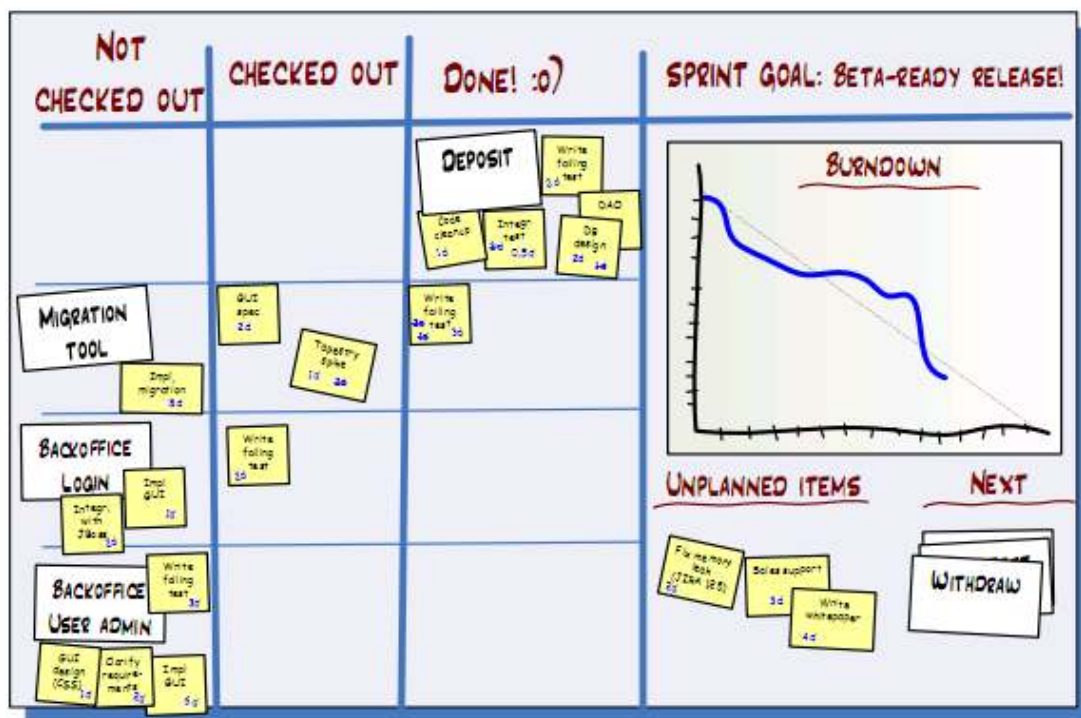


סקראם ו XP מהשוחות

איך אנחנו מתנהלים בסקראם

תרגום: חברת AgileSparks



Henrik Kniberg

henrik.kniberg@crisp.se

www.crisp.se

הקדמה למהדורה העברית

I'm very happy to see my book translated to Hebrew.

If you bump into Danko or any of the other translators
give them a big pat on the back,
because they have done this for you, and they have done it for free!

I suspect that was more work than it was for me to write the book :o)

Happy reading!
Henrik Kniberg, November 2010

דברי המתרגמים

זהו כבוד גדול להיות אלו שמתרגמים את אחד הספרים, אם לא הספר הראשון, שתיאר בצורה מאוד קורקטית, פשוטה ומדהימה כיצד צוותים מריצים סקראם בארגונים. הספר הזה נותן תיאור של צעד אחרי צעד כיצד ניתן להטמיע סקראם ואנחנו בטוחים שתמצאו אותו מועיל, גם בעברית, לפחות כמו המהדורה האנגלית.

מטבע הדברים המון מושגים בסקראם הינם באנגלית. חלקם תורגמו וחלקם הושארו בשמם המקורי (ניסיתם פעם לתרגם לעברית את המושג "סקראם מסטר"?...) ולכן הוספנו מילון מונחים בסופו של הספר. אין הדבר מעיד על כך שלפעמים השמות נראים מאולצים (Product Backlog הוא רשימת המוצר), ולדעתנו זהו המחיר הכבד שנאלצנו לשלם שכן אחרת הספר היה הופך להיות שעטנז של מילים באנגלית ובעברית.

כמו בסקארם, עבודת התרגום הייתה מאמץ מרוכז של כל צוות אג'יילספרקס שעבדו ימים כלילות כדי להוציא את הספר המתורגם הזה וזאת למרות שעבודת תרגום איננה עבודת היום יום שלהם. תודה מיוחדת אני רוצה להגיד לענבר אורן, יעל גרומן רבינוביץ, שירלי גונן, בן רייך, עופר כהן ואלעד עמית.

ודבר אחרון, למרות שזו הייתה עבודת צוות כייפית ומאתגרת שכוונה להוציא ספר מתורגם בצורה איכותית, אם נקרתה איזושהי טעות, אני נושא את נטל האשמה עלי בלבד.

קריאה מהנה!

דני (דנקו) קובץ', אפריל 2011

1. מבוא – הי סקראם עבד!

סקראם עבד! לפחות אצלנו (ואני מתכוון אצל הלקוח שלי כרגע בשטוקהולם שאת שמו אני לא מתכוון להזכיר).

מקווה שזה יעבוד בשבילך גם! אולי המסמך הזה יעזור לך לאורך הדרך. זו הפעם הראשונה שראיתי מתודולוגיית פיתוח (סליחה קן*, הכוונה למסגרת) שעובדת בדיוק על פי הספר. הכנס ושחק** . כולנו מרוצים מזה – מפתחים, אנשי בדיקות, מנהלים. זה עזר לנו לצאת ממצב קשה ואפשר לנו לשמור על מיקוד ומומנטום למרות סערות קשות בשוק וקיצוצים בכוח האדם.

אני לא אגיד שהייתי מופתע, אבל זה נראה יותר מדי טוב מלהיות אמיתי (וכלנו מכירים את הביטוי "אם משהו נראה יותר מדי טוב מלהיות אמיתי***..."). אז הייתי די סקפטי. אבל אחרי שעבדתי שנה בסקראם אני התרשמתי לטובה במידה מספקת (וכך גם רוב האנשים בצוות שלי) עד כדי כך שכנראה אני אמשיך לעבוד בסקראם כברירת מחדל בפרויקטים חדשים אלא אם כן תהיה סיבה חזקה מספיק לא לעשות זאת.

*קן שוואבר, אחד משלושת ממציאי השיטה
** PlugNPlay ביטוי מעולם פיתוח התוכנה המתאר התקנת אפליקציה ללא צורך לקונפיגורציה אחרי ההתקנה שהפך לתיאור של משהו מהיר ללא צורך בהתערבות תוך כדי או אחרי התקנה
***"אם משהו נראה יותר מדי טוב מלהיות אמיתי, כנראה שזה נכון"

2. הקדמה

אתם עומדים להשתמש בסקראם בארגון שלכם. או אולי אתם עובדים בסקראם כמה חודשים, יש לכם את הבסיס, קראתם את הספרים, אולי אפילו הוסמכתם כסקראם מסטרים. איחול!

אבל עדיין אתם מרגישים מבולבלים.

כמו שקן שוואבר אומר, סקראם זו לא מתודולוגיה, אלא מסגרת. זה אומר שסקראם לא באמת עומד לספר לכם מה אתם בדיוק צריכים לעשות. באסה.

החדשות הטובות הם שאני עומד לספר לכם בדיוק איך אני עושה סקראם. זה לא אומר שאתם צריכים לעשות בדיוק אותו דבר. למעשה, אני אולי אעשה זאת בדרך אחרת עם אתקל במצב שונה.

הגישה העכשווית שלי לסקראם היא תוצאה של שנה אחת של ניסיון התנהלות בסקראם בצוות פיתוח המונה בערך 40 אנשים. החברה הייתה במצב קשה עם המון שעות נוספות, בעיות איכות קשות, טיפול מתמיד בשריפות, אי עמידה בתאריכי יעד וכו'.

החברה החליטה לעבור לסקראם אבל לא באמת סיימה את ההטמעה. זו היתה המשימה שלי. לרוב האנשים בצוות הפיתוח באותו הזמן, "סקראם" היתה רק מילה נחמדה שהם שמעו עליה מפעם לפעם במסדרונות, בלי השפעה על חיי העבודה היום יומיים שלהם.

אחרי שנה, הטמענו סקראם בכל החברה, ניסינו מספר שונה של אנשי צוות (3-12 אנשים), אורכי ספרינט שונים (6-2 שבועות), הגדרות שונות למונח "סיום", פורמטים שונים לרשימת מוצר ורשימת מוצר ספרינטית** (Excel, Jira, קלפי אינדקס***), אסטרטגיות שונות לבדיקות, דרכים שונות לבצע Demo****, דרכים שונות לסנכרון צוותי סקראם מרובים וכו'. בנוסף התנסינו בפרקטיקות מעולם XP***** - דרכים שונות לבצע Continuous build⁽⁶⁾ תכנות מונחה בדיקות⁽⁷⁾ וכו' ואיך לשלב זאת עם סקראם.

זהו תהליך מתמשך ולכן זה אינו סופו של הסיפור. אני משוכנע שהחברה הזו תמשיך ללמוד (אם הם ימשיכו לבצע ישיבת רטרוספקטיב) ויעלו על תובנות איך להטמיע את סקראם בהקשר הספציפי שלהם.

** Sprint Backlog – רשימה מתועדפת ומתומחרת של נושאים עליהם התחיייה הצוות לבצע במהלך הספרינט
*** ידועים גם בשם Planning Poker Cards ומשמשים את הצוות בהערכת משימות
**** Demo – גרסת תוכנה המיועדת להדגמה
***** eXtreme Programming – שיטה אג'ילית נוספת בעלת דמיון לסקראם שמתמקדת יותר באלמנטים הטכניים של התנהלות הצוות במהלך הספרינט
(6) – מנגנון שבונה את הגרסא בכל רגע נתון או באמצעות טריגר אוטומטי
(7) Test Driven Development – מתודולוגיית פיתוח בא הבדיקות נעשות במקביל ואפילו לפעמים לפני פיתוח הקוד

2.1. הסרת אחריות

מסמך זה לא מייצג את הדרך "הנכונה" לעשות סקראם! הוא רק מתאר דרך אחת לבצע סקראם, תוצאה של ליטוש השיטה לאורך שנה. אתה יכול להחליט שעשינו זאת בצורה לא נכונה.

כל מה שכתוב במסמך הזה מייצג את דעותיי האישיות הסובייקטיביות ובשום פנים ואופן איננו מייצג דעה רשמית של חברת CRISP או של אחד מהלקוחות הקיימים. בדיוק מסיבה זו אני נמנע בכוונה מלהציג פרוייקט ספציפי או שם של אדם.

2.2 למה כתבתי ספר זה

כשלמדתי על סקראם, קראתי את כל הספרים הרלוונטיים על סקראם ועל אג'יל, חרשתי אתרים ופורומים על סקראם, הוסמכתי על ידי קן שוואבר, שאלתי אותו שאלות מפולפלות, וביליתי המון זמן בדיונים עם העמיתים שלי. למרות זאת, אחד מהמקורות בעלי הערך הגבוה ביותר היו סיפור קרב אמיתיים. סיפורי הקרב הפכו את העקרונות והפרקטיקות ל...נו... "איך אתה באמת מבצע זאת". הם גם עזרו לי לזהות (ולפעמים להימנע מ...) טעויות סקראם של מתחילים. לכן, זהו הסיכוי שלי לתת משהו חזרה, הנה סיפור הקרב שלי

אני מקווה שסיפור הקרב שלי יגרום למשוב בעל ערך מאלו שנמצאים בסיטואציה דומה. בבקשה השכילו אותי!

2.3 אבל מה זה סקראם?

אה, סליחה, אתה לגמרי לא מכיר מה זה סקראם או XP? במקרה הזה מומלץ לך להסתכל באתרים הבאים

- [http://agilemanifesto.org /](http://agilemanifesto.org/)
- <http://www.mountangoatsoftware.com/scrum>
- <http://www.xprogramming.com/xpmag/whatisxp.htm>

אם אתה יותר מדי חסר סבלנות לעשות את זה, הרגש חופשי להמשיך לקרוא. רוב המכאניקה של סקראם מוסברת תוך כדי המסמך אז יש סיכוי שעדיין תמצא את זה מעניין.

3. איך אנחנו יוצרים רשימות מוצר

רשימת המוצר היא לב לבו של סקראם. זה המקום שבו הכול מתחיל. רשימת המוצר היא בבסיסה רשימה מתועדפת של דרישות, או סיפורים או תכונות מוצר או מה שזה לא יהיה. דברים שהלקוח רוצה, מתוארים בשפת הלקוח.

אנחנו קוראים להם סיפורים או לפעמים רק פריטי רשימה.

הסיפורים שלנו כוללים את השדות הבאים:

זיהוי: זיהוי יחידני, רק מספר שוטף אוטומטי על מנת לעקוב בצורה טובה אחרי הסיפורים גם כאשר הם משנים את השם שלהם

שם: תאור ממצה של הסיפור. לדוגמא "תסתכל על היסטוריית הטרנזקציה

שלך". מספיק ברור שהמקודד ובעל המוצר יבינו בערך על מה אנחנו מדברים ומספיק ברור להבדיל אותו מסיפורים אחרים. בדרך כלל 10-2 מילים.

חשיבות: דירוג חשיבות הסיפור על פי בעל המוצר. למשל 10 או 150. גבוה = יותר חשוב. אני נוטה להימנע מהביטוי "עדיפות" שכן עדיפות "1" מתארת בדרך כלל כחשובה "ביותר" מה שנעשה מוזר אם יותר מאוחר אתה מחליט שמשהו אחר יותר חשוב. איזה עדיפות זה יקבל? עדיפות 0? עדיפות 1?

הערכה ראשונית: הערכה ראשונית של הצוות כמה עבודה צריכים להשקיע בהשוואה לסיפורים אחרים. היחידות הן נקודות סיפור ובדרך כלל מתורגמות ל"ימים של אדם אידיאלי".

- שאל את הצוות "אם תיקח את מספר האנשים האופטימלי לסיפור הזה (לא מעט מדי ולא יותר מדי, בדרך כלל 2), תנעלו את עצמכם בחדר עם המון אוכל ותעבדו בלי הפרעות, אחרי כמה ימים תצאו עם סיפור גמור, שניתן להצגה, סיפור שהוא בדוק וניתן לשחרור? אם התשובה היא "עם 3 אנשים נעולים בתוך חדר זה יארך בערך 4 ימים" אז ההערכה הראשונית היא 12 נקודות סיפור"

- הדבר החשוב הוא לא להגיע להערכה אבסולוטית נכונה (כלומר ש2 נקודות סיפור צריכות לקחת יומיים), הדבר החשוב הוא לתת הערכה יחסית נכונה (כלומר שסיפור בעל 2 נקודות צריך לקחת כחצי עבודה מסיפור בעל 4 נקודות)

איך להציג: תיאור ברמה מאוד גסה איך הסיפור עומד להיות מוצג בישיבת הדמו של סוף הספרינט. זוהי בעצם בדיקה פשוטה של הדרישה. "תעשה את זה, אחר כך את זה, ואז תקבל תוצאה כזו".

אם אתה עובד בTDD (תכנות מונחה בדיקות) התיאור הזה יכול לשמש אותך כפסאודו קוד לבדיקות הנכונות של הקוד עצמו.

הערות כל מידע אחר, הבהרות, הפניות למקורות אחרים של מידע וכו'. בדרך כלל מאוד בקצרה.

דוגמא לרשימת מוצר:

ID	Name	Imp	Est	How to demo	Notes
1	Deposit	30	5	Log in, open deposit page , deposit €10, go to my balance page and check that it has increased by €10	Need a UML sequence diagram. No need to worry about encryption for now .
2	See your own transaction history	8	10	Log in, click on "transactions ." Do a deposit. Go back to transactions, check that the new deposit shows up .	Use paging to avoid large DB queries . Design similar to view users page

אנחנו ניסינו המון שדות אחרים אבל בסופו של יום, ששת השדות הללו היו היחידים שבאמת השתמשנו בהם ספרינט אחרי ספרינט. בדרך כלל אנחנו משתמשים במסמך Excel משותף (כלומר מספר משתמשים יכולים לעדכן אותו בעת ובעונה אחת). רשמית, בעל המוצר הינו בעל המסמך אבל אנחנו לא רוצים לנעול משתמשים אחרים. פעמים רבות מפתח רוצה לפתוח את המסמך להבהיר משהו או לשנות הערכה.

מאותה סיבה אנחנו לא שמים את המסמך הזה בכלי בקרת התצורה שלנו, אלא בכונן השיתופי. זה הפך להיות הפתרון הפשוט ביותר המאפשר מספר רב של מעדכנים בו זמנית בלי ליצור נעילות עקב קונפליקטים.

3.1 שדות נוספים לסיפורי משתמש

לפעמים אנחנו משתמשים בשדות נוספים ברשימת המוצר, בדרך כלל לנוחות בעל המוצר לעזור לו לסדר את העדיפויות.

מעקב: תאור גס של קטגוריית הסיפור, לדוגמא "עבודת משרד" או "אופטימיזציה". בדרך זו בעל המוצר יכול בקלות לסנן את כל ה "אופטימיזציות" ולתת להם עדיפות נמוכה, וכו'.

קומפוננטות: בדרך כלל הן מיוצגות כ"תיבות מסומנות" במסמך Excel, לדוגמא "בסיס המידע, סרבר, קליינט". כך הצוות או בעל המוצר יכולים לזהות אילו קומפוננטות טכניות יהיו מעורבות בביצוע הסיפור. זה מאוד מוכיח את עצמו כשקיימים כמה צוותי סקראם.

מבקש הבקשה: בעל המוצר יכול לרצות לדעת איזה לקוח או בעל עניין ביקש את הבקשה המקורית כדי לתת לו משוב על ההתקדמות.

זיהוי מעקב תקלה: אם יש לכם מערכת נפרדת למעקב אחרי תקלות, כמו שיש לנו עם JIRA, זה מאוד אפקטיבי לעקוב אחרי התכתובת בין הסיפור לבין תקלה אחת או יותר שדווחו.

3.2 איך לשמור על רשימת המוצר ברמה העסקית

אם לבעל המוצר יש רקע טכני, הוא יכול להוסיף סיפורים כמו "הוסף אינדקסים לטבלת הארועים". למה הוא רוצה את זה? כותרת הבקשה האמיתית היא בטח משהו כמו "לבצע את החיפוש בצורה יותר מהירה"

יכול להיות שבסוף יתברר שהאינדקסים לא היו צוואר הבקבוק שבגללו טבלאות איטיות נטענות באיטיות. זה יכול להיות בכלל משהו אחר. הצוות הוא זה שבדד כלל יודע מה הדרך הטובה ביותר לפתור משהו, לכן בעל המוצר צרי להתמקד ביעדים העסקיים.

כשאני רואה סיפורים בעלי אוריינטציה טכנית כמו זה, אני בדרך כלל שואל את בעל המוצר סדרת שאלות של "אבל למה" עד שאני מוצא את המטרה האמיתית ("לבצע את החיפוש בצורה יותר מהירה"). במקרה כגון זה, הבקשה הטכנית הראשונית ("הוסף אינדקסים לטבלת האירועים") תהפוך להערה בסיפור המשתמש.

4. כיצד להתכונן לישיבת תכנון הספרינט

אוקי, ישיבת תכנון הספרינט מגיעה במהירות. שיעור חשוב שלמדנו שוב ושוב: היו בטוחים שרשימת המוצר מוכנה למשעי לפני ישיבת תכנון הספרינט.

ומה זה אומר? שכל הסיפורים חייבים להיות מוגדרים בצורה מושלמת? שכל ההערכות נכונות? שכל התעדופים שניתנו מדויקים?

לא, לא ולא! כל מה שזה אומר זה:

- שרשימת המוצר אכן קיימת (תארו לעצמכם...!)
- שיש רשימת מוצר אחת ובעל מוצר אחד (לכל מוצר כמובן)
- לכל הפריטים החשובים יש תיעדוף רלוונטי ושונה
 - האמת, שזה בסדר גמור אם לפריטים עם תיעדוף נמוך יהיה ערך זהה, כוון שכנראה הם במילא לא יהיו חלק ממוקד הדיון בתכנון הספרינט
 - כל סיפור שבעל המוצר חושב שיהיה לו חלק בספרינט הבא חייב שיהיה לו ערך שונה בסולם העדיפויות
 - חשיבות התיעדוף היא רק לשם סידור הפריטים לפי החשיבות שלהם. כלומר אם פריט אחד יש לו ערך 20 ופריט שני ערך 100, זה בסך הכול אומר שהפריט השני יותר חשוב מהפריט הראשון. זה לא אומר שהפריט השני חשוב פי 5 מהפריט הראשון. אם הפריט השני היה בעל ערך 21 המשמעות היתה זהה!
 - זה שימושי להשאיר פערים בין הערכים למקרה שפריט שלישי מגיע שהוא יותר חשוב מהראשון ופחות מהשני. נכון, תמיד אפשר לכתוב ערך 20.5 אבל זה מכוער, אז במקום זה פשוט השאירו פערים.
- בעל המוצר חייב להבין כל סיפור (בדרך כלל הוא זה שכותב אותם, אבל בחלק מהמקרים אנשים אחרים מוסיפים בקשות שבעל המוצר יכול לתעדף). הוא לא צריך לדעת בדיוק מה צריך להיות ממומש, אבל הוא צריך להבין למה הסיפור הזה קיים.
- **הערה:** אנשים אחרים יכולים להוסיף סיפורים לרשימת המוצר. אבל הם לא יכולים לתעדף אותם, זהו התפקיד הבלעדי של בעל המוצר. הם גם לא יכולים לתת הערכה, זהו תפקידו הבלעדי של הצוות.

גישות אחרות שניסינו או הערכנו:

- שימוש ב JIRA (כלי מעקב הבאגים שלנו) כרשימת המוצר. רוב בעלי המוצר שלנו מצאו שהם צריכים להקליק יותר מדי. Excel הוא כלי נחמד ונוח. אתה יכול בקלות לקדד נושאים בצבעים, לארגן מחדש פריטים, להוסיף עמודות בצורה פרטנית, להוסיף הערות, ליבא מידע, ליצא אותו וכו'.
- שימוש בכלים אגיליים התומכים במהלך החיים האגילי של המוצר כמו XPLANNER, SCRUMWORKS, VERSIONONE וכו'. עוד לא בחנו לעומק את הכלים הללו אבל בוודאי נעשה זאת בעתיד

5 איך אנחנו מבצעים את ישיבת תכנון הספרינט

ישיבת תכנון הספרינט היא ישיבה קריטית, כנראה אחד האירועים החשובים ביותר בסקראם (לדעתי כמובן). ישיבה שמתבצעת בצורה גרועה יכולה להרוס את כל הספרינט.

משמעות הישיבה היא לתת לצוות מספיק מידע כדי שהוא יוכל לעבוד בשקט וללא הפרעות כמה שבועות, ולתת לבעל המוצר מספיק בטחון כדי לתת להם לעשות זאת.

אוקי, זה היה קצת באוויר, הנה מה שישיבת תכנון הספרינט צריכה לייצר:

- יעד הספרינט

- רשימת אנשי הצוות (ורמת המחויבות שלהם אם היא לא 100%)

- רשימת ספרינט (רשימה של סיפורים שנכנסים לתו הספרינט)

- תאריך מוגדר של הצגת הספרינט

- זמן מוגדר ומקום מוגדר של הישיבה היומית.

5.1 מדוע בעל המוצר צריך להיות נוכח בישיבת תכנון

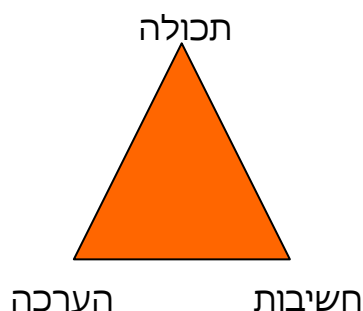
ספרינט

לפעמים בעלי המוצר אינם נלהבים להשקיע שעות עם הצוות בישיבת תכנון הספרינט.

"חברים, כבר ייצרתי לכם רשימה של הדרישות שלי. אין לי זמן להיות בישיבת תכנון הספרינט שלכם"

זו בעיה די חמורה.

הסיבה שהצוות ובעל המוצר צריכים להיות בישיבת תכנון הספרינט היא שכל סיפור מכיל בתוכו שלושה אלמנטים שמאוד תלויים אחד בשני



תכולה וחשיבות מוגדרים על ידי בעל המוצר. ההערכה ניתנת על ידי הצוות. תוך כדי ישיבת תכנון הספרינט, שלושת האלמנטים הללו מתגמשים דרך הדיאלוג שנעשה פנים מול פנים בין הצוות לבעל המוצר.

בדרך כלל הישיבה מתחילה כשבעל המוצר מתחיל בתיאור יעד הספרינט ויותר חשוב, בתיאור הסיפורים. אחר כך, הצוות נותן את הערכותיו לכל אחד מהסיפורים, כשמתחילים בסיפור החשוב ביותר. תוך כדי שהם עושים את זה, תעלה שאלת תכולה חשובה דוגמאת: "האם סיפור 'מחיקת המשתמש' כולל השהיית כל טרנזקציה של אותו משתמש וביטולה?". בחלק מהמקרים, התשובות יפתיעו את הצוות ויגרמו להם לשנות את הערכותיהם.

בחלק מהמקרים הערכות הזמנים לא יהיו מה שבעל המוצר יצפה. זה יגרום לו לשנות את חשיבות הסיפור, או את תכולת הסיפור, מה שיגרום לצוות לתת הערכה מחודשת וכו' וכו'.

כזה סוג של שיתוף פעולה ישיר הוא בבסיסו של מסגרת הסקראם ולמעשה בכל פיתוח תוכנה אג'ילי.

מה אם בעל המוצר עדיין מתעקש שאין לו זמן להצטרף לישיבת תכנון הספרינט? בדרך כלל אני משתמש באסטרטגיות הבאות, לפי הסדר הזה:

- מנסה לעזור לבעל המוצר להבין למה השתתפותו האישית חשובה כל כך וקריטית ומקווה שהוא ישנה את דעתו

- מנסה שמישהו מאנשי הצוות יתנדב להיות בא כוחו של בעל המוצר בישיבה. ואז אני אומר לבעל המוצר "מכוון שאתה לא יכול להצטרף לישיבה שלנו, אנחנו ניתן לענבר לייצג אותך כבא כוחך. הוא יהיה בעל סמכות מלאה לשנות עדיפויות ותכולה של בסיפורים בשמך במשך הישיבה. אני מציע שתסתנכרן איתו כמה שתוכל לפני הישיבה. אם אתה לא מוכן שענבר יהיה בא כוחך, אנא הצע מישהו חילופי ובלבד שהוא יוכל להצטרף לכל אורך הישיבה".

- מנסה לשכנע את ההנהלה להסמיך מישהו אחר כבעל המוצר.

- דוחה את תחילת הספרינט עד שבעל המוצר ימצא את הזמן להצטרף לפגישה. בינתיים, דוחה כל התחייבות להוציא תוצרים כלשהם. בינתיים נותן לצוות להשקיע זמן בכל מה שנראה להם חשוב באותו יום.

5.2 למה איכות איננה ניתנת למשא ומתן

במשולש למעלה, נמנעתי במכוון להתייחס לאלמנט רביעי של איכות.

אני אנסה להבדיל בין איכות פנימית ואיכות חיצונית

- איכות חיצונית היא מה שאנחנו מבינים מהמשתמש של המערכת. ממשק איטי ולא אינטואיטיבי הוא דוגמא לאיכות חיצונית ירודה.

- איכות פנימית מתייחסת לנושאים שבדרך כלל המשתמשים לא רואים, אבל יש לה אפקט משמעותי על תחזוקת המערכת. דברים כמו עיצוב מערכת קונסיסטנטי, כיסוי בדיקות, קריאות הקוד, שכתובי קוד, וכו'.

סיכומי של דבר, מערכת עם איכות פנימית גבוהה עדיין יכולה להיות בעלת איכות חיצונית ירודה אבל מערכת עם איכות פנימית נמוכה, נדיר שתהיה עם איכות חיצונית גבוהה. זה קשה לבנות משהו נחמד על בסיס יסודות רעועים. אני מתייחס לאיכות חיצונית כאל חלק מהתכולה.

בחלק מהמקרים, זה יהיה בעל הגיון עסקי מושלם להוציא גרסא של המערכת בעלת ממשק משתמש איטי ומסורבל ואז להוציא גרסא נקיה מאוחד יותר. אני משאיר את שקלול הדברים לבעל המוצר מכיוון שהוא האחראי לתכולת המוצר.

איכות פנימית לעומת זאת אינה נושא לדיון. הצוות אחראי לתחזוק איכות המערכת תחת כל הנסיבות ולכן נושא זה לעולם אינו נתון למשא ומתן. (נו, טוב, כמעט לעולם)

כיצד מפדילים בין איכות פנימית לאיכות חיצונית?

בוא נניח שבעל המוצר אומר "אוקי חברים, אני מכבד את הערכת הזמנים שלכם של 6 נקודות סיפור אבל אני בטוח שאתם יכולים לעשות מין תיקון-מהיר לזה במחצית מהזמן אם רק תקדישו את המחשבה הנכונה לזה"

אהה! הוא מנסה להשתמש באיכות פנימית כמשתנה. איך אני יודע? כי הוא רוצה שנוריד את ההערכה של הסיפור מבלי "לשלם את המחיר" של הורדת התכולה.

המילה "תיקון-מהיר" צריך ליצור צלצול צורם בראש שלכם...

ולמה אנחנו לא מרשים את זה?

הניסיון שלי מראה שהקרבת איכות פנימית היא כמעט תמיד רעיון איום. הזמן שאנחנו חוסכים הוא רחוק לאין שעור מהמחיר המידי והעתידי. כשלקוד מותר להתחיל להתדרדר, זה מאוד קשה להחזיר את האיכות חזרה.

במקום זה אני מנסה להוביל את הדיון לכוון תכולה. "מכוון שזה חשוב לך לקבל את התוצר מוקדם, אולי נוכל להוריד מהתכולה כך שנוכל לבצע את זה בצורה יותר מהירה? אולי נוכל לפשט את הטיפול בטעויות ולעשות 'טיפול מתקדם לטעויות' כסיפור נפרד שנשמור לעתיד לבוא? או אולי נוכל להוריד את העדיפות של סיפורים אחרים כדי שנוכל להתמקד בזה?"

כשבעל המוצר מבין שאיכות פנימית איננה נתונה למשא ומתן, הוא נעשה בדרך כלל די טוב במשחק עם שאר המשתנים כאלטרנטיבה להתפשרות על איכות פנימית.

5.3 ישיבות תכנון ספרינט שנמשכות לנצח....

הדבר הקשה ביותר בהקשר של ישיבת תכנון ספרינט הוא:

1. אנשים לא חושבים שהישיבה תיקח המון זמן

2. אבל היא כן לוקחת המון זמן !

כל דבר בסקראם חסום בזמן. אני אוהב את זה. חוק פשוט ועקבי. אנחנו ומנסים להיצמד לזה.

לכן מה אנחנו עושים כשהישיבה עומדת להגמר ואין סימן נראה לעין של יעדי ספרינט או רשימת ספרינט? האם אנחנו פשוט חותכים את הישיבה? או שאנחנו מאריכים אותה בשעה? או אולי מסיימים את הישיבה וממשיכים למחרת?

זה קורה פעם אחר פעם, במיוחד לצוותים חדשים. אז מה צריך לעשות?

אני לא יודע. אבל מה אנחנו עושים? אהה, מממ, ובכן בדרך כלל אני עוצר את הישיבה בברוטאליות. מסיים אותה. נותן לספרינט לסבול. ביתר פרוט, אני אומר לצוות ולבעל המוצר: " הישיבה עומדת להגמר עוד 10 דקות. אין לנו ממש תוכנית לספרינט. האם נוכל לצאת לספרינט עם מה שיש לנו עד כה או שנתזמן עוד 4 שעות ישיבת המשך ב 8 בבוקר?" אתם יכולים לנחש מה תהיה התשובה: ☺

ניסיתי לתת לצוותים להמשיך. בדרך כלל זה לא משיג שום דבר בגלל שאנשים מתעייפים. אם הם לא השיגו שום דבר במשך 2-8 שעות (או כמה שאתם מקצים להם בישיבה), כנראה הם לא ישיגו את זה בעוד שעה.

האופציה השניה היא די טובה למעשה, לקבוע ישיבה ליום הבא. צפו לזה שאנשים בדרך כלל חסרי סבלנות להתחיל את הספרינט ולכן לא ישקיעו עוד שעות בתכנון

.

אז אני חותך את זה. וכן, הספרינט סובל. היתרון הוא שהצוות למד שיעור חשוב. ישיבת תכנון הספרינט הבאה תהיה הרבה יותר יעילה. בנוסף אנשים פחות יתנגדו כאשר יציעו להם אורך ישיבה שבעבר הם חשבו שהיא ארוכה מאוד. למדו לשמור על חסם הזמן, למדו לתת חסמי זמן ריאליים. שיתאימו הן לאורך הישיבה והן לאורך הספרינט.

5.4 לוח הזמנים של ישיבת תכנון ספרינט

הצבת לוחות זמנים לישיבת תכנון הספרינט תוריד את רמת הסיכון שלא נעמוד באורך הישיבה.

הנה דוגמא ללוח זמנים טיפוסי שלנו:

ישיבת תכנון ספרינט: 13:00-17:00 (10 דקות הפסקה כל שעה)

- 13:00-13:30 בעל המוצר עובר על יעדי הספרינט ומסכם את רשימת המוצר. זמן ההדגמה, תאריך וזמן מוגדרים.

- 13:30-15:00 הצוות מעריך זמנים ומפצל את הסיפורים לתתי סיפורים אם צריך. בעל המוצר מעדכן על עדכונים חשובים. מבהירים פרטי מידע. "איך להציג" זה נושא שחשוב מאוד לכל פריט שכזה.

- 15:00-16:00 הצוות בוחר סיפורים שיכנסו לספרינט. עושים חישוב לגבי וקטור מהירות הספרינט כבוחן מציאות.

- 16:00-17:00 בחירת זמן ומקום לישיבת הסקראם היומית (אם שונה מהספרינט הקודם). המשך חלוקת סיפורים למשימות.

לוח הזמנים של הישיבה בשום אופן איננו קשיח. הסקראם מסטר יכול להעריך או לקצר את לוח הזמנים אם צריך בהתאם להתקדמות הישיבה.

5.5 הגדרת אורך הספרינט

אחד מהתוצרים של ישיבת תכנון הספרינט היא הגדרת זמן ישיבת ההדגמה. זה אומר שאתם צריכים להחליט על אורך הספרינט.

מהו האורך המומלץ לספרינט?

ובכן, ספרינטיים קצרים הם דבר מצויין כי הם מאפשרים לחברה להיות יותר "אג'ילית", כלומר לשנות את הכוון הרצוי בצורה תכופה יותר. ספרינטיים קצרים = משובים תכופים יותר = יותר תוצרים ליחידת זמן = יותר משובים מהלקוח = פחות זמן ריצה בכוונים לא נכונים = למידה ושיפור גבוהים יותר, וכו'.

אבל, ספרינטיים ארוכים גם הם דבר טוב. הצוות מקבל זמן לבנות את המומנטום שלו, הם מקבלים יותר זמן בהתאוששות מתקלות ועדיין לעמוד ביעדי הספרינט, אתה מקבל פחות זמן תקורה מבחינת ישיבות הסקראם וכו'.

ככלל, בעלי מוצר אוהבים ספרינטים קצרים וצוותים אוהבים ספרינטים ארוכים, לכן אורך הספרינט הוא פשרה. אנחנו התנסינו בזה המון ולבסוף יצאנו עם הנוסח הבא:

אורך: 3 שבועות. לרוב הצוותים (אבל לא לכולם) אורך הספרינט הוא 3 שבועות. מספיק קצר להחשב אגילי ומספיק ארוך לצוות להשיג קצב ומסוגלות להתאושש מבעיות שצצות בספרינט. דבר אחד הפנמנו: התנסו בהגדרת אורך הספרינט. אל תבזבזו זמן בלעשות אנליזה ארוכה. פשוט תבחרו אורך ותנו לזה הזדמנות ספרינט אחד או שניים, אז שנו את האורך בהתאם.

עם זאת, ברגע שהחלטתם מה אורך הספרינט שאתם אוהבים ומתאים לכם, השארו עם ההחלטה לתקופה יותר ארוכה. אחרי כמה חודשים של ניסיון מצאנו ש-3 שבועות מתאימים לנו, אז אנחנו מתנהלים ב-3 שבועות לספרינט. לפעמים זה יראה כזמן ארוך, לפעמים כזמן קצר. אבל על ידי שמירת הקצב, זה יהפך לפעימת הלב הארגונית וכולם יאמצו אותו. הויכוח על תאריך הוצאת הגרסה יעלם כי כולם ידעו שכל שלושה שבועות יש הוצאת גרסה. נקודה.

5.6 הגדרת יעדי הספרינט

זה קורה כמעט תמיד. בשלב מסוים בישיבת תכנון הספרינט אני שואל "אז מה היעדים לספרינט הזה?" וכולם מסתכלים עלי ועל בעל המוצר בעיני עגל...

מסיבה מסוימת קשה להגדיר יעדים לספרינט. אבל למרות זאת, גיליתי שזה משתלם לעשות כן. עדיף יעדים או אפילו יעד אחד גרוע מאשר אף יעד. היעד יכול להיות "לעשות יותר כסף" או "לסיים את שלושת הסיפורים הראשונים" או "להרשים את המנכ"ל" או "הפיכת המערכת למספיק טובה כך אפשר יהיה לשחרר אותה ברמת ראשונית", או הוספת פונקציונליות בסיסית" או מה שזה לא יהיה. הדבר החשוב הוא שהיעד יהיה מנוסח ברמת העסקית לא ברמה הטכנית. זה אומר שאנשים מחוץ לצוות יוכלו להבין את היעד.

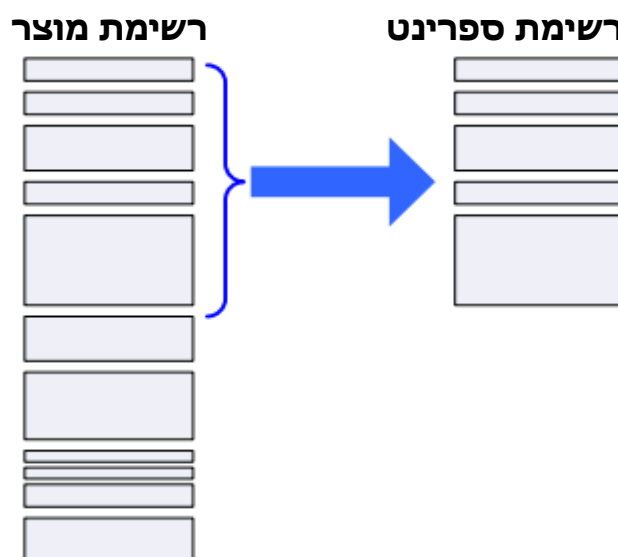
יעדי הספרינט צריכים לענות על השאלות הבסיסיות הבאות: "למה אנחנו עושים את הספרינט הזה? למה שלא נצא לחופשה במקום?". למעשה, דרך טובה מאוד להגיע להגדרת היעדים היא פשוט לשאול בדיוק את השאלות הללו.

לפעמים היעדים לא מושגים. "להרשים את המנכ"ל" יכול להיות אולי יעד מרשים אבל לא אם הוא כבר מורשם מהמערכת כמו שהיא. במקרה כזה כולם יכולים ללכת הביתה ויעדי הספרינט הושגו.

יעדי הספרינט יכולים להראות טיפשיים ומיותרים במהלך ישיבת תכנון הספרינט, אבל לעיתים קרובות הם שימושיים מאוד באמצע ספרינט, כאשר אנשים מתחילים להיות מבולבלים בקשר למה הם אמורים לעשות. אם יש לכם כמה צוותי סקראם (כמו שלנו יש) שעובדים על כמה פרויקטים, זה מאוד שימושי לקבל רשימת יעדים בשביל כל הצוותים בדף WIKI. (או מה שלא יהיה) ולשים אותם במקום אחד, כך שכולם בחברה (לא רק ההנהלה הבכירה) ידעו מה החברה עושה – ולמה!

5.7 ההחלטה אילו סיפורים יכנסו לספרינט

אחת מההחלטות העיקריות בישיבת תכנון הספרינט היא אילו סיפורים יכנסו לספרינט או במילים אחרות, אילו סיפורים מתוך רשימת המוצר יועתקו לרשימת הספרינט.



תסתכלו על התמונה למעלה. כל ריבוע מייצג סיפור. הריבועים מתועדפים לפי החשיבות שלהם. הסיפור החשוב ביותר, נמצא למעלה יותר ברשימה. גודל הריבוע מייצג את גודל הסיפור (כלומר הערכת זמן בנקודות סיפור). הגובה של הסוגריים הכחולות מייצג את הערכת וקטור המהירות של הצוות, כלומר כמה נקודות סיפור הצוות מאמין שהוא מסוגל לסיים בספרינט הבא.

רשימת הספרינט מימין היא העתקה של הסיפורים מרשימת המוצר. הוא מייצג את רשימת הסיפורים שהצוות מתחייב לספרינט הזה.

הצוות מחליט כמה סיפורים להכניס לספרינט. לא בעל המוצר או מישהו אחר. זה מעלה שתי שאלות:

1. איך הצוות מחליט אילו סיפורים להכניס לספרינט?

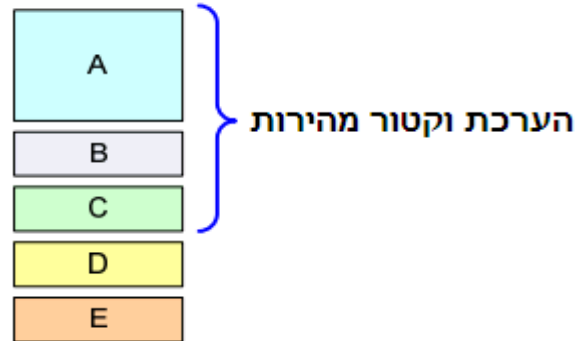
2. איך בעל המוצר יכול להשפיע איזה סיפור יכנס לספרינט?

אני אתחיל מהשאלה השנייה.

5.8 כיצד בעל המוצר יכול להשפיע איזה סיפור יכנס לספרינט?

נניח שיש לנו את הסיטואציה הבאה בישיבת תכנון הספרינט:

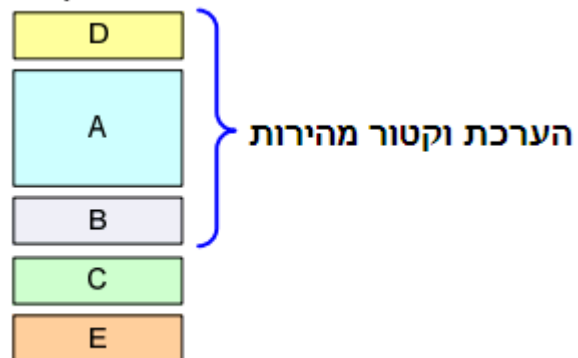
רשימת מוצר



כיוון שבעל המוצר די מאוכזב מזה שסיפור D לא יהיה כלול בספרינט, נשאלת השאלה מהם האפשרויות הפתוחות בפניו במהלך ישיבת תכנון הספרינט?

אפשרות אחת היא לתעדף מחדש. אם הוא ייתן לסיפור D את התעדוף הגבוה ביותר, הצוות יהיה מחויב להוסיף אותו לספרינט (ולהוציא את סיפור C במקרה הזה).

אפשרות ראשונה



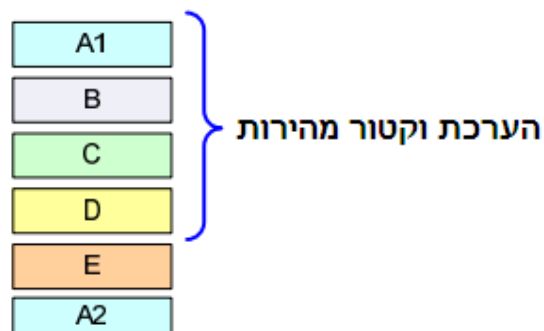
האפשרות השנייה היא לשנות את התכולה - הורדת התכולה של סיפור A עד לרגע שבו הצוות מרגיש שהוא יכול להכניס את סיפור D לתוך הספרינט

אפשרות שניה



האפשרות השלישית היא לפצל סיפור. בעל המוצר יכול להחליט שיש כמה היבטים בסיפור A שלא כאלו חשובים, אז הוא מפצל את סיפור A לשני תתי סיפורים A1 ו A2 עם תיעדוף שונה

אפשרות שלישית



כמו שאתם רואים, למרות שבעל המוצר בדרך כלל לא שולט על הערכת וקטור המהירות, ישנן דרכים אחרות שהוא יכול להשפיע מה נכנס בסופו של דבר לתוך הספרינט.

5.9 כיצד קובע הצוות אילו סיפורים יכנסו לספרינט?

משתמשים בשתי טכניקות:

1. תחושת בטן
2. חישוב וקטור המהירות

הערכה בעזרת תחושת בטן:

- סקראם מסטר: הי חברים נוכל לסיים את סיפור A בספרינט הזה? (מצביע על הסיפור בראש רשימת המוצר)
- ליסה: ברור! יש לנו ספרינט של שלושה שבועות והסיפור הזה מאוד טריביאלי!
- סקראם מסטר: אוקי, ומה אם נוסיף את סיפור B (מצביע על הסיפור השני בחשיבותו ברשימת המוצר)
- טום וליסה: עדיין, זה מאוד טריביאלי
- סקראם מסטר: אוקי, ומה עם סיפור A, סיפור B וכן סיפור C?
- סם (לבעל המוצר): האם סיפור C כולל טיפול מתקדם בהודאות שגיאה?
- בעל המוצר: לא, אתה יכול לוותר על זה כרגע, רק טיפול בסיסי אני צריך
- סם: אם כן, C יכול להכלל בספרינט

- סקראם מסטר: ואם נוסיף את D?
- ליסה: ממממ...
- טום: אני מניח שנוכל לעשות את זה
- סקראם מסטר: תחושת בטחון של 90% או 50%
- טום וליסה: יותר של 90%
- סקראם מסטר: בסדר, אז D נכנס גם. ומה אם נוסיף את E?
- סם: אולי
- סקראם מסטר: 50%? 90%?
- סם: יותר קרוב ל-50%
- ליסה: אני מסכימה
- סקראם מסטר: טוב, אז נשאיר את זה מחוץ לספרינט. נתחייב על A, B, C ו D. כמובן, אם נרגיש שנוכל להספיק את E אז נעשה את זה אבל אף אחד לא צריך להסתמך על זה ולכן נוציא אותו מתוכנית הספרינט. מה דעתכם?
- כולם: בסדר!

תחושת בטן עובדת מצוין לצוותים קטנים וספרינטים קצרים.

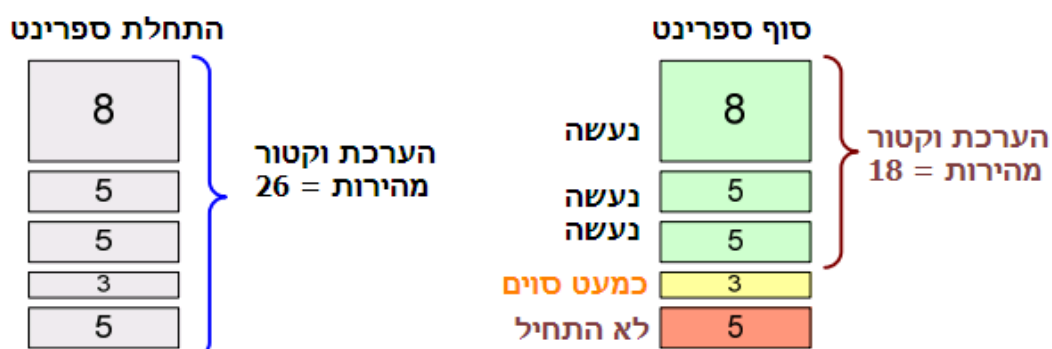
הערכה באמצעות חישוב וקטור המהירות

טכניקה זו מתחלקת לשני חלקים:

1. החלטה על הערכת וקטור המהירות
2. חישוב כמה סיפורים אתם יכולים להוסיף בלי לעבור את ההערכה הזו

וקטור המהירות הוא חישוב "כמות העבודה שנעשתה", כאשר כל פריט משוקלל על ידי ההערכה הראשונית שלו.

התרשים מלמטה מראה דוגמה של הערכת וקטור מהירות בתחילת ספרינט ואת וקטור המהירות הממשי בסוף הספרינט. כל ריבוע הוא סיפור והמספר בתוכו הוא ההערכה הראשונית של אותו סיפור



שימו לב שההערכת וקטור המהירות הממשי מתבססת על ההערכה הראשונית של כל סיפור. אנחנו מתעלמים מכל עדכון הערכת זמן של סיפור במשך הספרינט.

אני כבר יכול לשמוע את ההתנגדויות: "איך זה שימושי? וקטור מהירות גבוה או נמוך תלוי בכל כך הרבה גורמים! מקצוענות התוכניתנים, הערכות ראשוניות לא נכונות, תכולה מתווספת, הפרעות בלתי מתוכננות במהלך הספרינט וכו'!"

אני מסכים, זהו מספר גס. אבל עדיין זה מספר שימושי. בייחוד כשמשווים אותו לכלום. הוא נותן לך קצת עובדות ממשיות. "בלי קשר לסיבה, הנה ההערכה לשוני בין כמה חשבנו שנעשה וכמה עשינו בפועל".

מה עם סיפורים שכמעט סיימנו בספרינט? למה שלא נקבל ניקוד חלקי עליהם? ובכן, זה כדי להמחיש ולהחצין שבסקראם (ולאמיתו של דבר באג'ייל בכלל וב-LEAN גם) הסיפור הוא סביב הדברים שמושלמים ומוכנים לשחרור, באופן מוחלט. הערך בדברים שהם חצי מוכנים הוא אפס (אולי אפילו שליליים). קראו את "Managing the Design Factory" של Donald Reinertsen או את אחד הספרים של Poppendieck כדי ללמוד יותר.

מהי הדרך סודית שבה משתמשים כדי להעריך וקטור מהירות?

דרך מאוד טובה להעריך וקטור מהירות היא להסתכל על ההיסטוריה של הצוות. מה היה וקטור המהירות שלהם במהלך הספרינטים האחרונים? הניחו שוקטור המהירות לספרינט הנוכחי יהיה פחות או יותר זהה.

הטכניקה הזו נקראת *תחזית האתמול*. היא טובה רק לצוותים שעובדים יחדיו כמה ספרינטים (כך שהסטטיסטיקה קיימת) והם יבצעו את הספרינט פחות או יותר באותה צורה, עם אותו גודל צוות סביבת עבודה דומה וכו'. זה כמובן לא תמיד המצב.

משתנה יותר מורכב הוא חישוב משאבים. נניח שאנחנו מתכננים 3 שבועות ספרינט (15 ימי עבודה) עם צוות שמונה 4 אנשים. יעל תהיה בחופש יומיים, גיל יהיה זמין רק ב-50% משרה ויהיה בחופש יום אחד. נשים את כל הנתונים ביחד:

ארז 15
יעל 13
בן 15
גיל 7

סך הכל: 50 ימי עבודה

האם זהו וקטור המהירות? לא! בגלל שהיחידות ההערכה הם הימים אידיאליים. יום אידיאלי הוא יום אפקטיבי ללא הפרעות, מה שמאוד נדיר. יתר על כן, אנחנו צריכים לקחת בחשבון דברים כמו עבודה בלתי מתוכננת שמתווספת לספרינט, אנשים חולים וכו'.

אז ההערכות שלנו יהיו בבירור מתחת ל-50. אבל כמה מתחת ל-50? אנו משתמשים במונח "פקטור מיקוד" לצורך העניין הזה.

ימי עבודה זמינים X פקטור מיקוד = הערכת וקטור מהירות

פקטור מיקוד הוא הערכה של כמה הצוות ממוקד. מיקוד נמוך יכול להעיד על כך שהצוות צופה הרבה הפרעות או מעריך שההערכות שלו היו אופטימיות.

הדרך הטובה ביותר להחליט על פקטור מיקוד ריאלי הוא להסתכל על הספרינט האחרון (או יותר טוב, על ממוצע של כמה ספרינטים). פקטור מיקוד של ספרינט קודם:

$$\frac{\text{ווקטור המהירות הממשי}}{\text{פוקוס פקטור}} = \text{ימי עבודה זמינים}$$

וקטור המהירות הממשי הוא סכום של כל ההערכות הראשוניות של כל הסיפורים שהסתיימו בספרינט.

בואו נניח שבספרינט הסתיימו 18 נקודות סיפור על ידי 3 אנשים. הצוות מורכב מארז, שירלי ובן שיעבדו 3 שבועות שהם 45 ימי עבודה. ועכשיו אנחנו מנסים לחשב את וקטור המהירות של הספרינט המתקרב.

כדי לסבך מעט, בדוגמא זו, איש צוות חדש, אלעד, הצטרף לצוות לספרינט הזה. כשלוקחים בחשבון חופשים וכו', נותרים לנו 50 ימי אדם לספרינט הבא.

LAST SPRINT'S FOCUS FACTOR:

$$40\% = \frac{18 \text{ STORY POINTS}}{45 \text{ MAN-DAYS}}$$

THIS SPRINT'S ESTIMATED VELOCITY:

$$50 \text{ MAN-DAYS} \times 40\% = 20 \text{ STORY POINTS}$$

לכן הערכת וקטור המהירות שלנו לספרינט המתקרב הוא 20 נקודות סיפור. זה אומר שהצוות צריך להעריך את מספר הסיפורים לספרינט עד שהם יגיעו לבערך 20.

התחלת הספרינט



במקרה הזה הצוות יכול לבחור את 4 הסיפורים בסך הכולל של 19 נקודות סיפור או 5 הסיפורים בסך של 24 נקודות סיפור. בוא נניח שהם בחרו 4 סיפורים, מכון שזה יצא כמעט בדיוק 20. כשיש ספק, בחרו פחות סיפורים.

מכוון שסך כל הסיפורים הוא 19 נקודות סיפור, הערכת וקטור המהירות שלהם לספרינט יהיה 19.

תחזית האתמול היא טכניקה הניתנת לשימוש אבל אל תשכחו להשתמש בשכל הישר. אם הספרינט האחרון היה ממש גרוע בגלל שרוב הצוות היה חולה למשך שבוע, אז ניתן להניח שאותו חוסר מזל לא יחזור על עצמו גם בספרינט הזה ניתן לתת הערכה גבוהה יותר לספרינט הבא. אם הצוות התקין לאחרונה מערכת של התקנות אוטומטיות, אתה יכול להגדיל את הערכת וקטור המהירות די בביטחון. אם איש צוות חדש הצטרף לצוות, תצטרך להוריד את וקטור המהירות כדי להביא בחשבון את הלימוד שלו. וכו'.

כשזה אפשרי, הסתכל אחורה כמה ספרינטים ובצע ממוצע של המספרים כדי לקבל הערכות יותר מהימנות.

ומה אם הצוות לגמרי חדש ואין לו סטטיסטיקה? ניתן ללמוד מפקטור המיקוד של צוות אחר עם סביבת עבודה דומה.

ומה אם אין לך אף צוות אחר להסתכל עליו? פשוט תנחש את פקטור המיקוד. החדשות הטובות הן שבניחוש שלך ישתמשו רק בספרינט הראשון. אחרי כן, תהיה לך סטטיסטיקה ותוכלו למדוד ולשפר את פקטור המיקוד ואת הערכת וקטור המהירות.

ברירת המחדל של פקטור המהירות שאני משתמש בו לצוותים חדשים הוא 70%, מכוון שזהו המספר שרוב הצוותים שלנו מגיעים אליו במרוצת הזמן.

באיזו שיטת הערכה אנחנו משתמשים?

הזכרתי כמה וכמה טכניקות – תחושת בטן, חישוב וקטור מהירות המתבססת על תחזית האתמול וכן חישוב וקטור מהירות המתבססת על ימי אדם זמינים והערכת פקטור המיקוד.

אז באיזו טכניקה אנחנו משתמשים?

אנחנו בדרך כלל משתמשים בקומבינציה מסוימת של כל השיטות. זה לא ממש אורך זמן רב.

אנחנו מסתכלים על פקטור המיקוד ועל וקטור המהירות של הספרינט הקודם. אנחנו רואים כמה משאבים זמינים יש לנו לספרינט ומעריכים פקטור מיקוד. אנחנו מדברים על השונות בין שני פקטורי המיקוד ואז עושים התאמות אם צריך.

כשיש לנו רשימה לא מחייבת של סיפורים להכניס לספרינט, אני משתמש ב"תחושת הבטן" כמדד. אני מבקש מהצוות להתעלם מהמספרים לרגע ולחשוב האם בתחושתם זה נראה להם סביר. אם זה נראה להם יותר מדי, אנחנו מורידים סיפור אחד או שניים, והפוך.

בסופו של יום, היעד הוא להחליט כמה סיפורים להכניס לספרינט. פקטור מיקוד, זמינות משאבים והערכת וקטור מהירות הם רק אמצעים להשיג את היעד הזה.

5.10 למה אנחנו משתמשים בכרטיסיות

רוב פגישת תכנון הספרינט קשורה בעבודה על סיפורים מרשימת המוצר, הערכה שלהם, שינוי העדיפות שלהם, הבהרה שלהם, שבירה שלהם לסיפורים קטנים יותר וכו'.

כיצד אנו עושים זאת?

ברירת המחדל היא שהצוותים היו מפעילים את המקרן, מקרינים רשימת מוצר בExcel, ואחד האנשים (בדרך כלל אחראי המוצר או הסקראם מסטר) יושב ליד המקלדת, עובר על התוכן של כל סיפור ומזמין דיון. בזמן שהצוות ואחראי המוצר היו דנים על העדיפויות והפרטים, האדם ליד המקלדת היה מעדכן את הסיפור ישירות בExcel.

נשמע טוב? זהו, שלא. זה בדך כלל נוראי מביא לתוצאה איומה. מה שיותר גרוע, זה שהצוות, בדרך כלל, לא מבחין בכך עד שמגיעים לסוף זמן הישיבה ומבינים שהם עדיין לא הספיקו לעבור על כל רשימת הסיפורים.

פתרון שעובד יותר טוב הוא ליצור כרטיסיות ולשים אותם על הקיר (או על שולחן גדול).



- זה ממשק משתמש טוב בהרבה ממחשב ומקרן מפני ש:
- אנשים קמים ומסתובבים ולכן נשארים ערים ומעורבים לאורך זמן רב יותר.
 - כולם מרגישים יותר מעורבים (בניגוד רק לאדם עם המקלדת)
 - אפשר לערוך מספר סיפורים בו זמנית.
 - שינוי עדיפות הוא קל – פשוט מזיזים את הכרטיסיה.
 - אחרי הפגישה, ניתן לקחת את הכרטיסיות מידית לחדר של הצוות ולהניח על לוח הצוות (ראו עמוד ?? "איך אנחנו עושים ספרינט בקלוג")

אפשר לכתוב את הכרטיסיות ביד או (כמו שאנחנו עושים לרוב) להשתמש בסקריפט פשוט כדי לחולל כרטיסיות מודפסות ישירות מרשימת המוצר.

הפקדה

תיאור

חשיבות

30

הערכה

צריך תרשים UML. אין צורך לדאוג להצפנה בשלב זה.

מה להדגים

הכנס למערכת, פתח עמוד הפקדה, הפקד \$10, לך לעמוד היתרה שלי ובדוק שהיא גדלה ב-\$10.

דרך אגב - ניתן למצוא את הסקריפט (לכרטיסיות באנגלית) באתר שלי
<http://blog.crisp.se/henrikkniberg>

חשוב: אחרי כל ישיבת תכנון הספרינט, הסקראם מסטר שלנו מעדכן ידנית את רשימת המוצר ב Excel בכל השינויים שנעשו בכרטיסיות הסיפורים בזמן. כן, יש בזה מטרד ניהולי אבל אנחנו מצאנו שזה מוצדק, מכיוון ששיבת תכנון הספרינט כל כך יותר יעילה עם כרטיסיות פיזיות.

הערה אחת בקשר לשדה ה"חשיבות" שבדוגמא: זאת החשיבות שנמצאת ברשימת המוצר ברגע ההדפסה.

העובדה שזה מופיע על הכרטיסייה מקלה על המיון הידני לפי חשיבות (בדרך כלל פריטים חשובים יותר נמצאים בצד שמאל של הלוח ופריטים פחות חשובים נמצאים בצד ימין).

אבל, מרגע שהכרטיסיות נמצאות על הלוח, ניתן להתעלם מהשדה הזה ולהתייחס רק למיקום הפיזי על הלוח בשביל לציין חשיבות יחסית. אם אחראי המוצר מחליף בין שני פריטים אין צורך לבזבז זמן בעדכון החשיבות על הנייר. צריך רק לוודא שהחשיבות מעודכנת ברשימת המוצר ב Excel לאחר הפגישה.

הערכות זמן הן, בדרך כלל, קלות יותר (ויותר מדויקות) אם הסיפור נשבר למשימות. למעשה אנחנו משתמשים במילה "פעילות" מכיוון שלמילה "משימה" (Task) יש משמעות אחרת לגמרי בשבדית o:

גם חלוקה למשימות קל ונוח לעשות עם הכרטיסיות שלנו. אפשר לבקש מהצוות להתחלק לזוגות וכל הזוגות יפרקו סיפורים במקביל.



אנחנו עושים את זה על ידי הוספת פתקיות קטנות מתחת לכל סיפור, כל פתקית היא משימה אחת בתוך הסיפור.

- אנחנו לא מעדכנים את רשימת Excel שלנו במשימות משתי סיבות:
 • הפירוק למשימות משתנה ומתבהר במהלך הספרינט, כך שזה יותר מדי טרחה לעדכן את Excel.
- אחראי המוצר לא צריך להיות מעורב ברמת הפירוט הזאת ממילא.

בדיוק כמו בכרטיסיות הסיפורים, גם בפתקיות הפירוק למשימות ניתן להשתמש מחדש ברשימת הספרינט (עמוד XX) "איך אנחנו יוצרים רשימת ספרינט".



5.11 הגדרת הסתיים (Definition of Done)

זה חשוב שאחראי המוצר והצוות יסכימו על הגדרה ברורה של "הסתיים". האם סיפור הסתיים כשכל הקוד Checked in? או האם הוא הסתיים רק כשהוא הותקן על סביבת בדיקות ואושר על ידי צוות הבדיקות האינטגרטיביות? מתי שאפשר אנחנו משתמשים בהגדרת הסתיים של "מוכן להתקן בסביבת הלקוח", אבל לפעמים אנחנו חייבים להסתפק בהגדרת הסתיים של "הותקן על סביבת בדיקות ומוכן לבדיקות קבלה".

בהתחלה, היו לנו רשימות קריטריונים לסיום. עכשיו, אנחנו לרוב פשוט אומרים, "זה הסתיים כשהבודק בצוות הסקראם אומר שזה הסתיים." זה באחריות הבודק לוודא שכוונת אחראי המוצר הובנה ע"י הצוות, ושהפריט "הסתיים" מספיק בכדי לעמוד בהגדרה המוסכמת של "הסתיים".

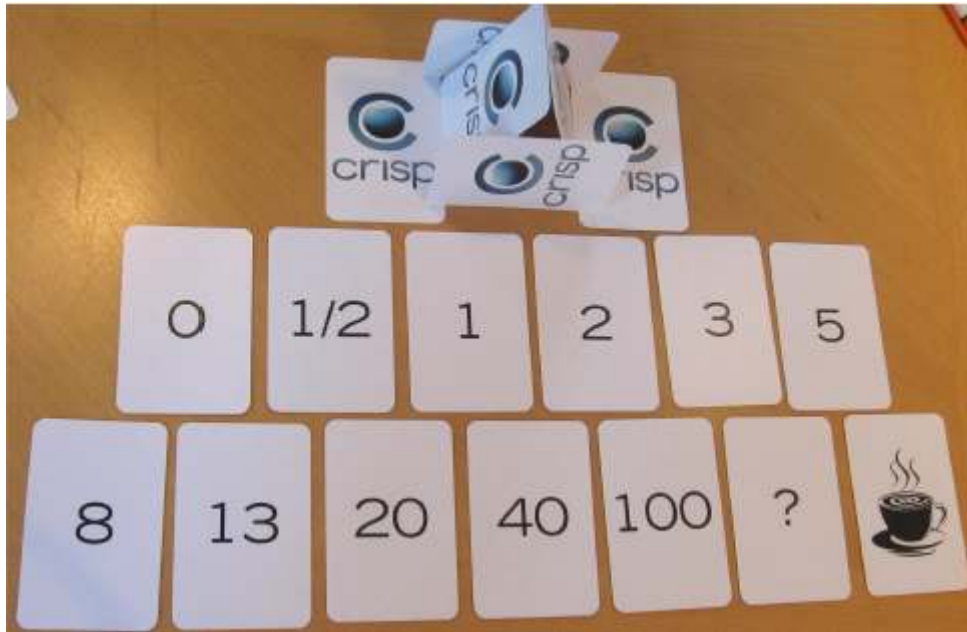
התחלנו להבין שלא כל הסיפורים יכולים לקבל טיפול זהה. סיפור ששמו "טופס שאילתת משתמשים" יקבל טיפול אחר לחלוטין מסיפור בשם "מדריך תפעול". במקרה השני, הגדרת "הסתיים" יכולה פשוט להיות "התקבל ע"י צוות התפעול". לכן הגיון בריא, בדרך כלל עדיף מרשימת בדיקות רשמית.

אם יש לכם, בעיית בהירות סביב הגדרת "הסתיים" (שעשינו בהתחלה) צריך כנראה שדה "הגדת הסתיים" בכל סיפור.

5.12 הערכות זמנים בעזרת פוקר תכנון

- הערכה היא פעילות צוותית – וכל חבר צוות, לרוב, מעורב בהערכת כל סיפור. למה?
- בזמן התכנון אנחנו בדרך כלל לא יודעים בדיוק מי יבצע איזה חלק של הסיפור.
 - בסיפור אחד מעורבים, לעיתים קרובות, מספר אנשים ומספר סוגי מומחים (עיצוב ממשק משתמש, קידוד, בדיקות, וכו').
 - בשביל לתת הערכה, חבר הצוות צריך הבנה כלשהי על מה הסיפור. על ידי כך שאנחנו מבקשים מכולם להעריך כל סיפור, אנחנו מוודאים שכל חבר צוות מבין מה משמעות של כל פריט. זה מעלה את הסיכוי שחברי צוות יעזרו זה לזה במהלך הספרינט. זה גם מעלה את הסיכוי ששאלות חשובות לגבי הסיפור יעלו מוקדם.
 - כשאנחנו מבקשים מכולם להעריך סיפור אנחנו, לרוב, מגלים פערים. כשלשני חברי צוות יש הערכות מאוד שונות של אותו הסיפור. עדיף לגלות ולדון על הדברים בהקדם ולא באיחור.
- אם תבקש מחבר צוות לתת הערכה, בדרך כלל זה שמבין את הסיפור הכי טוב יפלוט הערכה ראשונית. לרוע המזל, זה ישפיע בצורה חזקה על הערכות של האחרים.

יש טכניקה מצוינת בכדי למנוע השפעות אלה – היא נקראת פוקר תכנון (המונח נטבע, כמדומני, על ידי מייק כהן).



כל חבר צוות מקבל חבילה של 13 קלפים, כמו שלמעלה. כל פעם שיש צורך להעריך סיפור, כל חבר צוות בוחר קלף שמסמן את הערכת הזמן שלו (בנקודות סיפור) ומניח אותו עם הפנים למטה על השולחן.

לאחר שכל חברי הצוות סיימו, הופכים את הקלפים באותו הזמן. בצורה זאת כל חבר צוות חייב לחשוב בעצמו במקום להסתמך על הערכות של מישהו אחר.

אם יש פער גדול בין שתי הערכות, הצוות דן על ההפרשים ומנסה לבנות תמונה משותפת של העבודה שנדרשת בשביל הסיפור. יכול להיות שהם יבצעו שבירה מסוימת למשימות. לאחר מכן, הצוות מעריך שוב. הלולאה הזאת ממשיכה עד שההערכות מתכנסות, כלומר כל ההערכות הן בערך אותו הדבר לאותו הסיפור.

זה חשוב להזכיר לחברי הצוות שהם צריכים להעריך את כלל העבודה שדרושה לסיפור. לא רק את החלק "שלהם" של העבודה. הבודק לא צריך להעריך רק את כמות עבודת הבדיקות.

שימו לב שרצף המספרים הוא אינו לינארי. למשל אין שום דבר בין 40 ו-100. מדוע?

זה נועד למנוע תחושת דיוק מזויפת בהערכות זמנים גדולות. אם סיפור מוערך 20 נקודות סיפור בקירוב, זה לא חשוב לדון האם זה 20, 18 או 21. כל מה שאנו יודעים זה שזה סיפור גדול ושקשה להעריך אותו ולכן 20 זה הניחוש שלנו. רוצים הערכה יותר מדויקת? חלקו את הסיפור לסיפורים קטנים יותר והעריכו את כל הסיפורים הקטנים כדי להגיע להערכת הסיפור הגדול!

ולא, אסור לרמות ע"י חיבור של 5 ו-2 בכדי ליצור 7. אתה צריך לבחור בין 5 ל-8, אין 7.

להלן כמה קלפים מיוחדים שיש לשים עליהם לב:

- 0 = "הסיפור הזה כבר גמור" או "הסיפור הזה הוא ממש כלום, כמה דקות של עבודה."
- ? = "אין לי מושג בכלל. שום מושג."
- כוס קפה – "אני עייף מדי לחשוב. בואו ניקח הפסקה קצרה."

5.13 הבהרת משמעות הסיפורים

הגרוע ביותר זה כשצוות מדגים בגאווה תכונה חדשה בישיבת ההדגמה שבסוף הספרינט, ואחראי המוצר אומר במבט מוטרד, "זה יפה, אבל זה לא מה שביקשתי!".

איך אתה מוודא שההבנה של אחראי המוצר ושל הצוות לגבי הסיפור תואמת ושלכל חברי הצוות יש את אותה הבנה של כל סיפור? ובכן, אתה לא יכול, אבל יש מספר טכניקות פשוטות בשביל לזהות את חוסר ההבנות הבוטות ביותר. הטכניקה הפשוטה ביותר היא לדאוג שכל השדות מלאים בכל סיפור (או ליתר דיוק, לכל סיפור שחשיבותו גבוהה מספיק בכדי להיות מועמד להיכנס לספרינט).

דוגמא 1:

הצוות ואחראי המוצר מרוצים לגבי תוכנית הספרינט ומוכנים לסיים את הפגישה. הסקראם מסטר אומר, "חכו רגע, אין הערכה לסיפור 'הוסף משתמש'. בואו נעריך אותו!". אחרי מספר סיבובים של פוקר תכנון הצוות מסכים על 20 נקודות סיפור ואז אחראי המוצר נעמד בזעם "מהההה?!". אחרי כמה דקות של דיון לוהט, מתברר שהצוות לא הבין את הנדרש מהמשימה "הוסף משתמש", הם חשבו שהכוונה היא "ממשק WEB יפה, להוספת משתמשים הורדתם ושליפתם", בשעה שאחראי המוצר התכוון רק "הוסף משתמשים ידנית על SQL מול בסיס הנתונים". הם העריכו שוב והגיעו ל-5 נקודות סיפור.

דוגמא 2:

הצוות ואחראי המוצר מרוצים לגבי תוכנית הספרינט ומוכנים לסיים את הפגישה. הסקראם מסטר אומר, "חכו רגע, לגבי הסיפור 'הוסף משתמש', איך נדגים אותו?". יש קצת לחשופים ואחרי דקה מישחי נעמדת ואומרת "טוב, קודם כל נעשה לוג-אין לאתר, ואז..." אחראי המוצר מפריע "לוג-אין לאתר? לא, לא, לא, לא, הפונקציונאליות הזאת לא צריכה להיות חלק מהאתר בכלל, זה צריך להיות סקריפט SQL טיפשי, רק בשביל מנהל המערכת".

שדה ה-"איך להדגים" של הסיפור יכול (וצריך) להיות מאוד קצר! אחרת לא נסיים את ישיבת התכנון בזמן. זה תיאור כללי בעברית פשוטה של איך לבצע את תסריט הבדיקה הבסיסי ביותר, בצורה ידנית. "עשה את זה, ואז את זה, ואז את זה."

מצאתי שהתיאור הפשוט הזה, בהרבה מקרים, חושף חוסר הבנות חשובות לגבי היקף הסיפור. טוב לגלות אותן מוקדם, נכון?

5.14 פירוק סיפורים לסיפורים קטנים

סיפורים לא צריכים להיות קטנים מדי או גדולים מידי (במונחים של הערכת גודל). אם יש לך המון סיפורים של 0.5 נקודות סיפור, אתה, כנראה, קורבן של ניהול תחת מיקרוסקופ. מאידך, סיפור של 40 נקודות סיפור מהווה סיכון גדול להסתיים חלקית, מה שיגרום לכך שהסיפור לא יביא שום ערך ורק יגדיל את תקורת הניהול.

יותר מזה, אם המהירות המוערכת שלך היא 70, ושני הסיפורים החשובים ביותר הם 40 נקודות סיפור כל אחד, התכנון נהיה קשה. צריך לבחור בין התחייבות חסר (לקיחת רק סיפור אחד), או התחייבות יתר (לקיחת שני הסיפורים). אני מוצא, שכמעט תמיד, ניתן לפרק סיפור גדול לסיפורים קטנים. צריך רק לוודא שהסיפורים הקטנים מייצגים תוצר בעל ערך עסקי.

אנחנו, לרוב, שואפים לסיפורים בגודל של 2-8 ימי אדם. המהירות שלנו היא, בדרך כלל, סביב 40-60, לצוות ממוצע, אז זה אומר סביב 10 סיפורים בספרינט. לפעמים רק 5, לפעמים 15. מספר כרטיסיות כזה ניתן לנהל.

5.15 פירוק סיפורים למשימות

רגע, מה ההבדל בין "משימה" לבין "סיפור"? שאלה הוגנת. ההבדל הוא פשוט. סיפורים הם תוצרים שאכפת לאחראי המוצר מהם. משימות, או שאינם תוצרים כלל, או שהם תוצרים שלאחראי המוצר לא אכפת מהם.

דוגמת פירוק סיפור לסיפורים קטנים:



דוגמא לפירוק סיפור למשימות



להלן מספר תובנות מעניינות:

- צוותי סקראם חדשים נרתעים מהזמן הנדרש לפירוק סיפורים למשימות מראש. חלקם מרגישים שזוגישה שהיא יותר מידי מעולם ה-Waterfall.
- אם הסיפור מובן וברור קל במידה שווה לפרק את הספור מראש או בהמשך התהליך.
- שבירה הסיפורים מגלה עבודה נוספת שגורמת להערכה לעלות, ולכן מייצרת תוכנית ספרינט יותר מציאותית.

- שבירה שכזאת מראש הופכת את ישיבות הסקראם היומיות למשמעותית ויעילות יותר (ראה עמוד XX) "איך אנחנו עושים פגישות סקראם יומיות".
- אפילו אם הפירוק לא מדויק, וישתנה לאחר שהעבודה תתחיל, היתרונות לעיל עדיין תקפים.

אנחנו מנסים שזמן פגישת התכנון יספיק גם עבור פירוק, אבל אם הזמן נגמר, אנחנו מוותרים (ראו "איפה אנחנו מותחים את הקו" למטה).

הערה – אנחנו עובדים ב-TDD (Test Driven Development) מה שאומר למעשה שהמשימה הראשונה בכמעט כל סיפור היא "כתוב בדיקה שנכשלת" והמשימה האחרונה היא Refactor (=שיפור קריאות הקוד והורדת כפילויות).

5.16 החלטה על מיקום וזמן פגישת הסקראם היומית

תוצר אחד של פגישת התכנון, שלרוב נשכח, הוא "מיקום וזמן פגישת הסקראם היומית". בלי זה הספרינט יתחיל בצורה גרועה. פגישת הסקראם היומית הראשונה היא בעצם פגישת ההתנעה שבה כולם מחליטים היכן להתחיל לעבוד.

אני מעדיף פגישות בוקר. אבל, אני חייב להודות, שלא ממש ניסינו לערוך את פגישות הסקראם היומיות אחרי הצהריים או באמצע היום.

חסרונות של סקראם אחרי הצהריים: כשאתה מגיע בבוקר לעבודה, אתה צריך לנסות להיזכר מה אמרת לאנשים אתמול לגבי תכנון העבודה שלך להיום.

חסרונות של סקראם בבוקר: כשאתה מגיע בבוקר לעבודה, אתה צריך לנסות להיזכר מה עשית אתמול כדי שתוכל לדווח על כך. לדעתי, החיסרון הראשון גרוע יותר, כי יותר חשוב מה שאתה הולך לעשות ממה עשית.

דרך ברירת המחדל שלנו היא לבחור את השעה המוקדמת ביותר שלא גורמת לאף חבר צוות להיאנח. בדרך כלל, 9:00, 9:30 או 10:00. החשוב ביותר, שתיקבע שעה שכל חברי הצוות מסכימים לה.

5.17 איפה מותחים את הקו

טוב, הזמן אוזל. מכל הדברים שאנחנו רוצים להספיק בזמן פגישת התכנון, על מה מוותרים אם נגמר הזמן? אני משתמש ברשימת העדיפויות הבאה:

עדיפות 1: מטרת הספרינט ותאריך ההדגמה. זה המעט ביותר הדרוש בכדי להתחיל ספרינט. לצוות יש מטרה, תאריך סיום, והם יכולים לעבוד ישירות מרשימת המוצר. זו אלטרנטיבה גרועה, וכדאי לחשוב ברצינות על קביעת ישיבת תכנון נוספת למחרת, אבל אם חייבים להתחיל את הספרינט זה כנראה יספיק. בשביל להיות הוגן, עם זאת, אני מעולם לא התחלתי ספרינט עם כל כך מעט מידע.

עדיפות 2: רשימת סיפורים שהצוות קיבל לספרינט.

עדיפות 3: שדה הערכת זמנים מולא לכל סיפור בספרינט.

עדיפות 4: שדה "איך להדגים" מולא לכל סיפור בספרינט.

עדיפות 5: חישובי מהירות ומשאבים, כבדיקת שפיות לתוכנית הספרינט. כולל רשימת חברי הצוות וההתחייבויות שלהם (אחרת לא ניתן לחשב את המהירות).

עדיפות 6: זמן ומיקום לפגישת הסקראם היומית. זה לוקח רק רגע להחליט, אבל אם נגמר הזמן, הסקראם מסטר יחליט וישלח דוא"ל לכולם.

עדיפות 7: סיפורים מפורקים למשימות. הפירוק הזה ליכול להתבצע ברמה יומית בשילוב עם פגישות הסקראם היומיות, אך זה יפריע קצת לזרימת הספרינט.

5.18 סיפורים טכניים

ועכשיו לנושא מסובך: סיפורים טכניים. או פריטים לא פונקציונאליים, או איך שתמצאו לקרוא להם. אני מתכוון לדברים שצריכים להיעשות, אבל הם לא מותקנים, לא קשורים ישירות לאף סיפור, ולא מעניינים, ישירות, את אחראי המוצר. אנחנו קוראים להם "סיפורים טכניים". לדוגמא:

- **להתקין שרת ל-Continuous Build**

- למה צריך לעשות את זה: כי זה חוסך כמויות עצומות של זמן למפתחים ומקטין את הסיכון של אינטגרציות גדולות בסוף האיטרציה.

- **כתוב מסמך Design של המערכת**

- למה צריך לעשות את זה: כי מפתחים שוכחים את התכנון הכללי, ולכן כותבים קוד לא עקבי. צריך מסמך של "התמונה הגדולה" בשביל שכולם יישרו קו מבחינת ה-Design.

• Refactor לשכבת ה-DAO

- למה צריך לעשות את זה: כי שכבת ה-DAO נהייתה מאוד מלוכלכת, וגזלת מכולם זמן בגלל בלבול ותקלות. ניקוי הקוד יחסוך זמן לכולם ויגדיל את חוסן המערכת.

• שדרוג של Jira (מערכת ניהול באגים)

- למה צריך לעשות את זה: הגרסה הנוכחית מלאה באגים ואיטית, שדרוג יחסוך לכולם זמן.

האם אלו סיפורים במובן הרגיל של המילה? האם אלה משימות שלא קשורות לשום סיפור? מי מתעדף אותם? האם אחראי המוצר צריך להיות מעורב בדברים כאלו?

ניסינו המון דרכים שונות לטפל בסיפורים טכניים. ניסינו להתייחס אליהם כאל סיפורים כמו כל סיפור אחר. זה לא עבד, כשאחראי המוצר תעדף את רשימת המוצר זה היה כמו להשוות תפוחים ותפוזים.

למעשה, מסיבות ברורות, סיפורים טכניים קיבלו לרוב עדיפות נמוכה מסיבות כגון "כן חברה, אני בטוח ששרת ל-Continuous Build זה דבר חשוב, אבל בוא נבנה יכולות שידחפו הכנסות קודם, בסדר? אחר כך נוסיף את הממתקים הטכניים שלכם, טוב?"

לעיתים אחראי המוצר צודק, אבל לרוב לא. הגענו למסקנה שאחראי המוצר לא תמיד מסוגל להעריך מה עדיף. הנה מה אנחנו עושים:

1. מנסים להימנע מסיפורים טכניים. חפש היטב דרך להפוך כל סיפור טכני לסיפור רגיל עם ערך עסקי. בדרך זאת אחראי המוצר יכול להעריך בצורה טובה מה עדיף על פני מה.
2. אם אנחנו לא יכולים להפוך את הסיפור הטכני לסיפור רגיל, נראה האם העבודה יכולה להיות משימה בתוך סיפור אחר. לדוגמה "Refactor לשכבת ה-DAO" יכולה להיות משימה בתוך הסיפור "ערוך משתמש", מכיוון שזה נוגע בשכבת ה-DAO.
3. אם שתי האופציות לעיל נכשלות, הגדר סיפור טכני, והחזק רשימה נפרדת של סיפורים אלו. תנו לאחראי המוצר לראות את הרשימה אך לא לערוך אותה. השתמשו ב"מקדם המיקוד" וב-"מהירות המוערכת" כדי לנהל משא ומתן עם אחראי המוצר ולהשיג זמן בספרינט ליישם סיפורים טכניים.

דוגמא (שיחה דומה לזאת התקיימה באחת משיבות תכנון הספרינט שלנו)

- צוות: "יש לנו כמה דברים טכניים פנימיים שצריך לעשות. היינו רוצים להקצות לזה 10% מהזמן שלנו, כלומר להוריד את מקדם הריכוז מ-75% ל-65%. זה בסדר?"
- אחראי מוצר: "בשום פנים ואופן לא! אין לנו זמן!"
- צוות: "תסתכל על הספרינט האחרון (כל הראשים מסתובבים להסתכל על ציון המהירות על הלוח) המהירות המוערכת שלנו הייתה 80, והמהירות האמיתית שלנו הייתה 30, נכון?"

- אחראי מוצר: "בדיוק! אז אין לנו זמן לעשות דברים טכניים פנימיים! צריכים יכולות חדשות!"
- צוות: "אבל, הסיבה שהמהירות שלנו הייתה כל כך נמוכה הייתה בגלל שבזבזנו הרבה זמן בניסיון להרים גרסאות יציבות לבדיקות."
- אחראי מוצר: "כן, אז?"
- צוות: "המהירות שלנו תמשיך להיות כזאת נמוכה אם לא נעשה משהו בקשר לזה."
- אחראי מוצר: "כן, אז?"
- צוות: "אז אנחנו מציעים להקצות 10% מהזמן להקים שרת Continuous Build ועוד דברים שיקלו על כאב האינטגרציה. זה, כנראה, יגביר את המהירות שלנו בסביבות 20% בכל הספרינטים הבאים, לנצח!"
- אחראי מוצר: "באמת? למה לא עשינו את זה בספרינט הקודם?!"
- צוות: "אממ...כי לא רצית שנעשה את זה..."
- אחראי מוצר: "אה, בסדר, זה נשמע כמו רעיון טוב. תעשו את זה עכשיו!"

כמובן שהאפשרות השנייה היא להוציא את אחראי המוצר מהדיון או להציב בפניו מקדם מיקוד שלא פתוח למשא ומתן. אבל אין סיבה לא לנסות להגיע קודם כל להסכמה.

אם אחראי המוצר הוא אדם הגיוני ומקצועי (והיה לנו מזל בדברים הללו), אני מציע לידע אותו ככל האפשר ולתת לו להחליט על תעדוף. שקיפות היא אחד מעקרונות הבסיס של סקראם, נכון?

5.19 מערכת לניהול באגים לעומת רשימת מוצר

הנה נושא מורכב. Excel הוא כלי מצוין לרשימת המוצר. אבל עדיין צריך מערכת למעקב אחרי באגים, Exceli כנראה לא יועיל. אנחנו משתמשים ב-Jira. אז איך מגיעים מקרים מ-Jira לפגישת תכנון הספרינט? הכוונה היא שזה לא טוב להתעלם מהם ולהתמקד רק בסיפורים. ניסינו מספר אסטרטגיות:

1. אחראי המוצר מדפיס את המקרים בעלי העדיפות הגבוהה ביותר מ-Jira, מביא אותם לשיבת תכנון הספרינט, ומניח אותם על הקיר ביחד עם הסיפורים האחרים (ובכך מראה את העדיפות שלהם ביחס לסיפורים אחרים).
2. אחראי המוצר יוצר סיפורים שמתייחסים למקרים ב-Jira. לדוגמא "תקנו את הבאגים החשובים ביותר במשרד האחורי, Jira-124, Jira-126 ו-Jira180".
3. תיקון באגים נחשב מחוץ לספרינט, כלומר הצוות שומר על מקדם מיקוד נמוך מספיק (לדוגמא 50%) כדי להבטיח שיהיה להם זמן לתקן באגים. עכשיו, ניתן להניח שהצוות יבלה כמות של זמן בכל ספרינט בתיקון באגים.
4. לשים את רשימת המוצר ב-Jira (כלומר, לזרוק את Excel). להתייחס לבאגים כמו אל כל סיפור.

לא הגענו, באמת, למסקנה איזו אסטרטגיה היא הטובה ביותר בשבילנו, למעשה זה משתנה מצוות לצוות ומספרינט לספרינט. אני נוטה לכיוון האסטרטגיה הראשונה. היא פשוטה ונוחה.

5.20 ישיבת תכנון הספרינט סוף סוף מסתיימת!

ואו, לעולם לא הייתי חושב שהפרק הזה על ישיבת תכנון הספרינט יהיה כזה ארוך! אני מניח שזה משקף את דעתי שיש ישיבת תכנון הספרינט, היא הדבר החשוב ביותר שעושים בסקראם. אם תקדישו מספיק זמן לעשות זאת בצורה טובה, שאר העבודה תהיה פשוטה הרבה יותר.

ישיבת תכנון הספרינט היא מוצלחת אם כולם (כל חברי הצוות ואחראי המוצר) יוצאים מהישיבה בחיוך, קמים למחרת בבוקר עם חיוך, ועושים את פגישת הסקראם היומית הראשונה שלהם עם חיוך.

אז, כמובן, הרבה דברים יכולים ללכת ממש לא טוב בהמשך הדרך, אבל לפחות לא תוכלו להאשים את תוכנית הספרינט (o)

6 איך אנחנו מתקשרים ספרינטים

חשוב ליידע את כל החברה לגבי מה שקורה. אחרת אנשים יתלוננו, או גרוע מכך – יניחו הנחות שגויות לגבי מה שקורה.
אנחנו משתמשים ב"דף מידע לספרינט" לשם כך:

צוות גלדיאטור, ספרינט 15

מטרת הספרינט

- גרסא מוכנה לביתא!

באקלוג הספרינט (הערכות בסוגריים)

- הפקדה (3)
- כלי מיגרציה (8)
- הזדהות (5)
- משתמש אדמניסטרטיבי (5)

הערכת מהירות: 21

זמנים

- תקופת הספרינט: 6/11-24/11
- פגישה יומית: 09:30-09:45 בחדר הצוות
- הדגמה: 24/11, 13:00 בקפיטריה

הצוות

- גדי
- יעל (סקראם מסטר)
- אבי (75%)
- מיכל
- דניאל

לעתים אנו כוללים מידע לגבי איך יודגם כל סיפור.
מיד לאחר פגישת התכנון הסקראם מסטר יוצר את הדף, מעלה אותו לאתר הצוותי ושולח דואר זבל לכל החברה.

נושא: ספרינט 15 גלדיאטורים התחיל

הי כולם! צוות גלדיאטורים התחיל את ספרינט 15. מטרתנו היא להדגים גרסא מוכנה לביתא ב-24 בנובמבר.
לפרטים צפו בדף המידע לספרינט:

<http://wiki.mycompany.com/gladiator/sprint15>

יש לנו גם לוח ארגוני באתר החברה עם לינקים לכל הספרינטים הנוכחיים:

לוח החברה
ספרינטים נוכחיים:

- [צוות X ספרינט 15](#)
- [צוות Y ספרינט 12](#)
- [צוות Z ספרינט 1](#)

בנוסף, הסקראם מסטר מדפיס את דף המידע לספרינט ותולה אותו מחוץ לחדר הצוות שלו. כך כל אחד יכול לראות את הדף ולדעת מה עושה הצוות. כיוון שהדף כולל את הזמן והמקום לפגישה היומית ולהדגמה, הוא יודע לאן ללכת על מנת לברר מידע נוסף. לקראת סיום הספרינט, הסקראם מסטר מזכיר לכולם על פגישה ההדגמה:

נושא: הדגמת ספרינט גלדיאטורים מחר ב-13:00 בקפיטריה

הי כולם! אתם מוזמנים להשתתף בהדגמת הספרינט שלנו ב-13:00 בקפיטריה מחר (חמישי). נדגים גרסא מוכנה לביתא. לפרטים צפו בדף המידע לספרינט:
<http://wiki.mycompany.com/gladiator/sprint15>

לאחר כל כך הרבה מאמצים, לאף אחד אין תירוץ אמיתי לא לדעת מה קורה.

7 איך אנחנו מבצעים ספרינט

הגעת עד כה ? וויפי, כל הכבוד.

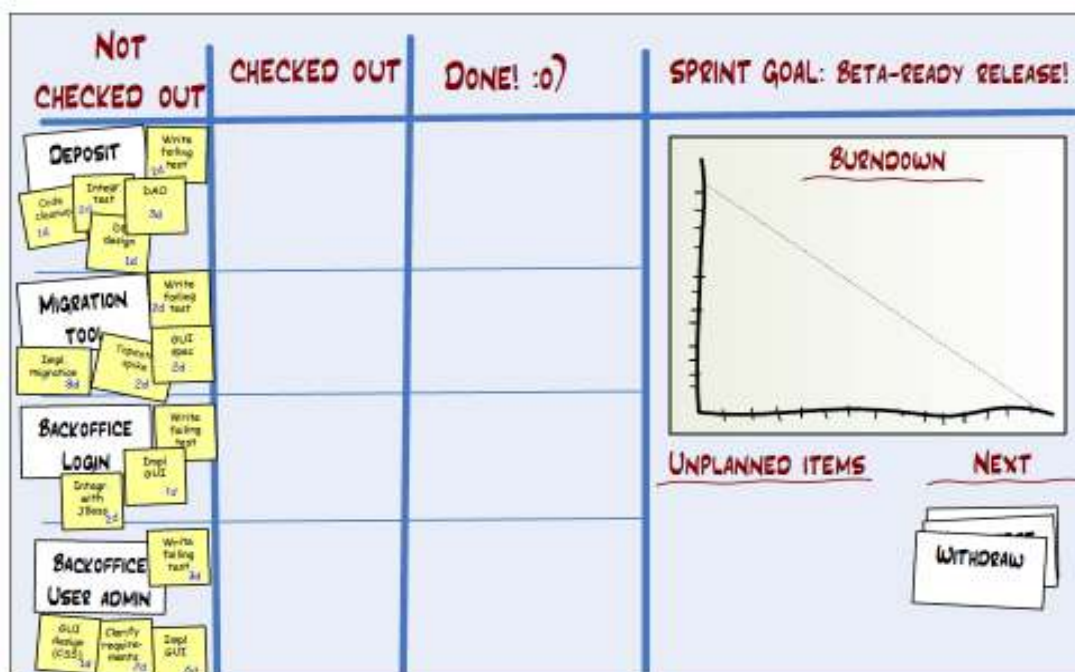
אז עכשיו לאחר כשסיימנו את ישיבת תכנון הספרינט וסיפרנו לעולם על הספרינט החדש והנוצץ שלנו, הגיע הזמן שהסקראם מסטר יכין את רשימת הספרינט. זה צריך להעשות אחרי ישיבת התכנון אבל לפני הישיבה היומית הראשונה.

7.1 מיבנה רשימת הספרינט

התנסנו במספר פורמטים של רשימת המוצר, כולל Jira, Excel, ולוח פיזי על הקיר. בהתחלה השתמשנו בעיקר ב Excel, יש הרבה תבניות זמינות ברשת לניהול רשימת המוצר, כולל ייצור אוטומטי של ה burndown, (תרשימים ודברים כאלו. אני יכול לדבר הרבה על האופן שבו שיפרנו את רשימת המוצר שלנו ב Excel אך אני לא אעשה זאת. אני אפילו לא אוסיף את הדוגמא שלנו כאן.

במקום זה, אני אתאר את מה שמצאנו שעובד הכי טוב והכי אפקטיבי עבור רשימת הספרינט – לוח משימות על הקיר (wall-based taskboard)!

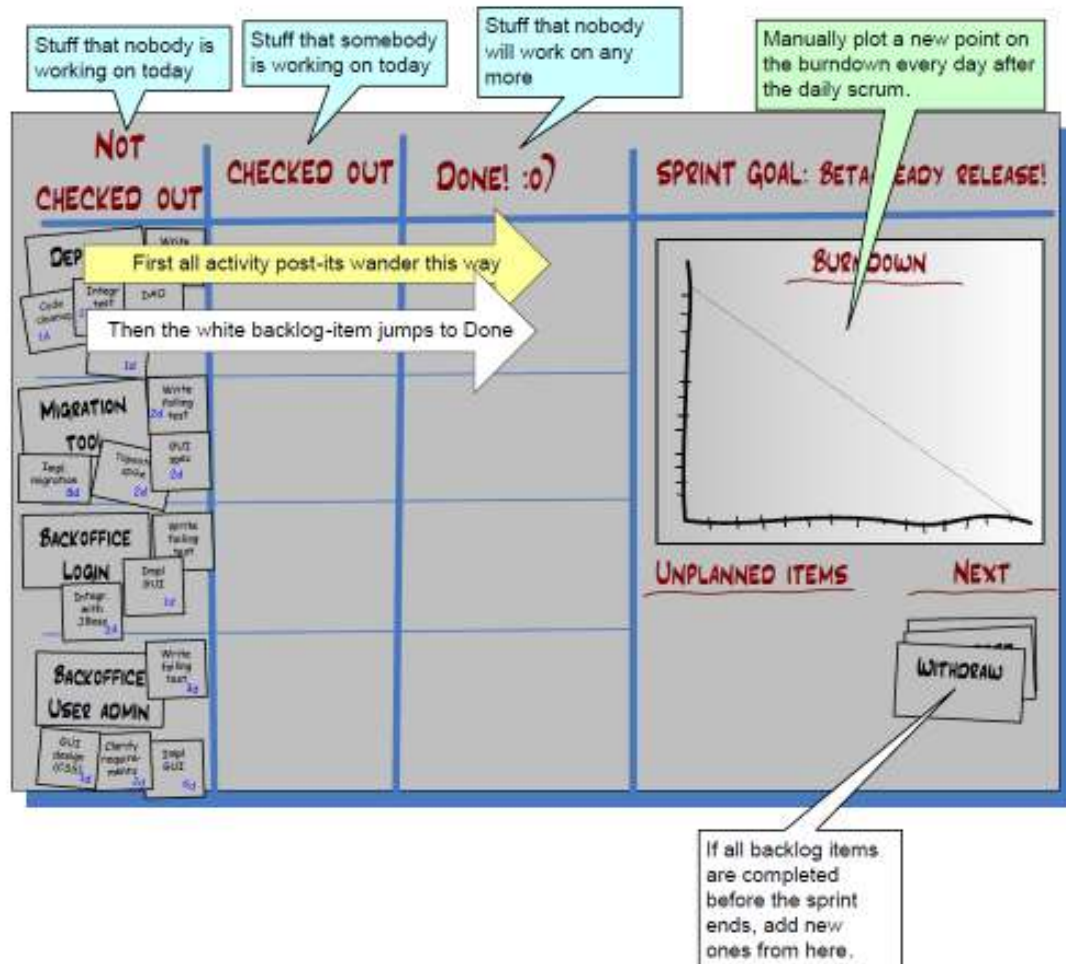
מצא קיר גדול שלא משתמשים בו או שתלוי עליו הלוגו של החברה או תמונות מכוערות. נקה את הקיר (בקש אישור אם אתה חייב) ואז תכין עליו את הדבר הבא:



אתה יכול כמובן להשתמש בלוח מחיק (whiteboard). אבל זה קצת ביזבז. אם אפשר, שמור את הלוחות לקישקושים של דיוני עיצוב ותשתמש בקיר רגיל עבור לוח משימות (taskboards)

שים לב – אם אתה משתמש בפתקים דביקים, אל תשכח להצמיד אותם לקיר עם דבק אמיתי, כדי שלא תמצא אותם על הרצפה יום אחד.

7.2 איך לוח המשימות (taskboard) עובד

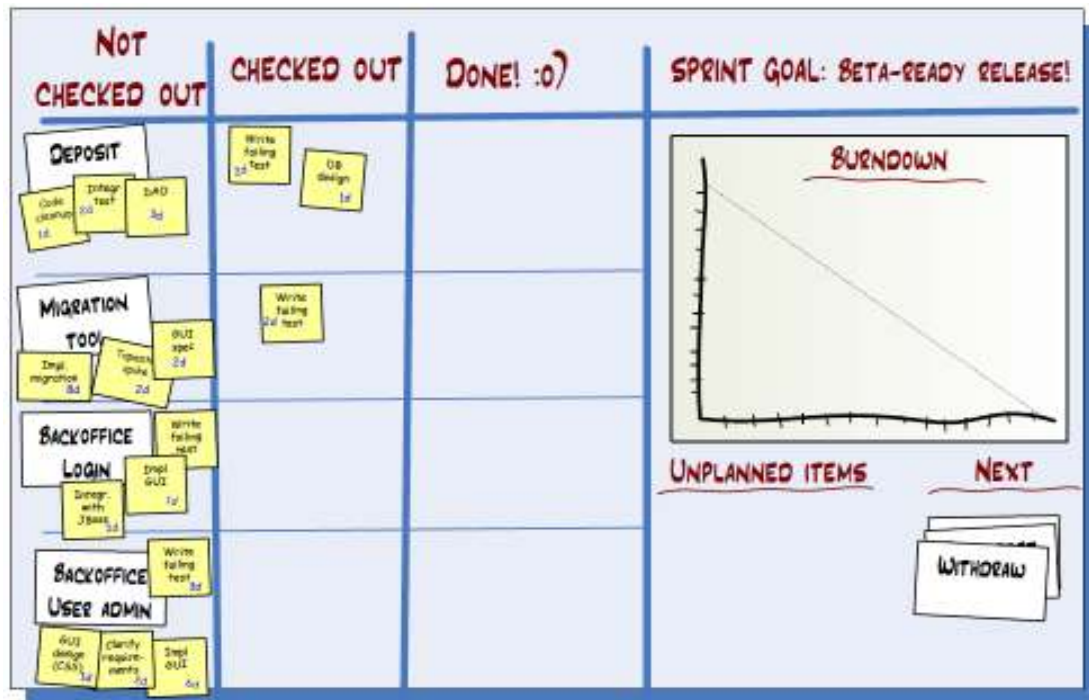


אתה יכול כמובן להוסיף כל מיני עמודות נוספות. "מחכה לאינטגרציה" למשל. או "מבוטל". בכל מיקרה לפני שאתה מסבך את העניינים, חשוב לעומק.

האם התוספת הזאת באמת באמת נחוצה? אני גיליתי שלפשוט יש ערך גבוה במקרים שכאלה. הייתי מוסיף סיבוכיות רק כשהעלות שלא להוסיף אותה גבוהה.

7.3 דוגמא 1 – אחרי ה daily הראשון

אחרי ה daily הראשון הלוח יכול להראות ככה:

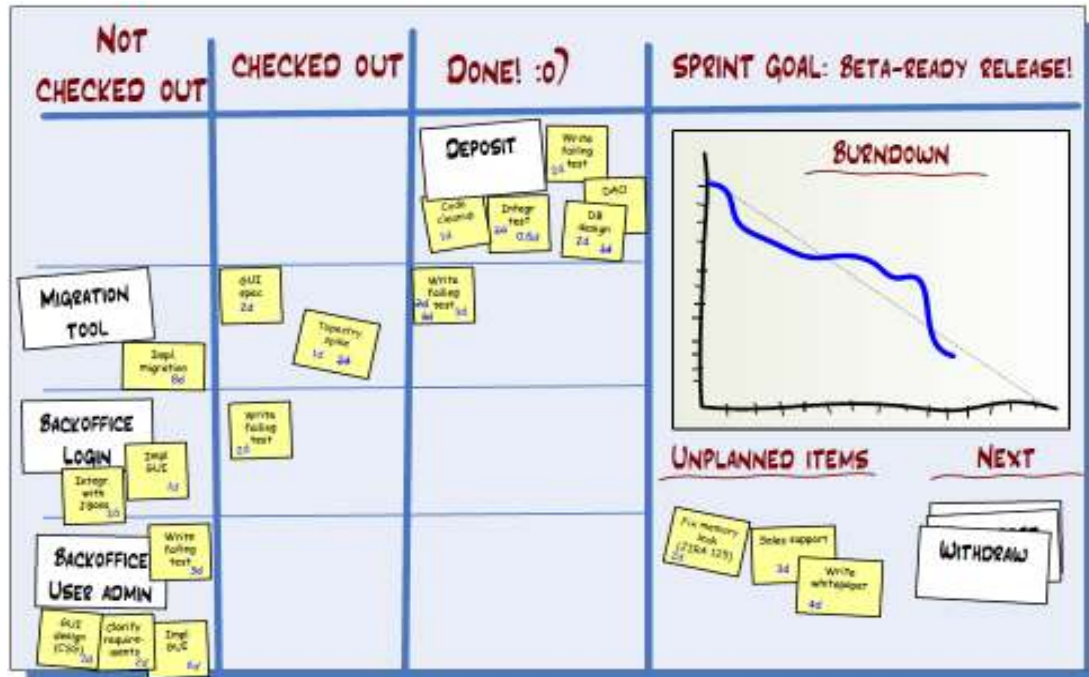


כמו שאתה רואה, שלוש משימות עברו ל "checked-out", ז"א, הצוות יעבוד על הפריטים הללו היום. לפעמים, בצוותים גדולים, משימה נתקעת ב "checked out" משום שאף אחד לא זוכר מי עבד עליה.

אם זה קורה הצוות מגדיר מדיניות חדשה, כגון לסמן על המשימה את שמו של מי שהעביר ל "checked-out",

7.4 דוגמא 2 – כעבור מספר ימים

מספר ימים לאחר מכן הלוח יכול להראות כך:



כמו שניתן לראות סיימנו את הסיפור שנקרא "Deposit" (ז"א שזה הוכנס ל source control ונבדק). הסיפור Migration Tool מוכן חלקית והסיפור הבא הותחל. את האחרון עוד לא התחילו.

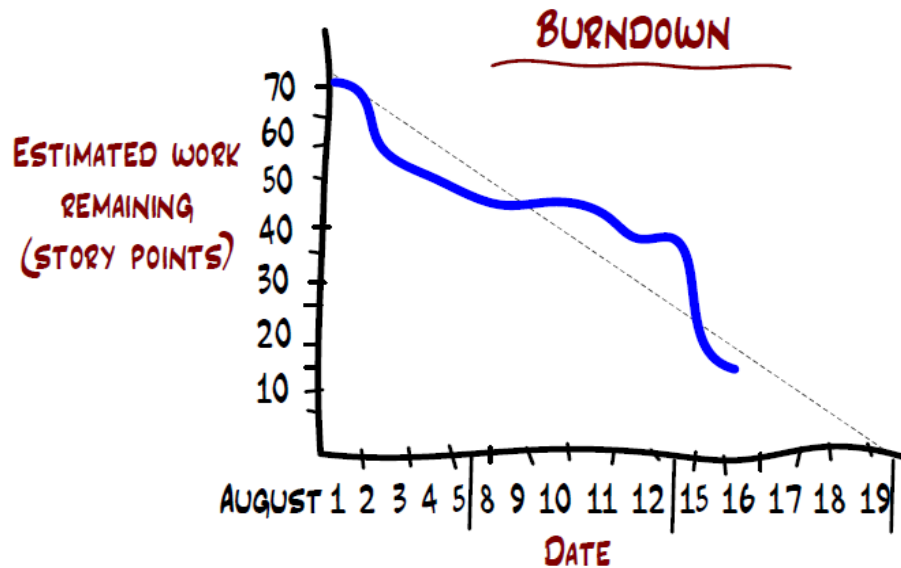
היו לנו 3 פריטים שלא תוכננו. זה שימושי לזכור זאת כאשר עושים את retrospective לספרינט.

הנה דוגמא של לוח הספרינט לקראת סיום הספרינט. הוא אכן ניהיה עמוס ככל שהספרינט מתקדם, אבל זה בסדר יש לו חיים קצרים. בכל ספרינט חדש. אנחנו יוצרים לוח חדש ונקי.



7.5 איך עובד תרשים Burndown

בואו נתמקד בתרשים ה Burndown

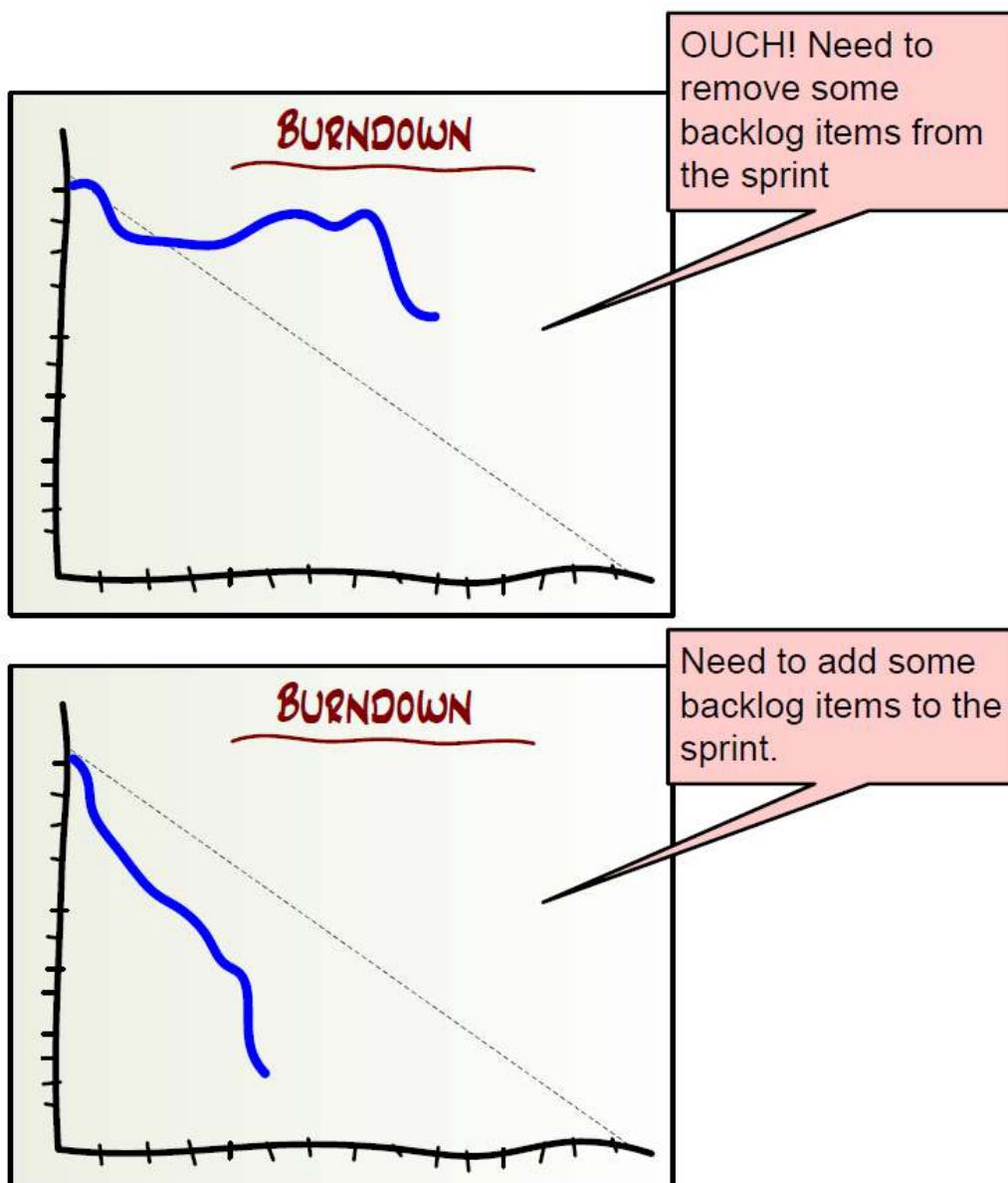


התרשים מראה:

- ביום הראשון של הספרינט (1 באוגוסט) הצוות העריך שנשארו בערך 70 סטורי פוינטס על מנת לסיים את הספרינט. זה בעצם היה הערכה של וקטור המהירות של כל הספרינט.
 - ב 16 לאוגוסט הצוות העריך שנשארו 15 סטורי פוינטס. הקו המקווקו מראה שהצוות מתקדם לפי המצופה, ז"א שבקצב הזה הצוות יסיים את כל המשימות עד סוף הספרינט.
- אנחנו לא מראים סופי שבוע על ציר ה-X, משום שבדרך כלל עבודה אינה מתבצעת בסופי שבוע.

7.6 סימני אזהרה על הלוח

מבט מהיר על הלוח אמור לתת לכל אחד אינדיקציה על רמת ההתקדמות של הספרינט. הסקראם מסטר צריך לוודא שהצוות פועל כאשר סימני אזהרה כמו אלו מופיעים



OUCH! Team is doing stuff in the wrong order, and neglecting the top-priority items!

CHECKLIST

- DEPOSIT
 - Driver test 2d
 - Unit test 1d
 - UI 1d
- MIGRATION TOOL
 - Integration test 1d
 - Unit test 1d
 - UI 1d
- BACKOFFICE LOGIN
 - Unit test 1d
 - UI 1d

BURNDOWN

UNPLANNED ITEMS

- For security audit (2d)
- Setup system 1d
- Write unit tests 1d

NEXT

- WITHDRAW

BACKOFFICE USER ADMIN

- Unit test 1d
- UI 1d
- Unit test 1d
- UI 1d

NOT CHECKED OUT

CHECKED OUT

DONE! :o)

SPRINT GOAL: BETA-READY RELEASE!

OUCH! Unplanned items are killing the sprint!

BURNDOWN

UNPLANNED ITEMS

NEXT

WITHDRAW

7.7 הי, מה אם היכולת לעקוב אחר שינויים?!

הדרך הטובה ביותר במודל הזה היא לצלם באופן דיגיטלי כל יום. אני עושה את זה לפעמים אבל אף פעם לא מוצא את הצורך לחפור ולהשתמש בתמונות אלו.

אם זאת, במידה וזה נושא שחשוב לך אז אולי הפתרון של הלוח לא מתאים עבורך.

אני מציע שתנסה להעריך מה באמת הערך של היכולת לאחזר את הנתונים של הספרינט. נשאלות השאלות: ברגע שהספרינט הסתיים והתוכנה סופקה, למישהו באמת אכפת כמה סיפורים הסתיימו ביום החמישי של הספרינט? למישהו באמת אכפת מה הייתה הערכת זמנים הראשונית של המשימות?

7.8 הערכה בימים לעומת בשעות

ברוב הספרים על סקראם תמצא כי המשימות מוערכות בזמנים ברזולוציה של שעות, לא ימים. הנוסחא הכללית שלנו הייתה 1 יום אדם אפקטיבי=6 שעות אפקטיביות.

הפסקנו לעשות את זה, לפחות ברוב הצוותים, מהסיבות הבאות:

- זה היה ברזולוציה גבוהה מידי, ולכן מייצר נטייה לניהול יתר (micromanagement).
- התברר שכולם חושבים במושגים של ימי אדם (man days) בכל מיקרה. ורק להכפיל את זה ב-6 לפני שכותבים שעות אדם....

"ממממ....המשימה הזו תעריך בערך יום עבודה, טוב אני צריך לכתוב

שעות, אז אני אכתוב 6 שעות"

- שתי יחידות זמן מייצרות בלבול: "האם זה הוערך בימי אדם או בשעות אדם?"

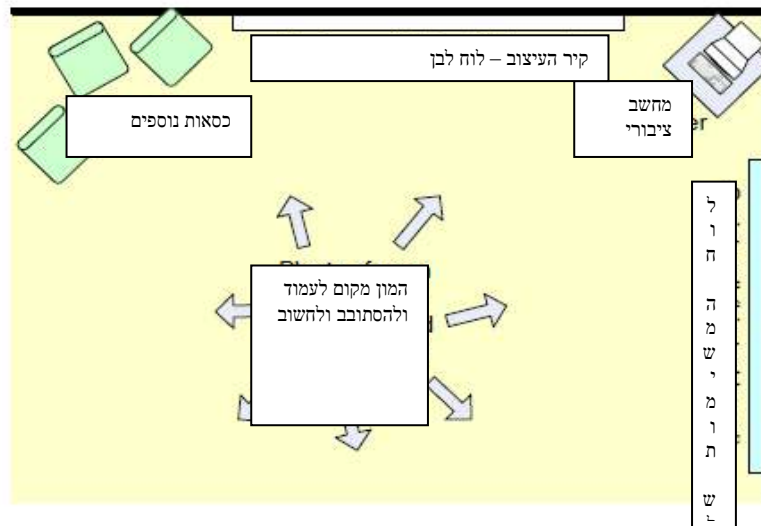
אז עכשיו אנחנו משתמשים בימי אדם עבור כל הערכות שלנו. הערך הקטן ביותר שלנו הוא חצי יום.

8. כיצד מארגנים את חדר הצוות.

8.1 פינת ה DESIGN (עיצוב)

שמתי לב שהדיונים המעניינים והחשובים ביותר בענייני Design (עיצוב) קורים באופן ספונטני מול לוח המשימות.

מהסיבה הזו, אנו מנסים לארגן אזור זה שיהיה מיועד להיות פינת העיצוב של הצוות.



למעשה, זה דבר מאד יעיל.

אין דרך טובה יותר לקבל מושג על מצב המערכת מאשר לעמוד בפינת העיצוב ולהתבונן בשני הקירות, ואז להציץ לעבר המחשב ובו זמנית לנסות את ה build הבא. (אם יתמזל מזלך, אז יש לך Continuous build, בניה רציפה) - ראה עמ' 62 "כיצד משלבים xpi scrum -"

קיר העיצוב "Design wall" הוא בסה"כ לוח לבן גדול המכיל את הפרטים החשובים ביותר לגבי העיצוב, כגון סירטוטים, הדפסות חשובות, אב טיפוס של ממשק משתמש, וכ"ו.



בתמונה למעלה ניתן לראות כיצד פגישת הסקראם היומית מתקיימת בפניה המוזכרת בעמוד קודם.

מממ.. שימו לב לגרף ההתקדמות בצד , הוא נראה לי ישר מידי , אבל הצוות מתעקש שזה אמיתי : 0)

8.2 הושיבו את הצוות ביחד

כשחשמדובר במקומות ישיבה , סידור שולחנות בחלל , חשוב מאד להתעקש בצורה חזקה וברורה:

להושיב את כל הצוות ביחד!

על מנת להבהיר את זה מעט , מה שאני אומר זה

להושיב את הצוות ביחד!

אנשים מסרבים לזוז. לפחות במקומות שבהם אני עבדתי. לא בא להם להרים את כל חפציהם , לנתק את המחשבים , להזיז את כל ה"זבל" שלהם לשולחן חדש , ולחבר ולארגן הכל מהתחלה. ככל שהמרחק קצר יותר , כך ההתנגדות לעבור גדולה יותר: "בחיך , בוס ,מה הטעם לעבור חמישה מטרים?". במידה ואנו מעונינים לבנות צוותי סקראם אפקטיביים , אין אלטרטיבה מלבד להושיב את כולם ביחד.

כאשר בונים צוותי סקראם אפקטיביים אין אפשרות אחרת. קח את כל הצוות והעבירו אותם למקום אחד. גם אם זה אומר שצריך לשוחח באופן אישי אם כל אחד ואחד , לאיים , לסחוב את חפציו , או לנקות את כיתמי הקפה הישנים מהשולחן שלו. אם אין מרחב מספיק להושיב את כל הצוות , תיצור את המרחב הזה. איפשרו , לא משנה היכן , אפילו אם זה אומר להושיב אותם במרתף. תזיז שולחנות , תשחד את מנהל המשרד , תעשה כל מה שצריך. העיקר להושיב את כל הצוות ביחד.

ברגע שכל אנשי הצוות ישבו ביחד , ההחזר יהיה מידי. לאחר ספרינט אחד בלבד הצוות כבר יסכים שזה היה רעיון טוב לעבור לשבת ביחד (כמובן שאני מדבר מניסיוני האישי ,אלא אם כן הצוות באמת יהיה עקשן מכדי להודות בזה).

מה אומר ה"ביחד" הזה? איך אמורים השולחנות להיות מסודרים? בעצם , אין לי ממש דעה חד משמעית לגבי הסידור הזה. ואפילו אם היה לי , אני מניח שלרוב הצוותים אין את הפריבילגיה להחליט בדיוק איך ימוקמו השולחנות בחדר.בדרך כלל יש אילוצים פיזיים – הצוות השכן, דלת השירותים ,מכונת המזל במרכז החדר , ועוד כל מיני. משמעות ה"ביחד" היא :

- **שמע** - היכולת של חברי צוות לדבר בניהם מבלי לקום מהשולחן.
- **ניראות** - כל אחד בצוות יכול לראות כל אחד אחר. כל אחד יכול לראות את לוח המשימות. לא בהכרח קרוב מספיק על מנת 'לקרוא' את הלוח אבל בהחלט 'לראות' אותו.

- **בידוד** - במידה וכל הצוות נעמד לפתע באופן ספונטני ומחליט לבצע דיון חי בענייני Design, לא יהיה מישהו אחר מחוץ לצוות קרוב מספיק שהעניין עלול יפריע לו. וההפך גם הוא נכון. "בידוד", אינו אומר שהצוות צריך להיות מבודד לחלוטין. בסביבה של Cubicles, מספיק שלצוות יהיה את cubicle- שלו, וקירות ה-cubicle רצוי שיהיו גבוהים מספיק ורחבים על מנת לסנן את רוב הרעשים שאינם רעשי הצוות. ומה קורה במצב של צוותים מבוזרים? ובכן, במצב כזה נגמר לך המזל. על מנת למזער את הנזק, רצוי להשתמש בכמה שיותר עזרים טכניים כגון – ישיבות ועידה דרך וידאו, מצלמות אינטרנט, שיתוף מחשבים וכו'.

8.3 שמור שבעל המוצר יהיה אליך במרחק סביר.

צריך שבעל המוצר יהיה קרוב מספיק על מנת שהצוות יוכל בפשטות להגיע אליו ולשאול שאלות, ועל מנת שזה האחרון יוכל גם לגשת בקלות ללוח המשימות של הצוות. אבל אסור שהוא ישב עם הצוות. למה? בגלל שהסיכוי שהוא יוכל לעצור בעצמו מ'להתערב' בפרטים שבהם עסוק הצוות הוא קטן ביותר, ובכך להעלות את הסיכון שהצוות לא "יתגבש" כראוי (כלומר: לא יגיע לניהול עצמי, היפר פרודוקטיביות וכו').

למען האמת, זו רק ספקולציה. לא ממש ראיתי מקרה בו בעל המוצר יושב עם הצוות, כך שאין לי ממש סיבה אמפירית לומר שזה רעיון גרוע. זוהי בעיקר תחושת בטן או שמועה ששמעתי מסקראם מסטרים אחרים.

8.4 שמור את המנהלים ומאמני הצוותים אליך במרחק סביר.

קצת קשה לי לכתוב על הנושא הזה מכיוון שהייתי גם מנהל וגם מאמן סקראם.

זה היה התפקיד שלי לעבוד קרוב לצוות כמה שיותר. אני הרכבתי את הצוותים, זזתי ביניהם, עשיתי Pair Programming איתם, אימנתי סקראם מסטרים, ארגנתי את הספרינטים, תכננתי ישיבות, וכו'. בראיה לאחור, רוב האנשים חשבו שזה רעיון טוב, כי היה לי ניסיון קודם בפיתוח תוכנה אגילי.

אבל במקרה שלי הייתי (ונוסיף מוסיקה של Darth Vader) גם בתפקיד מנהל הפיתוח וגם בתפקיד מנהל בפועל. שזה אומר, שכשאני נכנס לצוות, הוא אוטומטית הופך להיות פחות Self Managed.

"אויש, הבוס פה, בטח יהיה לו עכשיו הרבה מה להגיד על מה שאנחנו צריכים לעשות ואיך אנחנו צריכים לעשות את זה. ניתן לו לדבר".

דעתי היא זו: אם אתה מאמן סקראם (ואולי אף מנהל), כדאי שתכנס לצוות ותהיה מעורב כמה שיותר. אבל רק לתקופה מוגבלת, ואז רצוי שתצא ותיתן לצוות להתגבש ולהגיע למצב של ניהול עצמי (Self Managed). בדוק את הצוות מידי פעם (לא יותר מידי) על ידי כך שתגיע לישיבת של ספרינט או על ידי כך שתתבונן בלוח המשימות ותקשיב בישיבות הסקראם היומיות. אם אתה רואה מקום לשיפור, קח את הסקראם מסטר הצידה ותאמן אותו לשינוי הזה. לא מול הצוות.

רעיון טוב נוסף הוא להיות נוכח בישיבת הרטרוספקטיבה של הצוות (ראה ע"מ 65 "כיצד מבצעים רטרוספקטיבה של ספרינט"), אם הצוות סומך עליך עד כדי שהנוכחות שלך לא 'תרגיע' אותם יותר מידי. בהקשר של צוות סקראם שמתפקד היטב, כדאי לוודא שהם מקבלים את כל מה שהם צריכים, ואז להתרחק מהם כמה שיותר (חוץ משיבות ההדגמה של הספרינט).

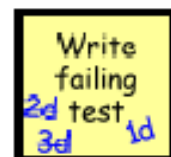
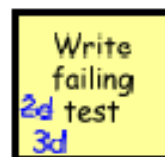
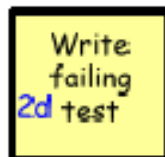
9 איך עושים את פגישת הסקראם יומית.

פגישות הסקראם היומיות שלנו, למעשה קרובות מאד להיות ישיבות "על פי הספר". הן מתחילות בדיוק באותו הזמן כל יום ובאותו המקום. בהתחלה, נהגנו לפרוש לחדר צדדי לעשות את תכנון הספרינט (אלו היו הימים בהם השתמשנו בלוח משימות אלקטרוני), אבל כעת אנו עושים את הפגישה היומית בחדר הצוות ממש אל מול לוח המשימות שלנו. אין דבר שיכול לנצח את זה.

בדרך כלל אנו עושים את הפגישות הללו כשאנו עומדים, מכיוון שזה מקטין את הסיכון של לעבור את 15 הדקות המוקצבות.

9.1 – כיצד מעדכנים את לוח המשימות ?

בדרך כלל אנו מעדכנים את לוח המשימות תוך כדי הפגישה היומית. חבר צוות מתאר מה הוא עשה אתמול ומה בכוונתו לעשות היום, תוך שהוא מושך מדבקות על הלוח. כאשר חבר צוות מתאר משימה שאינה מתוכננת, הוא שם מדבקה רלוונטית לכך על הלוח. כאשר חבר צוות מעדכן את הערכת הזמנים שלו, הוא כותב הערכת זמנים מעודכנת על המדבקה המתאימה ומוחק את ההערכה הקודמת. לפעמים הסקראם מסטר עושה את הפעילויות עם המדבקות בזמן שאחרים מדברים.



יש צוותים שקבעו כי כל חבר צוות חייב לעדכן את הלוח לפני הפגישה היומית. גם זה עובד טוב. העניין הוא פשוט להחליט על המדיניות ולדבוק בה.

חשוב לנסות להביא את כל הצוות להיות מעורב בעדכון לוח המשימות, וזאת ללא קשר למדיניות או לפורמט של הבקלוג של הספרינט. ניסינו לבצע ספרינטים שבהם הסקראם מסטר הוא המבצע הבלעדי של כל פעולות עדכון הבקלוג, ודואג לעבור בין האנשים במשך היום ולשאול לגבי עדכון הערכות הזמנים שלהם. החסרונות של השיטה הזו הם :

- הסקראם מסטר מבזבז יותר מידי זמן על אדמיניסטרציה, וזאת במקום לתמוך בצוות ולהסיר מכשולים.
- חברי הצוות לא מודעים לסטטוס של הספרינט מכיוון שלוח המשימות אינו משהו שהם צריכים לדאוג לו. מצב כזה של חוסר היזון חוזר בין חברי הצוות מקטין למעשה, את רמת הגמישות ואת רמת הפוקוס של הצוות.

רשימת מוצר שמעוצבת בצורה טובה ופשוטה, יכולה לאפשר לכל חבר צוות לעדכן את משימותיו בפשטות בעצמו.

מיד עם סיום הפגישה היומית , מישהו מחברי הצוות מסכם את כל הערכות הזמנים (מלבד המשימות שבסטטוס "הסתיים") ומיצר נקודה חדשה על לוח גרף ההתקדמות של הצוות.

9.2 – התמודדות עם מאחרים.

לחלק מהצוותים יש צנצנת מטבעות ושטרות. כאשר חבר צוות מאחר , וגם אם זה רק בדקה , הוא מוסיף סכום כסף מוסכם לצנצנת. בלי לשאול שאלות. גם אם מישהו מתקשר לפני הישיבה ומודיע על איחור , עליו עדיין לשלם.

רק במידה ויש הסבר ממש טוב לאי הגעה או איחור לישיבה , יאפשר הצוות אי תשלום. כגון : פגישה אצל רופא , חתונה שלך או משהו כזה. הכסף בצנצנת יכול לשמש למטרות חברתיות של הצוות. לקנות המבורגר בלילות שיש משחק חשוב בטלוויזיה למשל (0)

זו שיטה שעובדת טוב. אבל השיטה הזו נחוצה רק לצוותים שבהם אנשים נוטים לאחר באופן תכוף. יש צוותים שאינם זקוקים לסוג כזה של סכמת התנהגות.

9.3 – התמודדות עם " אני לא יודע מה לעשות היום".

זה בכלל לא נדיר שמישהו מחברי הצוות יגיד "אתמול עשיתי בלה בלה בלה ואין לי מושג מה אני אמור לעשות היום." (???). עכשיו מה?

נניח שגיל ויעל הם אלו שלא יודעים מה עליהם לעשות היום. ושאני הסקראם מסטר . אני אמשך הלאה לחבר הצוות הבא על מנת שידבר , אבל , אני יעיר וארשום לעצמי שיש אנשים שאין להם מה לעשות היום. אחרי שכולם סיימו לדבר , אני עובר על לוח המשימות עם כל הצוות , מלמעלה למטה ובודק שהכל מסונכרן , ושכולם יודעים מהי הכוונה בכל משימה וכ".ו.

לאחר מכן אני אומין את חברי הצוות לשים עוד מדבקות על הלוח. ואז אני פונה אל אותם חברי צוות שלא ידעו מה עליהם לעשות "עכשיו כשעברנו על כל הלוח , האם יש לכם רעיון לגבי מה אתם יכולים לעשות היום?" בתקווה , יהיה להם רעיון.

אם לא , אני שואל אם יש כאן איזשהו פוטנציאל ל Pair Programming . נניח שארז אמור היום , להטמיע את ממשק המשתמש של ה Back Office , במקרה כזה אני אציע בנימוס שאולי גיל ויעל יצטרפו אליו ל Pair Programming. בדרך כלל זה עובד.

אם גם זה לא עובד , אז ישנו הטריק הבא:
סקראם מסטר : "בסדר, מי רוצה להדגים לנו את גרסת הבטה ?" (בהנחה שזה היה היעד של הספרינט)
צוות: שתיקה מבולבלת...
סקראם מסטר : "מה, לא סיימנו עם זה?"
צוות : "מממ...לא.."
סקראם מסטר : "או, באמת? , למה לא? מה נשאר לעשות?"
צוות : "האמת , שאין לנו מחשב בדיוק להריץ עליו , והרמת הגרסה לא עובדת."

סקראם מסטר : "אה. (מוסיף שתי מדבקות על הלוח). גיל ויעל , איך תוכלו לעזור לנו היום ?"

גיל: "אמ... אני מניח שאני אנסה למצוא איזה מחשב בדיקות."
יעל : "...ואני אנסה לתקן את הבעיה שהגרסא לא עולה."

אם יתמזל מזלכם, מישהו באמת יעשה הדגמה לגרסת הבטא שביקשתם. מעולה! השגנו את יעדי הספרינט. אבל מה אם זה קורה כשאנחנו באמצע הספרינט? זה קל. תברך את הצוות על הצלחתו, משוך כמה מסיפורי המשתמש מהחלק "הבא" של הלוח , בצידו הימני התחתון, ותעביר אותם לעמודה משמאל " עדיין לא התחיל". ואז , יש לבצע שוב את הפגישה היומית ולהודיע לבעל המוצר שהוספת עוד כמה דברים לספריט.

אבל מה אם נניח כי הצוות עוד לא השיג את יעד הספרינט וגיל ויעל עדין מסרבים לבוא עם משהו מועיל לעשות? בדרך כלל , במצב כזה , אני אשקול את אחת משתי האסטרטגיות הבאות (אף אחת מהן לא ממש נעימה אבל זה באמת המוצא האחרון).

- **בושה :** "טוב , אם באמת אין לכם מה לעשות בספרינט , אני מציע שתלכו הביתה ותקראו איזה ספר או משהו. או סתם תשבו כאן עד שמישהו יקרא לכם לעזרה"
- **בית ספר בשיטה הישנה :** פשוט תן להם משימה.
- **לחץ חברתי :** " תרגישו חופשי , גיל ויעל , קחו ת'זמן ,אנחנו נעמוד פה ונחכה עד שתמצאו משהו לעשות שיעזור לנו לעמוד ביעדים של הספרינט"
- **מילצור** - "אתם יכולים לעזור היום בעקיפין לצוות על ידי זה שתהיו מלצרי היום. תביאו קפה , תעבירו הודעות לאנשים , תוציאו זבל , תבשלו צהרים , וכל דבר שנבקש היום". יתכן ותהיו מופתעים כמה מהר יעל וגיל יצליחו למצוא משהו טכני ומועיל לעשות היום (-:

במידה ויש חבר צוות שבאופן תכוף גורם לנו ללכת כל כך רחוק , סביר להניח שכדאי שניקח את חבר הצוות הזה הצידה ונעביר אותו אימון רציני . אם הבעיה עדין קיימת , כדאי שנעריך עד כמה חבר צוות זה חשוב לנו לצוות. אם הוא איננו כה חשוב , נסה להוציא אותו מהצוות שלך.

אם חבר צוות זה אכן חשוב לצוות , אז כדאי לנסות להצמיד אותו למישהו אחר בצוות שיקח על עצמו תפקיד של 'רועה' או 'מנחה'. גיל יכול להיות ארכיטקט ומפתח מעולה , אלא שהוא מעדיף שאנשים אחרים יגידו לו מה לעשות. אין בעיה . נצמיד לארז את תפקיד ה 'רועה' של גיל. או שניקח את התפקיד הזה על עצמנו. אם גיל כל כך חשוב לצוות המאמץ ישתלם. היו לנו מקרים כאלו והפתרון האחרון עובד פחות או יותר.

10 איך מבצעים ישיבת הדגמה של Sprint Demo/Review

הדגמת התוצרים בסוף הספרינט היא חלק מאד חשוב בסקראם ויש אנשים הנוטים להמעיט בערכו:

"או, האם אנו באמת חייבים לעשות הדגמה? אין הרבה דברים מעניינים להראות!"

"אין לנו מן להכין הדגמה ל %&^*"

"אין לנו זמן להגיע לישיבות הדגמה של קבוצות אחרות"

10.1 מדוע אנו מתעקשים שכל ספרינט יסתיים עם הדגמה?

להדגמת ספרינט טובה, גם אם אינה דרמטית מידי, יש אפקט מאד משמעותי.

- הצוות מקבל קרדיט על הישגיו. הם מרגישים טוב.
- אחרים לומדים מה הצוות שלך עושה.
- ההדגמות הן אירוע חברתי (או לפחות, אמורות להיות) בו הצוותים השונים מתקשרים זה עם זה סביב דברים שנעשו. זה אירוע בעל ערך.
- ההדגמות, למעשה, מאלצות את הצוות, לסיים דברים ולשחרר אותם. (גם אם זה רק לסביבת הבדיקות). ללא ההדגמות, נמשיך לקבל ערימה של דברים שגמורים רק ב 99%. בעזרת ההדגמות נקבל אומנם מעט יותר, אולם המעט הזה יהיה גמור באמת. מצב זה טוב הרבה יותר (במקרה שלנו) מלקבל ערימה של דברים שרק בערך הסתיימה וככל הנראה 'תזהם' את הספרינט הבא.

אם נכריח את הצוות לבצע הדגמה, גם אם אין לו באמת משהו שעובד, מצפה לנו הדגמה שכולה מבוכה. סביר להניח שהצוות יגמגם ויתפתל בכיסאו, ומחיאיות הכפים בסוף, בקושי ישמעו. אנשים ירחמו על הצוות, או יתרגזו שהם בזבוזו זמן על הדגמה שכזו.

זה כואב. אבל האפקט הוא כמו אפקט של תרופה מרה. בספרינט הבא, הצוות כבר ינסה להביא משהו עובד וגמור להדגמה. הם ירגישו ש "אנחנו יכולים להדגים שני דברים במקום חמישה, אבל לפחות הפעם, זה יעבוד." הצוות יודע שהוא יצטרך לבצע הדגמה, ויהי מה. מה שמגדיל באופן משמעותי את הסיכוי שהם יציגו משהו מועיל. ראיתי את זה קורה מספר פעמים.

10.2 רשימת תכולה להדגמה של ספרינט.

- תודאו שאתם מציגים בצורה ברורה את היעד של הספרינט. אם יש אנשים בקהל שלא יודעים כלום לגבי המוצר שלכם, קחו את הזמן ותארו אותו בקצרה.
- אל תשקיעו יותר מידי זמן בהכנה להדגמה, במיוחד לא במצגות נוצצות. ותרו על כל הזבל מסביב וגשו להצגה ממוקדת של קוד שפותח ועובד.
- שימרו על קצב הדגמה גבוה. התמקדו בהכנות על הדרך בה תציגו הכי מהר במקום על 'הכי יפה'.
- שדאגו שההדגמה תהיה ממוקדת מבחינה עסקית. השאירו את הפרטים הטכניים מחוץ להדגמה. התמקדו ב"מה עשינו" ולא ב"איך עשינו".
- אם זה אפשרי, תנו לקהל לנסות את המוצר בעצמם.
- אל תציגו חבילה של תיקוני באגים ויכולות מינוריות. אפשר להזכיר אותם אבל לא להדגים אותם, מכיוון שזה ייקח יותר מידי זמן ויגרור את תשומת הלב מהדברים החשובים באמת.

10.3 התמודדות עם דברים ש "לא ניתנים להדגמה"

חבר צוות : " אני לא הולך להדגים את הדבר הזה מכיוון שהוא בלתי ניתן להדגמה. סיפור המשתמש היה לעשות הרחבה למערכת על מנת שתוכל לתמוך ב 10,000 משתמשים בו זמנית. אין מצב שאני יכול להזמין עשרת אלפים אנשים להשתמש בו זמנית במערכת לצורך הדגמה. נכון?!"

סקראם מסטר : "האם סיימת את סיפור המשתמש הזה?"

חבר צוות : "כן, בטח"

סקראם מסטר : "איך אתה יודע?"

חבר צוות : " הרמתי את מערכת הבדיקות כך שהיא תתמוך בהתחלה בשמונה שרתים לצורך עומס והפצצתי אותה בבקשות משתמשים"
סקראם מסטר : אבל , האם יש לך איזושהי אינדיקציה שהמערכת תעמוד בעומס של 10,000 משתמשים?"

חבר צוות : "כן, המכונות של הבדיקות הן קצת 'עגלות' אבל הן יכלו לשאת 50 אלף משתמשים במהלך הבדיקה"

סקראם מסטר : "איך אתה יודע?"

חבר צוות (מתוסכל) : " ובכן , הפקתי דוח! אתה יכול לראות בעצמך , זה מראה כיצד נעשתה הבדיקה וכמה בקשות עברו!"

סקראם מסטר : "מעולה! אז, הנה ההדגמה שלך. רק תראה את הדוח , תעבור עליו עם הקהל. יתר טוב מכלום , לא?"

חבר צוות : " אה, וזה מספיק? אבל זה מכוער, צריך קצת ללטש את זה."

סקראם מסטר : "בסדר, אבל אל תשקיע יותר מידי זמן בזה. זה לא צריך להיות יפה אלא להעביר מסר."

11 - כיצד עושים רטרופקטיבה של ספרינט.

11.1 מדוע אנו מתעקשים שכל הצוותים יבצעו רטרופקטיבה.

הדבר הכי חשוב ברטרופקטיבה זה שהיא תתקיים.

משום מה, צוותים, לא תמיד נראים נלהבים לבצע את הטקס הזה. רוב הצוותים יעדיפו לדלג על הטקס הזה ולעבור לספרינט הבא. יתכן שזה משהו תרבותי, אני לא בטוח.

למרות כל זאת, נראה שכולם מסכימים שהרטרופקטיבה היא דבר יעיל במיוחד. למעשה, אני אציין שזה לדעתי הדבר השני בחשיבותו באירועי הסקראם (הראשון הוא ישיבת תכנון הספרינט), מכיוון שזו ההזדמנות הטובה ביותר שלנו להשתפר.

ברור, שאנחנו לא צריכים רטרופקטיבה על מנת לבוא עם רעיונות טובים. אנחנו יכולים לעשות את זה באמבטיה! האם הצוות יקבל את הרעיונות שלנו? אולי, אבל הסיכוי שכולם יהיו מחויבים לרעיון, גדול הרבה יותר, אם זה בא "מתוך הצוות". כלומר, עולה במהלך הרטרופקטיבה כאשר כולם רשאים לתרום לדיון ולרעיונות. בלי רטרופקטיבה, נמצא שהצוות ממשיך לשגות את אותם שגיאות שוב ושוב.

11.2 כיצד מארגנים רטרופקטיבה.

הפורמט משתנה לעיתים ממקום למקום, אבל בדרך כלל הוא נראה משהו כזה:

- מקצים שעה עד שלוש שעות. תלוי בכמות הדיון הצפויה.
- משתתפים: בעל המוצר, כל הצוות ואני.
- מתמקמים בחדר סגור, במקום עם ספות נעימות, על פאטיו בגג, או משהו כזה. העיקר שנוכל לקיים דיון ללא הפרעה.
- בדרך כלל לא נקיים את הישיבה בחדר הצוות, מכיוון שתשומת הלב של האנשים יכולה להיות מוסתת מהעיקר.
- נמנה משהו כמזכיר.
- הסקראם מסטר יראה את רשימת הספרינט, ובעזרת החברים יסכם את הספרינט. אירועים חשובים, החלטות וכ"ו
- נבצע את ה"סבב" הרגיל. כל אחד בתורו יגיד, ללא הפרעה, מה לדעתו היה טוב בספרינט, מה לדעתו אפשר היה לעשות טוב יותר, ומה הוא היה מציע לעשות אחרת בספרינט הבא.
- נסתכל על היכולות של הצוות בפועל בספרינט מול מה שהם העריכו שיוכלו לבצע. במידה ויש פער גדול ננסה לנתח את הגורמים לפער.
- מעט לקראת הסוף, הסקראם מסטר יסכם בצורה קונקרטית את הדברים שניתן לעשות טוב יותר בפעם הבאה.

בדרך כלל הפגישה הזו לא כל כך מובנית. למעשה המסר הוא כל הזמן אותו הדבר "מה נוכל לעשות יותר טוב בספרינט הבא".

להלן דוגמה ללוח, איך שהוא נראה אחרי פגישת רטרוספקטיבה שעשיתי לאחרונה.



שלוש עמודות:

דברים טובים: אם היינו יכולים לחזור על כל הספרינט מתחילתו, היינו עושים שוב גם את הדברים בעמודה הזו?
יכולנו לעשות יותר טוב: אם היינו יכולים לחזור על כל הספרינט מתחילתו, היינו עושים אחרת את הדברים בעמודה זו?
שיפורים: רעיונות ממשיים לגבי דברים שיש ביכולתנו לשפר בעתיד.

ניתן לראות שעמודות 1 ו 2 מסתכלות לעבר, בעוד עמודה 3 מתבוננת אל תוך העתיד.

לאחר שהצוות, בסיעור מוחות העלה את כל המדבקות על הלוח, הם השתמשו בשיטה של "הצבעת נקודות", על מנת להחליט באיזה שיפור כדאי להתמקד בספרינט הבא. כל איש צוות קיבל לידיו שלשה מגנטים ונתבקש ל להצביע בעזרתם לגבי השיפור או השיפורים שלדעתו צריכים לקבל עדיפות בספרינט הבא. כל איש צוות יכול לחלק את הנקודות שבידיו איך שבא לו, אפילו לשים את כל הקופה על נושא אחד בלבד.

בהתבסס על השיטה הזו, נבחרו חמישה תהליכים לשיפור, ואנחנו נעקוב אחריהם במהלך הספרינט.

חשוב מאד לא להיות במצב שיש יותר מידי משימות לשיפור לספרינט הבא, אלא להתמקד בדברים החשובים באמת.

11.3 – הפצת הלקחים שנלמדו בצוות.

המידע שמועלה בישיבת הרטרוספקטיבה הוא בדך כלל בעל ערך רב מאד. אם הצוות הזה מתקשה להתמקד במשימות שלו כי מנהל המכירות כל הזמן "גונב" אנשי תוכנה מומחים על מנת להשתתף בישיבות של אנשי המכירות? זו אינפורמציה חשובה ביותר. אולי צוות אחר חווה את אותה הבעיה בדיוק? אולי כדאי שנעבוד עם מנהלי המוצרים על הכרות עם המוצר על מנת שהם אלו שיוכלו ללכת לישובות המכירה ולא אנשי פיתוח?

מטרת הרטרוספקטיבה אינה רק בעניין הצוות עצמו ומה הוא יכול לעשות טוב יותר בפעם הבאה, אלא בעלת משמעות רחבה הרבה יותר מזה.

האסטרטגיה שלנו להפצת הלקחים היא מאד פשוטה. מישהו (במקרה הזה אני) משתתף בכל פגישות הרטרוספקטיבה ומתפקד כ'גשר' להעברת המידע. מאד לא פורמאלי.

דרך נוספת יכולה להיות, שכל צוות מפרסם דוח לגבי תוצאות המפגש שלו. ניסינו את הדרך הזו, אולם גילינו שאנשים לא ממש קוראים את הדוחות הללו ואת המלצותיהם ופחות מיישמים את הלקחים. לכן החלטנו על הדרך הפשוטה הנ"ל.

חוקים אשר חשובים למי שבתפקיד 'מגשר הידע':

- הוא חייב להיות מישהו שיודע להקשיב.
- אם הרטרוספקטיבה שקטה מידי, הוא חייב להיות מוכן לשאול שאלות עדינות אם כי ממוקדות ומכוונות על מנת להניע את הדיון. למשל "בהנחה ויכולתם להחזיר את הזמן לאחור, לחזור ליום הראשון של הספרינט, מה הייתם עושים אחרת?"
- עליו להשתתף בכל הפגישות של כל הצוותים.
- הוא חייב להיות מישהו בעל סמכות בארגון, על מנת שהוא יוכל לפעול בנושאים שהינם בלקחים שהם מחוץ לפוקוס של צוות מסוים בלבד.

הגישה המתוארת כאן, עובדת די יפה, יתכן שישנן גישות אחרות לעשות את אותו הדבר ואולי אפילו יותר טוב. במקרה כזה, אנא העשירו אותי.

11.4 - לשנות או לא לשנות.

נניח שהצוות הסיק ש: "אנחנו מתקשרים מעט מדי בתוך הצוות ולכן דורסים אחד לשני את התכנונים והעבודה"
מה צריך לעשות לגבי זה? להכניס ישיבות Design יומיות? לחפש כלים אחרים לשיפור תקשורת? להוסיף עוד דפי wiki? טוב, אולי זה פתרון, אבל אולי גם לא.

מצאנו כי שבמקרים רבים, עצם זיהוי הבעיה בצורה נכונה, כבר פותר אותה בספרינט הבא במיוחד אם תוצאות הרטרוספקטיבה מתפרסמות על הקיר בחדר הצוות (משהו שאנחנו תמיד שוכחים לעשות. שנתבייש לנו!).

לכל שינוי שנרצה לבצע, יש איזשהו מחיר, כך, שלפני שאנו מחליטים לשנות אולי כדאי שנשקול לא לעשות דבר ולקוות שהבעיה תיפתר מעצמה (או תיקטן).

הדוגמא הנ"ל ("אנחנו מתקשרים מעט מידי...") היא דוגמא די מאפיינת, ולפעמים עדיף לפתור את הבעיות על ידי כך שלא נעשה דבר בעניין.

אם בכל פעם שמישהו מתלונן על משהו, נציג שינוי כלשהוא, אנשים יחששו לחשוף קשיים, אפילו קטנים, וזה נורא.

11.5 – דוגמאות לדברים שיכולים לעלות בישיבות הרטרופקטיבה.

להלן מספר דוגמאות לדברים טיפוסיים שעולים בישיבת התכנון, ופעולות טיפוסיות לפתרון.

"היינו צריכים להשקיע יותר זמן בשבירת סיפורי משתמש למשימות קטנות"

זו אמירה די נפוצה. בכל יום בפגישה היומית, חברי צוות מוצאים עצמם אומרים "אני לא ממש יודע מה אני אמור לעשות היום". אז אחרי כל פגישה יומית, אנו משקעים זמן למצוא משימות נכונות להיום. יותר אפקטיבי לעשות את זה מראש. **פעולה טיפוסית לתיקון:** כלום. סביר להניח שהצוות יפתור את זה כבר בעצמו במהלך ישיבת התכנון של הספרינט הבא. במידה והבעיה חוזרת על עצמה, הגדל את זמן ישיבת תכנון הספרינט.

"יותר מידי הפרעות חיצוניות" **פעולה טיפוסית לתיקון:**

- בקש מהצוות, לספרינט הבא, להכין תוכנית שמשקפת יותר את המציאות שבה יש הפרעות.
- בקש מהצוות לתעד את ההפרעות הללו יותר טוב בספרינט הבא. מי מפריע, לכמה זמן. זה יעזור להבין את הבעיה טוב יותר ובכך יעזור לפתור אותה.
- תבקש מהצוות להפנות את כל ההפרעות בספרינט לטיפול של מנהל המוצר או הסקראם מסטר.
- תבקש מהצוות למנות משהו שהוא "שומר יעד הספרינט", כל ההפרעות יונתבו דרכו, כך ששאר הצוות יוכל להתמקד ביעדי הספרינט. יכול להיות שזה יהיה הסקראם מסטר או רוטציה כלשהיא בצוות.

"התחייבנו ליותר מדי דברים והספקנו לסיים רק חצי"

פעולה טיפוסית לתיקון: כלום. סביר להניח שהצוות כבר לא יתחייב בספרינט הבא על יותר ממה שהוא מסוגל. או לפחות לא יתחייב יותר בפער כל כך גדול.

"יש יותר מידי רעש ובאלאגאן במשרד" **פעולה טיפוסית לתיקון:**

- תנסה ליצר סביבת עבודה טובה יותר. קח את הצוות מחוץ למשרד, תשכור חדר במלון. ראה ע"מ **47** "כיצד מארגנים את סביבת העבודה של הצוות".
- אם זה אינו אפשרי, אפשר לצוות להקטין את אלמנט הפוקוס שלהם בספרינט הבא, ולציין במפורש שזה בגלל הסביבה הרועשת והמבולגנת שהם חיים בה. בתקווה, שזה יגרום למנהלי מוצר ללחוץ על ההנהלה הבכירה בעניין.

למזלי, מעולם לא הייתי צריך לאיים שאני אזיז את הצוות מחוץ למשרד, אבל הייתי עושה זאת אם הייתי חייב. ☺

12. זמן הפוגה בין ספרינטים

בחיים האמיתיים, אתה לא יכול כל הזמן לרוץ בספרינט, אתה צריך מנוחה בין הספרינטים. אם אתה כל הזמן תעשה ספרינטים, אתה למעשה עושה ג'וגינג.

אותו הדבר קורה בסקראם ופיתוח תוכנה בצורה כללית. ספרינטים הם די אינטנסיביים. כמפתח אין לך באמת זמן הפוגה. כל יום אתה צריך לעמוד בישיבה היומית ולספר מה הצלחת להשיג אתמול. מעט מאוד אנשים יגידו שהם העבירו את היום עם רגליים על השולחן, הסתכלות בבלוגים ולגימת קפוצ'ינו.

בנוסף למנוחה עצמה, ישנה עוד סיבה טובה לבצע הפוגה בין הספרינטים. אחרי ישיבת ההדגמה וישיבת התכנון, גם הצוות וגם בעל המוצר יהיו עמוסים במידע ורעיונות לעכל. אם הם יעברו מיידית לתכנן את הספרינט הבא, הסיכוי שאף אחד לא באמת יעכל באמת את הרעיונות, יגדל.

לכן אג'נדה כזו היא די רעה:

09:00-10:00 – ישיבת תכנון

10:00-11:00 – ישיבת רטרוספקטיבה

13:00-16:00 – ישיבת תכנון

אני אנסה להציג לכם אפשרות של זמן הפוגה לפני הספרינט החדש (ספציפית, הזמן אחרי ישיבת הרטרוספקטיבה ולפני ישיבת התכנון). רק שתדעו, אנחנו לא תמיד מצליחים לעשות את זה.

לפחות אנחנו מנסים שישבת הרטרוספקטיבה וישיבת התכנון לא יקרו באותו יום. כל אחד צריך שיהיה לו לפחות לילה אחד נטול ספרינטים לפני תחילת ספרינט חדש.

אג'נדה טובה יותר:

יום א':

09:00-10:00 – ישיבת תכנון

10:00-11:00 – ישיבת רטרוספקטיבה

יום ב':

09:00-15:00 – ישיבת תכנון

אג'נדה אפילו יותר טובה:

יום ה':

09:00-10:00 – ישיבת תכנון

10:00-11:00 – ישיבת רטרוספקטיבה

יום שישי:

יום שבת:

יום א':

09:00-15:00 – ישיבת תכנון

דרך אחת לעשות את "ימי המעבדה" (או איך שתרצו אתם לקרוא להם), כלומר הימים שבהם המפתחים יכולים לעשות למעשה כל דבר שהם חושבים שנחוץ (אוקי, אני מודה שאני שואב את הרעיון מגוגל). למשל, לקרוא על הכלים ו APIs האחרונים, ללמוד להסמכות מקצועיות, לשוחח על דברים גיקיים עם החברים לעבודה, לכתוב פרוייקט "הובי" מהצד, וכו'.

המטרה שלנו לבצע ימי מעבדה בין ספרינטים. בצורה כזו אתה מקבל מנוחה טבעית בין ספרינטים, ואתה מקבל צוות מפתחים שמקבל הזדמנות אמיתית לשפר את הידע שלהם ולשמור אותו מעודכן. ובנוסף זוהי צורת עבודה אטרקטיבית לעובד.

אג'נדה הכי טובה?

יום ד':

09:00-10:00 – ישיבת תכנון

10:00-11:00 – ישיבת רטרוספקטיבה

יום ה': יום מעבדה

יום שישי:

יום שבת:

יום א':

09:00-15:00 – ישיבת תכנון

כרגע יש לנו ימי מעבדה פעם בחודש. יום החמישי האחרון בכל חודש ליתר דיוק. למה לא בין ספרינטים? ובכן, בגלל שחשבתי שזה יותר חשוב שכל החברה תיקח יום מעבדה באותו הזמן. אחרת, אנשים לא יקחו את זה ברצינות. ומכיוון שאין לנו (עד כה) ספרינטים מסונכרנים לכל הפרויקטים שלנו, הייתי צריך לבחור באפשרות הזו.

אולי ביום מן הימים, ננסה לסנכרן את הספרינטים לכל הפרויקטים שלנו (כלומר התחלת הספרינטים מתחילה באותו היום לכל הפרויקטים והצוותים). במקרה הזה אנחנו בצורה ודאית נמקם את יום המעבדה בין הספרינטים.

13. איך אנחנו מתמודדים עם תכנון הגרסא וחוזים

קבועי מחירי

לפעמים אנחנו צריכים לתכנן יותר מאשר רק הספרינט הבא. בדרך כלל יש מתאם בין הצורך לתכנן בצורה כזו וחוזים קבועי מחיר (קבלנות) כדי שנמזער את הסיכון לא להוציא את הגרסא לא בזמן. בדרך כלל תכנון גרסא זה בעצם לענות על השאלות "מתי, לכל המאוחר, נוכל להוציא את גרסא 1.0 של המערכת החדשה." אם אתם באמת רוצים ללמוד על תכנון גרסא, אני מציע שתדלגו על הפרק הזה ובמקום זה תקנו את הספר של מייק כהן "הערכות ותכנונים באג'יל"*

הלוואי והייתי קורא את הספר הזה קודם (קראתי אותו אחרי שכבר המצאנו בעצמנו את מה שנכתב). השיטה שלי לתכנון גרסא יחסית פשוטה אבל מהווה רק נקודת התחלה.

13.1 הגדר את סיפי הקבלה

בנוסף לרשימת המוצר הרגילה, בעל המוצר מגדיר גם סיפי קבלה שהם תיעדופים פשוטים של הפריטים ברמה העסקית ובהקשר של החוזה. הנה דוגמא של חוקי סיפי קבלה:

- כל הפריטים עם ציון תיעדוף גדול או שווה ל 100 חייבים להיות כלולים בגרסא 1.0 אחרת אנחנו נקנס למוות.

* באנגלית "Agile Estimating and Planning". מופיע גם בסוף הספר תצלום של הספר

- כל הפריטים עם ציון תיעדוף בין 55 ל 99 חייבים להיות כלולים בגרסא 1.0 אבל אנחנו יכולים להניח שניתן יהיה להשלים אותם בגרסאות מאוחרות יותר.
- כל הפריטים עם ציון תיעדוף בין 25 ל 49 מחוייבים אבל יכולים להיות מבוצעים גם בגרסא 1.0
- כל הפריטים עם ציון תיעדוף הקטנים מ 25 הם ספקולטיביים ועלולים לא להיות נחוצים בכלל.

הנה דוגמא לרשימת מוצר:

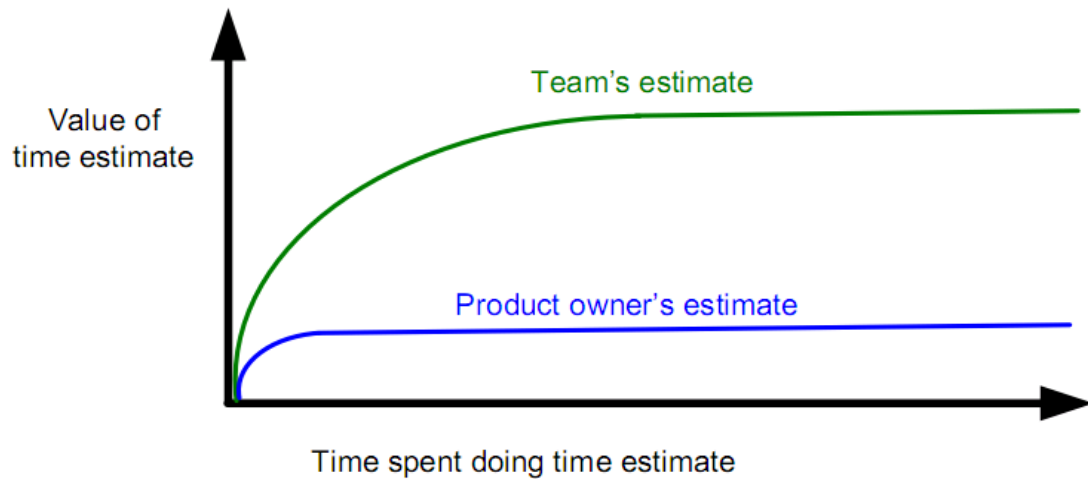
חשיבות	שם
130	בננה
120	תפוח
115	תפוז
110	גויאבה
100	אפרסק
95	צימוקים
80	אגוזים
70	סופגניות
60	בצל
40	מלון
35	פפאיה
10	חמוציות
10	חציל

אדום – חייב להיכלל בגרסא 1.0
צהוב – כדאי שיהיה כלול בגרסא 1.0
ירוק – יכול להתבצע יותר מאוחר
ובכן אם אנחנו מבצעים את כל האדומים והצהובים, אנחנו בטוחים. אם הזמן לא מאפשר לנו נוותר על הצהובים, כל השאר הוא בונוס.

13.2 הערכות זמנים של הפריטים החשובים ביותר

כדי לבצע תכנון גרסא, בעל המוצר צריך הערכות זמנים, לפחות לכל הסיפורים שנמצאים בחוזה. כמו תכנון הספרינט גם כאן יש שיתוף מאמצים בין בעל המוצר והצוות. הצוות מעריך ובעל המוצר מתאר את הפריטים ועונה על שאלות. הערכות זמנים הן בעלות ערך אם בסופו של דבר הן קרובות למציאות. אם הן, נניח, שגויות ב-30% הן בעלות ערך מופחת. הן חסרות ערך אם אין להן קשר למציאות.

הנה המתאם בין הגורמים השונים בראייה שלי:



כל זה היה בעצם להגיד באריכות את הדברים הבאים:
 - תנו לצוות לתת את ההערכות

- אל תתנו להם להשקיע יותר מדי זמן בזה

- וודאו שהם מבינים שההערכות הן הערכות גסות ולא מחייבות.

בדרך כלל בעל המוצר אוסף את כל הצוות בחדר, מביא כמה חטיפים ואומר להם שהמשימה של הישיבה היא להעריך את 20 (או מה שזה לא יהיה) הסיפורים החשובים ביותר ברשימת המוצר (הסיפורים העליונים). הוא עובר על כל סיפור וסיפור ואז נותן לצוות לעבוד. בעל המוצר נשאר בחדר לענות על שאלות והבהרות לגבי כל פריט אם צריך. כמו בתכנון ספרינט, נושא ה"איך להציג" הוא מאוד שימושי כדי למזער את הסיכון בחוסר הבנה. הישיבה צריכה להיות תחומה בזמן, אחרת תשקיעו יותר מדי זמן בהערכת של פחות מדי סיפורים.

אם בעל המוצר רוצה להשקיע יותר זמן, כל מה שהוא צריך לעשות זה לקבוע ישיבה נוספת מאוחר יותר. הצוות צריך לקחת בחשבון את הזמן ש"נלקח" מהם בתכנון הספרינט ולהחזין זאת בצורה מאוד ברורה לבעל המוצר, כך שהוא יבין שזה לא בא בחינם אלא על חשבון משהו אחר. הנה דוגמא לצורת התוצאה של הערכת זמנים. (בנקודות סיפור):

הערכה	שם	חשיבות
12	בננה	130
9	תפוח	120
20	תפוח	115
8	גויאבה	110
20	אפרסק	100
12	צימוקים	95
10	אגוזים	80
8	סופגניות	70
10	בצל	60
14	מלון	40
4	פפאיה	35
	חמוציות	10

	חציל	10
--	------	----

13.3 הערכת וקטור מהירות

אוקי, אז עכשיו יש לנו הערכה גסה כלשהי לגבי רוב הסיפורים החשובים. הצעד הבא הוא להעריך מהו סדר גודל המהירות הממוצעת לספרינט. זה אומר שאנחנו צריכים להחליט על פקטור המיקוד שלנו. תסתכלו בעמוד XX "איך הצוות מחליט אילו סיפורים להכניס לספרינט"

פקטור המיקוד הוא בבסיסו "כמה זמן הצוות משקיע בהתמקדות על הסיפורים שהתחייבו עליהם". זה אף פעם לא 100% מכיוון שהצוות מפסיד זמן בעשיית דברים שלא תוכננו, מעבר ממשימה אחת לשנייה, עזרה לצוותים אחרים, אימיילים, תיקון המחשבים השבורים שלהם, ויכוחים פוליטיים במטבח וכו'. בוא נניח פקטור מיקוד של 50% (די נמוך, בדרך כלל אנחנו מגדירים משהו סביב ה 70%). ובוא נאמר שאורך הספרינט שלנו הוא 3 שבועות (15 ימים) ושהצוות מורכב משישה אנשים.

אי לכך, כל ספרינט שווה ל-90 ימי עבודה, אבל ניתן לצפות ל-45 ימי עבודה זמינים לעבודה על סיפורים (לאור פקטור המיקוד של 50%). ובכן, הערכת כוון המהירות שלנו היא 45 נקודות סיפור. אם לכל סיפור היתה הערכה של 5 ימים (מה שאינו נכון), אז הצוות יפתח כ-9 סיפורים לספרינט.

13.4 בתוך הרכבת תכנית הגרסא

עכשיו כשיש לנו הערכות זמנים ווקטור מהירות, אנחנו יכולים בקלות לחלק את רשימת המוצר לספרינטים:

חשיבות	שם	הערכה
ספרינט 1		
130	בונה	12
120	תפוח	9
115	תפוח	20
ספרינט 2		
110	גויאבה	8
100	אפרסק	20
95	צימוקים	12
ספרינט 3		
80	אגוזים	10
70	סופגניות	8
60	בצל	10
40	מלון	14
ספרינט 4		
35	פפאיה	4
10	חמוציות	
10	חציל	

כל ספרינט מכיל את כמות הסיפורים הגדולה ביותר האפשרית ללא גלישה מגבול וקטור המהירות של 45.

עכשיו ניתן לראות שנצטרך כנראה 3 ספרינטים לסיים את כל "מה שחייבים" ו"צריכים".

3 ספרינטים = 9 שבועות = חודשיים. ובכן, האם זהו התאריך שהבטחנו ללקוח? תלוי לגמרי בטבעו של החוזה, עד כמה התכולה סופית וכו'. אנחנו בדרך כלל מוסיפים buffer גדול כדי להגן על הערכות זמנים שגויות, בעיות בלתי מתוכננות, מאפיינים בלתי צפויים וכו'. לכן, במקרה הזה נוכל להסכים שתאריך שחרור הגרסה הוא בעוד 3 חודשים, שנותן לנו חודש כעתודה. הדבר היפה הוא שאנחנו יכולים להציג ללקוח משהו שימושי כל 3 שבועות ולהזמין אותו לשנות את הדרישות לאורך הדרך (בתלות בחוזה כמובן).

13.5 התאמת תוכנית הגרסה

המציאות לא תתאים את עצמה לתוכנית לכן כנראה יקרה ההפך. אחרי כל ספרינט אנו בודקים את וקטור המהירות של הספרינט. אם מסתבר שזה ממש שונה מהערכת וקטור המהירות, אנחנו משפרים את הערכת וקטור המהירות לספרינטים עתידיים ומעדכנים את תוכנית הגרסה. אם כתוצאה משינוי ההערכה אנחנו בבעיה מול הלקוח, בעל המוצר יכול לנהל משא ומתן עם הלקוח או להתחיל לבדוק אם אפשר להוריד תכולה בלי להפר את החוזה. אולי ייתכן שבעל המוצר או הצוות יציעו רעיון לשיפור וקטור המהירות או פקטור המיקוד על ידי הסרת מכשולים עיקריים שנתגלו בספרינט.

בעל המוצר יכול ליצור קשר עם הלקוח ולהגיד לו "הי, אנחנו קצת מאחרים בלוחות הזמנים אבל אני מאמין שאם נוריד את "הפקמן המשובץ" שלוקח לנו המון זמן לפתח, נוכל לעמוד בזמן. נוכל להוסיף אותו שלושה שבועות אחרי שחרור הגרסה שאנחנו עובדים עליה כרגע אם זה בסדר מבחינתך". ייתכן שאלה לא חדשות טובות עבור הלקוח אבל לפחות אנחנו כנים ונותנים ללקוח את אפשרות החלטה מספיק מוקדם – האם אנחנו מספקים את החומר החשוב ביותר בזמן או מספקים הכל אבל מאוחר. בדרך כלל לא החלטה קשה (0:

14 איך משלבים סקראם ו XP

להגיד שסקראם ו XP (eXtreme Programming) יכולים להפרות אחד את השני זה לא ממש אמירה שמתנגדים לה. רוב החומר שאני רואה ברשת תומך בהנחה הזו ולכן אני לא אשקיע זמן בלתאר מדוע זה כך.

בכל זאת אזכיר דבר אחד, סקראם מתמקד בהנהלה ופרקטיקה ארגונית בעוד ש XP מתמקד בפרקטיקה תכנותית. זוהי הסיבה שהם מותאמים אחד לשני בצורה נפלאה – הם פונים לאזורים שונים ומשלימים. בזאת אני מוסיף את קולי לכל הקולות הקיימים ולעדויות שסקראם ו XP עובדים ביחד!

אדגיש כמה מהתכונות היותר שימושיות של XP ואיך הן מיושמות ביום יום. לא כל הצוותים אצלנו הצליחו להטמיע את כולן אבל ככלל ניתן לומר שהטמענו את כל ההיבטים של XP/סקראם. חלק מהיישומים הנזכרים ב XP מיושמים גם בסקראם ולכן יש חפיפה למשל "צוות כולו יושב ביחד", "סיפורים", "משחק תכנון". במקרים הללו פשוט דבקנו בסקראם.

14.1 תכנות זוגי

התחלנו לעשות זאת באחד הצוותים שלנו. עובד די טוב למעשה. רוב הצוותים לא מבצעים זאת אבל בראותי את הצוות שכן מבצע זאת, אני די שואף לאמן את שאר הצוותים האפשרות כדי שייתנו הזדמנות לתכנות זוגי.

כמה מסקנות לגבי תכנות זוגי:

- תכנות זוגי אכן משפר את איכות הקוד
- תכנות זוגי אכן משפר את מיקוד הצוות (למשל כאשר הבחור שמאחוריך אומר "הי, זה באמת נחוץ לספרינט?").
- באופן מפתיע הרבה מהתוכניתנים שנגד תכנות זוגי, לא באמת ניסו את זה ואחרי שמכירים את זה ומתנסים בזה אוהבים את זה.
- תכנות זוגי מתיש ולא כדאי לעשות אותו כל היום
- זה טוב מאוד להחליף זוגות בתכיפות
- תכנות זוגי משפר את התפשטות הידע בקבוצה. מהירה במהירות מפתיעה.
- חלק מהאנשים פשוט לא מרגישים בנוחות עם תכנות זוגי. אל תזרקו תוכניתנים מעולים רק בגלל זה.
- מעבר על קוד (code review) הוא טוב מאוד כאלטרנטיבה לתכנות זוגי.
- גם "נווט" (זה שלא משתמש במקלדת) צריך שיהיה מחשב משלו. לא בשביל פיתוח אלא בשביל פעילויות נחוצות כגון מעבר על מסמכים כאשר ה"נהג" (זה שמשתמש במקלדת) נתקע.
- אל תכריחו אנשים לעשות תכנות זוגי. עודדו אותם והציעו את הכלים המתאימים אבל תנו להם להתנסות בזה בקצב שלהם.

14.2 תכנות מונחה בדיקות

אמן! זה יותר חשוב בעיניי מסקראם וXP ביחד. אתם יכולים לקחת את הבית שלי עם הטלויזיה והכלב אבל אל תעצרו אותי מלעשות תכנות מונחה בדיקות! אם אתם לא אוהבים תכנות מונחה בדיקות, אל תתנו לי להיות בבנין, בגלל שאני אנסה להכניס את זה בדרך כזו או אחרת ☺
הנה סיכום ה-10 שניות שלי בנושא:

תכנות מונחה בדיקות אומר שאתה כותב בדיקה אוטומטית, כותב מספיק קוד שהבדיקה תצליח ואז משפר את הקוד כך שיהיה יותר קריא, מסיר כפילויות. וחוזר חלילה.

קצת על תכנות מונחה בדיקות (תמ"ב)

- תמ"ב קשה לביצוע. לוקח זמן מה למפתח להפנים תמ"ב. למעשה, במקרים רבים זה לא משנה כמה אתה משקיע בללמד, להדריך או להראות – בדרך כלל הדרך היחידה היא לצרף מפתח למישהו שמצויין בתמ"ב. מצד שני, כשמפתח כבר קלט, הוירוס הזה ידבק בו והוא לא ירצה לעבוד בדרך אחרת.

- לתמ"ב יש תרומה מאוד משמעותית לעיצוב מערכת.

- לוקח זמן להעלות ולהריץ תמ"ב בצורה יעילה בתחילת מוצר, במיוחד באינטגרציה בדיקות של "קופסא שחורה", אבל החזר ההוצאות הוא מהיר.

- ודא שאתה משקיע את הזמן המספק לכתוב בדיקת בצורה קלה. זה אומר, להשתמש בכלים הנכונים, להשקיע בלימוד האנשים וכו'.

אנחנו משתמשים בכלים הללו לתכנות מונחה בדיקות:

· Selenium ואת TestNG / httpUnit / jUnit ואנחנו שוקלים ב-

· HSQLDB לבדיקות כמסד נתונים מובנה בזכרון לבדיקות

· Jetty כ- web container מובנה בזכרון לבדיקות

· Cobertura למדידת כיסוי בדיקות ו

ברוב המוצרים שלנו שהם מורכבים (מבחינת תמ"ב), יש לנו בדיקות אוטומטיות. הבדיקות מאתחלות את כל המערכת בזיכרון, כולל בסיס המידע, שרתי הרשת ונכנסות למערכת רק דרך הממשקים החיצוניים שלה (למשל HTTP).

זה מאפשר למחזור החיים של פיתוח-בניה-בדיקה להיות מאוד מהיר. זה גם נותן רשת בטחון למפתחים שרוצים לבנות מחדש קוד ולכן העיצוב נשאר נקי ופשוט כשהמערכת גדלה.

תמ"ב על קוד חדש

אנחנו מבצעים תמ"ב בכל פיתוח חדש, אפילו אם זה אומר שאיתחול פרוייקט יקח יותר זמן (מכוון שאולי נצטרך כלים אחרים לבדיקות). זה נעשה בצורה כל כך מובנת מאליה מכוון שאנחנו יודעים כמה הדבר נחוץ ולכן אין תירוץ לא להשתמש בתמ"ב.

תמ"ב בקוד ישן

תמ"ב קשה, אבל לעשות תמ"ב על קוד שלא פותח בעזרת תמ"ב מההתחלה.. ממש קשה! למה? למעשה אני יכול לכתוב עמודים שלמים על הנושא הזה, אז

אני מציע שאעצור כאן. אני אשמור את זה לספר הבא שלי: "תמ"ב מהשוחות" (0).

השקענו זמן די רב בבדיקות אוטומטיות אינטגרטיביות באחת מהמערכות היותר מורכבות שלנו, קטע קוד שלא השתמשו בו זמן מה ושהיה במצב גרוע וסבוך וללא בדיקות.

לכל גרסא של המערכת היה לנו צוות שהתמקד בבדיקות וביצע סט של בדיקות מורכבות של רגרסיות ופרוגרסיות. הרגרסיות ברוב המקרים היו יותר ידניות. מה שהאט באופן משמעותי את הפיתוח ואת מחזור חיי הגרסא. היעד שלנו היה להפוך את הבדיקות הללו לאוטומטיות. אחרי שהטחנו את הראשים שלנו בקיר כמה חודשים, לא הגענו יותר קרוב למטרה.

אחרי זה, שינינו גישה, הנחנו שאנחנו תקועים עם בדיקות רגרסיה ידניות והתחלנו לשאול את עצמנו "איך אנחנו יכולים לחסוך זמן בביצוע הבדיקות הידניות?". זו היתה מערכת של משחקים אז הבנו שהרבה מזמן הבודק מושקע בהרמת הסביבה, למשל הגדרת סוגי תחרויות או המתנה לתיזמון תחרות. לכן פיתחנו כלי שיעשה את זה. סקריפט קטן, פשוט וזמין שעושה את כל העבודה השחורה ושמשאירה לבודק רק את העבודה הממשית. המאמץ ממש השתלם לנו! למעשה, זה מה שהיינו צריכים לעשות מלכתחילה. היינו כל כך ממוקדים בלעבור לבדיקות אוטומטיות ששכחנו לעשות את זה עקב בצד אגודל, כשהצעד הראשון היה ליצור דברים שיגרמו לבדיקות הידניות להיות יותר יעילות.

מוסר השכל: אם אתה תקוע עם יותר מדי בדיקות רגרסיה ידניות ורוצה לעבור לבדיקות אוטומטיות, אל (אלא אם כן זה ממש קל) תעשה את זה. בנה סביבה שתהפוך את הבדיקות הידניות יותר קלות. או אז, שקול למכן את הבדיקות.

14.3 עיצוב מתווסף

המשמעות היא לשמור על העיצוב פשוט מההתחלה ובאופן מתמיד לשפר אותו במקום לנסות לתפוס את הכל מההתחלה ואז להקפיא. אנחנו פחות או יותר עושים את זה, כלומר אנחנו משקיעים זמן סביר ב Refactoring ושיפור העיצוב הקיים. אנחנו כמעט שלא משקיעים זמן בעיצוב כולל מלכתחילה. לפעמים אנחנו טועים כמובן. למשל לתת לעיצוב ארעי "להתגנב" בצורה חזקה למוצר ואז תהליך Refactoring הופך להיות פרוייקט גדול. אבל באופן כללי אנחנו די מרוצים. עיצוב מתמשך ומשתפר הוא בעצם, תופעת לוואי אוטומטית של תכנות מונחה בבדיקות.

14.4 אינטגרציה מתמשכת

לרוב המוצרים שלנו יש אינטגרציה מתמשכת די מתוחכמת שמבוססת על QuickBuild ו Maven. זה שימושי מאוד וחוסך זמן משמעותי. זה מהווה פתרון אולטימטיבי למשפט המוכר והאהוב "הי, אבל זה עבד על המחשב שלי". שרת האינטגרציה משמש כ"שופט" או סימוכין לרמת של הקוד שלנו. כל פעם שמישהו מכניס קוד כלשהו לסביבת בקרת התצורה שלנו, השרת מתעורר, בונה את הכל מההתחלה על שרת משותף ומריץ את כל הבדיקות. אם משהו לא עובד כשורה הוא ישלח אימייל לכל הצוות שיש כשלון בניה תוך כדי תיאור המיקום שבו הקוד שונה, הבדיקה שנכשלה וכו'. כל לילה השרת בונה מחדש את הפרוייקט מההתחלה ומוציא את כל התוצרים הדרושים, מסמכים, דוחות בדיקה, דוחות אחוזי בדיקה, דוחות תלויות, וכו' לפורטל המסמכים הפרטי שלנו. חלק מהמוצרים יותקנו בצורה אוטומטית

בסביבת הבדיקה שלנו. להגיע למצב הזה לקח המון זמן אבל זה היה שווה כל דקה שהשקענו.

14.5 בעלות שיתופית על קוד

אנחנו מאוד מעודדים בעלות שיתופים על קוד אבל לא כל הצוותים אימצו את זה עדיין. מצאנו שתכנות זוגי עם תחלופה תכופה של הזוגות מובילה לרמה גבוהה מאוד של בעלות משותפת על הקוד. צוותים כאלו מוכיחים את עצמם כחזקים מאוד, לדוגמא, הספרינט שלהם לא פתאום נגזז אם אחד מאנשי המפתח חולה.

14.6 שטח העבודה מבוסס מידע

לכל הצוותים יש גישה ללוחות מחיקים וקירות ריקים והם עושים בהם שימושים טובים מאוד. ברוב החדרים תמצאו על הקירות דפים תלויים עם כל מיני סוגי מידע על המוצר ועל הפרוייקט. הבעיה הגדולה ביותר היא כמות הזבל שנערמת על הקירות. ייתכן שנגדיר מישור שיהיה אחראי לנקות מכל צוות. אנחנו מעודדים שימוש בלוחות מחיקים אבל לא כל הצוותים משתמשים בהם. עברו לעמוד **xx** "איך אנחנו מארגנים את חדר הצוות".

14.7 תקן קוד

לאחרונה התחלנו להגדיר תקני קוד. מאוד שימושי. הלוואי והיינו עושים זאת יותר מוקדם. זה לוקח כמעט אפס זמן. פשוט תתחילו בקטן ותגדלו לאט לאט. תכתבו רק דברים שהם לא מובנים מאליהם לכולם ותקשרו אותם לדברים הרלוונטיים כשזה אפשרי. לרוב התוכניתנים יש סגנון תכנות ייחודי להם. פרטים קטנים כמו איך להתמודד עם הודעות שגיאה של הקוד עצמו, איך הם מתעדים את הקוד, מתי הם מחזירים NULL, וכו'. בחלק מהמקרים ההבדל לא משנה, במקרים אחרים זה יכול להוביל לחוסר תאימות בעיצוב וחוסר קריאות של הקוד. תקן של קוד הוא מאוד שימושי במקרה כזה, כל עוד אתם מתמקדים בדברים שחשובים באמת. הנה כמה דוגמאות מתקן הקוד שלנו:

- אתה יכול לשבור כל אחד מהחוקים אבל ודא שיש סיבה טובה לכך ותעד את זה
- כברירת מחדל, השתמש בתקן של Sun:
<http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html>
- לעולם, לעולם, לעולם אל תתפוס Exception בלי לעשות Login ל Stack Trace או Rethrowing. להשתמש ב(log.debug) זה בסדר אבל אל תאבד את ה Stack trace.
- הימנע משימוש בראשי תיבות. שימוש בראשי תיבות ידועים כמו DAO זה בסדר.
- מתודות שמחזירות מערכים חייבות לא להחזיר Null. החזירו מערכים ריקים במקום להחזיר Null

14.8 מקצב אחיד / עבודה אנרגטית

המון ספרים בנושא פתוח תוכנה באג'ייל טוענים שהערכת זמן העבודה לשעות נוספות פוגעת בפרודוקטיביות.

אחרי כמה ניסיונות שהוכרחתי לעשות, לא נותר לי אלא להסכים בצורה גורפת! לפני כשנה אחד הצוותים (הגדול ביותר) עבד שעות מטורפות.

איכות הקוד שהם ייצרו היתה מתחת לכל ביקורת והם היו צריכים להשקיע המון זמן בכבוי ה-"אש" שנוצרה. לצוות הבדיקות (שגם עשה שעות נוספות) לא היתה האפשרות הממשית לבצע בדיקות איכותיות. המשתמשים שלנו התעצבנו עלינו והעיתונים אכלו אותנו חיים.

אחרי כמה חודשים הצלחנו להוריד את כמות השעות אצל העובדים לרמה סבירה. אנשים עבדו שעות נורמאליות (חוץ מהפעמים שהפרויקט קרס). ולהפתעתי, הפרודוקטיביות והאיכות עלו פלאים.

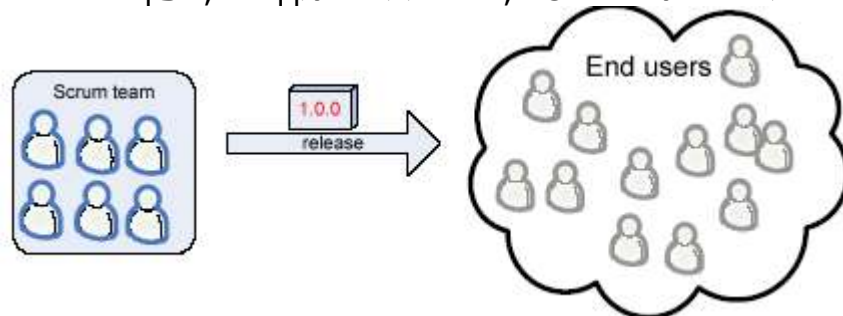
כמובן, הורדת כמות השעות לא היתה הדבר היחידי שגרם לשיפור, אבל לכולם היה ברור שהירידה היוותה סיבה עיקרית לכך.

15. איך עושים בדיקות

זה החלק הקשה ביותר. אני לא בטוח אם זה החלק הקשה ביותר ב-סקראם או שזה פשוט החלק הקשה ביותר בפיתוח תוכנה. כנראה שבדיקות זהו החלק שמשתנה ביותר בין ארגונים. תלוי במספר הבודקים שיש בצוות, כמה בדיקות אוטומטיות יש, מבנה המערכת (רק Server ו אפליקצית WEB או תוכנה ארוזה בקופסא), תדירות שחרור גירסאות למשתמשים, האם התוכנה מסופקת ללקוחות ארוזה, האם המערכת היא מערכת קריטית (Blog server לעומת Flight control) וכו... התנסינו רבות באופן שיש לבצע בדיקות בסקראם. אני אנסה לתאר את מה שעשינו ואת מה שלמדנו על כך עד כה.

15.1 אתה כנראה לא מצליח להפטר משלב בדיקות הקבלה

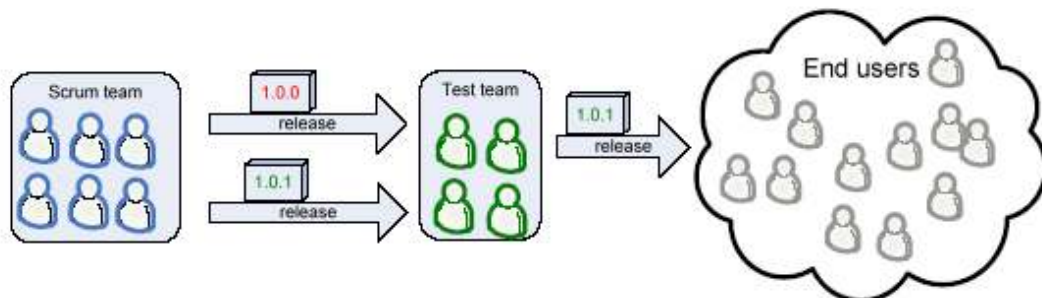
בעולם סקראם אידיאלי, תוצאות ספרינט הן פוטנציאלית מוכנות לשחרור והתקנה אצל המשתמשים. טוב, אז יאללה נתקין זהו, נכון ?



זהו שלא!

מנסיוננו זה כנראה לא יעבוד. תהליך כזה עלול ליצור תקלות מכוערות. במידה ואיכות זה דבר חשוב עבורך תהליך כלשהוא של בדיקות קבלה ידניות מחוייב המציאות.

זהו תהליך שבו צוות של בודקים ייעודי, אשר אינם חלק מצוות ה סקראם "מתעללים" במערכת עם בדיקות שהצוות לא יכול היה לחשוב עליהם או שלא היה להם את הזמן לבצע או את החומרה המתאימה. הבודקים במסגרת תהליך זה ייגשו לתוכנה באופן שהמשתמשים ניגשים אליה מה שאומר שהבדיקות יעשו בצורה ידנית (בהנחה שהמערכת בנויה למשתמשים אנושיים).



צוות הבדיקות ימצא תקלות, צוות הסקראם ייאלץ לתקן את התקלות ולהוציא גירסא מתוקנת של התוכנה. ואז במוקדם או במאוחר (עדיף כמה שיותר מוקדם) יתאפשר שחרור גרסא מתוקנת למשתמשים - 1.0.1 במקום הגירסא הלא יציבה 1.0.0

כאשר אני מתייחס לתהליך בדיקות הקבלה, אני מתכוון לכל התקופה של הבדיקות והתקונים וייצור גירסאות חדשות עד שיש ביד גירסא מספיק טובה לשיחרור למשתמשים.

15.2 צמצם את שלב בדיקות הקבלה

שלב זה של בדיקות הקבלה הוא שלב כואב. יש בו תחושה לא אג'ילית. למרות שלא נוכל להפתר משלב זה אנחנו יכולים לצמצמו ואכן מצליחים.. באופן יותר ספציפי אנחנו מקצרים את משך הזמן של השלב הזה. ניתן לעשות זאת ע"י:

- שיפור למקסימום אפשרי את איכות הקוד המועברת ע"י צוות הסקראם
- ייעול תהליך הבדיקות הידניות. לדוגמא שמוש בבודקים הטובים ביותר, שמוש בכלים טובים יותר, דווח על מבזבי זמן שניתן לעשות להם אוטומציה.

אבל איך משפרים למקסימום את איכות הקוד המועברת ע"י צוות הסקראם? יש הרבה דרכים, הינה שתי דרכים שמצאנו שעובדות מצויין:

- לשים בודקים בצוות הסקראם
- לעשות פחות בכל ספרינט

15.3 שיפור האיכות ע"י השמת בודקים בצוות הסקראם



כן, אני שומע את הטענות:

- "אבל זה ברור מעליו, צוות הסקראם אמור להיות רב תחומי"
- "צוות הסקראם אמורים להיות ללא הגדרה מראש של תפקידים"

תנו לי להבהיר. הכוונה שלי כאשר אני אומר בודק במיקרה זה היא: מישהו שהכישורים המרכזיים שלו הם בדיקות ולא מישהו שתפקידו לבצע רק בדיקות. מפתחים הם לרוב בודקים גרועים במיוחד כשמדובר בקוד שניכתב על ידם.

הבודק הוא החותם הסופי

בנוסף להיותו "רק" חבר בצוות, לבודק יש אחריות חשובה. הוא החותם הסופי. שום דבר אינו נחשב גמור (Done) בספרינט עד אשר הוא מאשר שזה אכן מוכן. מצאתי שמפתחים אומרים על תכנה שהיא מוכנה כשלמעשה הם לא ממש מוכנים עדיין. אפילו אם יש לך הגדרה ברורה של מהו דבר מוכן (DOD) ואתה חייב שתהיה לך הגדרה כזו (ראה עמוד 26, הגדרה של DOD), מפתחים בד"כ ישכחו אותה.

אנחנו המפתחים אנשים חסרי סבלנות שרוצים לעבור למשימה הבאה כמה שיותר מהר.

אז איך אדון T (הבודק שלנו) יודע כשמשהו מוכן ? טוב, אז קודם כל, הוא צריך (הפתעה) לבדוק את זה !! בהרבה מקרים מסתבר שמשהו שהמפתח החשיב כגמור לא ניתן אפילו להבדק! בגלל שהוא לא עשה check-in לקוד או שהוא לא הותקן על server הבדיקות או משהו כזה. ברגע שהבודק סיים את הבדיקות עליו לעבור על ה- checklist של DOD. למשל אם הרשימה מחייבת כי לכל פיתוח חדש צריך להיות הסבר ב- Release Notes אז הבודק מוודא שזה אכן

קיים. אם יש עוד אי אלו מסמכים רישמיים שאמורים להתעדכן אז הבדוק גם אותם וכו...
הצד הנחמד של זה הוא שהבדוק עכשיו הוא המועמד המושלם להכנת המצגת של סוף הספרינט.

מה עושה הבדוק כאשר אין עדיין מה לבדוק ?

השאלה הזו עולה כל הזמן, אדון T: "הי, סקראם מסטר, שום דבר לא מוכן עדיין לבדיקות, אז מה עליי לעשות כרגע?"
זה יקח שבוע עד שהצוות יסיים את הסיפור הראשון, אז מה אמור הבדוק לעשות במהלך שבוע זה ?
טוב, דבר ראשון, הוא אמור להתכונן לבדיקות. הכוונה לתכנן את תסריטי הבדיקות, להכין סביבה ועוד... ואז כשיש משהו מוכן לבדיקות אפשר להתחיל לבדוק מיד ללא עיכובים – אדון T יכול מיד לצלול לבדיקות.
אם הצוות עושה TDD חברי הצוות כותבים בדיקות מהיום הראשון. הבדוק עושה Pair programming עם המפתחים, רק שהוא מנחה ומוביל ונותן למפתח לכתוב את הקוד. בודק טוב בד"כ יעלה תסריטי בדיקות שונים מאלו שיעלה מפתח טוב, וכך הם משלימים זה את זה.
אם הצוות לא עושה TDD, או שאין מספיק תסריטי בדיקה על מנת למלא את זמנו של הבדוק, הוא צריך פשוט לעשות כל מה שצריך כדי לעזור לצוות להשיג את מטרת הספרינט. אם הבדוק יכול לפתח אז מעולה. אם לא אז הצוות צריך לזהות את כל המשימות בספרינט שלא קשורות לקידוד.
כאשר שוברים את הסיפורים למשימות במהלך פגישת תכנון הספרינט, הצוות נוטה להתרכז במשימות פיתוח. אך בד"כ יש הרבה משימות לא פיתוחיות שצריכות להתבצע בספרינט. אם מקדישים זמן במהלך הפגישה לזהות את המשימות האלו יש סיכויים גבוהים שהבדוק יוכל לתרום הרבה אפילו אם הוא לא מקודד וגם אם אין משימת בדיקות ברגענתון.
דוגמאות למשימות שאינן משימות קידוד שאמורות בד"כ להתבצע במהלך ספרינט:

- הכנת סביבת בדיקות/אינטגרציה
- הבהרה של דרישות מול מנהלי המוצר
- השגת פרטים מקבוצת התפעול על התקנות/פריסה
- כתיבת מסמכי Release Notes (למשל)
- יצירת קשר עם גורמים מחוץ לצוות (למשל מעצבי GUI)
- שיפור ה – Build Scripts
- המשך שבירה של סיפורים למשימות
- זיהוי שאלות פתוחות של מפתחים והשגת התשובות מהגורמים הרלוונטים

ובמקרה ההפוך, מה נעשה עם אדון T, במידה והוא הופך לצוואר הבקבוק ? בו נאמר שאנחנו ביום האחרון של הספרינט ופתאום הרבה דברים מוכנים לבדיקות ולבדוק אין סיכוי לסיים לבדוק הכל. מה נעשה ? אז כל הצוות יוכל להתגייס לעזרתו של אדון T. אדון T יחליט מה הכרחי שהוא יבצע בעצמו ומה הוא מעביר לידי חברי הצוות האחרים – זו המשמעות של צוות רב תחומי!
אז כן, לבדוק יש תפקיד ספציפי בצוות, אבל הוא מורשה ונידרש לעשות דברים נוספים וגם אנשי הצוות האחרים מורשים ויכולים לעשות את עבודתו.

15.4 העלאת האיכות ע"י עשייה של פחות סיפורים לספרינט

כאן אנחנו חוזרים לפגישת תכנון הספרינט. פשוט אל תכניסו יותר מדי סיפורים! אם יש בעיות איכות, או תהליך בדיקות קבלה ארוך – תעשו פחות בכל ספרינט.

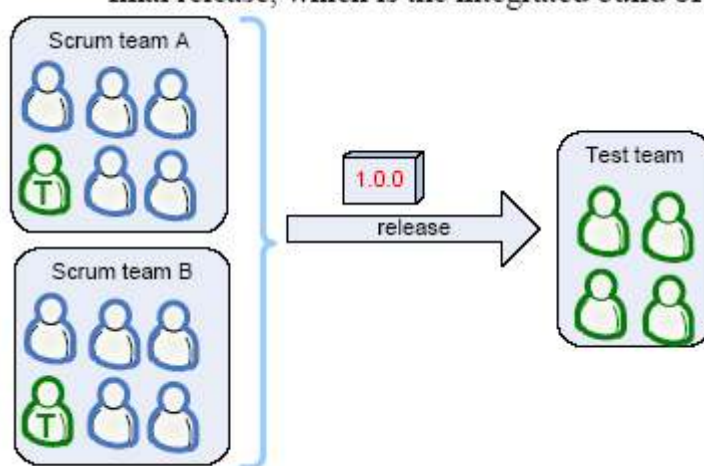
זה יוביל כמעט אוטומטית לעלייה באיכות, קיצור תקופת בדיקות הקבלה, פחות תקלות שמשפיעות על המשתמש, ובטווח הרחוק עלייה בפרודוקטיביות משום שהצוות יכול יותר להתרכז בדברים חדשים ולא בתיקון של דברים ישנים ששוב ושוב מתקלקלים.

זה כמעט תמיד זול יותר לבנות פחות, אך יציב לעומת בניית המון ואז להזדקק לייצר בפאניקה תיקונים דחופים (hotfixes).

15.5 האם שלב בדיקות הקבלה צריך להיות חלק מהספרינט?

אנחנו מתלבטים בזה המון. חלק מהצוותים שלנו כוללים חלק זה בתוך הספרינט אך רוב הצוותים שלנו לא. משתי סיבות:

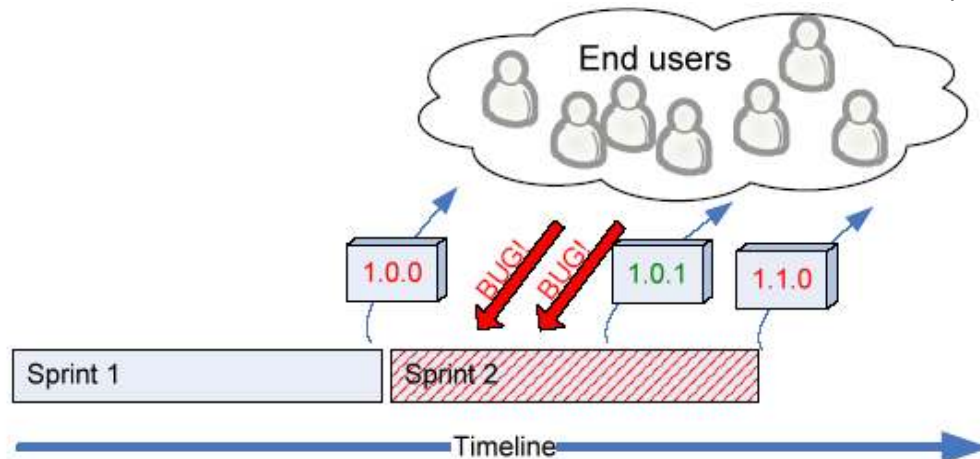
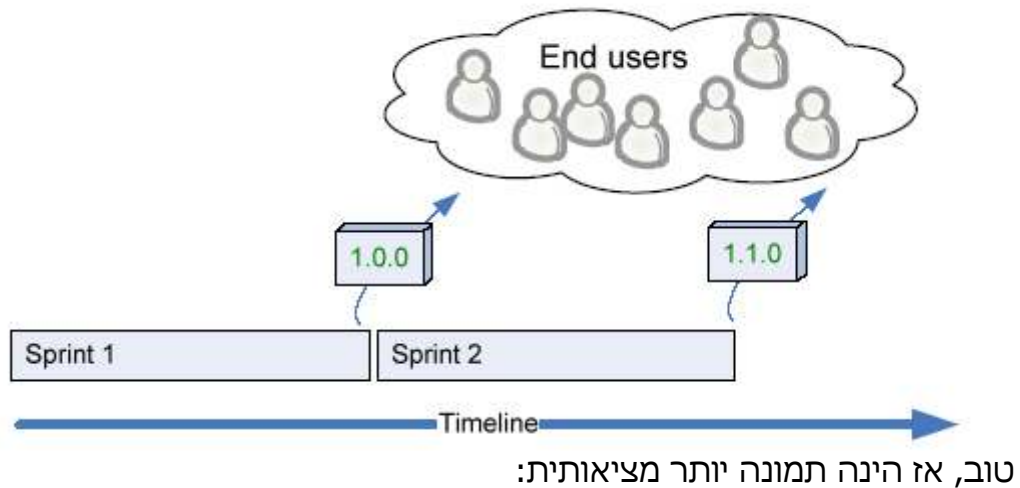
- ספרינט מוגבל בזמן (time boxed). בדיקות הקבלה (בהגדרה שלי הכוללת סבבים של בדיקות, תיקונים והוצאת גירסא מחדש) זה דבר שקשה מאוד להכניס לזמן מוגדר. מה קורה אם נגמר הזמן ועדיין יש תקלה קריטית דחופה? האם נשחרר את הגירסא עם תקלה זו? האם נחכה לספרינט הבא? ברוב המיקרים שלנו הפתרונות האלו לא אפשריים ולכן נשאיר את השלב הזה מחוץ לספרינט (מחוץ ל timebox).
- אם יש לך מספר צוותי סקראם שעובדים על אותו מוצר, בדיקות הקבלה הידניות חייבות להתבצע על התוצר המשותף של כל הצוותים. אם הצוותים עשו את הבדיקות הידניות במהלך הספרינט, עדיין יש צורך בצוות שיבדוק את הגירסא הסופית, שהיא התוצר המשותף של הצוותים.



זה אינו פתרון מושלם אבל זה פתרון מספיק טוב ברוב המקרים.

15.6 סבבי ספרינטים לעומת סבבי בדיקות קבלה

בעולם סקראם מושלם, אין צורך בשלב בדיקות הקבלה משום שכל צוות סקראם מספק בסיום כל ספרינט גירסא מוכנה להפצה (PSP-potentially shippable product)

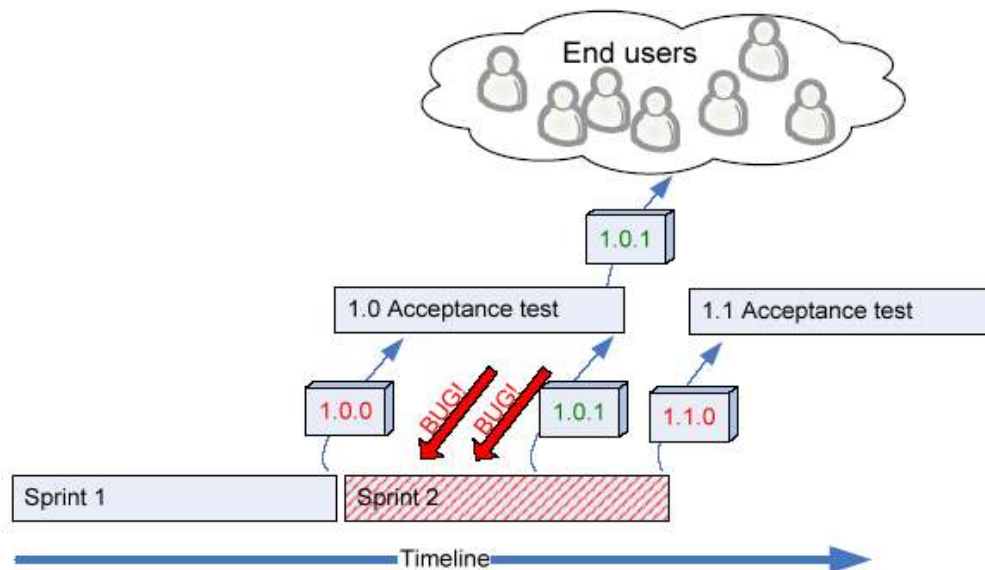


וכך בסוף הספרינט השני משחרר הצוות גירסא שאף יותר מלאה בתקלות מקודמתה משום שלצוות היה עוד פחות זמן לעשות דברים נכון בשל כל ההפרעות וחוזר חלילה...

הקווים האלכסוניים באדום מסמנים כאוס.

תמונה לא יפה במיוחד, נכון? טוב, הדבר העצוב הוא שהבעיה נותרת בעינה גם אם יש לך צוות מיוחד לבדיקות הקבלה.

ההבדל היחידי הוא שרוב התקלות ידווחו ע"י צוות הבדיקות ולא ע"י לקוחות זועמים. זה הבדל עצום מנקודת מבט עיסקית, אבל עבור הצוותים זה מצטבר לאותו הדבר. חוץ מזה שהבודקים הם בד"כ קצת פחות אגרסיביים מלקוחות (משתמשי קצה). בד"כ.



לא מצאנו פתרון פשוט לבעיה זו. אבל התנסינו הרבה עם מודלים שונים של פתרונות. קודם כל, שוב, שפר את איכות הקוד שהצוות משחרר. העלות של מציאה ותיקון תקלות במהלך ספרינט, היא באופן משמעותי הרבה יותר נמוכה מאשר מציאה ותיקון לאחר מכן. אך העובדה נותרת, גם אם נפחית את כמות התקלות, עדיין יגיעו דיווחים על תקלות לאחר סיום הספרינט. איך מתמודדים עם זה?

גישה 1: אל תתחיל לבנות דברים חדשים עד שהדברים הישנים משוחררים.

נישמע נחמד, נכון? היינו קרובים לאמון בגישה זו מספר פעמים, וגם יצרנו מודלים מרשימים של האופן שזה יכול להתבצע. אך תמיד שינינו את דעתנו כשהבנו את החסרונות. נאלץ להוסיף בין הספרינטים תקופה לא מוגבלת בזמן שבה נעשה רק בדיקות ותיקונים עד שנוכל להוציא גרסא.

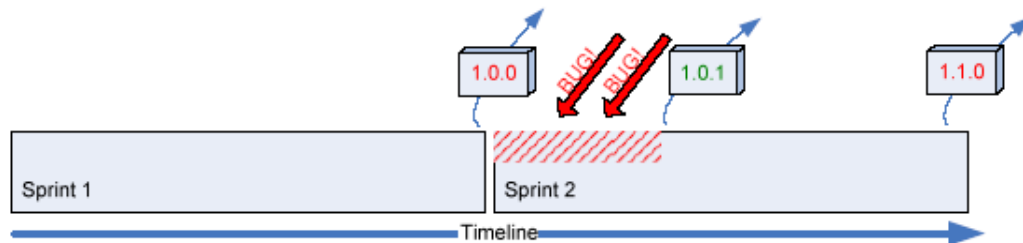


לא אהבנו את הרעיון שיש תקופות לא מוגבלות בזמן בין ספרינטים. בעיקר בגלל שזה ישבור את הפעימה הקבועה של הספרינטים. לא נוכל לומר יותר שכל 3 שבועות אנחנו מתחילים ספרינט חדש. וחץ מזה, זה לא לגמרי פותר את הבעיה. אפילו אם תהיה תקופת release כזו, עדיין יהיו דיווחים על תקלות דחופות מפעם לפעם, ואנחנו צריכים להיות מוכנים להתמודד איתם.

גישה 2: "זה בסדר להתחיל לבנות דברים חדשים, אבל תינתן עדיפות גבוהה יותר לתיקון הדברים הישנים ששוחררו"

זו הגישה המועדפת עלינו. לפחות בינתיים. באופן בסיסי, כשאנחנו מסיימים ספרינט אנחנו מתקדמים לספרינט שאחריו. אבל אנחנו מצפים שחלק מהזמן במהלך הספרינט יוקדש לתיקונים של הספרינט הקודם. אם הספרינט ניפגע מאוד משום שהיינו עסוקים מדי בתיקונים של הספרינט הקודם, צריך לבצע הערכה של הסיבות והדרך לשפר את איכות

הקוד. יש לוודא שאורך הספרינט מספיק על מנת להיות מסוגלים לבצע תיקונים וגם להתקדם. באופן הדרגתי, לאורך תקופה של מספר רב של חודשים, אורך הזמן שהשקענו בתיקונים של הספרינט הקודם קטן. בנוסף הצלחנו לערב פחות מאנשי הצוות במקרה של תקלה, כך שלא היה צורך להפריע לכל הצוות בכל תקלה. כעת אנחנו ברמה מקובלת יותר של כמות תקלות.



במהלך פגישת תיכנון הספרינט, אנחנו מציבים את ה- focus factor נמוך מספיק על מנת לקחת בחשבון את כמות התקלות שאנחנו מצפים שנצטרך לתקן מהספרינט הקודם. עם הזמן הצוותים נעשו ממש טובים בהערכה זו. מטריקת ה velocity עוזרת מאוד (ראה עמוד 19 "איך צוות מחליט אילו סיפורים לכלול בספרינט")

גישה גרועה – התמקד בבנייה של דברים חדשים

למעשה הכוונה בכך היא "התמקד בבנייה של דברים חדשים במקום להביא את הדברים הקודמים לשיחרור". מי ירצה לעשות דבר כזה? למרות זאת עשינו את הטעות הזו הרבה בהתחלה, ואני בטוח שהרבה חברות אחרות עשו זאת גם כן. זו מחלה שקשורה ללחץ. הרבה מנהלים לא באמת מבינים את זה, שכשכל הקידוד הסתיים אתה עדיין רחוק משיחרור הגירסא. לפחות במערכות מורכבות. אז המנהל (או מנהל המוצר) מבקש מהצוות להמשיך להוסיף דברים חדשים בזמן ש"התיק" של הדברים שכמעט מוכנים לשיחרור נעשה כבד מנשוא וגורם להאטה כללית.

15.7 אל תברח מהחלק האיטי ביותר בשרשרת שלך

נאמר שבדיקות הקבלה הם השלב האיטי ביותר שלך. יש לך מעט מידי בודקים, או שהשלב לוקח הרבה זמן בגלל איכות הקוד. נאמר שצוות הבודקים שלך יכול לבדוק עד 3 features בשבוע (לא, אנחנו לא משתמשים במדד של features לשבוע, זה רק לצורך הדוגמא הזו). נניח שהמפתחים יכולים לפתח 6 features לשבוע. אל תעשה את זה! המציאות תשיג אותך בסוף, בדרך כזו או אחרת, וזה יכאב! במקום זה, פתח 3 features לשבוע ובשאר הזמן תטפל בצווארי בקבוק של הבדיקות. למשל:

- קח מספר מפתחים שיעשו בדיקות (הם מאוד יאהבו אותך על כך...)
- יישם כלים ו scripts להפוך את הבדיקות לקלות יותר
- הוסף בדיקות אוטומטיות
- הארך את אורך הספרינט והוסף את בדיקות הקבלה לתוך הספרינט
- הגדר מספר ספרינטים כספרינטים של בדיקות והקדש את כל הצוות לבדיקות
- גייס עוד בודקים (גם אם זה אומר פחות מפתחים)

ניסינו את כל ההצעות הנ"ל (חוץ מהאחרונה). הפיתרון הארוך טווח היחידי הוא כמובן השני והשלישי, ז"א כלים טובים יותר ובדיקות אוטומטיות. פגישות הרטרוספקטיבה הם המקום הטוב ביותר לזהות את השלב האיטי בשרשרת.

15.8 חזרה למציאות

בטח קיבלתם את הרושם שיש לנו בודקים בכל צוותי הסקראם שלנו, שיש לנו צוות בודקים ענק לצורך בדיקות קבלה לכל מוצר, ושנחנו משחררים גירסא אחרי כל ספרינט. אז לא.

לפעמים אנחנו מצליחים לעשות את הדברים הללו, וראינו את התוצאות הטובות של זה. אבל אנחנו עדיין רחוקים מתהליך QA מקובל וטוב, ויש לנו עוד הרבה מה ללמוד כאן.

16. איך מטפלים במספר צוותי סקראם

הרבה דברים הופכים להיות מסובכים וקשים יותר כאשר מספר צוותים עובדים על אותו מוצר. הבעיה הזו היא אוניברסלית ואין לה באמת קשר לסקראם. יותר מפתחים = יותר סיבוכיות.

ערכנו (כרגיל) מספר ניסויים בזה. היה לנו צוות שהגיע בשיא לסביבות 40 איש שעבדו על אותו מוצר. שאלות המפתח הן:

- כמה צוותים לבנות
- איך להקצות אנשים לצוותים

16.1 כמה צוותים לבנות

אם עבודה עם מספר צוותי סקראם היא כל כך מורכבת, אז מדוע לטרוח? למה לא פשוט לשים את כולם בצוות אחד? היו 11 איש בצוות הסקראם הגדול ביותר שהיה לנו. זה עבד, אבל לא כל כך טוב. פגישות ה Daily Scrum נטו להתארך מעבר ל-15 דקות. חברי הצוות לא ידעו מה מה נעשה בצוותים האחרים. נוצר בלבול. לסקראם מסטר היה מאוד קשה למקד את כולם לעבר היעדים, וקשה היה לו למצוא זמן על מנת להתמודד עם כל המכשולים שדווחו.

האפשרות האחרת היא לחלק לשני צוותים. אך האם זה יותר טוב? לא בהכרח.

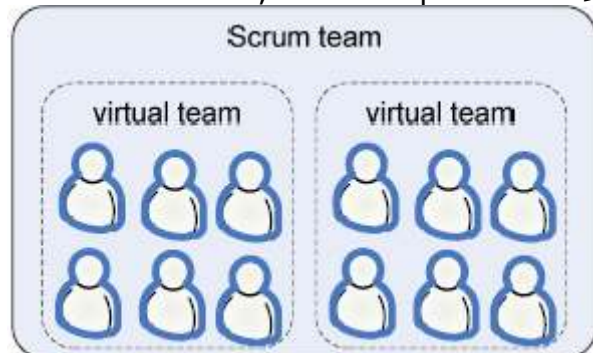
אם הצוות מנוסה ומרגיש נוח עם סקראם, וקיימת דרך הגיונית להפריד את רשימת המוצר לשני מסלולים נפרדים, ושני המסלולים הללו לא קשורים לאותו source code, אז הייתי אומר שזה רעיון טוב לפצל לשני צוותים. אחרת הייתי שוקל להשאר עם צוות אחד, למרות החסרונות של צוותים גדולים. מנסיוני טוב יותר שיש מעט צוותים גדולים מדי, מאשר הרבה צוותים קטנים שכל הזמן מפריעים זה לזה. בנה צוותים קטנים רק כאשר הם לא צריכים להפריע אחד לשני!

צוותים ווירטואלים

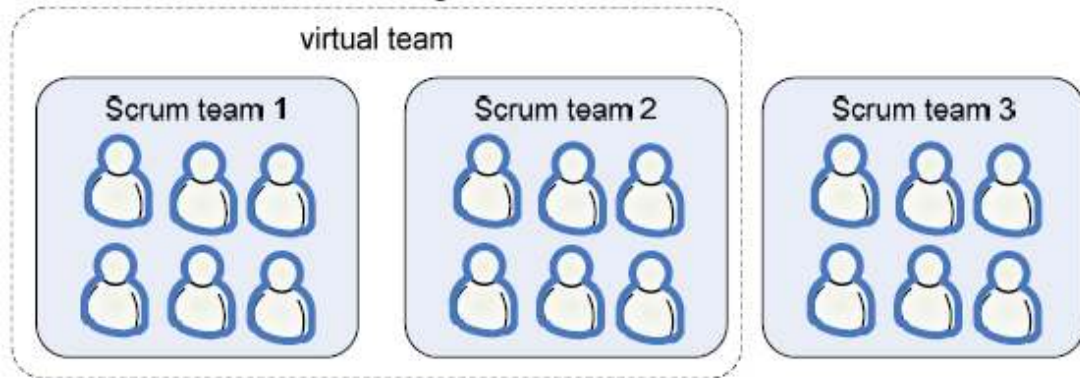
איך תוכל לדעת שעשית את הבחירה הנכונה לאור השיקולים בין "צוות גדול" ל"הרבה צוותים"?

אם תפתח את העיניים והאוזניים תוכל לשים לב שנוצרים "צוותים ווירטואלים".

דוגמא 1: בחרת שיהיה צוות אחד גדול. אבל כאשר אתה שם לב למי מדבר עם מי במהלך הספרינט, מתברר שהצוות למעשה התחלק לשני תתי צוותים.



דוגמא 2: אתה בוחר שיהיו לך 3 צוותים קטנים יותר. אבל כאשר אתה שם לב למי מדבר עם מי במהלך הספרינט, מתברר שצוות 1 וצוות 2 מדברים ביניהם כל הזמן אבל צוות 3 בבידוד.



מה המשמעות של זה? שאסטרטגיית חלוקת הצוותים שבחרת הייתה שגויה? כן, במידה והצוותים הוירטואליים הם קבועים. לא, במידה והצוותים הוירטואליים הם זמניים.

התבונן בדוגמא 1 שוב. אם שני תתי הצוותים הוירטואליים נוטים להשתנות מזמן לזמן (זאת אומרת שאנשים עוברים מצוות וירטואלי אחד לשני) כנראה זו תהיה בחירה נכונה להשאיר אותם כצוות אחד גדול. אם הצוותים נשארים אותו הדבר לאורך כל הספרינט, אז כנראה שצריך לשבור אותם לשני צוותים אמיתיים בספרינט שאחריו.

כעת, התבונן בדוגמא 2 שוב. אם צוות 1 וצוות 2 מדברים זה עם זה כל הזמן (ולא צוות 3) לאורך כל הספרינט, אז כנראה שצריך לאחד את צוות 1 ו 2 לצוות אחד בספרינט הבא. אם צוות 1 ו 2 מדברים זה עם זה הרבה בחלק הראשון של הספרינט ואז בחלק השני של הספרינט צוות 1 ו 3 מדברים זה עם זה, אז עליך לשקול לאחד אותם לצוות אחד גדול. העלה את הבעיה בסוף הספרינט בפגישת ה-retrospective ותן לצוות להחליט.

חלוקה לצוותים היא אחת החלקים הקשים של סקראם. אל תחשוב יותר מדי ותעשה יותר מדי אופטימיזציות. תתנסה, שים עין על צוותים וירטואליים וודא שאתה לוקח את הזמן לדון בזה עם הצוות בסוף הספרינט במהלך פגישת ה-retrospective. במוקדם או במאוחר אתה תמצא פתרון למצב המיוחד שלכם. הדבר החשוב ביותר הוא שהצוותים ירגישו בנוח ולא יפריעו זה לזה יותר מדי.

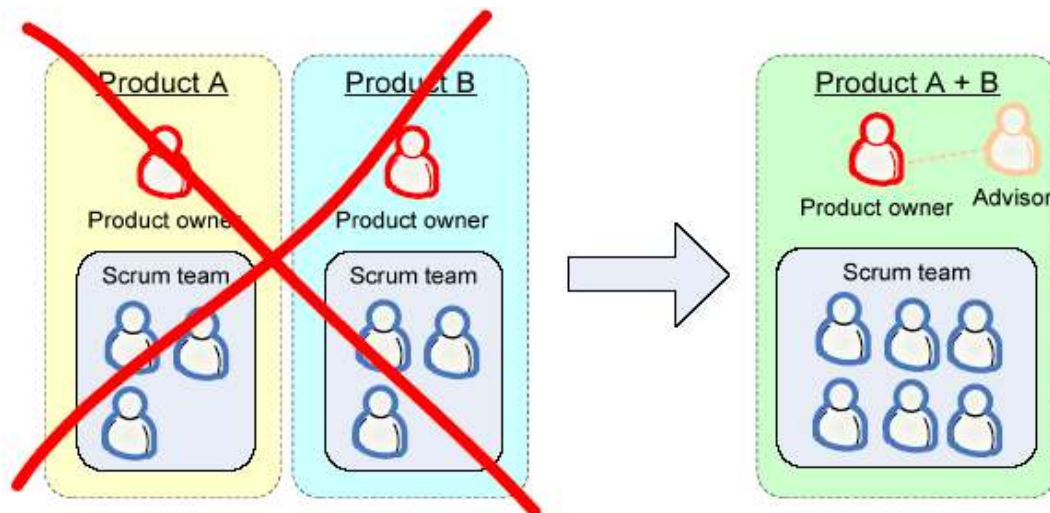
הגודל האופטימאלי לצוות

רוב הספרים שקראתי טוענים שהגודל ה"אופטימאלי" לצוות הוא בסביבות 5-9 אנשים.

ממה שאני ראיתי עד כה אני יכול רק להסכים. למרות שאני הייתי אומר 3-8 אנשים. למעשה אני מאמין שכדאי לסבול קצת קשיים על מנת להשיג צוותים בגודל זה.

נגיד שיש לך צוות יחיד של 10 אנשים. תשקול להוציא את שני חברי הצוות החלשים ביותר. אופס, האם אני אמרתי את זה?

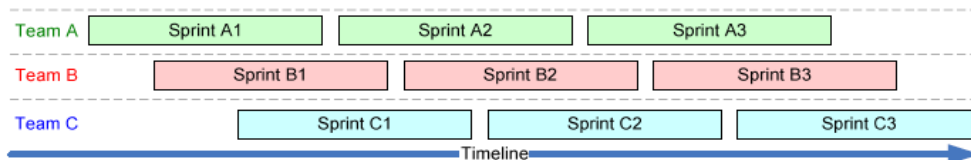
נגיד שיש לך שני מוצרים שונים, עם צוות של 3 אנשים עבור כל מוצר, ושניהם מתקדמים לאט מדי. זה יכול להיות רעיון טוב לצרף אותם לצוות אחד של 6 אנשים שאחראי על שני המוצרים. במקרה זה שחרר את אחד מבעלי המוצר (או שתיתן לו תפקיד של יועץ או משהו כזה).



נניח שיש לך צוות יחיד של 12 אנשים, משום שהקוד במצב כל כך רעוע שאין כל דרך ששני צוותים יעבדו עליו בצורה עצמאית. הקדש מאמץ משמעותי לתיקון הקוד (במקום לבנות דברים חדשים) עד שתגיע לנקודה שבה תוכל לפצל את הצוות. השקעה זו כנראה תשתלם מהר מאוד.

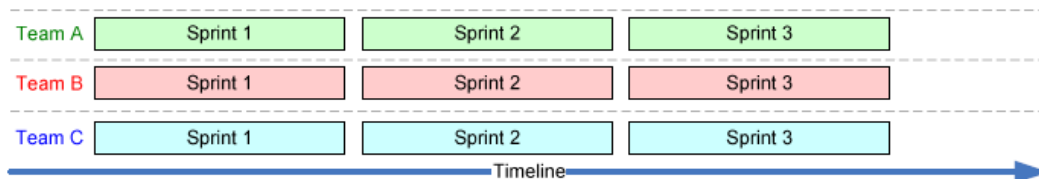
16.2 האם לסנכרן ספרינטים – או לא?

נגיד שיש לך שלושה צוותים שעובדים על אותו מוצר. האם הספרינטים שלהם צריכים להיות מתואמים, זאת אומרת, להתחיל ולהסתיים באותו הזמן? או שהם צריכים להיות חופפים? הגישה הראשונה שלנו הייתה שיהיו לנו ספרינטים חופפים (בהיבט של הזמן).



זה נשמע נחמד, בכל רגע נתון ישנו ספרינט רץ שעומד להסתיים וספרינט חדש שעומד להתחיל. עומס העבודה של בעל המוצר יהיה פרוש באופן שווה בזמן. ניצור גירסאות חדשות כל הזמן. תהיה הדגמה כל שבוע. הללויה. כן, אני יודע, בזמן ההוא זה באמת נשמע משכנע!

כשהתחלנו להתנהל כך והייתה לי ההזדמנות לדבר עם קן שוואבר (במהלך ההסמכה שלי בסקראם). הוא הצביע על כך שזה היה רעיון גרוע, ז"א זה הרבה יותר טוב לתאם בין הספרינטים. אני לא זוכר את הסיבות הספציפיות, אבל אחרי השיחה עימו שוכנעתי.

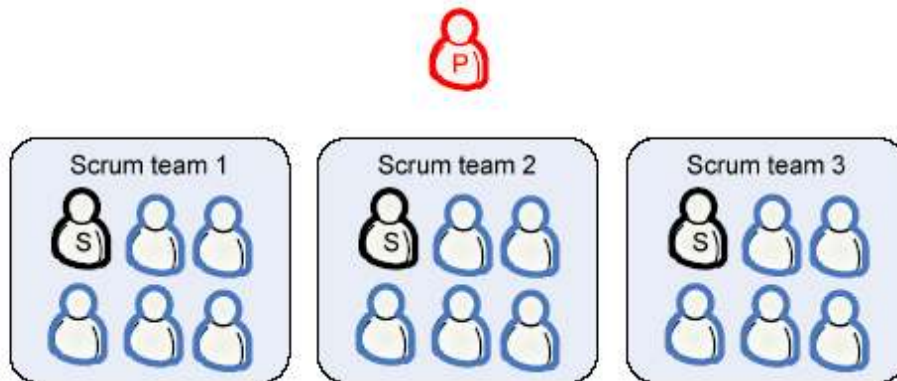


זה הפתרון שהשתמשנו מאז, ואף פעם לא חזרנו בנו. לעולם לא אדע אם אסטרטגית הספרינטים החופפים הייתה נכשלת, אבל ניראה לי שכן. היתרונות בספרינטים מתואמים הם:

- יש זמן טבעי שבו אפשר לארגן מחדש את הצוותים במידת הצורך – בין ספרינטים! עם ספרינטים חופפים, אין דרך לארגן מחדש את הצוותים מבלי שנפריע לפחות לצוות אחד באמצע הספרינט.
- כל הצוותים יכולים לעבוד לעבר יעד זהה בספרינט ולעשות את פגישת התכנון ביחד, דבר שמוביל גם לשיתוף פעולה טוב יותר בין הצוותים.
- פחות תקורה ניהולית, ז"א פחות פגישות תכנון, פחות הדגמות וגרסאות.

16.3 מדוע הוספנו את תפקיד "ראש הצוות"

נגיד שיש לנו מוצר אחד עם שלושה צוותים.



הבחור המסומן באדום הוא בעל המוצר. הדמויות שמסומנות בשחור הן הסקראם מסטרים. שאר הדמויות הן חברי צוות מכובדים. אז עם ההרכב הזה, מי מחליט מי צריך להיות באיזה צוות? בעל המוצר? הסקראם מסטרים ביחד? או שכל אחד בוחר את הצוות שלו? אבל אז אם כולם רוצים להיות בצוות אחד (כי סקראם מסטר 1 זה כל כך טוב)? ומה אם לאחר זמן מה מתגלה שזה לא אפשרי שיותר משני צוותים יעבדו במקביל על אותו בסיס קוד, אז נצטרך להפוך לשני צוותים עם 9 אנשים במקום שלושה צוותים עם שישה אנשים כל אחד. זה אומר שני סקראם מסטרים. את איזה משלושת הסקראם מסטרים נשחרר מתפקידו? ברוב החברות אלו יהיו עניינים מאוד רגישים.

זה מפתה לתת לבעל המוצר לעשות את השיוך של האנשים. אבל זו לא ממש עניין בעל המוצר, נכון? בעל המוצר הוא המומחה שאומר לצוות לאיזה כיוון הם צריכים לרוץ. הוא לא באמת אמור להתערב בעניינים האלו. במיוחד משום שהוא "תרנגולת" (אם לא שמעת את המטאפורה של התרנגולות והחזירים, חפש "chickens and pigs" בגוגל).

פתרנו את הבעיה ע"י הוספת תפקיד "ראש הצוות". זה מתאים למה שיכול להיקרא לפעמים "מסטר ה-Scrum of Scrums" או "הבוס" או הסקראם מסטר הראשי". הוא לא צריך להוביל אף צוות ספציפי, אבל הוא אחראי על עניינים שהם חוצי צוותים כמו למשל מי יהיה הסקראם מסטר של הצוותים, איך אנשים יצוותו וכולי.

היה לנו קשה למצוא שם עבור התפקיד הזה. השם "ראש הצוות" היה השם הכי פחות גרוע שמצאנו.

הפתרון הזה עבד עבורנו ואני ממליץ עליו (לא משנה איך תחליטו לקרוא לתפקיד הזה).

16.4 איך אנחנו מקצים אנשים לצוותים

ישנן שתי אסטרטגיות כלליות להקצאת אנשים לצוותים כאשר יש לך מספר צוותים אשר עובדים על אותו מוצר.

- אדם מסוים יבצע את ההקצאה, למשל "ראש הצוות" שהזכרתי קודם, בעל המוצר או המנהל הפונקציונאלי (במידה והוא מספיק מעורב על מנת לאפשר לו לבצע החלטה טובה כאן).
- תן לצוות לעשות זאת בעצמם בדרך כל שהיא.

התנסינו בכל שלושת האפשרויות. שלוש כן. אסטרטגיה 1, אסטרטגיה 2, ושילוב של השתיים.

גילינו ששילוב של השתיים הוא הטוב ביותר.

לפני פגישת תכנון הספרינט, ראש הצוות עורך פגישת הקצאה לצוותים ביחד עם בעל המוצר וכל הסקראם מסטרים. בפגישה מדברים על הספרינט שעבר ומחליטים אם יש צורך באי אילו הקצאות חדשות ומוצדקות. אולי אנחנו רוצים לחבר שני צוותים או להעביר אנשים מצוות אחד לאחר. אנחנו מחליטים ומביאים את ההחלטה לישיבת תכנון הספרינט כהצעת הקצאה לצוותים.

הדבר הראשון שאנחנו עושים ישיבת תכנון הספרינט זה לעבור על הפריטים בעדיפות הגבוהה ביותר ברשימת מוצר. ראש הצוות אומר משהו כגון:

"שלום לכולם, אנחנו מציעים את ההקצאות הבאות לצוותים לספרינט הבא:"

<i>Preliminary team allocation</i>		
Team 1 - tom - jerry - donald - mickey	Team 2 - goofy - daffy - humpty - dumpty	Team 3 - minnie - scrooge - winnie - roo

"כמו שאתם רואים, המשמעות של החלוקה הזו היא ירידה מ-4 צוותים ל-3 צוותים. פירטנו את החברים בכל צוות. אנא התקבצו ותפסו לכם מקטע ליד הקיר."

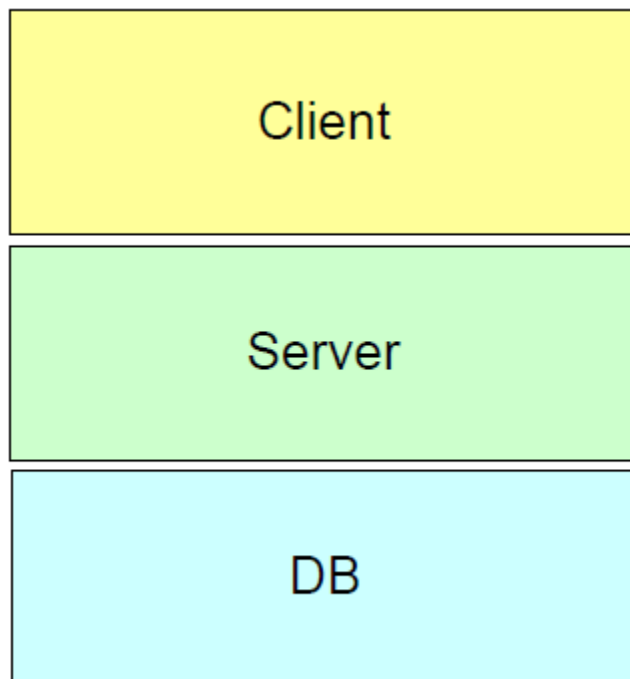
ראש הצוות מחכה בזמן שכולם משוטטים בחדר, לאחר כמה זמן ישנן 3 קבוצות של אנשים, כל אחת ליד מקטע ריק של קיר).

"זו חלוקה ראשונית! זו נקודת התחלה, על מנת לחסוך בזמן. במהלך הפגישה אתם חופשיים לעבור לקבוצה אחרת, לחלק את הקבוצה לשתיים, לצרף שתי קבוצות לקבוצה אחת. השתמשו בהגיון פשוט על בסיס העדיפויות של בעל המוצר."

זה מה שמצאנו שעובד הכי טוב. רמה מסויימת של שליטה מרכזית בהתחלה, שבעקבותיה נעשית אופטימיזציה של האנשים בשטח.

6.5 צוותים מקצועיים או לא?

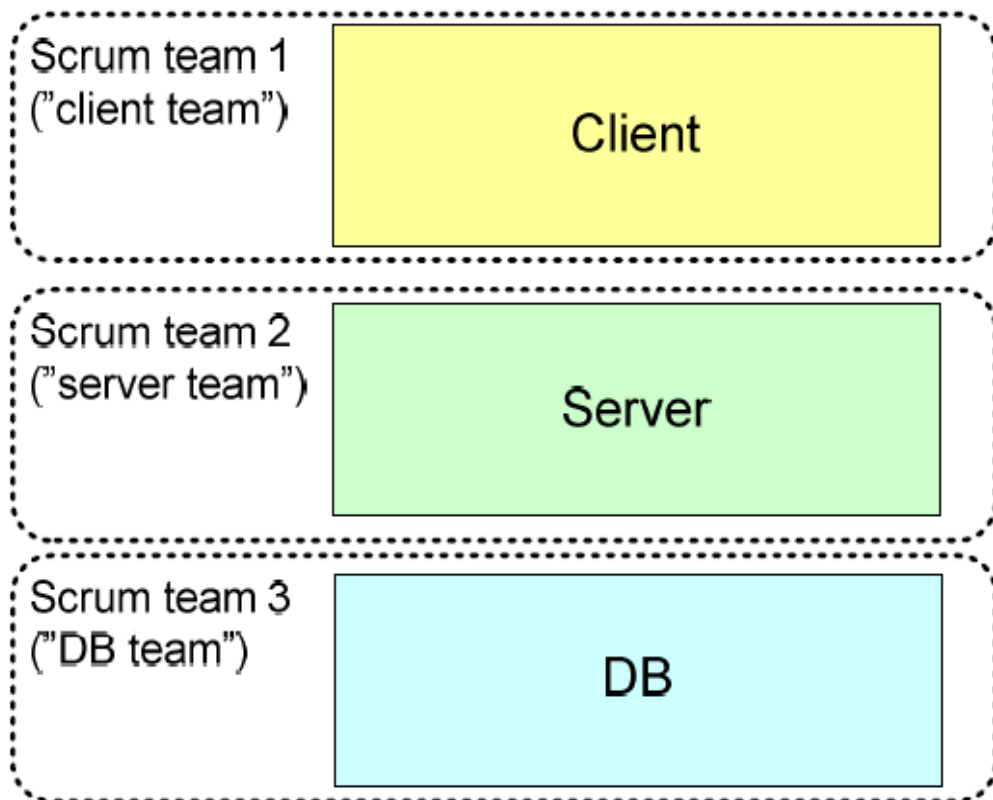
נניח שהטכנולוגיה שלך מורכבת משלושה רכיבים מרכזיים:



ונניח שיש לך 15 אנשים שעובדים על פיתוח המוצר הזה. אתה לא באמת רוצה להתנהל כצוות סקראם אחד. אילו צוותים תיצור?

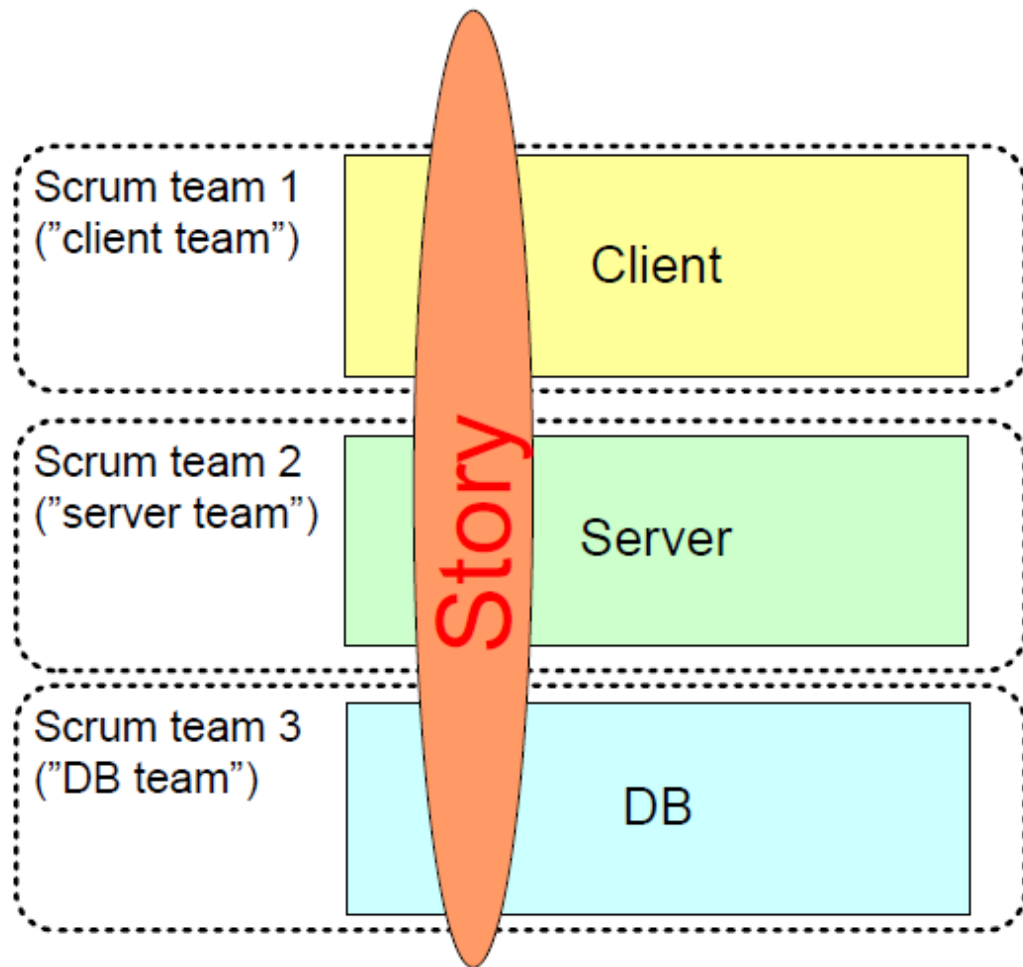
גישה 1: צוותים מומחים לפי רכיבים

גישה אחת היא ליצור צוותים לפי רכיבים כגון צוות Client, צוות Server, וצוות DB



זה המקום שבו התחלנו. זה לא עובד יותר מדי טוב, לפחות כאשר ברוב הסיפורים שיש לפתח מעורבים שלושת הרכיבים.

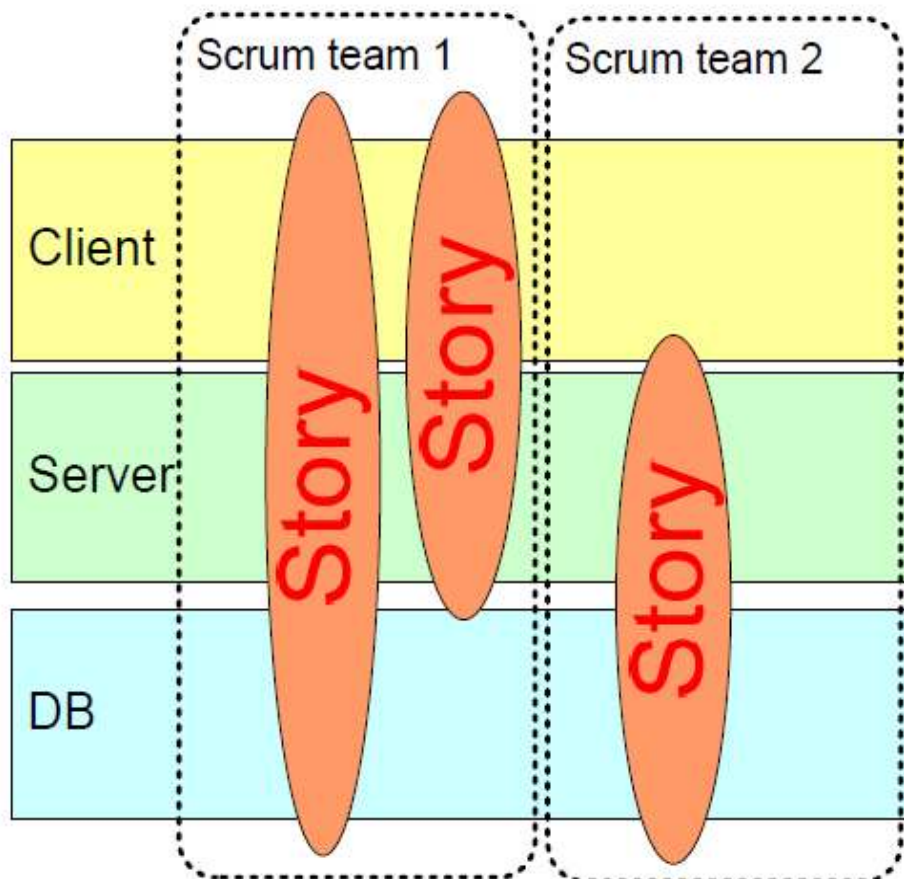
נניח למשל שיש לנו סיפור שניקרא "לוח מודעות שבו משתמשים יכולים לפרסם הודעות אחד לשני". על מנת לפתח את הסיפור הזה יהיה צורך לעדכן את ממשק המשתמש ב-Client, להוסיף לוגיקה ב-Server, ולהוסיף טבלאות בבסיס הנתונים.



המשמעות היא שכל שלושת הצוותים יצטרכו לשתף פעולה על מנת לסיים את הסיפור. לא טוב.

גישה 2: צוותים חוצי רכיבים

הגישה השניה היא ליצור צוותים חוצי רכיבים, ז"א צוותים שלא קשורים לאף רכיב ספציפי.



במידה והסיפורים מערבים מספר רכיבים אז חלוקה זו לצוותים תעבוד טוב יותר. כל צוות יכול ליישם סיפור שלם כולל החלקים של ה-client, ה-server ובסיס הנתונים. הצוותים יוכלו לעבוד בצורה עצמאית ללא תלות ביניהם, וזה דבר טוב.

אחד הדברים הראשונים שעשינו כאשר התחלנו לעבוד בסקראם היה לשבור את הצוותים הקיימים עם מומחיות רכיבית (גישה 1) ולבנות צוותים חוצי רכיבים במקום (גישה 2). זה הוריד משמעותית את המקרים של: "אנחנו לא יכולים לסיים את הפריט הזה משום שאנחנו מחכים שאנשי ה-server יעשו את החלק שלהם".

עם זאת, אנחנו לפעמים מקבצים צוותים מומחי רכיבים באופן זמני אם מתעורר צורך משמעותי.

16.6 האם לצוות מחדש את האנשים בין ספרינטים ?

כל ספרינט הוא למעשה די שונה מקודמו, תלוי בסיפורים שבעדיפות גבוהה באותו זמן. כתוצאה מכך, סידור הצוותים האופטימלי יכול להיות שונה בין הספרינטים השונים.

למעשה, כמעט כל ספרינט מצאנו את עצמנו אומרים משהו כמו "הספרינט הזה הוא לא ממש ספרינט רגיל בגלל (בלה בלה בלה)...". לאחר זמן מה פשוט ויתרנו על התפישה של "ספרינט רגיל". אין ספרינטים רגילים. בדיוק כמו שאין משפחות "רגילות" או אנשים "רגילים".

בספרינט אחד זה יהיה רעיון טוב שיהיה צוות Client שמורכב מאנשים שמכירים את הקוד של ה-Client ממש טוב. בספרינט שאחריו זה יהיה רעיון טוב להקים צוותים חוצי רכיבים ולפזר את אנשי ה-Client ביניהם.

אחד ההיבטים המרכזיים ב Scrum הוא "גיבוש הצוות". ז"א אם צוות עובד ביחד במהלך מספר ספרינטים הם בדרך כלל יהיו מאוד *מגובשים*. הם ילמדו להשיג זרימה קבוצתית טובה וישיגו פרודוקטיביות גבוהה. אבל זה לוקח מספר ספרינטים על מנת להגיע לשם. אם כל הזמן תשנה את הרכב הצוותים לא תשיג לעולם "גיבוש צוות" חזק.

לכן, במידה ואתה מעוניין לארגן את הצוותים מחדש, תוודא שאתה לוקח בחשבון את המשמעויות. האם זה שינוי קצר טווח או ארוך טווח? אם זה קצר טווח, תשקול לוותר עליו. במידה וזה שינוי ארוך טווח, אז לך על זה.

מקרה יוצא דופן הוא כשאתה מתחיל לעבוד בסקראם עם צוות גדול בפעם הראשונה. במקרה זה כדאי להתנסות קצת עם חלוקה לתתי צוותים עד שמוצאים משהו שנוח לכולם. תוודא שכולם מבינים שזה בסדר לפשל בפעמים הראשונות כל עוד משתפרים.

16.7 אנשי צוות בחלקי מישרה

אני יכול רק לאשר את מה שספרי הסקראם אומרים – לא מומלץ שיהיו אנשי צוות בחלקי מישרה.

בוא נאמר שאתה עומד לקחת את ג'ו כחבר במשרה חלקית בצוות הסקראם שלך. תחשוב טוב קודם כל. האם אתה באמת צריך אותו בצוות? אתה בטוח שאתה לא יכול לקבל את ג'ו במשרה מלאה? מהם ההתחייבויות האחרות של ג'ו? אולי משהו אחר יכול לקחת על עצמו את ההתחייבויות האלו וג'ו יכול לקחת חלק יותר פסיבי, תומך, ביחס להתחייבויות אלו? אולי אפשר שג'ו יעבור *מהספרינט הבא* לצוות שלך באופן מלא ובינתיים הוא יעביר את האחריות שלו למישהו אחר.

לפעמים פשוט אין דרך להחליץ מזה. אתה פשוט זקוק לג'ו כי הוא ה-DBA היחיד בבניין, וצוותים אחרים זקוקים לו באותה נאשות כך שאף פעם הוא לא יעבור לצוות שלך באופן מלא, והחברה לא יכולה לשכור DBA נוסף.

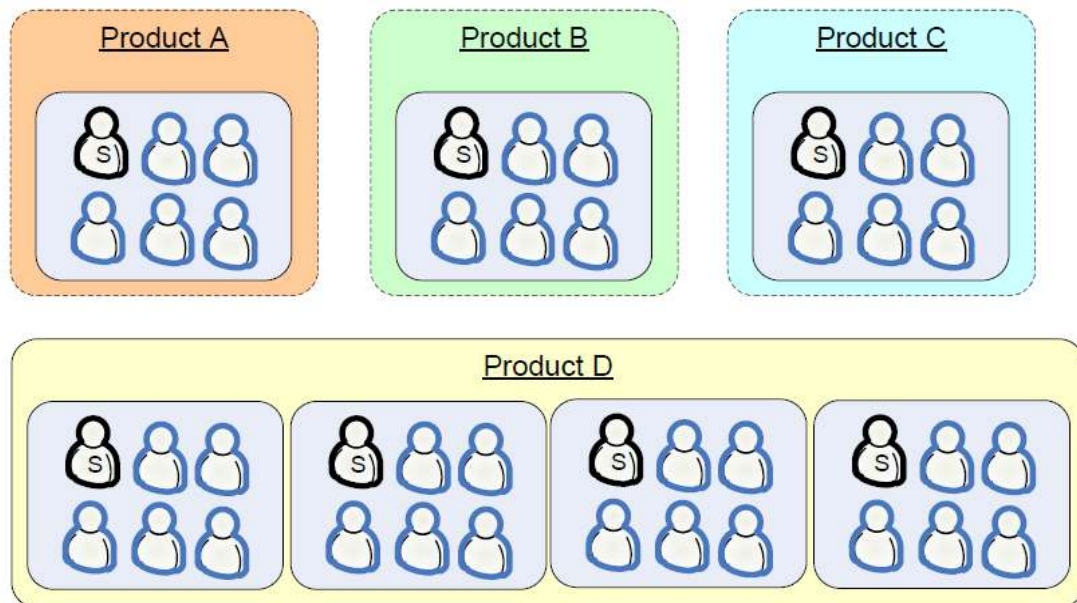
בסדר, זה מקרה שאפשר בו להחזיק את ג'ו במשרה חלקית בצוות שלך (מה שקרה לנו בדיוק). אבל תוודא שאתה עושה את הערכה הזו בכל פעם.

באופן כללי אני אעדיף צוות של 3 אנשים במשרה מלאה מ-8 במשרה חלקית.

במידה ויש לך מישהו שמחלק את זמנו בין מספר צוותים, כמו ה-DBA המדובר, זה יהיה רעיון טוב שהוא ישתייך לצוות אחד. ברר מי הצוות שיזדקק לו הכי הרבה ועשה את הצוות הזה "צוות הבית" שלו. כשאף אחד אחר לא "גורר" אותו, הוא יגיע ל-Daily, לישיבת התכנון, ול-retrospective של הצוות שלו.

16.8 איך אנחנו עושים Scrum-of-Scrums

Scrum-of-Scrums היא פגישה רגילה שבה כל הסקראם מסטרים נאספים ביחד לדבר. בשלב מסוים היו לנו ארבעה מוצרים, כאשר לשלושה מהמוצרים היה צוות אחד לכל מוצר ולמוצר אחד היו 25 אנשים מחולקים למספר צוותים. משהו כזה:



זה אומר שהיו לנו שתי רמות של Scrum-of-Scrums. רמת המוצר שבה היו כל הצוותים ממוצר D ואחד ברמת התאגיד שכלל את כל המוצרים.

Scrum-of-Scrums ברמת המוצר

הפגישה הזו היתה מאוד חשובה. עשינו את זה פעם בשבוע, לפעמים יותר תדיר מזה. דיברנו על ענייני אינטגרציה, הכנות לקראת הplanning של הספרינט הבא ועוד. הקצאנו לפגישה 30 דקות אבל לעיתים קרובות היינו זקוקים ליותר זמן. האלטרנטיבה הייתה לעשות Scrum-of-Scrums כל יום אבל לא הגענו לעשות זאת.

האג'נדה של ה Scrum-of-Scrums שלנו הייתה:

1. מסביב לשולחן, כל אחד מספר מה הצוות שלו השיג בשבוע שעבר, מה הם מתכננים להשיג לשבוע הבא ומה היו המיכשולים שבהם ניתקלו.
 2. כל נושא אחר שצריך להעלות, למשל נושאים של אינטגרציה.
- האג'נדה לא ממש חשובה לי, העיקר שיהיו לכם Scrum-of-Scrums באופן סדיר.

Scrum-of-Scrums ברמת החברה

קראנו לפגישה הזאת "הדופק". עשינו את הפגישה הזו בדרכים שונות באופנים שונים עם משתתפים שונים. לאחרונה נטשנו את זה והחלפנו אותה בפגישה שבועית כשכל הפיתוח מוזמן. 15 דקות? עם כולם? כל חברי הצוותים של כל המוצרים? האם זה עובד?

כן, זה עובד אם אתה (או מי שמעביר את הפגישה) מקפידים לשמור שתהיה קצרה.

המבנה של הפגישה הוא:

1. חדשות ועדכונים ממנהל הפיתוח. לדוגמא: מידע לגבי ארועים קרובים.
2. נציג מכל מוצר מציג מה הם השיגו עד כה ומה הם מתכוונים להשיג השבוע והאם היו בעיות. גם אנשים אחרים מדווחים (ראש צוות בדיקות ו CM)

3. כל אחד חופשי להוסיף כל מידע ולשאול שאלות זהו פורום למידע קצר, בלי דיון או שיקוף. בהקפדה על כך, 15 דקות בדרך כלל עובד. לפעמים אנחנו חורגים, אבל באופן חריג ליותר מ 30 דקות בסך הכל. במידה ודיון מעניין מתחיל אני מפסיק אותם ומזמין את כל מי שמעוניין להשאר אח"כ.

מדוע אנחנו עושים פגישה שכולם מוזמנים? משום ששמנו לב ש-Scrum-of-Scrums ברמת החברה הוא בדרך כלל באופי של דיווח. דיונים בקבוצה הזו היו נדירים. בנוסף הרבה אנשים מחוץ לקבוצה הזו היו רעבים למידע הזה. באופן בסיסי, הצוותים מעוניינים לדעת מה עושים בצוותים אחרים. אז הבנו שאם נפגש ונעדכן אחד את השני למה שלא נאפשר לכולם להגיע.

16.9 שיבוץ וחפיפה בסקראם היומי

אם יש לך הרבה צוותי Scrum במסגרת מוצר אחד, וכולם עושים Daily באותו הזמן, יש לך בעיה. בעל המוצר (ואנשים חטטניים כמוני) יכולים להיות בפגישת ה-Daily רק אצל צוות אחד ביום. אז אנחנו מבקשים מהצוותים להמנע מלעשות את ה-Daily באותו הזמן.

	Room 1	Room 2
9:00	Team 1	
9:15		Team 2
9:30	Team 3	
9:45		Team 4
10:00	Team 5	

לוח הזמנים למעלה הוא דוגמא לתקופה שבה היו לנו פגישות בחדרים נפרדים, לעומת מצב שזה בחדר של הצוות. הפגישות הן 15 דקות אבל כל צוות מקבל מרווח זמן של 30 דקות למקרה שהם צריכים להאריך מעט. זה מאוד שימושי משתי סיבות:

1. אנשים כמו בעל המוצר ואני יכולים להגיע לכול פגישות ה-Daily בבוקר אחד. אין דרך טובה יותר לקבל תמונה מדויקת על התקדמות הספרינט, ומהם האיומים והסיכונים המרכזיים.
2. צוותים יכולים לבקר בפגישת ה-Daily של צוותים אחרים. זה לא קורה הרבה אבל מידי פעם שני צוותים עובדים על אזור משותף, אז חלק מחברי הצוות מגיעים לפגישה על מנת לשמור על תיאום.

החיסרון הוא שיש פחות חופש לצוותים להחליט על הזמן שהכי נוח להם לקיים את הפגישה. אבל זה לא היה בעייתי עבורנו כלל.

16.10 צוותי "כיבוי שריפות"

היה לנו מצב שמוצר גדול לא יכול היה לאמץ שיטת Scrum משום שהצוותים היו רוב הזמן "מכבים שריפות", ז"א תיקוני באגים בפאניקה על סביבת ה Release שלהם. זה היה מעגל אימה, הם היו כל כך עסוקים בכיבוי שריפות שלא היה להם זמן לעבוד באופן פרואקטיבי על מנת למנוע שריפות (ז"א שיפור design, בדיקות אוטומטיות, יצירת כלי ניטור ועוד).

התמודדנו עם זה ע"י כך שבנינו צוותי "כיבוי שריפות" יעודיים, וצוות Scrum יעודי. תפקידו של צוות ה Scrum היה (בברכתו של בעל המוצר) לנסות לייצב את המערכת ובאופן אפקטיבי למנוע שריפות.

צוות "כיבוי שריפות" (קראנו להם למעשה "תמיכה") היה בעל שני תפקידים:

1. כיבוי שריפות
 2. להגן על צוות ה Scrum מכל מיני רעשים והפרעות, כולל דברים כמו דרישות ad-hoc שמגיעות משום מקום.
- צוות כיבוי השריפות מוקם ליד הדלת, צוות ה Scrum מוקם בחלק האחורי של החדר. כך שצוות "כיבוי השריפות" יוכל להגן פיזית על צוות ה Scrum מהפרעות כמו אנשי מכירות או לקוחות כועסים.
- מפתחים בכירים הושמו בשני הצוותים על מנת להוריד למינימום את התלות בין שני הצוותים. בבסיס זה היה נסיון לפתור בעיית Scrum bootstrapping. איך אפשר להתחיל לעשות Scrum אם לצוות אין הזדמנות לתכנן יותר מיום אחד קדימה? האסטרטגיה שלנו הייתה, כמו שציינתי, לפצל את הקבוצה. עבד לא רע. בזכות זה שניתן לצוות ה Scrum המקום לעבוד באופן פרואקטיבי הם הצליחו לבסוף לייצב את המערכת. ובאותו זמן צוות "כיבוי השריפות" ויתר לגמרי על התכנון קדימה ועבדו באופן ראקטיבי לחלוטין, פשוט לתקן את הבעיה הדחופה הבאה שהופיעה.
- ברור, שצוות ה Scrum לא היה מוגן באופן מוחלט. לעיתים צוות כיבוי השריפות עירב אנשים מצוות ה Scrum, ובמקרים הגרועים ביותר את כולם. בכל אופן, לאחר מספר חודשים המערכת הייתה יציבה מספיק על מנת שנוכל לפרק את צוות כיבוי השריפות ולבנות צוות Scrum נוסף. מכבי השריפות היו מאוד שמחים לוותר על הקסדות ולאחסן אותן ולהצטרף לצוות Scrum במקום.

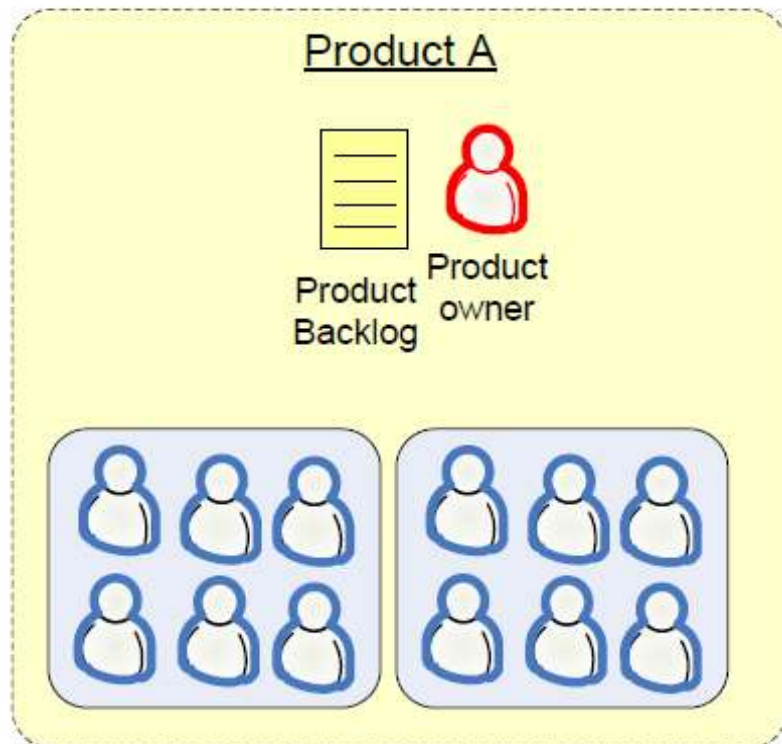
16.11 האם לפצל את רשימת המוצר – או לא ?

בוא נאמר שיש לך מוצר אחד ושני צוותי Scrum. כמה רשימות מוצר צריכות להיות לך ? כמה בעלי מוצר?

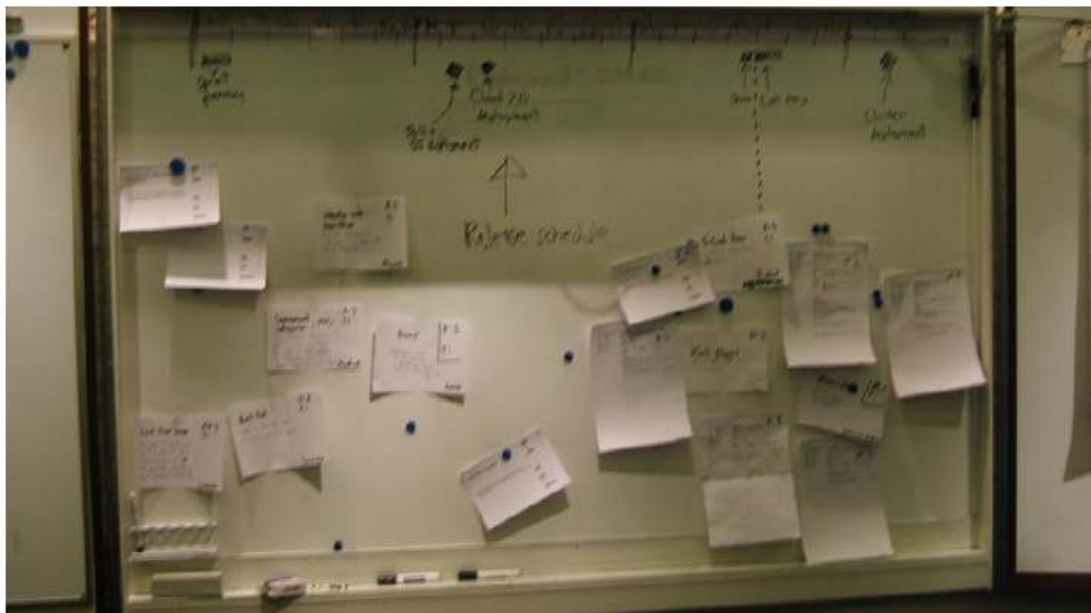
בדקנו מספר מודלים עבור זה. הבחירות השונות משפיעות משמעותית על האופן שמנהלים את פגישות ה planning.

אסטרטגיה 1: בעל מוצר אחד ורשימת מוצר אחת

הדבר הטוב במודל הזה הוא שניתן לתת לצוותים להחליט בעצמם על חלוקת העבודה ביניהם בהתבסס על העדיפויות שנקבעו ע"י בעל המוצר.

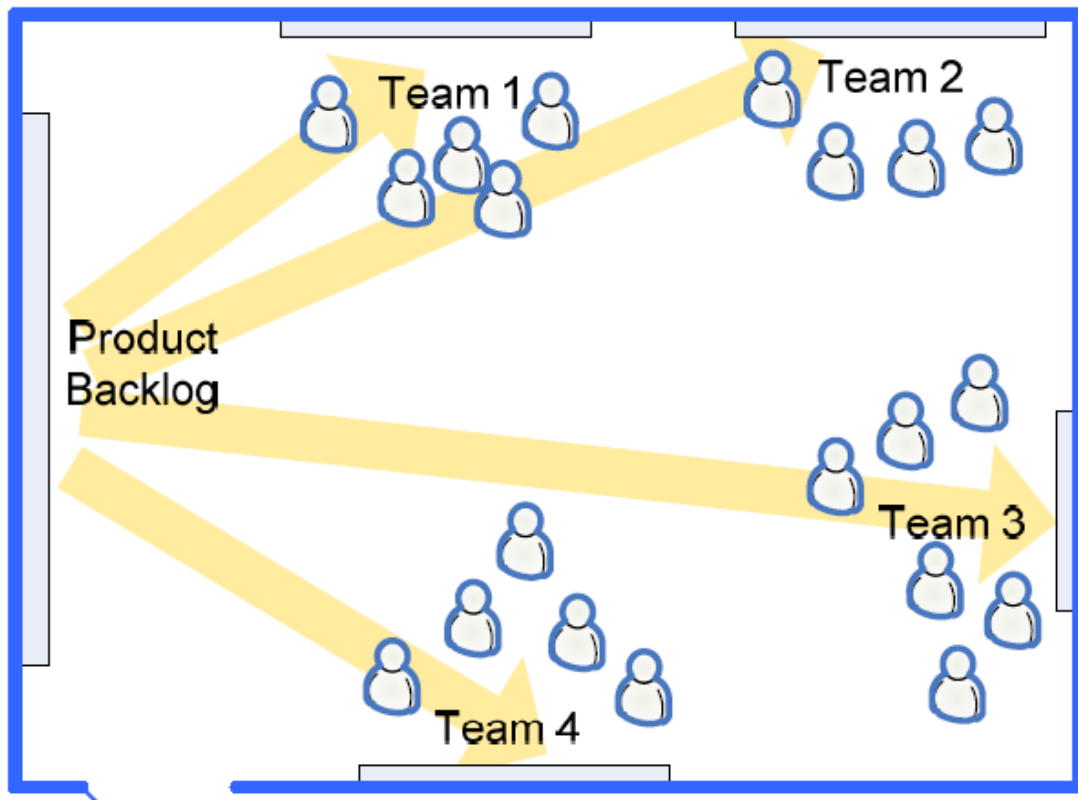


להיות יותר קונקרטי, ככה תעבוד פגישת התכנון עבור הצוות הזה. הפגישה תתנהל בחדר ישיבות חיצוני. מעט לפני הישיבה בעל המוצר מכריז על אחד הקירות להיות קיר רשימת המוצר. מניח על הקיר את הסיפורים (פתקיות דביקות) מסודרים לפי עדיפויות. הוא ממשיך לשים עוד ועוד סיפורים עד שהקיר מלא, וזה בדרך כלל מכיל מספיק פריטים לספרינט אחד.



כל צוות בוחר חלק ריק של הקיר ומסמן את שם הצוות על החלק הזה. זה הקיר של הצוות. כל צוות לוקח סיפורים מקיר רשימת המוצר לפי סדר העדיפויות ומניח על הקיר של הצוות שלו.

זה מתואר בתמונה שמתחת, כאשר החיצים הצהובים מסמלים את זרימת הסיפורים מקיר רשימת המוצר לקירות של הצוותים.



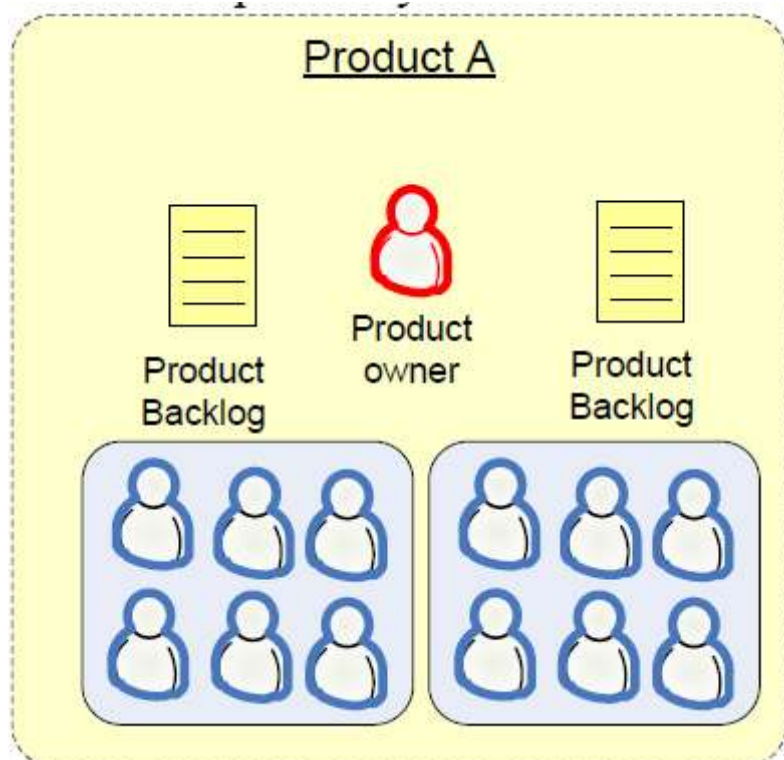
במהלך הפגישה, בעל המוצר והצוותים מתגודדים סביב הפתקים, מזיזים אותם בין הצוותים, מעלים ומורידים אותם לשנות עדיפויות ושוברים אותם לפריטים קטנים יותר. לאחר בערך שעה, לכל צוות יש את רשימת המוצר המועמדת שלו לספרינט הקרוב על קיר הצוות. לאחר מיכן כל צוות עובד באופן עצמאי על הערכות זמנים ושבירה של הסיפורים למשימות.



זה כאוטי ומטיש אבל גם אפקטיבי, כיף וחברתי. כשנגמר הזמן, בד"כ לכל הצוותים יש מספיק מידע על מנת להתחיל את הספרינט.

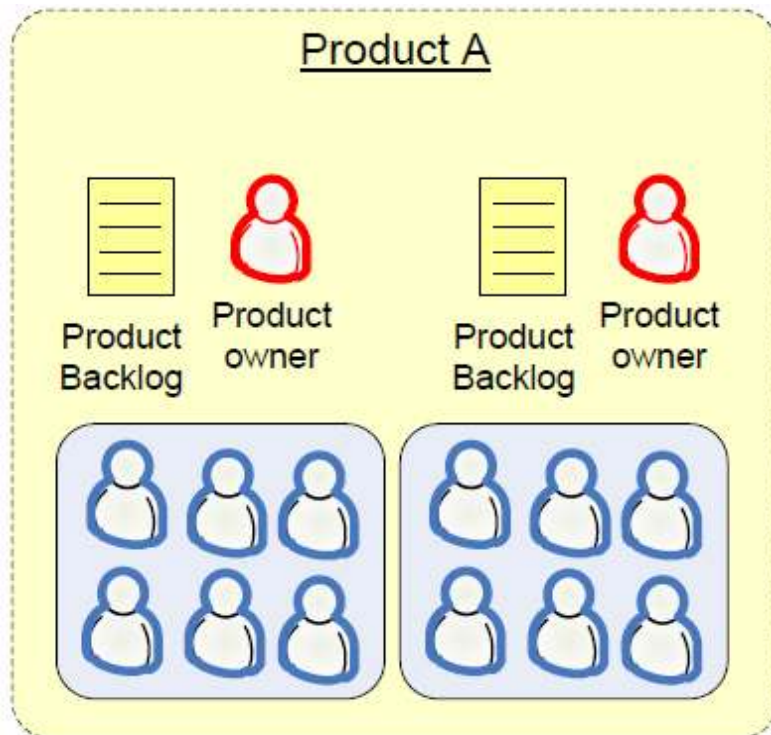
אסטרטגיה 2: בעל מוצר אחד, מספר רשימות מוצר

באסטרטגיה זו, בעל המוצר מתחזק רשימת מוצר אחת עבור כל צוות. לא ממש התנסו בגישה זו, אבל היינו קרובים. בעיקרון זו תוכנית "המגירה" שלנו למקרה שהגישה הראשונה תכשל. החולשה של אסטרטגיה זו היא שבעל המוצר מקצה את הסיפורים לצוותים, משימה שכנראה הצוותים מסוגלים לעשות בצורה טובה יותר.



אסטרטגיה 3: מספר בעלי מוצר, רשימת מוצר אחת לכל בעל מוצר

זה כמו האסטרטגיה השנייה, רשימת מוצר לכל צוות, רק שגם בעל מוצר לכל צוות



לא התנסינו בזה אבל כנראה שגם לא נתנסה.
 במידה ששתי רשימות מוצר מעורבות באותו codebase, זה ודאי יגרום קונפליקט רציני בין שני בעלי המוצר. במידה ורשימות המוצר הן לא על codebase משותף, זה במהות כמו לחלק את המוצר לתתי מוצרים אשר מנוהלים באופן עצמאי. זה אומר שאנחנו בחזרה במצב של צוות אחד עם רשימת מוצר אחת, שזה נחמד וקל.

Code branching 16.12

כשיש מספר צוותים העובדים על אותו code base זה בלתי נימנע להתמודד עם Code branches במערכת SCM (Software Configuration Management). ישנם הרבה ספרים ומאמרים על הנושא של איך להתמודד עם הרבה אנשים שעובדים על אותו codebase ולכן לא ארחיב כאן. אין לי שום דבר חדש או מהפכני להוסיף. עם זאת, אני אסכם חלק מהמסקנות החשובות ביותר שנלמדו עד כה ע"י הצוותים שלנו:

- הקפידו לשמור על mainline (או trunk) במצב קונסיסטנטי. המשמעות היא שלפחות, הכל צריך לעבור קומפילציה וכל ה unit tests עוברים. צריכים להיות מסוגלים לייצר release עובד בכל רגע נתון. עדיף שמערכת ה continuous build תתקין באופן אוטומטי על סביבת הבדיקות כל לילה.
- עבור כל Release שיהיה לכם Tag. בכל פעם שאתם משחררים גירסא לבדיקות קבלה או production, תוודא שיש לכם Tag שבעזרתו ניתן לזהות בדיוק מה שוחרר. זה אומר שבכל זמן שנדרש בעתיד ניתן יהיה ליצור branch תחזוקה מנקודה זו.
- צור Branch חדש רק במקרים שיש צורך. כלל אצבע טוב הוא ליצור branch חדש רק כאשר אי אפשר להשתמש ב codeline הנוכחי להפר את

המדיניות שלו. במיקרה של ספק, אל תיצור branch מפני שכל branch פעיל משמעותו ניהול תחזוקה ומוכבות.

- השתמש ב branches עבור מחזורי חיים שונים. ייתכן ותחליט או לא תחליט שכל צוות Scrum יעבוד על branch נפרד. אבל אם תערבב תיקונים קצרי טווח עם שינויים ארוכי טווח, תמצא שזה מאוד קשה לשחרר את התיקונים קצרי הטווח!
- סנכרן באופן תדיר. במידה ואתה עובד על branch, סנכרן ל mainline בכל פעם שיש לך משהו שעובר build. בכל יום שאתה מגיע לעבודה, סנכרן מה mainline ל branch שלך, כך שה branch שלך יהיה מעודכן עם כל השינויים שעושים בו שאר הצוותים. במידה וזה גורם לך גיהנום פשוט הבן שהמצב יהיה גרוע בהרבה אם תחכה.

16.13 Retrospective למספר צוותים

איך אנחנו עושים retrospective כשיש לנו מספר צוותים שעובדים על אותו מוצר? מיד לאחר פגישת הספרינט demo, אחרי מחיאות הכפיים, כל צוות הולך לחדר משלו, או לאיזור נוח מחוץ למשרד. הם עושים את ה retrospective שלהם. פחות או יותר כמו שמתואר בע"מ 56 "איך אנחנו עושים retrospective". משהו רושם הערות.

במהלך פגישת ת ה planning (אשר כל הצוותים משתתפים בה מישום שאנחנו עושים ספרינטים מסונכרנים בתוך כל מוצר), הדבר הראשון שאנחנו עושים הוא לתת זכות דיבור לנציג מכל צוות שיעמוד ויסכם את הנקודות העקריות מה retrospective שלהם. זה לוקח בערך 5 דקות עבור כל צוות. אז עושים דיון פתוח של 10-20 דקות ולאחר מכן עושים הפסקה שלאחריה אנחנו מתחילים את פגישת ה planning.

לא ניסינו דרך אחרת עבור מספר צוותים. דרך זו עבדה מספיק טוב. הבעיה הגדולה ביותר היא שאין מרווח זמן אחרי החלק של ה retrospective ולפני ה planning. ראה עמוד 60 "מרווח זמן בין ספרינטים". עבור מוצר של צוות אחד, אנחנו לא עושים את החלק שבו עוברים על הנקודות מה retrospective ב-planning. אין צורך כולם היו נוכחים בפגישת retrospective עצמה.

17 כיצד אנחנו מתמודדים עם צוותים מבוזרים

מה בעצם קורה כאשר חברי הצוות נמצאים במיקומים שונים? הרבה מה"קסם" של סקראם ו-XP מבוסס על חברי צוות אשר קרובים פיזית אחד לשני, משתפים פעולה בעבודה, נפגשים פנים אל פנים כל יום ואף מקודדים יחדיו ב-pair programming.

לנו יש מספר צוותים מבוזרים, ואף מספר אנשים אשר עובדים מהבית מדי פעם בפעם.

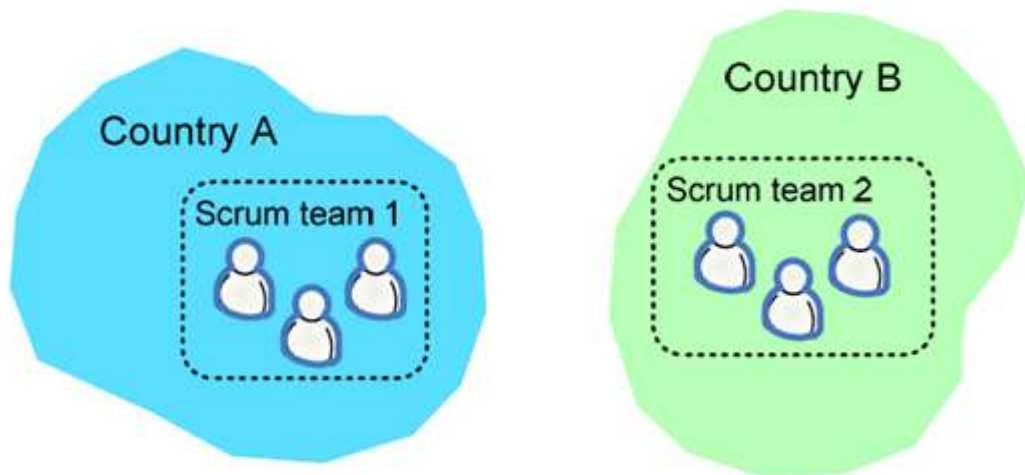
האסטרטגיה שלנו בנושא היא מאוד פשוטה. אנחנו משתמשים בכל דרך שאנחנו מסוגלים לדמיין על מנת להרחיב את ערוצי התקשורת שלנו בין חברי הצוות המרוחקים. אני כמובן לא מדבר אך ורק על רוחב פס שנמדד בביטים לשניה (למרות שזה חשוב לא פחות). אני מדבר על ערוצי תקשורת במובן הרחב יותר:

- היכולת לתכנת ביחד, פעילות הידועה כ-pair programming.
 - היכולת להיפגש פנים אל פנים בפגישה היומית.
 - היכולת להיפגש פנים אל פנים בפגישות עבודה בכל זמן במהלך היום.
 - היכולת להיפגש פנים אל פנים ולבנות מערכות יחסים חבריות.
 - היכולת להיפגש פנים אל פנים לטובת דיונית ספונטנים.
 - היכולת לראות את אותה "תמונת מצב" של הפרוייקט/ספרינט דרך ה-sprint backlog, ה-sprint burndown, רשימת המוצר וכל כלי נראות נוסף שהצוות משתמש בו.
- חלק מהאמצעים שאנחנו משתמשים בהם (או בתהליך הטמעה שלהם, לא עשינו עדיין את הכל) הם:
- מצלמות רשת בכל תחנת עבודה.
 - חדרי ישיבות עם מצלמות רשת, מיקרופונים לשיחות ועידה, מחשבים שתמיד דלוקים ומוכנים לשיחות ועידה, תוכנות לשיתוף מחשבים/שולחנות העבודה וכו'.
 - "חלונות וירטואלים". מסכים גדולים בכל משרד אשר מראים באופן שותף וחי, שידור של המתרחש במיקום השני. מסכים אלו מהווים חלון וירטואלי בין שני המיקומים. אתה יכול ממש לעמוד שם ולנופף לשלום. אתה יכול לראות מי נמצא ליד שולחן העבודה ומי מדבר עם מי. חלונות אלו יוצרים תחושת חיבור נפלאה אשר מובילה להבנה ש"אנחנו ביחד בפרוייקט הזה".
 - תוכניות "חילופי סטודנטים", שבהן אנשים ממיקומים שונים מבליים פרקי זמן קצרים במשרדים השונים באופן תדיר וקבוע.
- בעזרת טכניקות אלו, ואחרות, אנחנו לאט לאט לומדים איך ניתן וכדאי לבצע פגישות תכנון, הצגה, retrospectives, פגישות יומיות, וכו' עם צוותים מבוזרים. כמו תמיד, הבסיס ללמידה הוא ניסוי. בחן <= התאם <= בחן <= התאם <= בחן <= התאם <= בחן <= התאם <= בחן <= התאם <=

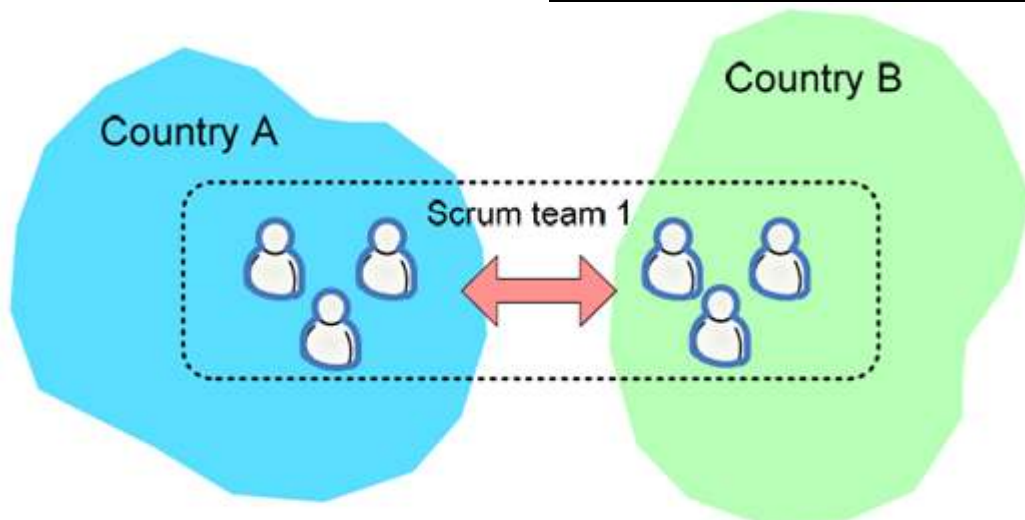
17.1 מיקור חוץ

יש לנו מספר צוותים מחוץ למשרד הראשי ואנחנו לומדים כיצד להתמודד עם הנושא בצורה יעילה בסקראם כבר זמן מה. ישנן שתי אסטרטגיות עיקריות: צוותים מופרדים או חברי צוות מבוזרים.

צוותים מופרדים



חברי צוות מבוזרים



האסטרטגיה הראשונה, צוותים מופרדים, היא בחירה קוסמת לעין. למרות זאת אנחנו החלטנו להתחיל עם האסטרטגיה השנייה, חברי צוות מבוזרים. ישנן מספר סיבות לבחירה זו:

1. אנחנו רוצים שחברי הצוות ילמדו להכיר אחד את השני היטב.
 2. אנחנו רוצים לייצר תשתית תקשורת מעולה בין שני המיקומים ורוצים לתת לצוותים תמריץ חזק לייצר תשתית זאת.
 3. בתחילת התהליך, הצוות המרוחק קטן מדי בכדי להיות צוות סקראם אפקטיבי בפני עצמו.
 4. אנחנו רוצים לייצר תקופת זמן של שיתוף ידע אינטנסיבי בין האנשים לפני שאנחנו מייצרים צוותים מרוחקים עצמאיים לחלוטין.
- יש סיכוי שבטווח הרחוק נחליט להחליף אסטרטגיה לכיוון "הצוותים המבוזרים".

17.2 חברי צוות שעובדים מהבית

עבודה מהבית יכולה להיות מעולה במקרים מסויימים. לפעמים אתה יכול להספיק לתכנת ביום אחד בבית מה שאתה מספיק בשבוע במשרד. אלא אם יש לך ילדים מתרוצצים לך בין הרגליים כמובן (o):
ובכל זאת, אחד העקרונות הבסיסיים בסקראם הוא שכל הצוות נמצא במקום משותף אחד.

אז מה אנחנו עושים?

אנחנו משאירים את ההחלטה הזאת בידיים של הצוות, הם מחליטים כמה לעבוד מהמשרד וכמה מהבית.

ישנם חברי צוות שעובדים באופן קבוע מהבית כיוון שעבורם הנסיעה למשרד ארוכה מאוד. למרות זאת, אנחנו כן מעודדים את הצוותים להיות ביחד "רוב" הזמן.

כאשר חברי צוות עובדים מהבית הם יכולים להצטרף לפגישה היומית ב-Skype, לפעמים אפילו בוידאו. הם זמינים כל היום דרך תוכנת מסרים מיידיים כלשהי. זה לא כמו להיות ביחד באותו חדר, אבל זה מספיק טוב.

ניסינו פעם את הרעיון שימי רביעי הם ימי מיקוד. זה אמר ש"אם אתה רוצה לעבוד מהבית, זה בסדר גמור, אבל תעשה את זה ביום רביעי, ותבדוק עם חברי הצוות שלך שזה בסדר". זה עבד די בסדר עם הצוות הראשון שבו ניסינו את זה. בד"כ רוב חברי הצוות נשארו בבית ביום רביעי, והם הספיקו הרבה מאוד עבודה תוך כדי שיתוף פעולה מספק. כיוון שזה היה רק יום אחד, חברי הצוות לא יצאו מסנכרון.

משום מה השיטה הזאת לא תפסה בשאר הצוותים.

בסופו של דבר, אנשים שעובדים מהבית הם לא באמת בעיה בשבילנו.

18 רשימת מטלות לסקראם מסטר

בפרק אחרון זה, אני אמנה את ה-checklist שלנו לסקראם מסטר, אשר מונה את הפעולות האדמיניסטרטיביות החוזרות ונשנות שיש אצלנו לסקראם מסטרים. דברים שקל לשכוח. אנחנו כמובן נדלג על הדברים הברורים כמו "הסרת מכשולים".

18.1 תחילת הספרינט

- לאחר פגישת התכנון, בנה דף מידע על הספרינט.
 - הוסף קישור לדף ב-Wiki.
 - הדפס את הדף והדבק אותו על הקיר כך שאנשים שעוברים ליד הצוות יוכלו לראות אותו.
- שלח לכולם מייל שמכריז על תחילתו של ספרינט חדש. אל תשכח לכלול בתוכו את מטרת הספרינט וקישור לדף המידע שהכנת.
- עדכן את מסמך סטטיסטיקות הספרינט. הוסף את המהירות המשוערכת שלכם, גודל הצוות, אורך הספרינט, וכו'.

18.2 כל יום

- וודא שהפגישה היומית מתחילה ונגמרת בזמן.
- וודא שהסיפורים מתעדכנים במהלך הספרינט ב-backlog באופן שתוכל לעמוד בל"ז.
 - וודא שבעל המוצר מיועד בכל העדכונים.
- וודא שרשימת המוצר וה-burndown chart מעודכנים ע"י הצוות.
- וודא שבעיות/מכשולים נפטרים ומדווחים לבעל המוצר ו/או מנהל הפיתוח.

18.3 סוף ספרינט

- ערוך מפגש פתוח להצגת תוצרי הספרינט.
 - יידע את כולם על קיום המפגש יום או יומיים לפני שהוא חל.
- ערוך רטרוספקטיבה עם כל הצוות ובעל המוצר. הזמן את מנהל הפיתוח כך שגם הוא יוכל ללמוד מהלקחים המופקים ולדאוג שיגיעו גם לצוותים אחרים.
- עדכן את מסמך סטטיסטיקות הספרינט. הוסף את המהירות הנצפית ונקודות חשובות שעלו ב-רטרוספקטיבה.

19 מילות פרידה

וואו! לא חשבתי שזה יהי כל כך ארוך.
אני מקווה שספר זה מספק לך מספר רעיונות, בין אם אתה חדש לסקראם, או שועל קרבות ותיק.
כיוון שצריך להתאים את סקראם לכל סביבה ומקרה, קשה לנהל דיון בונה שנוגע לפתרונות מומלצים כלליים. למרות זאת, אני מעוניין לשמוע מכולכם. ספרו לי כיצד הגישה שלכם שונה משלי. ספקו לי רעיונות כיצד להשתפר! תרגישו חופשיים ליצור עמי קשר ב-henrik.kniberg@crisp.se.
אני גם משתתף בקבוצת scrumdevelopment ב-yahoogroups.
אם אהבתם את הספר, אולי תרצו גם להעיף מבט בבלוג שלי מדי פעם לפעם.
אני מקווה שאוסיף עוד מספר מאמרים בנושאי Java, ופיתוח תוכנה אג'ילי.
<http://blog.crisp.se/henrikkniberg/>

והכי חשוב, אל תשכחו...

זאת בסה"כ עבודה, נכון?

19.1 הכרות תודה

הטיוטה הראשונה של ספר זה לקחה סופ"ש אחד, למרות היותו סופ"ש מאומץ ביותר! גורם מיקוד של 150% :o)

אני מבקש להודות לאשתי סופיה ולילדי דייה וג'ני על כך שסבלו את חוסר החברותיות שלי באותו סופ"ש, ולהוריה של סופיה, אווה ויורגן, על שבאו לעזור לטפל במשפחה.

תודה גם לחברי לעבודה ב-Crisp שבשטוקהולם, ולאנשים בקבוצת scrumdevelopment ב-Yahoo! על כך שסיפקו שירותי הגהה ועזרו לי לשפר את הספר.

ואחרונים חביבים, אני מבקש להודות לכל קוראי, אשר סיפקו לי משוב בונה. אני בעיקר שמח לשמוע שספר זה סיפק לכל כך הרבה אנשים את הניצוץ הדרוש לנסות את החיה הזאת שנקראת פיתוח תוכנה אג'יל!

19.2 קריאה מומלצת

הספרים הבאים סיפקו לי השראה ורעיונות רבים. אני ממליץ עליהם בחום!



נספח למהדורה העברית – מילון מונחים

בעל מוצר Product Owner
רשימת המוצר Product backlog
רשימת הספרינט Sprint backlog
סיפורים Stories
סיפורי לקוח User stories
ישיבת התכנון של הספרינט Sprint planning
ישיבת הדגמה Sprint review
ישיבת הרטרוספקטיבה Retrospective meeting
הצבעת נקודות Dot voting
סף קבלה Acceptance thresholds
נקודות סיפור Story points
וקטור מהירות – velocity
מאפיינים Features
תוכנית הגרסא – Release plan
תכנות זוגי – pair programming
משחק תכנון – planning game
תכנות מונחה בדיקות – TDD
עיצוב מתווסף – Incremental design