสการทำงาน ก็เหมือนทุกอย่างในชีวิตที่คำตอบไม่ได้เกิดจากการอ่านแล้วถกเถียงกัน แต่ต้องเกิดจากเอาไปใช้จริงแล้วกลับมาปรับนิยามให้เข้ากับของตัวเองไปเรื่อยๆ ทีละนิดๆ จนกลายเป็นนิยามและคำตอบให้ทีมของตัวเอง

ทีมใหม่ถ้ายังไม่เข้าใจอไจล์เลยรวดเดียวทีเดียวเป็นเรื่องปกติ อย่าให้ความคิดที่ว่า “เราต้องเข้าใจทุกอย่างให้ทะลุปรุโปร่งก่อน” มาเป็นอุปสรรคทำให้ทีมเราติดไม่ไปไหน เหมือนกับการที่ “เราต้องเข้าใจ Requirement ทุกอย่าง” ทำให้เราไม่ได้เริ่มลงมือทำงานสักที

[5] http://agilemanifesto.org/

ถึงทีมยังไม่เข้าใจอไจล์เต็มที่ร้อยเปอร์เซ็นต์ แทนที่จะมาติดหล่มเถียงกัน หันมาเปิดใจทดลองกันดีกว่า เกิดมีข้อเสนออะไรขึ้นมา ให้บอกทีมเสมอว่า “ลองทำดูซัก 1 สปรินท์นะ แล้วกลับมาคุยกันว่าดีขึ้นไหมยังไง”

เพราะหัวใจหลักของอไจล์ คือ การสร้างคุณค่าอย่างสม่ำเสมอผ่านการส่งมอบของที่ใช้งานได้จริง ซึ่งทำได้กับทั้งสร้างผลงานและการพัฒนาทีมให้ดีขึ้นไปเรื่อยๆ อย่างไม่มีที่สิ้นสุดนั่นเอง

แค่ Scrum พอไหม?

หลายทีมที่ไปคุยด้วย ถ้าอยู่ในบริษัทเริ่มอไจล์แบบจริงจัง กระโดดลงมาทำเต็มตัว ไม่ได้ทำตามแฟชั่นเล่นๆ บริษัทพวกนี้จะให้ความสนับสนุนเต็มที่ กล้าจะลองปรับเปลี่ยนโครงสร้างองค์กรให้คนที่เคยอยู่กระจายกันมารวมตัวกันเป็นทีมนั่งอยู่ที่เดียวกัน หาตำรามาให้อ่าน หาคนเก่งๆมารวมตัวกันเต็มที่ กำหนดลงมาจากเบื้องบนเลยว่า “พวกเจ้าจงหัด Scrum กันให้เป็น”

คนที่หัดใหม่ๆ ทุกคนก็เลยคิดว่าอไจล์มีแต่ในด้านของที่ Scrum กำหนดไว้ เพราะถูกกำหนดภารกิจมาแค่นั้น

ถ้าอไจล์ทั้งหมดมีแค่นั้นก็คงจะดีเหมือนกันนะ ทุกทีมก็คงเป็นอไจล์ไปหมดแล้วง่ายๆ เพราะ Scrum กำหนดหลักการและกิจกรรมให้เราทำไม่กี่อย่างเอง...

ลองมาดูสิ่งต่างๆ ที่ Scrumเขียนไว้ดูสิ ทั้งหมดจะเป็นเรื่องของการจัดการ การประสานงาน และการคุยกันเข้าใจตรงกันทั้งนั้น พูดง่ายๆ คือ เป็นแค่กฎและกรอบการทำงานกว้างๆที่เราตกลงร่วมกันในตอนเริ่มต้น

ยังจำได้ไหมว่า มันเหมือนเป็น ยาชุด แก้ไขอาการที่คนส่วนมากเป็นตอนเริ่มต้นเป็นอไจล์ ไม่ได้เป็นยาสารพัดโรคแก้ทุกอย่างซักหน่อย

บางสิ่งที่ขาดหาย

ทีมไหนได้ลอง Scrum เอง ไม่ได้นั่งอ่านเฉยๆ ก็จะเจอได้เองอย่างรวดเร็วว่า Scrum แค่บอกกว้างๆว่าจะต้องทำให้มีอะไรบ้าง เหมือนลูกค้าส่ง Requirement มาให้

แต่ไม่ได้บอกว่าจะต้องทำกิจกรรมหรือใช้เทคนิคอะไรยังไง ถึงจะทำให้ทีมทำได้ตามที่ Scrum บอกไว้

ตัวอย่างเช่น การสร้าง Backlog ที่กำหนดว่าจะต้องประกอบไปด้วย Sprint Item ที่มีขนาดเล็ก (แบบวันหรือสองวันจบ) แต่ก็ไม่ได้บอกว่า จะต้องทำยังไงให้เป็นชิ้นเล็กๆ ต้องมีเทคนิคอะไรในการปรับตัวจาก ทีมที่เคยวาดดีไซน์ทั้งระบบเยอะๆ ให้จบก่อนลงมือทำงาน กลายมาเขียนตกลงดีไซน์ของการ์ดเล็กๆ ให้พอในหนึ่งชิ้นจะทำได้ยังไง รวมไปถึงการประเมินเวลาการทำงาน (estimate) ของงานที่ถูกแตกเป็นชิ้นย่อยๆ จะต้องทำยังไงให้ได้มีประสิทธิภาพ

และที่สำคัญ ถ้างานแต่ละชิ้นมีขนาดเล็กๆ และต้องส่งผลงานที่ได้ในแต่ละสปรินท์ไปให้ลูกค้าได้ใช้บ่อยๆ ทุกสัปดาห์ ทีมจะจัดการอย่างไรให้มีเวลาทดสอบระบบย่อย (unit test) จะต้องทดสอบทั้งระบบ (regresstion test) ให้ได้ทันได้ไง?

สิ่งเล็กๆ ที่เป็นเรื่องใหญ่

นิทานเรื่องนี้สอนให้รู้ว่า โลกของอไจล์ยังมีอีกเยอะมากกว่าแค่กรอบของการทำงานร่วมกันแบบที่ Scrum กำหนดไว้ มีเรื่องอื่นอีกมากมายที่จะนำมาเสริมให้ทีมทำงานได้อย่างสมบูรณ์ทั้งหลักการและกระบวนการ

มีเทคนิคต่างๆ ที่กลุ่มอไจล์ทั่วโลกทดลอง แบ่งปัน และทะเลาะกัน อยู่มากมายไม่รู้จบ

ทั้งในแง่ของ technical เช่น TDD, BDD, Continuous Integration, Continuous Delivery, Automated Tests, Planning Poker

ทั้ง ใน แง่ ของ วัฒนธรรม การ ทำงาน ร่วม กัน เช่น การ บริหาร จัดการ เอง (self-managed), การ เคารพ คน ที่ ทำงาน ใน ทีม, การ สื่อสาร ปัญหา และ สถานะ ให้ ทีม และอีกมากมาย

สิ่งเหล่านี้ในวงการอไจล์มาเข่าคุยแลกเปลี่ยนกันบ่อยๆ ก็เพื่อตอบโจทย์ว่า เราจะทำสิ่งที่ Scrum กำหนดไว้ได้อย่างไรได้บ้าง เพื่อตอบโจทย์สุดท้ายของอไจล์ว่าเราจะส่งของให้ลูกค้าใช้ได้บ่อยๆได้อย่างไร?

อาจจะดูเป็นเรื่องเล็กๆขำๆ แต่นี่เป็นสาเหตุหลักอย่างนึงที่อไจล์และ Scrum ไปไม่รอดในหลายๆ ที่ เพราะหัวหน้ามักจะเอาหนังสือ Scrum มาอ่าน แล้วเจอว่าก็ไม่มีอะไรเลยนิ มีหลักการมีกรอบการคร่าวๆ นิดหน่อย แล้วก็เรียกคนมา 6-7 คน จากฝ่ายต่างๆ แล้วก็โยนหนังสือให้เลย “อ่ะ เอาไปอ่าน แล้วก็ไปทำซะ” ซึ่งทีมที่มาลองทำใหม่ก็ไปไม่ถูกแน่ๆ เพราะไม่รู้จะเอาอะไรมาใส่ในกรอบ

โดยเฉพาะทีมที่ยังไม่ได้สร้างตนเองให้สามารถบริหารตัวเองได้ (self-managed) ก็จะทำตัวไม่ถูกแน่ๆ เพราะปกติต้องรอสั่งการว่าต้องไปศึกษาเรื่องอะไร ทำตัวตอนไหนยังไง

หลังจากนั้นทีมบริหารก็จะกลับมางงกันเองว่า ทำไมมันไปไม่รอด หรือไปไม่ถูกไม่คืบหน้าไปไหน เพราะเน้นแต่กรอบกว้างๆ แต่ไม่ได้ดูเรื่องการสร้างรากฐานทางด้านเทคนิค และวัฒนธรรมขององค์กรให้พร้อมไปคู่ๆกัน

ถ้าสามารถหา Scrum Master ที่มีประสบการณ์จะช่วยมองภาพกว้างและวางแผนการปรับเปลี่ยนองค์กรได้อย่างเป็นขั้นตอน

ถ้ายังหาไม่ได้ แนะนำให้ลองเปิดโลกให้กว้างกว่าแค่หนังสือ Scrum Guide เล่มเดียว ลองหาหนังสือที่เล่าเรื่องประสบการณ์จริง เช่น Scrum and XP from the trenches[6] มาลองอ่านหรือ ลองติดตามใน Twitter ของคนดังว่าคุยอะไรกัน จะช่วยได้มาก ถึงแม้ว่าจะอ่านไม่รู้เรื่องก็คงจะได้คำสำคัญไปหาอ่านต่อได้อีกเยอะ

——>

[6] http://www.infoq.com/minibooks/scrum-xp-from-the-trenches

เรียนรู้จากก้าวเล็กๆ

ทีมใหม่ๆ ยังไม่เข้าใจอไจล์เต็มที่ ความคิดเห็นก็จะกระจัดกระจายกัน เห็นมันไปคนละทางสองทาง ทุกคนก็จะมีมุมมองของตนเอง ทั้งที่อ่านหนังสือมา หรือ ฟังคนที่รักที่ชอบบอกมา ทีมจะไม่ค่อยมีหลักยึดหรือจุดยืนร่วมกันเท่าไหร่ว่า กำลังไปถูกทางรึยัง

พอเวลาต้องมาช่วยกันปรับวิธีการทำงานของตัวเองให้เป็นอไจล์มากขึ้น ซึ่งส่วนมากก็คือ หยิบเอาสกรัมทีละส่วนๆ มาใช้นั่นแหละ แต่อย่างที่เห็นในตอนที่แล้ว เวลาเราเอาสกรัมมาใช้จริงเนี่ย ก็ต้องลงรายละเอียดกันเองมากกว่าในหนังสือเยอะ ต้องมาคุยกัน ปรับแต่งสิ่งที่อยู่ข้างในให้เหมาะกับทีมของเรา อ๊ะ เราจะต้องทำ planning แบบไหนดีล่ะ แล้วจะใช้วิธี estimate แบบเป็น story points หรือเป็น ideal hours ดีและอื่นๆ อีกมากมาย

ทุกอย่างอยู่ในหัว

แต่บังเอิญด้วยลักษณะโดยทั่วไปของมนุษย์โปรแกรมเมอร์ มนุษย์คอมพิวเตอร์แบบเราๆ ทั้งหลาย มักจะพยายามค้นหาคำตอบหนึ่งเดียวของจักรวาลที่จะหยิบมาใช้เหมาะกับเราได้พอดี ทันที ทันใจ แล้วตอนนี้ก็จะเริ่มจับกลุ่มเถียงกันละ ว่าแนวทางไหนมันเหมาะ หรือไม่เหมาะ ดีไม่ดียังไง พยายามยกเหตุการณ์สมมติมาเยอะแยะว่าถ้าเป็นอย่างนั้นอย่างนี้ มันจะติดอะไรบ้าง เหมือนพยายามรัน simulation อะไรบางอย่าง ว่าในหัว ว่าถ้าเอาแนวทางแบบไหนมาใช้กับทีม จะเกิดผลดีผลเสียอย่างไรในระยะยาวมาก

สุดท้ายก็มาลงที่ประโยคเด็ดพวกนี้

“ทำอย่างนี้มันไม่อไจล์นะ”

“ไม่เอาๆ เพราะทำแบบนี้มันกลับไปเป็น Waterfall”

ต่อมาทุกคนก็จะเริ่มลากไปถึงว่า นิยามคำว่าอไจล์มันคืออะไร มันต่างกับแบบเดิมยังไง มันคือหลักการคร่าวๆ หรือเปล่า เอ๊ะแล้วทำ Scrum หมดแล้วมันจะเป็นอไจล์ไหมนะ จบหมดนี่สุดท้ายเสียเวลาไปเป็นชั่วโมงเหมือนกันแล้วก็ไม่ได้ข้อสรุปซักที แล้วที่อยากจะลองทำเนี่ย ก็ไม่ได้ลองซักที

Product และ Process

ใครที่หยิบหนังสือหรืออ่านบล็อกเรื่องอไจล์มาซักพักน่าจะเห็นสองวลีนี้บ่อยๆ

Fail fast and iterate

Release Early, Release Often

จริงๆ แล้ว ข้อความนี้มักจะหมายถึงตัวผลงาน (product) ที่ออกมาทีละเล็กๆ เอาไปให้ลูกต้าได้ดู ได้สัมผัสจะสามารถตอบสนองความต้องการลูกค้าได้ดีกว่าพยายามนั่งทางใน เถียงกันหาคำตอบหนึ่งเดียวของจักรวาลมากเปลี่ยนจากการพยายามวางแผนบนกระดาษ กลายเป็นการเน้นไปที่การทดลองและเรียนรู้ไปเรื่อยๆ ค่อยๆ ใส่ทีละนิดแล้วก็ค่อยๆ เอาไปให้ลูกค้าดู เรียนรู้จากความคิดเห็นของลูกค้าและคนใช้งาน

ทีมหลายทีมก็มักจะลืมไปว่า วิธีการทำงานร่วมกัน (process) ก็เป็นหนึ่งในผลผลิตของทีมเหมือนกัน ในอไจล์มันเป็นสิ่งที่เราร่วมกันค้นหามาอย่างยากลำบากว่า ต้องทำอย่างไร มากเท่าไหร่ จึงจะเหมาะพอดีกับทีม ซึ่งเราจะไปคาดหวังให้วันหนึ่ง เราไปคิดจินตนาการขั้นตอนทุกรูปแบบในหัว แล้วนึกออกเป็นสิ่งที่เหมาะพอดีกับทีมเลยทันที มันก็เป็นไปไม่ได้เหมือนกัน

ลองเปลี่ยนจากความพยายามวางแผนว่าจะหยิบอะไรเข้ามาแล้วเหมาะในทีเดียว มาเป็นการทดลองแล้วปรับปรุงร่วมกันค้นหาสิ่งที่เหมาะกับทีมไปด้วยกันครับ ทีมหลายทีมหนีออกมาจากการติดล็อกเถียงกันไม่ไปไหนด้วยกับประโยคง่ายๆ

“ลองดูซัก sprint นึงนะ แล้วมาคุยกันว่าชอบหรือเปล่า”

จงอย่าหยุดทดลองและเรียนรู้ ครับ

หัวข้อที่วางแผนไว้

รายการหัวข้อที่จะเขียนต่อไป อาจจะไม่ได้มีชื่อตามนี้เป๊ะ แต่น่าจะเขียนถึงแน่ๆ

Quality

- ไม่จำเป็นต้องเทสเมื่อจบเท่านั้น
- เวลาเทสไม่พอจะรีลีส
- ทำอะไรอย่าลืม retrospective
- อไจล์แล้ว tester อยู่ยากจริงหรือ
- Explicit policy
- Definition of done

Requirements

- As a product owner...
- Definition of not done
- Story มาจากไหน
- งานไม่พอทำ
- acceptance criteria ที่ดีเป็นยังไง
- requirement เป็นเรื่องของทุกคน
- Make story INVEST

Planning and Estimation

- ทำไมต้อง fibonacci
- ทุกอย่างเป็น high priorities
- แตกไม่ได้อีกแล้ว
- Bring projects to teams
- Sustainable pace
- Put task into time, not time into tasks
- Incremental
- อย่าเครียดมากกับ Estimation
- ซอฟต์แวร์กล่องประเมินยังไง
- Whole team planning ไม่ใช่เอาไปแบ่งๆ กัน

Sprint Review

- Retro
- Talks business. feedback. celebrate.
- Team Agreement

การทำตัวใน Roles ต่างๆ

- Scrum Master
- PO
- Customer
- Team

Agile in Enterprise

- Bringing in QA
- Customer with Extra work
- Promotions
- Team helping culture

หลักการเบื้องหลังทีมอไจล์ที่ดี