

Agile2

["親方 Project"] 編

2024-05-26 版 親方 Project 発行

はじめに

この本の目的

この本を手にとっていただき、ありがとうございます。

2024年5月

編集長 親方@親方 Project 拝

免責事項

- 本書の内容は、情報提供のみを目的としております。著者一同、正確性には留意しておりますが、正確性を保証するものではありません。この本の記載内容に基づく一切の結果について、著者、編集者とも一切の責任を負いません。
- 会社名、商品名については、一般に各社の登録商標です。TM表記等については記載しておりません。また、特定の会社、製品、案件について、不当に貶める意図はありません。
- 本書の一部あるいは全部について、無断での複写・複製はお断りします。

目次

はじめに	2
この本の目的	2
免責事項	2
第1章 アジャイルで幸せになれるのか～IT古典良書を読み解く：「初めてのアジャイル開発」編～	9
1.1 本書との出会い	9
1.2 アジャイルは誇大広告か	10
1.3 アジャイルを導入する主な理由	10
1.4 アジャイルを使って失敗する方法	11
1.5 莫邪の剣も持ち手による	12
第2章 アジャイルの始め方・続け方	14
2.1 背景	14
2.2 初めてのアジャイル開発	15
2.3 地道な改善	15
2.4 学んだこと	16
2.5 その後の現場で	16
2.5.1 事例1：前職の現場にて	16
2.5.2 事例2：現職の現場にて	17
2.6 これから始める人へ	17
2.7 まとめ	17
第3章 非エンジニアの人から「アジャイルって早く作れるんでしょ？」って聞かれた時の私の一次回答	19
3.1 「アジャイルって早く作れるんでしょ？」	19
3.1.1 「早く作れる」とは何か？	19
3.1.2 どっちのプロセスが正しいか	20
3.2 アジャイルの誤解と定着しない理由	20
3.2.1 アジャイルはいつ完成するかわからない	20
3.2.2 アジャイルがなかなか定着しない理由	21
3.3 アジャイルになるには	22
3.3.1 面倒だけどやるという覚悟	22

目次

3.4	まとめ	22
3.4.1	頑張って作ったものは売れて欲しいわけです	23
第4章	ぼくのアジャイル Q&A 100 本ノック	24
4.1	ドキュメントに関する Q&A	24
4.2	プロダクトオーナーの役割に関する Q&A	24
4.3	見積り（時間とポイント）に関する Q&A	25
4.4	ペロシティに関する Q&A	25
4.5	スプリントに関する Q&A	26
4.6	メンバーのモチベーションに関する Q&A	26
4.7	スクラム全般に関する Q&A	27
4.8	ノックのつづき	27
第5章	第3章 アジャイルの実践	28
5.1	6.1 アイデアを出す	28
5.1.1	6.1.1 カスタマージャーニーマップ	29
5.1.2	6.1.2 リーンキャンバス	29
5.1.3	6.1.3 ブレインライティング	29
5.2	6.2 アイデアをかたちづくる	30
5.2.1	6.2.1 エレベーターピッチ	30
5.2.2	6.2.2 顧客インタビュー	30
5.3	6.3 スクラム	31
5.3.1	6.3.1 スクラムの作成物	31
5.3.2	6.3.2 スクラムイベントとスプリント	32
5.3.3	6.3.3 スプリントプランニング	32
5.3.4	6.3.4 デイリースクラム	32
5.3.5	6.3.5 スプリントレビュー	33
5.3.6	6.3.6 スプリントレトロスペクティブ	33
5.4	6.4 アジャイルの実践あとがき	33
第6章	アジャイルではリカバリーってどうやるの？	34
6.1	アジャイルではアウトプットよりアウトカムが大事	34
6.2	スプリントの目的は「スプリントゴール」	36
6.3	やらなくても良いことを最大化せよ	36
6.4	アジャイルでリカバリーってどうやるの？	37
6.5	まとめ	37
第7章	アジャイルな案件を受託する組織職の一事例	39
7.1	アジャイルという制約	39
7.2	しかし部下はそうじゃない	40
7.3	苛烈な戦場で組織職ができるここと	40
7.3.1	誰よりも楽しむ	40
7.3.2	コンディションこそ全て	41

7.3.3	仕事に学習を組み込む	41
7.4	まとめ	41
第8章	関係の質を改善させるためにやってよかった8つのこと	43
8.1	はじめに	43
8.2	いいチームの作り方	43
8.3	まとめ	46
第9章	不安とうまく付き合う	48
9.1	とあるチームのストーリー	48
9.2	不安感に負けてしまうチームと周りの人たち	49
9.3	リスクを早く炙り出した方がいいアジャイル	49
9.4	「デザイン」が更に不安を募らせる	49
9.5	不安感とうまく付き合うには	50
9.5.1	フレームワーク通りにやっているのに、しっくりこない時	50
9.5.2	デザインがボトルネックに感じる場合	50
9.5.3	チームの周りに不安感が漂う場合	51
9.6	終わりに	51
第10章	アジャイルは反復してナンボ！	52
10.1	アジャイルとはなんぞや？	52
10.2	「反復」とは何か	53
10.3	「反復」なきアジャイル	53
10.4	プロダクトバックログは「プロダクト」のバックログ	54
10.5	プロダクトバックログはToDoリストではない	55
10.6	小さな価値とは	55
10.7	まとめ	56
第11章	インセプションデッキからはじめる アジャイル入門	58
11.1	はじめに	58
11.2	開発チームに新規着任	58
11.3	朝会の目的...？	59
11.4	インセプションデッキとの出会い	59
11.5	チーム対面キックオフと得られたもの	60
11.6	キックオフのその後	61
11.7	まとめ	61
第12章	独習アジャイル	62
12.1	独りで学ぶ	62
12.2	独習アジャイルの進め方	62
12.2.1	準備をする	63
12.2.2	スプリントを回す	63
12.2.3	くり返し、進化・深化させる	64

目次

12.3	つまり、タスク管理なんです	64
12.4	もうひとつの重要なポイント	64
第 13 章 アジャイルの始め方～化石エンジニアがアジャイルの「本の探し方」（自己流）を 共有します～		66
13.1	テンケー！（天啓 or 典型）	66
13.2	無知の知	66
13.3	きりがない	67
13.4	やってみた	67
13.5	「本の探し方」説明	67
13.6	まとめ	68
第 14 章 コミュニティの探し方		70
14.1	コミュニティに参加する	70
14.2	コミュニティに入るメリット	70
14.2.1	デメリットはあるか？	70
14.3	どうやって探す？	71
14.3.1	人づてに探す	71
14.3.2	なければ作る	71
14.4	コミュニティの居心地をよくする	72
14.4.1	雑談する	72
14.4.2	アンチハラスメントポリシーを定める	73
14.5	まとめ	73
第 15 章 Agile イベントまとめ		74
15.1	良いカンファレンスに出会う、ということ	76
15.2	他にもたくさんあります	77
第 16 章 スクラムマスターの資格の選び方		78
16.1	そもそもスクラムマスターに資格は必要？	78
16.2	スクラムマスターの資格について	78
16.2.1	認定スクラムマスター (Certified ScrumMaster® /CSM®)	79
16.2.2	認定スクラムマスター (Registered Scrum Master®)	79
16.2.3	Professional Scrum Master™	79
16.3	資格の選び方	80
16.3.1	講師の考え方や理念で選ぶ	80
16.3.2	講師の得意分野で選ぶ	80
16.3.3	日本語で講師と対話できるかで選ぶ	81
16.3.4	研修方式で選ぶ	81
16.3.5	他の選び方について	82
16.4	おわりに	82
第 17 章 アジャイルを勉強した後のキャリアの 5 つのロールモデル		84

17.1	最初に	84
17.2	キャリアの5つのロールモデルを考えることになったきっかけ	84
17.3	アジャイルに関するキャリアの5つのロールモデル	85
17.4	最後に	87
あとがき		88

第1章

アジャイルで幸せになれるのか～IT古典良書を読み解く：「初めてのアジャイル開発」編～

伊藤慶紀

1.1 本書との出会い

こんにちは。SHIFT の伊藤と申します。

「健康とは、できる限りゆっくりとした速度で死に向かうことでしかない。」 - 作者不明

IT 古典良書を読み解くということで、Craig Larman（クレーグ・ラーマン）の『初めてのアジャイル開発 スクラム、XP、UP、Evo で学ぶ反復開発の進め方』を紹介します。何故か、各章の最初に名言が表示されていて、冒頭の名言は第3章 アジャイルからの引用になります。本書は2004年9月発売です。



▲図 1.1 初めてのアジャイル開発書影

当時、筆者はまだまだ若輩者のエンジニアでしたが、いくつかのプロジェクト開発を経験し、規模の差はありますがあくまでも炎上していました。原因は終盤で仕様変更が入ったり、追加

～

漏れがあったり大きな障害が見つかったりと様々でした。何とか納期に間に合っても開発陣はボロボロ、顧客も望んでいたシステムとはズレがあるようですが、これでヨシとする風潮が当時の（ひょっとして、今も!?）ソフトウェア業界でした。

なんとかならないものかと考え、アジャイル開発やテスト駆動開発等に興味を持ち独学で勉強し始めます。当時、「@ IT IT アーキテクト塾 テストファーストの実践」^{*1}でパネラーから紹介されていた書籍の1つが本書で、色々と疑問に思っていたことがスッキリしたことを覚えています。

探したら、該当記事がありました。懐かしいですが本質は変わっていないですね。アジャイルは黎明期でスクラムとXPを組み合わせるのが流行っていました。ペアプログラミング以外は今でも使うべきかと。また、テスターが必要とも書かれています。

1.2 アジャイルは誇大広告か

アジャイルは誇大広告という話があります。以前からある考え方（反復型など）を誇大広告して再利用しているだけなのでは？という問い合わせに対し、本書では「YESでもありNOでもある（P.40）」と書かれています。

どういうことかというと、いわゆるアジャイルな考え方は一昔前の再利用であるため一面ではYESだが、スクラムなどの原則やプラクティスを全体としてみると新しいものなのでNOということです。

ここで学べることは、アジャイルをバズワードのように用いるといわゆる「なんちゃってアジャイル」となり、「話が違うじゃないか」とまさに誇大広告になってしまい誰も幸せになれない結果になります（そそのかしたコンサルは幸せになるケースあり）。正しくアジャイルを用いることで誰しも幸せになれる可能性があがるということです。

今回はアジャイルの代表的な手法である「スクラム」に関する用語が出てきますので用語の意味を学んでおくとより分かりやすいですが、知らなくても考え方は伝わるかと思います。

《よもやま話》

誇大広告といえば個人的には「ビッグデータ」を思い出します。バズワード化した時期に導入して効果をあげた企業が一体何社あるのか… 「M2M」がいつの間にか「IoT」と言葉を変えて領域を増やしたりと、IT業界は不思議な用語が飛び交います。

1.3 アジャイルを導入する主な理由

よく聞かれる質問に「じゃあ、アジャイルにはどんなメリットがあるの？」があります。これにどう答えるのがいいのでしょうか。「第5章 導入理由（P.62）」にあるアジャイル導入理由の見出しから特に大事だというものをみていきましょう。

- ・反復型開発の方がリスクが低く、ウォーターフォール型の方がリスクが高い

→ リスクが高い工程が先に来るのがアジャイル、後に来るのがウォーターフォールとなっています。リスクグラフを見ると分かりやすいです。

^{*1} <https://www.itmedia.co.jp/im/articles/0602/24/news137.html>



▲図 1.2 リスクグラフ：ウォーターフォール



▲図 1.3 リスクグラフ：アジャイル

・最終製品がクライアントの真の希望に適ったものになる

→ 早い段階で評価やフィードバックを繰り返すため、製品が望んだものになる可能性が高くなります。これが、いつも私がウォーターフォールでモヤモヤしていた顧客が望まないものがリリースされる点を解決する方法になると考えています。

・タイムボックスの利点

→ アジャイルは短い期間で区切って開発をするのですが、調査によるタイムボックスを導入するだけで生産性が上がるという利点があるそうです。いくつか理由があるそうですが1つ目は「集中」。締め切りギリギリのときに驚くべき生産性を出す方も多いでしょうが。締切が長いと人はだらけてしまうようです。また、タスクを現実的に対処できる程度のものに縮小し、困難な決定を早く行うようになる効果があるようです。

そして、もう一つタイムボックスの価値は人間の不思議な習性に関連することです。それは人は期限を守れなかったことはよく覚えているが、内容が少しぐらい不足していても気にはしないというものです。100%を要求するウォーターフォールと、優先順位が高い機能から作成し75%でも納期にリリースできるアジャイルを表しているようです。

1.4 アジャイルを使って失敗する方法

いざ、アジャイル開発を始めても失敗することはあります。それでは、どうすると失敗するのかを知っておけば事前に防げるかも知れません。ここでは代表的なアジャイル手法であるスクラム

～

が失敗する方法の見出しをみてみましょう。（第7章 スクラム P.155）

- ・自律的なチームでない。マネージャーまたはスクラムマスターを指揮・編成している

→ こちらは立ち上げではとても難しいです。マネージャーは解決策を提示して指示しないといけないと思いますし、メンバーは逆に指示を仰ぎがちになってしまいます。徐々にでもいいので自律的なチームを作っていくましょう。個人的に一番良くないことが、決めたスプリントバックログを必ず終えるために、スプリント内でスコープ調整などをせずにウォータフォールのように稼働をあげて対応してしまうことです。

- ・スプリントや個人に対して新しい作業が追加される

→ スプリント中は作業を中断させないことが大前提ですが、緊急でどうしてもという場合はバックロググルーミングなどを使いタスク調整をすることが大事だと考えます。単に追加では溢れてしまうだけです。

- ・プロダクトオーナーが参加していない、あるいは判断を下していない

→ 他にもプロダクトオーナーが複数存在したり、最終意思決定が出来なかつたりといった問題も散見されます。個人的にはプロダクトオーナーの意識、熱量不足があるとうまく回らないことが多いようです。

- ・ドキュメントが不十分である

→ アジャイル開発はドキュメントを作らなくていいという話を聞いた方も多いかと思いますが、反文章主義ではなく、成果物として定義していないだけであり、価値があるのであれば作成するべきです。

既にアジャイル開発を行っている方もいると思いますが、上記のリストに思い当たる節が無かつた方！おめでとうございます！「アジャイルで幸せになれる」を体現できるはずです。逆に思い当たる節だらけの方、既に様々な歪みが起きていると思いますが勇気を持って変革していきましょう。

1.5 莫邪の剣も持ち手による

「莫邪の剣（ばくやのつるぎ）も持ち手による」^{*2}という言葉がありますが、どんなに優れた名刀でも持ち手が臆病であったりすると、その真価が発揮できないという意味になります。アジャイルは確かに優れた武器ですが、使い方を誤るとその真価が発揮できません。これは、結果が出ないからと次々とサービスや商品を乗り換える方もいますが、使う側に問題があるというのは往々にしてあることです。

結論としては「アジャイルで幸せになれるが、持ち手による」ということでしょう。

最後に、スクラムが成功する価値について特に大事な個所を引用します。良い持ち手になりましょう。（第7章 スクラム P.153 一部の用語を分かりやすいよう現代風に改めています）

- ・コミットすること

スクラムチームは、そのスプリントの目標を達成することをコミットする代わりに、達成するにはどうするのが一番よいかを自分たちで判断する権限と自治権が与えられる。経営陣とスクラムマスターは、スプリントに新しい作業を追加しないこと、チームに指図しな

^{*2} 「莫邪の剣も持ち手による」。武器は良くても使う人がダメだと効果が発揮できないという意味。「宝の持ち腐れ」ではしきりこないのでよりよいことわざは無いかなと探したところ、ぴったりのものを見つけた次第です。筆者も初めて使いました。

いこと、リソースを提供しディリースクラムで挙げられた障害を迅速に取り除くことをコミットする。プロダクトオーナーは、プロダクトバックログを定義して、その優先順位を付け、次のスプリントの目標を選択するのに際してチームを導き、各スプリントの結果をレビューしてフィードバックすることをコミットする。

・敬意を払うこと

または責任転嫁するのではなく、チームで責任を持つこと。チームのメンバーがそれぞれの長所/短所に敬意を払い、スプリントが失敗しても誰か一人の責任にしない。マネージャーではなくチーム全体が、自己組織化と自律によって、グループで解決策を調べて「個人の」問題を解決するという姿勢をとる。また、専門のコンサルタントを雇って足りない知識を補うなどといった難問に対応するための権限とリソースを与えられる。

・勇気を出すこと

経営陣は、勇気を持って、適用型の計画や方向性を示し、メンバー個人やチームを信用してスプリントを行う方法に口出ししないようにする。チームは自主性と自己管理を必要とする仕事に勇気をもってあたる。

アジャイルを成功させる方法は「コミットすること」に集約されていますが、最終的には、敬意を払う、勇気を出すといった技術ではなくマインドが大事なんだなということが再認識すると共に、IT 業界に○○信者や○○教といった言葉がはびこっている理由もなんとなく分かってしまいました。アジャイル開発をしていない人にも敬意を払いましょう！

そして、本書を執筆していただいた Craig Larman 氏に敬意を払いたいと思います。本書では、紹介した項目以外にもウォーターフォールが新規製品開発に向かない理由、アジャイルを導入する理由・失敗する方法、スクラムが成功する価値について詳細がまとめられていますので、是非手にとってみてはいかがでしょうか。



伊藤 慶紀

大手 SIer にて業務用アプリケーションの開発に従事。ウォーターフォールは何故炎上するのか疑問を感じ、アジャイルに目覚め、一時期、休職してアメリカに語学留学。Facebook の勢いを目の当たりにしたのち、帰国後、クラウド関連のサービス・プロダクト企画・立ち上げを行う。その後、ベンチャーに転職し、個人向けアプリ・Web サービスの PM、社内システム刷新など様々なプロジェクト経験を経て SHIFT に入社。趣味は将棋、ドライブ、ラーメン、花火、読書など

第2章

アジャイルの始め方-続け方

砂田 文宏 @orinbou

対象読者（読んでみて欲しい人）

この内容は、既にバリバリ濃いアジャイルを実践されている人にはあまり役に立たないと思います。できれば以下のような人に読んでいただき、何かヒントになるものがあれば幸いです。（もし無かったらごめんなさい）

- 過去アジャイルに取り組んだがうまくいかなかった人
- 現在アジャイルに取り組んでいるがうまくいっていない人
- これからアジャイルに取り組もうと思っているが、どう始めたら良いか分からず困っている人

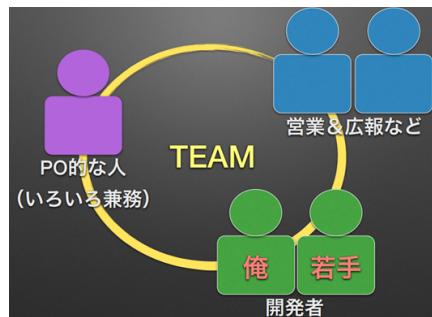
2.1 背景

今よりもっとうまくソフトウェア開発したい。価値あるソフトウェアを利用者に届けたい。ソフトウェア開発を志す人ならば一度くらい、そんなことを考えたことがあるのではないでしょうか？ そんなことができたら本当にどんなに素晴らしいことでしょう。でも現実は厳しく、誰かが（いつ？ どんな目的で？）決めたのかよく分からない開発標準という枠組みの中で、あまり役に立たないと感じながらも決まり事だからと渋々、あるいは必死に不合理と戦い苦しんでいるケースも未だに多いかもしれません。そんな状況に当時の自分は耐えかねてソフトウェア開発から逃亡しようと画策しました。しかし、幸運にも（？）2008年秋に発生したリーマンショックで初めての転職と言う名の逃亡が見事に失敗した後、生きる（食いつなぐ）ために再びソフトウェア開発の仕事に戻ることにしました。日々の作業に追われる中、2012年頃たまたま本屋で目にし、手に取ったのが書籍「アジャイルサムライ — 達人開発者への道」でした。今から遡ること、約10年前のことでした。

衝撃的な内容に興奮して一気に読み終え、気になった部分をマーキングし、付箋を貼りまくったことを今でも覚えています。

2.2 初めてのアジャイル開発

当時は少し景気が悪かったことが幸いし、アサイン先がなく社内で浮いていた人員を活用して自社サービスのプロトタイプを開発するというプロジェクトが立ち上りました。アジャイルサムライを読んで熱に浮かされていた私にとって、これは千載一遇のチャンスだと感じました。上司には是非アジャイル開発でやらせて欲しいと懇願して、何とか受け入れてもらいました。今後は約2週間（スプリント）ごとに動くソフトウェアを届けますと（実際できるかどうか分からなかつたけど）約束しました。当時のスタートアップ時の体制（ご近所さん達）はこんな感じでした。



▲図2.1 スタートアップ時の体制（ご近所さん達）

当時のインプットは、アジャイルサムライとネットを漁って調べた情報だけでした。とにかくアジャイルだかスクラムだかもよく分からぬまま手探りで始めました。まず初めにやったこと（プラクティス？）は次の3つでした。

- 朝会
- スプリント（約2週間）
- ふりかえり（レトロスペクティブ）

たったこれだけ？と思うかもしれません、たったこれだけでも序盤からいきなり躊躇まくりました。まず、朝会が時間どおりに始まりません。上司に声をかけられたPO（的な人）が、頻繁に朝会をスキップしてきました。さらに、スプリント開始時に合意した（絶対したはず！）WIPを無視してどんどんと追加タスクを積んできました。プラクティスが理解されず、乱れるリズム。つのるイライラ。駄々下がる開発者のモチベーション。そんな感じで序盤は、思った以上に思ったように進みませんでした。加えて、私のバディ（相方）である若手開発者が「僕は過去を振り返らない男なので」と（頑なに）ふりかえりを拒否したりもしました。新しいことを始める場合は「たったこれだけ？」でも本当に大変だと感じました。

2.3 地道な改善

序盤の躊躇もありアジャイルごっこにもなっていない状況で何度も挫折しかけました。しかし、相方に懇願して「ふりかえり」を継続して地道に軌道修正を続けたことで、どうにか続けることができました。当時、効果が高かった改善点の例としては下記が挙げられます。

・ご近所さん（プロジェクト関係者）の見える化

ご近所さんの図を壁に貼り同じチームメンバだと再認識することで、課題解決に向けた建設的な相談がしやすくなった。また、スプリントの途中で動くものを見せて早い（そして忌憚のない）フィードバックによる軌道修正が可能になった。

・バックログをプロダクトとスプリントに明確に分けた

PO や利用者に価値がある単位をプロダクトバックログ、それを実現するためスプリントで必要な作業に細分化したものをスプリントバックログとして分けて管理し、かつ見える化したこと、PO のマイクロマネジメント（細かなタスク単位の指示）が減った。

地道な改善を継続的に行い、それが実感できるようになってくると、チームメンバの意識も少しずつ変わってきました。あの自称「過去を振り返らない男」の若手開発者が、ある時ふいに「環境って自分達で変えられるものなんですね。知りませんでした。」と言ってくれました。この言葉に自分も少し救われた気持ちになりました。それ以来、彼は社内の特別な協力者になり、社内勉強会の企画や社外コミュニティでの発表など事あるごとに積極的に協力してくれるようになりました。その後、成功も失敗も沢山ありましたが、それまでと決定的に違ったのは、彼の協力がなかった頃と比べて新しいことを試すまでに要する時間が圧倒的に短くなったことでした。

2.4 学んだこと

変革にリーダー（先達）が必要というのは誰しも思うところだと思います。しかし、それでだけは物事はなかなか思ったように進みません。変革の取り組みを支持するファーストフォロワー（最初の支持者）が1人居てくれることにより、新しい取り組みへの対応速度が全く違ってきます。また、成功も失敗も格好の「ふりかえり」のネタ（むしろ宝物）として前向きにとらえることができ、一人では躊躇して（人知れず）止めてしまうようなケースも激減します。

変革の取り組みを支持するファーストフォロワー（最初の支持者）が存在することで、最初は突飛に見えるような取り組みにも信憑性が生まれ、多くの人達が参加するきっかけとなり、やがて社会運動に繋がる。これは2010年のTEDカンファレンスでアメリカの起業家デレク・シヴァーズ（Derek Sivers）氏が紹介した「社会運動はどうやって起こすか（How to start a movement）」^{*1}で分かりやすく説明してくれています。

2.5 その後の現場で

その後も様々な困難に遭遇しましたが「ファーストフォロワー」を見つけて、まず「2人から始める」ことで、私なりのアジャイルな取り組みを続けることができました。

2.5.1 事例1：前職の現場にて

Step1：若手エンジニアと【2人】で朝会をはじめる

派遣先の現場で若手エンジニア（顧客）と2人で朝会をはじめる。週1でふりかえりを実施して継続的な改善をし続けたことで、その成果が注目され複数チーム（3チーム：10人）へ普及。

Step2：シニアエンジニアと【2人】で勉強会へ参加する

^{*1} https://www.ted.com/talks/derek_sivers_how_to_start_a_movement/ TED How to start a movement. Derek Sivers 社会運動はどうやって起こすか

成果に注目してくれた派遣先の現場でシニアエンジニア（顧客）を誘って2人で社外の勉強会へ参加する。新たな知見や気付きを共に得て、あるべき姿について（飲みながら何度も）議論した。社外から講師を招待してチームやエンジニアに向けた勉強会を開催して成長や改善の糧となるインプットの場をつくった。Jenkins を導入して、これまで手動で行っていたビルトやインストーラー作成などを自動化することで、エンジニアの作業負荷やストレスを低減し、尚且つヒューマンエラーによる品質低下も激減した。

2.5.2 事例2：現職の現場にて

Step1：担当プロジェクトで若手エンジニアと【2人】でスクラムを画策

自分が担当するプロジェクトでスクラムに興味を持ってくれた若手エンジニアとなんちゃってスクラムをはじめスクラムマスターの役割を担う。

Step2：次期プロジェクトでPMと【2人】で画策して顧客にスクラムを提案

なんちゃってスクラムのリズムに慣れた頃合いと、次期開発フェーズのプロジェクト開始タイミングを見計らって、これまでの成果に注目してくれたPMと画策して顧客にスクラムを提案して受け入れてもらう。（いろいろとあったが）結果として顧客に価値を届けることができた。

Step3：社内でアジャイルを推進する役割（CoE）を【2人】で担う

最近では社内へアジャイルを推進・啓蒙する役割を担っている。かつてアジャイルコミュニティで知り合った仲間が偶然同じ会社に（しかも驚くべきことにはほぼ同じタイミングで）入社していたこともあり、ここ2~3年は二人三脚で支え合いながら取り組んでいる。今度は、私がフォロワー側に回ることも多い。

2.6 これから始める人へ

正しいアジャイル、スクラムかどうかを気にし（過ぎ）ないで欲しいと思います。確かに「アジャイルソフトウェア開発宣言」や「XP」や「スクラム」などの原則や価値を理解することは重要です。同様に理論を学び理解することも重要だと思います。しかし物事は全てゼロイチで割り切れるほど単純ではありません。私は理想と現実の間でバランスを取っていくことが大切だと考えています。アジャイル柱の（あの）平鍋氏も「（アジャイルなんて）今の仕事で使えないじゃん！」という境遇にかつて陥ったことがあるそうです。そんな時「今の仕事を今よりうまくすることはできる。」という言葉に救われ、アジャイルに拘り過ぎることをやめ、アジャイルの要素を使って「いきいきと仕事がしたい」「お客様と喜び合える仕事がしたい」と気持ちを切り替えたそうです。私もその言葉に何度も救われました。もし興味があれば下記の記事も読んでみてください。（実は私の記事ですw）

<https://www.manaslink.com/articles/15011>

2.7 まとめ

これまでアジャイルを学び、拙いながら実践し続けてこれたのは、その時々で良き理解者（ファーストフォロワー）に恵まれたことが最大の要因だと考えています。与えられた境遇の中で「ファーストフォロワー」を見つけて、まず「2人から始める」ことで、今日まで実践し続けることができました。運もあるかもしれませんのが、自分の意志で前進し、試行錯誤して足掻き続けて困っている

と、不思議と良いフォロワーに出会えることが多かったように思います。社内だけでなく、社外コミュニティなどでも、助けてくれるフォロワーはきっと居ると思います。そう信じて、まずは1人フォロワーを探して2人からアジャイルをはじめてみてはいかがでしょうか。



砂田 文宏（すなだ ふみひろ） @orinbou <https://twitter.com/orinbou>
ブログ：<https://blog.orinbou.info/>

認定スクラムマスター（CSM） / 認定スクラムプロダクトオーナー（CSPO）

独立系SIerでテックリードやスクラムマスターなどを生業としています。社内へアジャイル開発を啓蒙するべくAgilityCoEメンバとしても奮闘中。リーダー塾7期卒業生。AWS(SAP、DOP)とk8s(CKA、CKAD)チョトデキル。趣味はキャンプ△とサイクリング。バスケも好き。(最近は専らNBA観戦)

第3章

非エンジニアの人から「アジャイルって早く作れるんでしょ？」って聞かれた時の私の一次回答

荒川健太郎@

3.1 「アジャイルって早く作れるんでしょ？」

こんにちは。荒川です。ウイングアーク1st株式会社でプロセス改善エンジニアの仕事をしています。

私はソフトウェアの開発に携わっていない社内の人とも幅広く関わらせていただいているのですが、先日いわゆる”非エンジニア”な仲間との雑談中にこんなことを聞かれました。

仲間「アジャイルって早く作れるんでしょ？」

私「う、うーん…（どう返答したらいいのだろう）」

しばらく言葉に詰まった私は、

私「ちょっと10分くらい時間もらって説明していいですか？」

と前置きしてその人にも伝わるようにアジャイルについて説明しました。その内容をまとめたものが今回の記事です。

3.1.1 「早く作れる」とは何か？

「アジャイル」という言葉は本当に厄介（親しみを込めて）で、ちゃんと説明しようとするととても10分では説明できません。もっといと私自身が全てを理論立てて説明できるとは言い切れない部分もあります。

したがって今回の説明については「早く作れるんでしょ？」の問い合わせに対する答えだけに注力しました。（アジャイルとアジャイル開発は違うだとか、アジャイルはプロセスじゃないとかそういうことはひとまず横に置いておきます）

まず私はその仲間に「早く作れる」の「早く」のイメージを確認しました。

- とりあえず動く（バージョン0.1みたいな）状態のモノが早く完成するのか
- 企画段階で盛り込んだ機能が全て備わっている（いわゆるバージョン1.0相当な）状態の

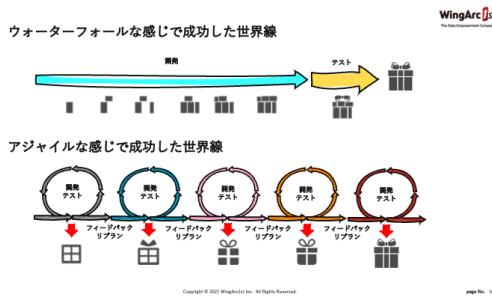
モノが早く完成するのか

仲間の回答は「後者」とのことでした。

やっぱりそうかと思った私は、夏頃の新人研修で使った資料を使いながら説明を開始しました。

3.1.2 どっちのプロセスが正しいか

注：コチラ (<https://flxy.jp/article/5984>) の記事にもある通り「ウォーターフォールとアジャイルは対となる概念ではない」っていうことは踏まえた上で、あえて今回はウォーターフォールと対比しています。



▲図 3.1 ウォーターフォールな感じで成功した世界線

上下で違うプロセスを踏んで同じプロダクトが完成させたことを表した図です。両プロセスとも一番右のプレゼントアイコンが完成品（バージョン 1.0）であると認識ください。

そして仮定の話として、この完成品はとてもよく出来ていて、ビジネス的に大成功（売り上げがとても良い）と言えるクオリティのモノであるとイメージしてください。

さて「アジャイルが良いぞ、アジャイルが良いぞ」と言われて久しい昨今、どちらのプロセスが正しいと言えるでしょうか。

答えは、

どちらも正しい。お客さん喜んでるんだから。売れてるんだから。

そうです。どちらも正しいんです。

良いモノが作れるならウォーターフォールだろうがアジャイルだろうが、ここで例示していないやり方だろうがなんだっていいんです。顧客に喜んでもらって儲かることが、われわれ民間企業の目的なわけですから。

プロセスはあくまでもプロセスであって、大切なのは顧客に価値を届けることだと私は考えます。

3.2 アジャイルの誤解と定着しない理由

3.2.1 アジャイルはいつ完成するかわからない

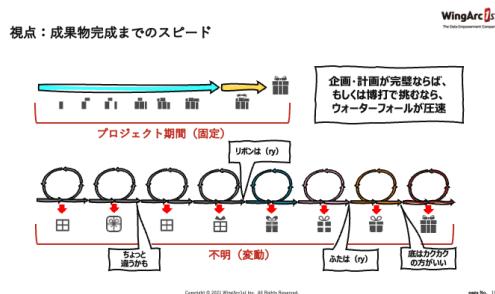
一般的にアジャイルな動きには以下のような利点があると言われています。

- 外部の変化に俊敏に対応できる（不確実性に強い）

- 早くから収益を見込める可能性がある
- 別プロダクトの開発の際、再現性の精度が高まる

こういった利点に注目すると「やっぱりアジャイルいいね」「アジャイルだといいモノ作れそう」となりそうですが、話はそんな単純ではありません。

前章で使用したスライドの下部、すなわちアジャイルなプロセスの図は便宜上 5 つの輪っか（イテレーション=繰り返し）でバージョン 1.0 が完成していますが、現実的にはもっとたくさんのイテレーションが必要となるでしょう。



▲図 3.2 成果物完成までのスピード

(この図を見せた時に非エンジニア仲間から「あ、そっか！」という反応をいただいたのは気持ちが良かった)

アジャイルはフィードバックを得る機会をたくさん作らないといけないため、開発に直結しない業務（プランニング、ふりかえり、レビューなどの非コーディング作業）の回数は増加し、確実にリードタイムは大きくなります。そして本当に顧客に満足してもらえるクオリティに持っていくまでの期間（輪っかの数）は未知なのです。

...

この時点でいったん「アジャイルって早く（完成形を）作れるんでしょ？」という仲間の問い合わせに対する答えを伝えることが出来ました。

答え：アジャイルはいつ完成するかわからないのです。

これで当初の目的は達成できたわけですが、興が乗った私はもう少し話を続けました。

3.2.2 アジャイルがなかなか定着しない理由

要は、アジャイルは「良いモノを作れる確率を上げられるかもしれない」「早期から収益が見込めるかもしれない」けれども「開発業務以外の仕事が多くて」「開発スピードは上がらないであろう」シロモノであるということです。

私は日本の組織でアジャイルがいまいち定着しない理由は、プロセスの方法論ばかり注目されて、それ以前のアジャイルマインドの醸成の重要性が軽視されているからだと考えています。

- 自分の業務以外のタスクが多くて忙しくてなんかジレンマ
- あまり開発に集中できない
- 結果、開発スピードが上がらない

といった感じでアジャイルに悩む声を私はたくさん聞いてきましたし、私自身非常に共感しています。

3.3 アジャイルになるには

3.3.1 面倒だけどやるという覚悟

ここまで説明したように、

「良いものが作れるかもしれない VS. 面倒くさい」

この図式で後者からの解放をとる組織が多い中、それでも前者を重んじアジャイルを採用したい場合どうすれば良いのでしょうか。

まずは関係者全員がアジャイルマニフェストに書かれている内容を理解し、自分たちが作ろうとしているプロダクトにはアジャイルが必要だと納得している状態になる必要があると思います。

繰り返しになりますが、アジャイルは「短いイテレーションで少しづつ作る」「決められた行事（プランニング、ふりかえり、レビュー）をこまめにやる」といった方法ではありません。アジャイルな開発を目指すには「アジャイルマニフェスト」とそこで謳われている価値や主義に基づいた12の原則を関係者全員が理解し実践する状態になる事を目指すという組織としての覚悟が必要となってきます。

こういったことからアジャイル界隈では有名な、Don't just do "Agile", be "Agile".（アジャイルするな。アジャイルになれ）という言葉があるんですね。

関係者全員が心技体でいう「心」の部分を理解しなければ、アジャイルは成り立たないです。

なので「それアジャイルで作る必要あるの？」と言った自身への問いかけもとても重要な私は思っています。作るモノ（完成形）は明らか、外部環境からの影響もない、それほどフィードバックも必要ない。そのようなモノは当初の企画通りに集中して作ってしまった方が、スピード、コスト、品質面でも成功の確度は高く、アジャイルが出る幕はありません。

3.4 まとめ

まとめます。

- 「Don't just do "Agile", be "Agile".」を関係者全員が理解する
- 「アジャイルは早く作れないし面倒くさいけど良いモノを作りたいからやる」という覚悟を関係者全員が持つ
- そのプロダクトの開発はアジャイルに進めた方が効果がある（ビジネスで勝てる）ことを関係者全員が納得している

この条件を満たして初めて組織はアジャイルなプロセスへの一歩を踏み出せるんだと思います。

...

仲間「なるほどですねー。全然思ってたのと違った。すごくよくわかった。」

と言ってくれたので嬉しかったです。

まあこの話はかなりデフォルメされていて多角的な視点にも欠けており、まだまだアジャイルの入り口にすぎないのですが…

その辺はまた別の機会にお話しできればと思います。

3.4.1 頑張って作ったものは売れて欲しいわけです

最後に私の立ち位置としては、

「良いモノが作れるならね…最初は面倒かもしれないけどね…それぞれのプラクティスにはちゃんと理由はあるわけだから…見方とかマインドとか変化させて…みんなアジャイルになろうよ！」
って感じです。

だって、

「良いものが作れるかもしれない < 面倒くさい」だからアジャイル嫌だ
こんなの勿体無いじゃないですか。日々頑張って作ったのに「お客様喜ばない＆売れない」っていう事態は避けたいですよね。

「良いものが作れるかもしれない > 面倒くさい」だからアジャイルになってみよっかな！

こうあるべきだと思います。みんなの頑張りは報われるべきです、絶対。



荒川健太郎 @

ウイングアーク1stでプロセス改善とソフトウェアテストの自動化に取り組んでます
/ Certified ScrumMasterR / Certified ScrumDeveloperR / JBA 公認 E 級審判 / 全日本スノーダイビング協会会長

第4章

ぼくのアジャイル Q&A 100 本ノック

木下 史彦

アジャイルコーチが「アジャイル」に関連する質問にひたすら答えていきます。

4.1 ドキュメントに関する Q&A

Q: アジャイル開発であってもドキュメントは作成すべきでしょうか？

A: ドキュメントとひとことに言ってもさまざまな種類のドキュメントがあります。手はじめに現状作成しているドキュメントを以下の3つに分類してみましょう。

1. 引き継ぎのためのドキュメント
2. 自己防衛のためのドキュメント
3. コンプライアンスのためのドキュメント

ウォーターフォール開発のように開発のフェーズによってチームを分けないのであれば引き継ぎのためのドキュメントは必要ないでしょう。

チームが信頼関係をもって仕事をできるなら、合意事項を事細かに文書化し訴訟リスクに対応するような、自己防衛のためのドキュメントも必要ないでしょう。そんなドキュメントを作っている時間があったら信頼関係を築くことに力を注いだ方が得です。

法令によって定められたドキュメントは作らないわけにはいきませんので、プロダクトバックログに積んでおきましょう。

これ以外にも重要なドキュメントはあります。ドキュメントはWordやExcelで書かれた形式的なドキュメントに限りません。たとえば、コミットログやPull Requestのコメントも立派なドキュメントです。これらには「なぜ変更したのか」あるいは「なぜしなかったのか」を記載しておくことで、ソフトウェアのメンテナンスが格段にやりやすくなります。

4.2 プロダクトオーナーの役割に関する Q&A

Q: POが忙しすぎて役割をまとうことができません。どのような対策を考えられますか？

A: まずPOの役割をいくつかに分割してみましょう。POの役割（ロール）はプロダクトの価値を最大化することですが、これを実現するために『スクラムガイド』にはいくつかの仕事（ジョ

ブ) が例として示されています。

1. ビジョン・ゴールの伝達
2. PBI の作成
3. PBI の優先順位づけ
4. PBL の見える化
5. ステークホルダーマネジメント

その上でこれらの仕事を「PO に絶対にやってもらいたいこと」と「PO 以外のチーム全体で支援するできること」の 2 つに分けましょう。

例えば、ビジョン・ゴールがしっかりと伝わっていれば PBI の作成や PBI の優先順位づけは開発者がきるかもしれません。PBI の細かな受入条件については開発者が作成して、PO が確認するというようにしてもよいでしょう。

『ビジョナリー・カンパニー ZERO』には OPUR (One Person Ultimately Responsible : 最終責任を負う者) を任命すべきだと書かれています。PO はまさに OPUR だと言えます。一方で、OPUR と同じぐらい重要なのが、誰もが隣のために「歩道を雪かきする」文化があることだと同書には書かれています。自分の仕事に対して完全な責任を引き受けつつ、お互いの歩道の雪かきを助ける文化が大切だということです。

4.3 見積り（時間とポイント）に関する Q&A

Q: 時間の見積りとポイントの見積りの違いがよく分かりません。相対的に見積もったとしても結局、時間に換算するのではないでしょうか？

A: スクラムで見積りをするのはベロシティを計測することが目的です。

少し長くなりますが、実例で説明します。

たとえば、ある仕事（仕事 A）があったときにそれを「3 ポイント」と見積もり、その仕事をこなすのに結果的に 6 時間かかったとします。1 ヶ月後、仕事 A と同じような仕事 B をやることになりました。仕事 A をやったことでコツが分かっていたので、2 時間でできそうだと思ったとします。ポイントも 3 分の 1 に減らして「1 ポイント」にしました。これをやってしまうとベロシティはどうなるでしょうか？6 時間でこなせる仕事は「3 ポイント」分のまま変わりません。

仕事 B をあくまでも規模を基準に「3 ポイント」と見積もったならどうなるでしょうか？3 ポイントを 2 時間でこなすことができるのであれば、6 時間あれば 9 ポイント分の仕事ができることがあります。

前者（時間で見積もる）の問題点はベロシティの変化が分からなくなるという点です（時間で見積もっているのでベロシティは常に一定）。後者（規模で見積もる）ですとチームの改善が進んでベロシティが上がっているのか、それとも何かチームに課題があってベロシティが下がっているのかが分かります。

4.4 ベロシティに関する Q&A

Q: ベロシティの考え方として、スプリント中に終わらなかった PBI は 0 としてベロシティにカウントしませんが、次のスプリントで終わらせた場合、そのストーリーのポイントを丸々カウントするのでしょうか？その場合、本当は 2 スプリントかけて終わらせた PBI が 1 スプリントの

成果として計上されるのでベロシティが極端に上下しませんか？

A: はい。上下して OK です。

スプリントで終わらなかったというのは異常な状態ですのでベロシティが極端に上下することによって、その異常状態が見える化されます。

スクラムは問題を明るみに出すためのフレームワークです。問題が分かるのはいいことです。もしも、問題があるのに問題がないかのように取り繕おうとするなら、最初からスクラムを使わない方がいいでしょう。

4.5 スプリントに関する Q&A

Q: 1週間のスプリントでは「出荷可能なインクリメント」を作成することができません。スプリント期間を2週間に延ばしてもよいものでしょうか？

A: そういう理由でスプリント期間を延ばすことはおすすめしません。

スプリント期間中に出荷可能なインクリメントを作れないということはよくあります。しかし、それをよしとはせず「どうすれば1スプリントで出荷可能なインクリメントができるか」を考える改善のきっかけにしていくことが重要です。

改善のポイント

- 開発スキルの向上
- スウォーミング（全員でひとつの PBI に集中）
- PBI をより細かく分割（抽象化能力、設計力、本質を見抜く力が必要です）

スクラムは問題を明るみに出すためのフレームワークです。スプリント期間を短くすることでより多くの問題を明るみに出すことができます。スプリント期間を延ばすことによって、せっかく明るみに出した問題を覆い隠してしまってはいけません。

4.6 メンバーのモチベーションに関する Q&A

Q: 私はスクラムマスターです。スクラムイベントに積極的でないメンバーがいるのですが、どう促すと良いでしょうか？

A: 積極的になれない理由によって対応が変わってくると思います。

イベント趣旨や参加の仕方を単に知らないのであれば

→スクラムやアジャイル開発に関するレクチャを行う

心理的安全性が確保されていないでなくして参加しづらいのであれば

→ワーキングアグリーメントをメンバー全員でつくる

特定の人が話しそぎることで会話に参加しづらいのであれば

→ワーキングアグリーメントを決める

→話しすぎる人に対して個別にコーチングする

グループワークが苦手で会話に参加しづらいのであれば

→ファシリテーションの工夫によって均等に話せるようにする

例) 考える時間と発言の時間を分ける

チェックイン・チェックアウトで参加のハードルを下げる

スクラムが嫌い、もしくはスクラムに疑いの目を持っているのであれば

→スクラムマスターが困りごとを個別に 1on1 を行い、信頼関係づくりからはじめる（スクラムをやってもらう前に、その人の関心事や困りごとを聞いて、抱えている問題を一緒に解決するような動きをするとうまくいく場合が多い）

SM として積極的でないメンバーの内面状況が全く予想できないのであれば

→スクラムマスターが困りごとを個別に 1on1 を行い、信頼関係づくりからはじめる

4.7 スクラム全般に関する Q&A

Q: 何ができていたら、自分たちはスクラムができていることになりますか？

A: 逆に質問です。何ができていたら、料理ができることになりますか？

インスタントカレーが作れたら OK？ フランス料理のフルコースが作れる必要がある？ それとも、独自のレシピを発明する？ 調理師の資格は必要？ 料理に限らず、英語の学習、スポーツ、などなど・・・。

何でも同じです。スクラムも同じことです。スクラムができていることを目標にしてしまうといつまで経っても自信を持って「できています」と言えるようにはならないでしょう。目的にあわせて目指すべき達成度（目標）があるはずです。目的にあわせて目標を自分たちで考えて決めていればよいと思います。

4.8 ノックのつづき

紙面の都合で 7 本しかノックできませんでした。

Agile Studio では視聴者の方から寄せられたアジャイルの悩みに、アジャイルコーチも一緒に悩みつつも解決策考えていく「アジャイルカフェ@オンライン」というイベントを定期開催しています。

<https://www.agile-studio.jp/post/agile-cafe>

続きはこちらでお待ちしてます。



木下 史彦 @fkino <https://twitter.com/fkino>

永和システムマネジメント Agile Studio アジャイルコーチ

居飛車党、ニコン党、日本酒党。純米酒をあたためて呑むのが好きです。

第5章

第3章 アジャイルの実践

この章では、アジャイルの実践として、アジャイルであるために用いられるいくつかのフレームワークや手法を説明します。決してこれらがアジャイルにおける全てではありませんが、アジャイルを実践する上でそれをイメージする助けになることを目的としています。また、陥りやすいポイントや重要なポイントにおいて必要に応じて補足し、本書で得たキーワードをもとに実践できる姿勢を整えます。第一歩というよりは、歩き始める前に「よーいドン」の「よーい」をしていただけるように執筆しております。

この章をどう使うかについては、もし読んでみてわからない単語や説明が多ければ、各節のタイトルを覚えていただき、アジャイルについて経験し学びを進めていく中で、その節の名前が出てきたときに立ち返ってみてください。何が重要だと書かれていたのかを改めて読み、実践に繋げることができるでしょう。そうではなく、なんとなく理解して読み進めることができたり、すでに実践していることがあれば、自分が想像していたものと何か違うところはないか、実践している中で書かれているものと違うところはないか、自分の理解と異なるところを意識して読んでみてください。この説明があなたの状況に必ずフィットするわけではありませんが、筆者の考えるポイントや概要と異なるところがあれば、その点について一考し、改善できる余地があるかもしれません。

この章のまえがきとして最後に、アジャイルを実践する上でのマインドについて記します。そのマインドとは、アジャイルを実践する方法もアジャイルに改善していくというものです。型に則るだけでなく、自分たちに合った型を探していくこともアジャイルに行ってみてください。筆者は筆者なりの考えを持って執筆しておりますが、アジャイルを実践する方法は多くあり、同じ名称でも説明やポイントが異なることがあります。アジャイルを実践する上で、知った方法をやってみるだけでなく、ぜひその方法を改善できないか考えてみてください。改善する上では、その方法が目的とするものや、実現したいことに立ち返ってみてください。実はチームでやってみているときに、型に則ることが最適ではないかもしれません。逆に、チームでやってみていることが型から外れすぎて、目的を達成しにくくなっているかもしれません。チームにとって、価値のあるプロダクトを提供することのできる、よりよい方法で実践できるように、アジャイルに振り返り調整してみてください。

5.1 6.1 アイデアを出す

プロダクト開発をする上で、まず初めには必ずアイデアから始まります。アジャイルを実践する上で、第一歩となるアイデアの出す方法について筆者の経験したいくつかを紹介します。

5.1.1 6.1.1 カスタマージャーニーマップ

カスタマージャーニーマップとは、一言で表した困りごとに対し、ペルソナを定めて、各フェーズごとに行動、関わるものや場所、思考や感情なども併せて書いていくというものです。困りごとを解決したいという前提がある中で、ユーザーの解決前の体験に焦点を当てて書き、イメージしやすくすることで「本当に課題があるのはどの部分の体験なのか」の目星をつけます。

このときのポイントが2つあります。1つ目は、最初に一言で表現した困りごとは、必ずしもユーザーの解決したいことの本質ではないということ。2つ目は、困りごとを解決する手段を前提にしないことです。前者については、カスタマージャーニーマップを客観的に書くことで、実は困りごとだと思っている「ここ」が、実は別の時系列にあるものが本質で、そこから派生した困りごとであるケースがあります。後者については、手段ありきのカスタマージャーニーマップになってしまふことで、困り事の本質に気づかず、今ある説を補強するだけになってしまふことがあります。この2つのポイントを押さえてカスタマージャーニーマップを書いていくことが、良質なアイデアを出す基板になります。

とはいって、特に2つ目にあるように手段ありきで考えてしまいやすいことが多いです。そんな時に便利な、The Job Story Format というものがあります。まずはカスタマージャーニーマップを一通り書いてから、困っているようなポイントを複数挙げます。それぞれのポイントにおいて、①それはどんな場面で、②ユーザーは何をしたいのか、③それをしたいのは何ができるからなのか、を書くことでユーザーが達成したい仕事を明確にします。困りごととその解決策への先入観のある状態から少し離れて、ユーザーが特定の場面では何をしたいのか、何ができるからそれをしたいのかと、具体的かつ客観的にユーザーのニーズを見極めます。

5.1.2 6.1.2 リーンキャンバス

リーンキャンバスとは、フォーマットに沿ってアイデアを書き出しながらそのアイデアを検証するというものです。すでに何かしらのアイデアがある前提で、そのアイデアをブラッシュアップしていくことに用いられます。その過程でアイデアが大きく転換する可能性もあります。

これにはいくつかの題目に沿った枠が用意されており、それを埋めていくことで活用します。1番目に顧客セグメントとアーリーアダプター、2番目に課題と既存の代替品、3番目に独自の価値提案を埋めていきます。誰の何をどんな独自性を持って解決したいのかを明確にし、ビジネス面におけるいくつかの事項を考慮に入れます。このとき、ソリューションを何にするかが1つの肝になりますが、ソリューションは次節.1.3で紹介するブレインライティングを活用してから定めることがおすすめの手段です。

リーンキャンバスについて短く語ることは困難ですが、初めて使用する際はぜひ枠を埋める順番に注意してみてください。枠が多いためどこから埋めたらいいかわからないといった疑問に対して、本書で書かれている埋め方で埋めることで埋めやすくなるでしょう。

5.1.3 6.1.3 ブレインライティング

ブレインライティングとは、フォーマットに沿った用紙をチーム内で回し、前の人と被らないようにしつつ、特定のテーマに沿って発想を派生させたり逆転させたりしてアイデアを出すものです。短い時間制限を設けた上で行い、1ターンの間にできる限り必ず全ての枠を埋めていき、その

アイデアを捻り出す過程でアイデアが柔軟なものになっていきます。

ブレインストーミングと同様、どんなアイデアでも歓迎され、突拍子のないアイデアからヒントを見つけられる可能性を大きくします。6.1.2で紹介したリーンキャンバスにおいて、今ある課題をテーマにし、そのソリューションについてブレインライティングをすることも可能です。

5.2 6.2 アイデアをかたちづくる

アイデアを出しただけでは、実際の開発に進むのは困難です。出したアイデアをかたちづくり、磨き上げ、開発に移る準備をしましょう。

もちろん、開発を始めてからもアイデアを検査する必要があります。いわゆるピボットをして、アイデアの方向性を変えることもあるでしょう。この節では、アイデアをかたちづくる2つの方法を紹介します。

5.2.1 6.2.1 エレベーターピッチ

エレベーターピッチとは、エレベーター内の短時間で上司や役員にアピールしても相手に必要な情報を伝えることのできる、1分程度の短いピッチのことです。様々出したアイデアを集約し、シンプルに要点をわかりやすく詰めることで、聴衆がそのアイデアの概要を短時間で把握することができるようになります。

フォーマットを調べると、表現にプレがあり一見するとどのフォーマットが良いかわかりにくいですが、一貫して次の7つを含むことが多いです。

- ①顧客の望むニーズや課題
- ②対象顧客
- ③プロジェクトやプロダクトの名称
- ④カテゴリやジャンル
- ⑤重要な利点と対価を払う説得力のある理由
- ⑥競合
- ⑦競合との違い

これらをつなぐと、次のような文章になります。「①を望む②向けの③は、④です。これは⑤ができ、⑥とは違って⑦があります。」という文章です。

5.2.2 6.2.2 顧客インタビュー

顧客インタビューは、その名の通り対象顧客へのインタビューのことです。アイデアをかたちづくる上で、客観的な声を取り入れ、必要な修正を加えながらアイデアを成長させます。顧客インタビューでのポイントは大きく2つです。1つ目は意見を誘導しないことで、例えば「こういう機能どうですか？」ではなく、「どんな機能が欲しいですか？」と聞くことです。2つ目は得た意見をただ全て取り入れるのではなく、チームでこれは確かにそうだと思ったものや、ユーザーに共通していた意見などを中心に取り入れていくことです。そうすることで素直な意見を集めながら、それでいて意見を全部取り入れて凡庸になることなく進められます。

5.3 6.3 スクラム

スクラムは、チームが価値を生み出すための軽量級フレームワークのことです。アジャイルと同様に価値基準が存在し、スクラムの理論に基づいています。アジャイルであるためによく採用されるフレームワークです。アジャイルマニフェストのようにスクラムガイドというガイドがあり、初版から数年おきに更新を繰り返しています。ガイドだけでなく、研修・認定資格・多数の本が世に出ています。

この節では、スクラムにおける 2 つ、作成物とイベントについてのみ触れます。スクラムの理論・価値基準・スクラムチームについてはほとんど触れておりません。アジャイルの実践として、6.1・6.2 で扱ったように、実戦における具体的なことを中心に扱います。

5.3.1 6.3.1 スクラムの作成物

スクラムでは 3 つの作成物が定義されています。それは、①プロダクトバックログ②スプリントバックログ③インクリメント、の 3 つです。これらの作成物を作成することを通して、プロダクト開発を進めます。この節では、作成物について説明しながら、付随する概念についても触れます。

1 つ目の作成物、プロダクトバックログ（以降、PBL）とは、プロダクトと 1 対 1 対応で存在し、一意なプロダクトバックログアイテムの優先順位を持っているリストのことです。プロダクトバックログアイテム（以降、PBI）は、PBL を構成する要素のことで、プロダクトの改善に必要なものです。最初期には、プロダクトとして何もない（＝何も価値がない）状態から、「プロダクトとして価値を生むために」必要なものが PBI になります。PBL はプロダクトゴールをもち、プロダクトの将来の状態として、計画におけるターゲットになります。

PBL において特に重要なのは PBI の粒度です。よくない状態として、PBL にある全ての PBI が粗い、もしくは全て細かい状態があります。理想的なのは、上は細かく、下に行けば行くほどだんだん粗くなっている状態です。なぜなら、まだ見通しが聞きやすい直近の PBI を細かい粒度で分割するのは適切でも、見通しが立たないようなまだまだ後に行う PBI を細かくしても、無駄になることが多いからです。また、必要以上に細かくすることは、コストを要する上に柔軟性を失い易くなります。また、直近の PBI を適切な粒度にすることは Ready な状態に置くと言います。Ready な状態とは、誰が見ても迷いなくタスクに分解でき、認識に齟齬も起きない程度の粒度のことです。これにより、その PBI に期待するプロダクトの価値を実現することをより確実にします。

PBL は定期的に、また随時、メンテナンスされることが望ましいです。そのメンテナンスのことをプロダクトバックログリファインメントと呼びます。リファインメントでは、着手が近い PBI を Ready な状態にしたり、PBL の中で PBI の位置を変更し、優先順位を調整します。

2 つ目の作成物、スプリントバックログとは、スプリント中の全てを集約するもののことです。のちに説明するスプリントプランニングで作成するスプリントゴールや、先ほどの PBL にある PBI とそれを分解したタスク、そのタスクの状態も、ここに記録されます。

3 つ目の作成物、インクリメントとは、1 つの PBI のタスクが全て DONE になり、PO によって受け入れられた後の PBI のことです。プロダクトの改善における踏み石とも呼ばれ、この踏み石を 1 つ 1 つ増やし、踏んでいくことでプロダクトがより価値を生むことが期待されます。

5.3.2 6.3.2 スクラムイベントとスプリント

スクラムでは5つのイベントも定義されています。それは、①スプリント②スプリントプランニング③デイリースクラム④スプリントレビュー⑤スプリントレトロスペクティブ、の5つです。スクラムイベントは、スクラムの作成物の検査と適応をするための公式の機会であり、このイベントを通して作成物を検査し、状況に適応します。スクラムイベントは、複雑さを低減するために、同じタイミングと時間で開催されることが望ましいです。

1つ目のイベント、スプリントとは、スクラムイベントの入れ物となるイベントのことです。これは期間でもあり、定期的なリズムを作るのに用いられます。多くのプロダクト開発では2週間としてスプリントを用意しますが、1ヶ月以内の決まった長さであれば問題ありません。期間を短くすることで、複雑さを低減し、できる限り予測可能な範囲で計画を立てて実行し振り返ります。

5.3.3 6.3.3 スプリントプランニング

2つ目のイベント、スプリントプランニングとは、スプリントにおける一番最初のイベントであり、そのスプリントにおける計画を行うイベントのことです。まず初めにスプリントゴールを策定し、それに応じて必要なPBIを選択します。

望ましくないパターンが2つあります。①PBIのいくつかを先に選択することが決まっていて、それをまとめたようなスプリントゴールになることと、②スプリントプランニングの時間を超過して、計画する時間を延長してしまうこと、です。①については、PBIがPBLの中で適切な順序に並んでいることを前提としてしまっていて、本来はそうではなく、前回のスプリントの振り返りも踏まえた上で、「今この瞬間、次のスプリントでは何を達成すべきなのか」というものをゴールとして設定することが望ましいです。②については、計画が時間内に終わらないのであればそもそもPBIがReadyに状態になっていないことや、あまりにも不確実性が高く計画しきれないため、早く動いたほうがいい状況があります。

この2パターンはどちらも、不確実性の高い中でアジャイルにプロダクトを開発していくために避けたいパターンです。①に陥ると、逐次状況が変わるような不確実性の高い中、前回のPBLリファインメント結果による先入観が入ってしまいます。スプリントプランニングは新しい1日の頭に置かれることが多いですが、それは一時的な先入観から逃れ、できるだけ最適な選択を取るためでもあります。②に陥ると、計画できる状況にない中で、最も計画を手助けする手段と言える、その計画を実施することを妨げてしまいます。また、そのような状況下で計画する時間を長くとっても、状況は好転することが一向にないことが多いです。

5.3.4 6.3.4 デイリースクラム

3つ目のイベント、デイリースクラムとは、毎日決まった時間に行われ、スプリントプランニングで計画したタスクを調整しながら、スプリントゴールに対する進捗を検査し、必要に応じてスプリントバックログを適応させるイベントのことです。具体的には例えば、昨日はスプリントゴールのために何を達成したのか、今日は何をスプリントゴールのために貢献する予定なのか、その達成の障害になっているものはないか、確認します。

5.3.5 6.3.5 スプリントレビュー

4つ目のイベント、スプリントレビューとは、インクリメントをデモして、さまざまな人からレビューを受ける場であるイベントのことです。このイベントの目的は、スプリントの成果をレビューを通して検査し、今後どう適応するかを決定することです。さまざまな意見が出たときは、顧客インタビューのように、適切にそれらを受け止めてから PBL に反映させます。

5.3.6 6.3.6 スプリントレトロスペクティブ

5つ目のイベント、スプリントレトロスペクティブとは、主にスクラムのプロセスとプロダクトに対して改善する方法を計画するイベントのことです。これはよく振り返りとしてイメージされますが、振り返りを通して次の意思決定や行動を改善することが重要です。実際に撮られる手法はとても多く、例えば、アジャイルマニフェストの4つの価値を指標として、その重なりを15象限のベン図にし、プラスとデルタについて記載するような方法もあります。一般的によく用いられているのは KPT や Fun Done Learn です。

5.4 6.4 アジャイルの実践あとがき

本書の冒頭で触れたように、アジャイルとはやり方ではなくあり方です。この章では、そのあり方を実践するための方法について紹介してきました。再三になりますが、決してこれらの方法だけが全てでなければ、紹介した方法そのものも記載した説明だけにとどまりません。しかしながらこれらは、アジャイルの実践として鍵となることが多い方法です。特に、最後に紹介したスクラムは、プロダクト開発をアジャイルに行う上で非常に有用なフレームワークです。ぜひ、本書のキーワードをもとに「よーいドン」の「よーい」をして、「ドン」といつでも一步目を踏み出せるようにしてみてください。アジャイルを実践する者として、共に歩む日を楽しみにしております。



amixedcolor/エイミ（保 龍児）

<https://twitter.com/amixedcolor>

株式会社 Relic エンジニア

新規事業開発、アジャイル、AWS、完全没入型仮想現実、歌、漫画が好きです。

第6章

アジャイルではリカバリーってどうやるの？

松永 広明

ある日の研修で、受講生のひとりからこんな質問をいただきました。

“アジャイルでは、遅れが出た場合のリカバリーはどうやるのでしょうか？”

うーん、実によい、趣のある深い質問です。実に深い。松崎しげ●の目尻のしわのように深いです（事実無根）。

この問題を理解するには、いくつかのステップが必要です。まずはそれを挙げてみます。

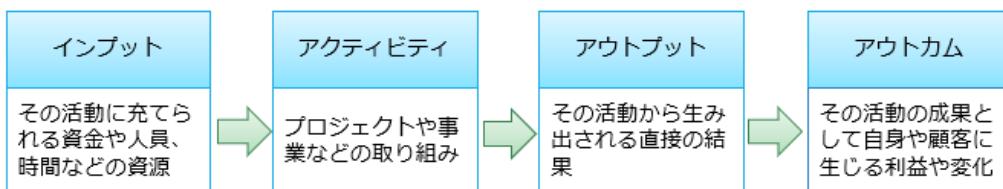
1. アジャイルは価値が大事。アウトプットよりアウトカム。
2. スプリントの目的は「スプリントゴール」
3. 価値をできるだけ毀損せず、やらなくても良いことを最大化せよ。

順に解説します。

6.1 アジャイルではアウトプットよりアウトカムが大事

「アウトプット」は、プロジェクトなどの活動から生み出される直接の結果や成果物のことです。一方で「アウトカム」は、その活動の成果として、自身や顧客に生じる利益や変化のことです。

図にするとこんな感じ。



▲図 6.1 ロジックモデル

ソフトウェア開発で言えば、インプットはお金だったり開発者たちの工数だったりします。ア

アウトプットはアクティビティ、すなわち日々の開発作業から作られたソフトウェアシステムそのものです。アウトカムは、そのシステムを使うことで得られる利益や、削減できるコストや、生活の変化だったりするわけです。

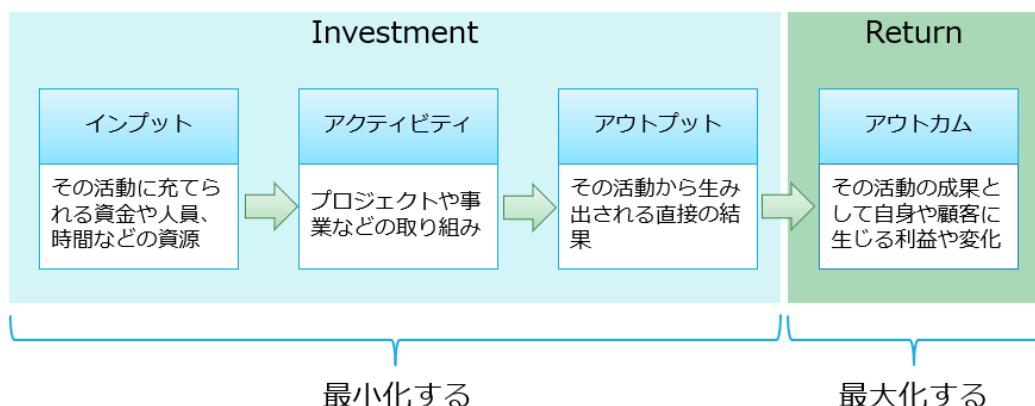
Return On Investment (ROI) という言葉があります。日本語だと投資対効果などと言ったりしますが、これはその投資に対してどれだけの利益が得られるのかを示す数値で、数値が大きいほど効果が高い投資だということを示します。計算式は下記の通りです。

$$\text{ROI} = \text{Return} / \text{Investment}$$

簡単ですね。いくら Return が大きくても、Investment も大きいと、ROI すなわち投資効果は減ってしまうことがわかります。つまり、ビジネスにおいては単に Return があることだけが大事なわけではなく、Investment が極力少ないものである必要があることがわかると思います。

では、図 1において、Investment はどれで、Return はどれでしょう？ Return というのはその活動の成果として得られる利益などを指すですから、この図で言えば”Return”は一番右の「アウトカム」だけとなります。わかりますか？ つまりみなさんが作っているソフトウェアシステムは、作って納入するだけで価値があるわけではなく、運用されて、何らかの効果が得られて初めて”Return”、すなわち価値となる得るわけです。ソフトウェアシステムは、それ単体では Investment、つまり投資にすぎないということです。ええ～～。

そしてさらに、分母である Investment は小さい方が ROI は大きくなるので、みんなはなるべく仕事をしないほど良いという事になりますね。やったぜ！



▲図 6.2 ROI の最大化

でも別の見方をすると、アウトプットをいくらたくさん生み出しても、それがアウトカムにつながらなければ何の意味も無いどころかお金の無駄遣いだったということになりますね。どれだけたくさん作ってもゴミはゴミということです。逆に言うと、アウトカムが最大化できるならアウトプットは小さくシンプルであるほうが良いということになります。

6.2 スプリントの目的は「スプリントゴール」

前項からわかるとおり、スプリントの目的は価値を生み出すことであり、ソフトウェアを作ることではありません。スクラムでは、スプリントで生み出されるこの「価値」を、スプリントゴールとして設定するわけです。つまり、スプリントの目的は、より多くの機能を作るとか、より長時間働くとか、計画された機能をすべて完成させるとかではなく、価値を届けることだと言えます。スプリントゴールとはスプリントの「Why」です。「なぜ」そのスプリントをやるのか？ です。つまり、スプリントゴールとは、そのスプリントが生み出そうとしている「価値」だということです。

私は大好きな LiSA さんが Catch the moment のリリースイベントで手渡してくれたサイン入りポストカードを持っていますが、彼女はこのサインをものの数秒で書き上げるわけです。これなんて売れば数万円ぐらいの値が付いてもおかしくありません（絶対にそんなことはしませんが）。数秒で数万円。たとえば5秒で5万円とすると、1時間で3600万円（！）です。ですがそんなことよりも、彼女とほんの短時間でも直接目を合わせて会話をしながら手渡してもらったという思い出が詰まっていて、実際のところ Priceless です。つまり何が言いたいかというと、価値というのは単純に作業時間や作業量とは単純にリニアに変換できるものではないということです。

ということはですよ？ 前項の説明によると、スプリントゴールさえ達成出来るならば、スプリントプランニングで作ると決めた機能が、決めた通りにすべて完成していなくても良いことになりますよね？ 目的が達成出来るなら手段はひとつじゃないよね？ ということです。

余談ですが、未だに世の中では、部下やベンダーに対して単に忙しく働くことだけを要求する管理職や顧客が多いのは嘆かわしいことです。たとえば毎日夜遅くまで働く部下が評価されたり、ベンダーの労働時間の多寡を測定したりする人知っています。また、そういった人たちに迎合して、言われたことだけをやるという態度を取るエンジニアもまだまだ多いのが現状です。コードはいかに多く書くかが大事なのではなく、いかに少ないコードで大きな価値を出すかが大事なのです。筆者は、そういったエンジニアが評価される社会を作りたいと考え、活動しています。

6.3 やらなくても良いことを最大化せよ

アジャイル宣言にはこういう記述があります。あえて英語版で。

"Simplicity--the art of maximizing the amount of work not done--is essential."¹¹

テキトー意訳すると、「やらなくても良いことを最大化する技術が不可欠です」とかそんな意味です。最大限にやらない、というのは、ストレートに言うとやることを最小限にしろということです。これは、上述の Investment を最小化せよという話と符合します。そのためにはシンプルさが必要だとアジャイル宣言は説いているわけです。一方で、いくらやることを減らすと言っても価値までは減らしてはいけないので、そのためによりシンプルなソリューションを考え、スプリントゴールで策定した価値を毀損することなくやることを最小化すべしと言っているわけです。

たとえば、当初フリーワード検索で考えていたけど、よく考えたら選択肢は5つしかないから、これだったらチェックボックスかラジオボタンで良くね？ とかそういうヤツです。たぶんその方が文字コードチェックとか要らないので実装も簡単ですし、なおかつ必要なものを検索するという当初の価値は全く失われていません。

これぐらいの単純な例なら良いですが、実際にはシンプルかつ価値の高いソフトウェアを作る

というのはけっこう大変です。まず顧客が何を価値だと感じているのかをよく知ることが必要です。顧客は自分にとっての価値が何なのか、実はよく分かっていないからたりするので、単にヒアリングするだけではなく本当の意味で顧客に寄り添う必要があります。また、シンプルな設計というのは実は最も難しかったりします。読みやすく理解しやすいコードだったり、結合度が低くて凝集度が高く、変更容易性の高い設計だったり、日々リファクタリングを重ねて常にシンプルさを保つ努力が必要だったりします。アジャイル宣言にもこう言及されています。

"技術的卓越性と優れた設計に対する不断の注意が機敏さを高めます。"

6.4 アジャイルでリカバリーってどうやるの？

で、冒頭のこの質問に戻ります。

極論すれば、アジャイルは「如何にして作らずに済ませるか」の技術です。よく「アジャイルでは何をやるかよりも、何をやらないかの方が大事」といったことが言われます。やらなくて良いことはやらない、作らなくても良いものは作らない。いたずらに仕事の量を追うのではなく、その価値を見極めて本当に必要なことだけをやっていけば、物事はシンプルになる、ということです。そうしていかに少ないコードで大きな価値を出すか（つまり ROI）を考えいくと、作業時間も短くすることができるかも知れません。アジャイル開発では、そうやってリカバリーを行うのです。まあもちろんそれだけとは限らないし、それが上手く行かない場合もあるのは否定しませんが。

そもそも、冒頭の質問の背景には、従来型開発で骨の髄まで染みついた、「コストと納期とスコープは守り通さねばならない」という考え方があると思うのです。つまり、スプリントプランニングで作ると決めた機能は、スプリント終了までに決めた通りにすべて作らなくてはならないという考え方をしているのだと思うのです。

こういう考え方だと、遅延が出てしまった場合は残業をしてリカバリーするしかなくなってしまいます。残業は最悪です。残業は、なにか問題があったときに、安易にその場しのぎのリカバリーをする方法です。こういった方法が恒常化しているチームは、問題（この場合は遅延）の根本的な原因にリーチしようとしないので、改善していかないので。アジャイル宣言にある「サステナブル・ベース」にも反しますね。なので、生産性が上がらないと言っているチームのコーチに入ったときに、私が最初にコーチすることは、残業を禁止することだったりします。こうすることで、彼らはいかに残業をせずに問題を解決するかを考えるようになります。

6.5 まとめ

まとめると、だいたい下記のようになります。

- リカバリーのために残業をするのは悪手。
- スプリントゴールは、そのスプリントの価値でありコミットメントである。
- 価値は、作業時間や作業量と比例しない。
- ROIを高めるためには、なるべく仕事をしないようにしよう。
- アジャイル開発でのリカバリーは、価値を毀損することなく、よりシンプルなソリューションを考えることで実現できる。



松永 広明 @qoragile <https://twitter.com/quoragile>

屋号：LSA CONSULTiNG

なんちゃってアジャイルをこよなく憎むフリーランスのアジャイルコーチ。LiSA っ子。

<https://www.facebook.com/lsaconsul>

<https://www.linkedin.com/in/lsaconsulting/>

第7章

アジャイルな案件を受託する組織職の 一事例

福田朋紀 (@chinmo)

僕はもう長いこと、受託開発の会社で、開発組織の組織職として働いていることになる。

うちの会社には営業がいないので、仕事を取ってくるのも僕の仕事だ。仕事を取ってきたら、部下に担当してもらう。以上。

あらためて文章にしてみると、あまりにあっさりしていて、虚しさすら覚えるけれど、存外、僕はこの日々を楽しんでいる。その秘密は、”アジャイル”にあるんだ。

7.1 アジャイルという制約

アジャイルに関する仕事しかしない。

これはね、もう「そうすることに決めちゃった」としか言いようがない。色々理屈はつけられるのだけど、そういった建前とか建て付けとは別にして、僕はもうそういう仕事しかしないし、周囲にもきっとそうに違いないと思われている。周囲にそう考えてもらうまでには随分長い時間がかかったけど、一度そうなってしまえば、誰も僕に口を挟むことは無くなり、それどころかそういった仕事は最初に僕に回してくれるようになった。

制約は、平凡な世界に刺激的な視点を提供し、一挙手一投足に、そうせざるを得ない理由を生む。結果として、僕という存在が、あるメッセージを周囲に受け取らせることになる。「あいつと関わると、好むと好まざるとに関わらず、アジャイルとやらに向き合わざるを得なくなる」というわけだ。

アジャイルな仕事がなかなかなくて、と嘯く組織職がいたとして、その人はつまるところ、「そうすること」にしていないだけなのだと思う。例えば僕の最初のアジャイルな仕事は、誰に求められたものでもなく、自分で提案したものだった。

アジャイルが向いていない、と誰かが考えている仕事があるとして、僕が「そうであること」をやめるかというとそんなことは一切ない。自分を曲げてまで何かに取り組むほど、勿体無い時間の使い方は無いじゃないか。それがソフトウェア開発であり、世の中に何かの価値を生み出すことを狙っているのなら、それは十分に複雑 (Complex) で、アジャイルな自分はしっかりきっちりお役に立てるはずだ、というのが「そうすることにした」僕の揺るがないスタンスなのだ。はっはっは。

7.2 しかし部下はそうじゃない

アジャイルであることは自分で決めたことだけど、ただ、部下はそうじゃない。たまたま、すとんきょうな事を言っては周囲と波風を立てる奴の下につくことになっただけだ。ご愁傷様です。

別のところに行きたくなつたらいつでもどこに行っても良いのだよと言ってみたところで、僕たち日本人は相当な条件が揃わない限りブツブツモヤモヤグチグチしながらもそこに留まり陰鬱な雰囲気づくりに貢献しがちなわけだ。となると組織職としてできることは、

我々の「旅」が、そこそこ趣に満ちている事をを目指すしかない。

お互い、楽しい！とか、やりがいがあるとか、そんな少年の様な爽やかなことが言える歳でもない。でもまあ、老後にポツポツと思い出しては人に話して聞かせたり、公園のベンチでフツ、とニヤついたりする様なことを体験してもらいたい。

7.3 苛烈な戦場で組織職ができること

ふと振り返った時に、何らかのポジティブな記憶を呼び起こす仕事を体験してもらいたい、というのが偽らざる思いではあるのだが、僕の部署のメンバーとして僕が取ってきた仕事を担当するということは、極めて困難な仕事に挑むことを意味している。

例えば、仕様が決まっていない、要求が曖昧、日程感がバグっている、顧客も課題も解決策もマーケットもフィットしていない、やりたいことに対して金が足りない、そもそもどうしたら悩みが解決するのかわからない、とにかく助けてほしい、この難局を乗り越えるための方法がアジャイルと聞いてここに来ました、などなど、靈媒師も頭を抱える様な危険な案件ばかりだからだ。

勝ち馬に乗るような仕事が無い、ということは、案件として担当するほぼ全てのプロダクト/サービスが、奮闘虚しく市場価値や問題の解決策を生み出せずに消滅する事を意味する。なんと、基本、負け戦前提なのだ！！

不安に苛まれ、それ聞いちゃダメだろ、みたいな問い合わせを投げかけ、利害関係者と衝突し、本気度合いのギャップに苛つき、形ばかりの作業に反対し、無茶苦茶なスケジュールに意を唱え、KPIの無い機能リリースロードマップを蹴り飛ばし、サポートへのクレームやネット上のバッシングに耐え、最後には人員削減や開発チームの解体を経て、サービス終了の屈辱に耐えながら少人数でのクロージング作業に従事する… そんな仕事したい奴なんているだろうか。

そしてそんな過酷な現場に放り込まれる部下たちに対して、他人のサラリーマン人生にそれなりに大きな影響を与える仕事を選択した組織職にできることとは、一体、何だろうか？

7.3.1 誰よりも楽しむ

目の前に広がるのが混沌とした戦場だとして、立ち尽くす部下たちの後ろから飛び出してきて「はいはい、盛り上がってますな！ 大好物です！」と真っ先に手を挙げて首を突っ込み、問題の構造を発見する度に小躍りし、何を面白いと思っているのか目をグルグルさせながらメンバーに話し、「絶対面白いから楽しんでおいで！」と送り出す、それが組織職の最初の仕事だと思っている。

面白そうだからやってみようぜ、と口説いている風には見えるが実際には上司部下の関係だ、任命しているに過ぎない。ただ、自分の熱意を相手にウザめに伝え、あわよくば相手の心に小さな火を灯したいのだ。

7.3.2 コンディションこそ全て

残業という概念を、自分の中から消そう。

部下の最高のパフォーマンスは、ポジティブな精神から生まれる。そのためには睡眠だ。部下は定時で家に帰っても、睡眠時間を削って何かしているかもしれないが、それは彼らの自由であり、組織職が彼らの睡眠時間を削って良い理由にはならない。

労働基準法ギリギリで部下の稼働を見事に回した経験があるなら、当たり前だが残業なしでも余裕で回せる。それだけで、部下のパフォーマンスは飛躍的に向上する。

ステークホルダーに残業してくれと言われても断ると良い。断れるか！と思うかもしれないが、普通に断れる。断ろう。当然、そのようなことを言われないように事前にさまざまな手を打つておくのがスマートだ。しかし覚えておいて欲しい。残業して稼ぐ一瞬の進捗は、リフレッシュした状態でやる場合よりも品質が低く、次のスプリントどころか翌日あっという間に取り返せる程度のものだ。

部下からもっと働きたい、とか言われることもある。そういう時は、その元気があるなら勉強して腕を上げて、同一時間で多くの成果を挙げられるようになって給与を上げるかもと給与もらえるところに行けるようになれ、残業すると利益が下がるから君の評価も下がるぞ、と答えよう。

新人なら1年、ベテランでも3年くらいそうしていたら、定時になるとスッとチャットからいなくなってくれる。たまに残ってる奴は仲間ともうちょっとだけお喋りしたい勢だ。

7.3.3 仕事に学習を組み込む

契約の中に、この開発チームは業務を遂行するために必要な学習を行う、ということを織り込もう。と言っても当たり前だけど。仕事上でわからないことがあれば学ばなければならぬ。ソフトウェア開発は、サービス開発は、価値の創出は、事業の創造は、すべて日々の学びの繰り返しだ。

仕事中に学ぶ、まあそうだよね、と思うかもしれない。しかしあなたも一步踏み込んで、部下に、「就業時間中に積極的に勉強会やトレーニングをしよう、できれば毎日やろう。勉強会や研修や、カンファレンス参加や、社会貢献活動に費やす時間以外で、自分達の計画をし、ステークホルダーとコミットし、ソフトウェアを作ろう」と言うとしたらどうだろう。最初、部下は戸惑うはずだ。

しかし、日々学び、学びを継続することを求めて学びの習慣をつけてもらうことが、彼らを自身に溢れるポジティブなエンジニア集団に変えていく。

指すは、3-4時間集中してプログラミングしたら、契約分の成果を挙げるチームだ。あとは学びを深めたり、他者を助けたり、雑談しているのが理想だ。継続していくば、そのうち、少々要領が良いくらいのエンジニアでは追いつかないレベルの実力とチームワークがあるチームになると信じてやっている。

7.4 まとめ

アジャイルな案件を受託するなら、覚悟を決めて困難を楽しみ、部下の最高のパフォーマンスを引き出すことに集中し、責任を引き受けて学習に投資しよう。

大丈夫、誰も死ないし、責任は組織職が取ればよいのだから。



福田 朋紀 @chinmo <https://twitter.com/chinmo>

リコー IT ソリューションズ株式会社のアジャイル推し。アジャイル鳥取、JISA アジャイル開発 G、CoderDojo 鳥取チャンピオン、演劇企画夢 Ores、TRPG のキャリアが一番長くて 30 年以上

第8章

関係の質を改善させるためにやってよ かった8のこと

小糸 悠平

8.1 はじめに

みなさんの考えるいいチームってどんなチームでしょうか。価値を早く提供できるチーム？ 自律したチーム？

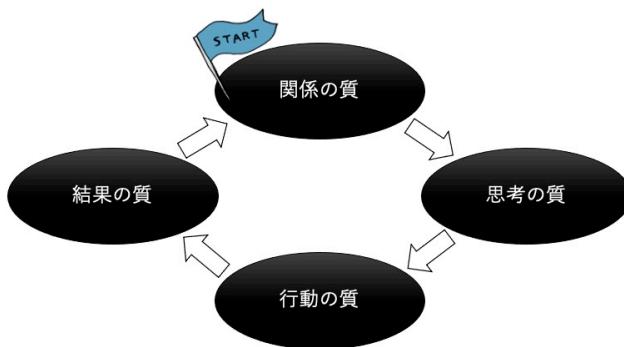
色々な考えがあると思うんですが、最初からいきなり「価値を早く提供できるチームになれ」とか「自律したチームになれ」と言われても何から手付ければ良いんだろうって感じませんか？ 私は長年そんな感想を持っていたのですが、スクラムマスター/ジャイルコーチという立場で多くのスクラムチームに携わってきた中で、最近ちょっとだけですがコツみたいなものが見えてきたので整理してみました。

8.2 いいチームの作り方

ダニエル・キム氏によって提唱された成功循環モデル^{*1}では、最初から成果（結果の質）を求めようとするとチームが直接的な数字作りに走り、チーム内に摩擦や対立が生まれ（関係の質の悪化）、心理的安全性が確保されず（思考の質の悪化）、行動が消極的になり（行動の質の悪化）、さらに結果が悪くなる（結果の質の悪化）という"失敗の循環"が生まれると述べられています。

反対に、"成功の循環"として、チームの関係性が良くなることから着手する（関係の質の向上）と、考え方が前向きになることで多くのメンバによる気付きが生まれ（思考の質の向上）、チームの積極的・自律的な行動に繋がり（行動の質の向上）、成果が生まれる（結果の質の向上）。成果が生まれることでますます関係性が良くなる（関係の質の更なる向上）と述べられています。

^{*1} 成功の循環 <https://www.humanvalue.co.jp/keywords/theory-of-success/>



▲図 8.1 成功の循環

これを最初の章で述べた良いチームに当てはめると、「価値を早く提供できる」や「自律した」は結果の質であるので、最初から結果を求めるのではなく、まずはチームメンバ間であったりチーム外のステークホルダとの関係性を良くする（関係の質の改善）ことで、結果の質にも良いものになっていくと言えます。関係の質の改善のため、チームの状況に合わせて適したアイデアを考えて実行していくのですが、私のこれまでの経験の中で、多くのチームに通用するであろう関係の質を改善するアイデア 8 つをご紹介したいと思います。

やってよかったこと① スクラムの勉強会を定期開催する

- 内容：
 - チームで 1 スプリントに 1 時間くらいスクラムについての勉強会を実施しました。
 - これまで実施してきたテーマは以下があります。
 - * 直近のスプリントで発生した事象に対するケーススタディ
 - * スクラム関連書籍の読書会
 - * スクラム関連イベントの動画鑑賞 など。
- 効果：
 - チームのスクラムへの理解度が一定の水準となり、議論の冒頭にチーム全員の認識を合わせる時間が減少します。これにより議論の本質に時間を割けるようになります。
 - メンバが知識を得ることで幸福度向上にも繋がります。

やってよかったこと② 競合システムの勉強会を開催する

- 内容：
 - プロダクトに先発の競合サイトがある場合は、そのサイトを皆で触って意見を交換する時間を設けました。操作者が画面を共有し、皆で気付いたことを言い合い、自分たちのプロダクトの価値向上に繋がるものは PBI に追加していくというものです。
- 効果：
 - PO とチームが対等な立場で意見を交換できるので距離が縮まります。
 - 皆で見ることで、気づきをチームで共有でき、チームの一体感が強くなります。
 - PO がこれまでうまく言語化出来ていなかったビジョンの共有に役立ちます。
 - サービスへの理解が深くなり、チーム内の認識齟齬が減ります。

やってよかったこと③ PBI をチームみんなで作る

- 内容：
 - PO がリファインメントまでにユーザーストーリー考えて PBI 書いてをほぼ 1 人で実施していたのをリファインメントに持ってきてもらうようにした。
 - リファインメントの時間をエピック単位のユーザーストーリーマッピングを作成するワークを行う時間にした。
- 効果：
 - チームでワークをすることで一体感が強くなります。
 - PO の負荷が下がり、開発チームとのコミュニケーションの時間が確保出来るようになります。
 - PO がチームメンバを信頼してチームに任せせるようになります。

やってよかったこと④ 失敗してみる

- 内容：
 - スプリントゴールを高くして失敗してみる。
 - シンプルですが、一番効果が高いです。私のチームでは 4 スプリントに 1 回くらい失敗しようとしてます(笑)
- 効果：
 - 失敗しても大丈夫と言うチームの心理的安全性に繋がり挑戦する雰囲気が生まれます。
 - 普段のスプリントより振り返りに熱が入り、コミュニケーションを改善させるアイデアが多く生まれます。
 - ステークホルダもチームの状況に興味を持ってくれるようになります。特にマネージャはめっちゃバーンダウンを見てくれるようになります。ステークホルダーがスクラムに過度な期待を持たなくなる効果もあります。

やってよかったこと⑤ サービスを使ってもらっている所を見る

- 内容：
 - サービスイン後、利用者が使っているところを見学させてもらい関係者とプロダクトの価値を向上させるためのワークを行った。
- 効果：
 - ステークホルダとの距離が縮まります。
 - チームのプロダクトへの思い入れが強くなりプロダクトの価値を高める議論が活性化されます。
 - 今後の開発に向けての意思統一が図れます。

やってよかったこと⑥ 順番に休む

- 内容：
 - チームメンバが順番に休む。(休暇を取らない場合でもその日はスクラムチームに関わらない)

- 私たちのチームではイベント有無に関係なく順番に休んでもらっていました。
- 効果：
 - なんでも知っているメンバがいなくなるので、お互いに質問する環境が出来ます。
 - 誰かに依存することなく、チームメンバ全員が自分の意見を発信するようになります。
 - 休んだメンバが翌日に第三者視点でおかしな点に気付き指摘することが出来ます。

やってよかったこと⑦ ステークホルダにチームの弱みを見せる

- 内容：
 - 振り返りで挙がったチーム、個人の問題点をステークホルダに共有する。
- 効果：
 - ステークホルダがチームに興味を持つようになります。
 - 続けて行くうちにステークホルダの悩みも打ち明けてもらえるようになり、チームがステークホルダのことを理解できるようになります。

やってよかったこと⑧ 每スプリント小さな変化を入れる

- 内容：
 - 同じ状況が続くと、チームのメンバは飽きてきてしまうので毎スプリントチームに変化を与えてみました。
 - これまでやってみた内容
 - * メンバの入れ替え (PO ⇄スクラムマスター、自チームのスクラムマスター ⇄他チームのスクラムマスターなど)
 - * ベロシティの数値をこっそり変えて多めの SP を積んでみる
 - * スクラムマスターが理由をつけてデイリースクラムに出ない
 - * 値段の殆どない PBI の優先度を上げてみる
 - * デイリースクラムに見学者を招く
 - ロールの入れ替えなど最初に気付くものもあれば、振り返り時に種明かしするものもあり。
- 効果：
 - 変化に適応するためにチーム内の会話が増えます。
 - 事前に変化の内容を明かさない場合は、メンバがゲーム感覚で変化の内容を見つけようとするので雑談が増えます。

8.3 まとめ

結果を出し続けられるチームを作るためには、チームの関係の質を上げていく必要があります。今回、8つのアイデアを紹介しましたが、皆さんのチームで実行している関係の質を向上させるためのアイデアについて情報交換させていただき、より良いチーム作りに繋げていければいいなあと思っています。



小糸 悠平 (Koito Yuhei)

<https://www.facebook.com/yuhei.koito/>

認定スクラムマスター（CSM） /

認定スクラムマスター（LSM） / 認定プロダクトオーナー（LSPO）

現在は、KDDI でアプリや Web サイト開発のスクラムマスターをやっています。

チームの幸福度を上げることを日々考えています。幸福にならないことはやりたくない。。。。

スクラム関連のイベントにはなるべく顔を出すようにしています。見かけたら声掛けてください。

第9章

不安とうまく付き合う

中村麻由

アジャイルに初めて取り組んだ時、期待していたような成果がすぐに出ず、焦りを感じたことはありませんか？プロジェクト序盤から問題が噴出し、むしろ進捗が遅く感じたことは？何より、周りから「アジャイルって速いんじゃなかったの？」などの批判を受け、チーム全体が意気消沈してしまったことはないでしょうか？

そして、なかなか改善する兆しが見られず、結局アジャイルの取り組み自体が終了してしまったことは？

プロジェクトの最初からつまづき、目に見える成果に辿り着けないと、チーム全体が不安に襲われると思います。しかしその不安が、実はアジャイルがうまくいっているサインだと考えるとどうでしょうか？

職場で初めてアジャイルが導入された際、私も同僚もかなり苦戦をしました。うまくいっていないかった原因は様々でしたが、当時、クライアントを含めチーム全体を覆いがちだったネガティブな雰囲気をよく覚えています。それは「不安」、そしてそれが解消されないことによる「不満」でした。

アジャイルを学ぶための書籍や研修では、主に「何をするべきか」を知る事ができます。しかし、チームとしてアジャイルのマインドセットを育み成長する過程で、誰もが体験しがちな「感情」についてはあまり触れられません。講習を受け、体系的な知識を得たチームですら実戦の中で不安を感じるならば、アジャイルのことをあまり知らずにステークホルダー（利害関係者）になった人は、更に大きな不安を感じるかもしれません。

アジャイル文化を根付かせる鍵は人間同士のコミュニケーションや信頼関係です。そのため、誰かが感じている「不安」など、感情に向き合う事は大切な一歩となります。筆者は主に実践を通してアジャイルを学びましたが、本稿ではデザイナーという立場から見えた景色と学び、そして初めてアジャイルに取り組むチームが陥りがちな「不安」について考察します。

9.1 とあるチームのストーリー

アジャイルを始めたばかりのチームが、スプリントデモを始める場面を思い浮かべてください。今回はスプリント2が終わったところです。

POが緊張した面持ちで、会を始めようとしています。前回は特に見せるものもなく、気まずい感じで終わってしまいました。部屋にはデザイナーと開発者が座っていますが、少し揉めている

ようには話を交わしています。部屋にステークホルダーが入ってきました。デモが始まります。

まずはこのスプリントで取り組んだストーリーのデモです。アプリを立ち上げるといいくつかのカード型 UI が表示されます。中のメッセージはまだダミーテキストで、見た目も簡素です。「これがデザインなの？」ステークホルダーの一人がコメントをしました。「あ、いえ、まだデザインは反映されていません。」慌ててデザイナーが答えます。「デザインが仕上がってこないので、後回しにして機能だけ先に作っています。」開発者が続けました。ステークホルダーは少し不満そうです。「開発のデモは以上です。」予定よりも早く終わりそうです。「デザインの方から、モックアップを見せられますか？」PO に聞かれ、デザイナーは少し焦った感じで返事をしました。「あ、はい…。」静まり返った時間が流れます。デザイナーが、しどろもどろになりながらデザインプロトタイプを見せ始めます。「まだラフなスケッチですが…どうでしょう？」「ボタンの位置をあと 2 ピクセル、上にあげた方がいいんじゃない？」デザインマネージャーがコメントします。「あ、でもまだラフなので…。」結局、特に有意義なフィードバックもなく、デモは終わりました。あまり盛り上がることもなく、次第に次のデモから参加する人は減っていきます——

9.2 不安全感に負けてしまうチームと周りの人たち

これは、私が体験した複数のプロジェクトの様子を組み合わせたものです。デザイナーは十分な時間がもらえず、まだ決めていない部分ばかりコメントされて不満でした。開発者はデザイナーがなかなか一つの案に決めてくれず、これ以上進められないと不満を漏らしていました。PO は終わるはずだったストーリーが終わらず焦っていて、デザインと開発のズレに気付いていませんでした。ステークホルダーは、アジャイルを「今までと同じやり方でとにかくスピードがあがる」と思っていました。

何より「アジャイルは正しいはず」という空気に、誰も「うまくいっていない」という声をあげられませんでした。結果思ったように開発は進まず、次第に自信を無くしたチームは、アジャイルはもうやめたほうがいいんじゃないかと思うようになりました。

しかし、本当はこのチームはいいスタートを切っていたはずです。

9.3 リスクを早く炙り出した方がいいアジャイル

「アジャイルな見積りと計画づくり」という本のなかでマイク・コーン氏は、従来の計画方法が崩れがちな理由の一つに「不確実性を無視している」ことをあげています。逆にアジャイルがうまく機能するチームが実践しているのは「検査と対応」です。この本の中では、架空のチームが出てくる課題に冷静に対応しながら計画を修正していく様子が描かれています。

アジャイルではこのチームのように早い段階で「うまくいかないこと」を可視化させ、対処することでリスクを減らしていくことが大切です。しかし、実際には不確実性が多い中で問題が噴出すると、人は不安になるものです。

9.4 「デザイン」が更に不安を募らせる

デザイン・開発の協業経験の有無によっても不安度は変わります。今まで「デザイン仕様」を受け取ってから開発を始めていた場合、真っ白な状態からデザインが始まることに不安を感じる人は多いでしょう。

もちろん開発者が作業を始める前に「ディスカバリー」を行い、コンセプト作りをしているケースがあるかもしれません。しかし、ある程度デザインの方向性が固まっていたとしても、基本デザインという作業は発散と収束の繰り返しです。しかも一度ではなく、視点を変えながら何度もその過程を繰り返すことで最終的なソリューションに落ち着きます。つまり、発散させている間は常に「何が正しいのかわからない」という状態が続きます。

開発も「模索」する時期があるとは思いますが、なるべく時間をかけず最適解を目指すでしょう。単純に性質の違いなのですが、これに気づいていないと、お互いに不満が出てきます。デザイナーは十分な時間がないと感じたまま、開発者を待たせているプレッシャーに悩み、開発はなかなかソリューションが決まらないことに苛立ちを覚えます。

9.5 不安全感とうまく付き合うには

こう聞くと「ならばデザインと開発は別に作業した方がいいのでは?」と思うかもしれません。しかし、その場合「技術的に実現できないデザインをしてしまう」「開発段階で落とされて、デザインの大変な意図が抜けてしまう」というリスクを後回しにすることになります。ここで疑問点や不協和音を見ないフリをしてしまうと、後々自分たちに戻ってくることになります。つまり不安になることが起った場合、多少時間がかかったとしても未来の自分たちを助けるつもりで対処してしまう方がいいのです。

それでは不安とうまく付き合いつつ、前に進むにはどうすればいいのでしょうか？ 全ての不安を取り除くのは難しいかもしれません、過去の取り組みでいくつかうまくいった例をあげます。

9.5.1 フレームワーク通りにやっているのに、しっくりこない時

アジャイル初心者のチームにありがちのが、研修で習ったプロセスをそのまま再現しようとし、理想通りに進めず「失敗した」と思ってしまうことです。あくまで理論は理論であり、実践では予測通りに行かないのは当たり前のことです。その際、チームが不安を口に出せずプロセスに従うことが優先されると、息苦しい時間が続くことになります。

振り返りで「うまくいっていない」ことを口に出せる心理的安全性が大切です。アジャイルの原則から離れすぎてはいけませんが、習ったプロセスに固執するのではなく、自分たちに最適な方法を考えましょう。どこまで崩していいのかわからないこともあるでしょうから、アジャイルコーチなどに参加してもらうのも有効です。

9.5.2 デザインがボトルネックに感じる場合

デザインがブロッカーになる状況を防ぐために、いくつかできることがあります。

- ディスカバリーから開発者が参加し、コンセプトのイメージを掴む。
- 体験設計のビジョンになるもの（ストーリーボードなど）を作業場所に貼り出し、誰でも見られるようにしておく。
- ムードボード、UIのサンプルなどをラフに作っておき、何となく出来上がりの雰囲気を掴んでおく。

密なコミュニケーションも重要です。デザイナーの中で決まっていない要素はどこなのか、そ

れはなぜか。開発者は先に作業できること、後回しにしてもいいことなどを伝えると、デザイナーの作業も進みやすくなります。

9.5.3 チームの周りに不安感が漂う場合

チーム以外の人との調整は難しいかも知れませんが、もしできことがあるとすれば：

- ・「アジャイルとは?」「デザインとは?」といった勉強会を、社内ステークホルダー向けに開催する。
- ・デモの中でデザインモックアップを見せる時間をきちんと組み込む。
- ・ある程度形になってからデモにステークホルダーを呼ぶ。

などでしょう。ここでも鍵となるのはコミュニケーションです。自分たちが何を意図していて、どんなものを目指しているのか、なるべく具体的に見せられる用意をしましょう。また、ステークホルダーが意見を押してくる場合、案そのものではなくその裏にある心配事を理解するのも効果的です。

9.6 終わりに

「小さく始め、改善させながら積み重ねていく」ことはチーム自体の成長にも言えます。まだ取り組み始めたばかりのチームが、フレームワークを試してみてもうまくいかなくて不安に思うことは当たり前のことです。誰かが不安を感じていたら、それはチームがステップアップするチャンスです。その理由に注目して、どうしたら解消できそうか考えてみてください。

不安が「チャンス」に見え始めたとき、みなさんはアジャイルな組織として大きな一步を踏み出した証拠です。

参考書籍

MIKE COHN (2009). アジャイルな見積もりと計画づくり 毎日コミュニケーションズ
Collin Lyons (2019). Make Learn Change ustwo ltd.



中村麻由@mayunak <https://twitter.com/mayunak/>
<https://mayunak.com/>

ustwo Tokyoでプリンシパルデザイナーをしています。イギリスに10年滞在し、ユーザー中心設計を学んだあと現在の会社のロンドン本社に入社しました。デザインと開発の距離をもっと縮めたいと考えています。

第 10 章

アジャイルは反復してナンボ！

松永 広明

10.1 アジャイルとはなんぞや？

まずアジャイルとはなんぞや？ というところから行ってみたいと思います。

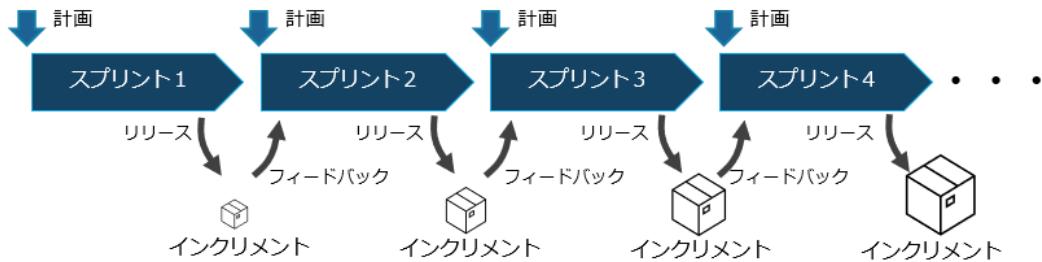
アジャイル宣言をひもとくと、こんなことが書いてあります。

「計画に従うことよりも変化への対応を」

また、同じくアジャイル宣言 12 の原則には下記のような記述もあります。

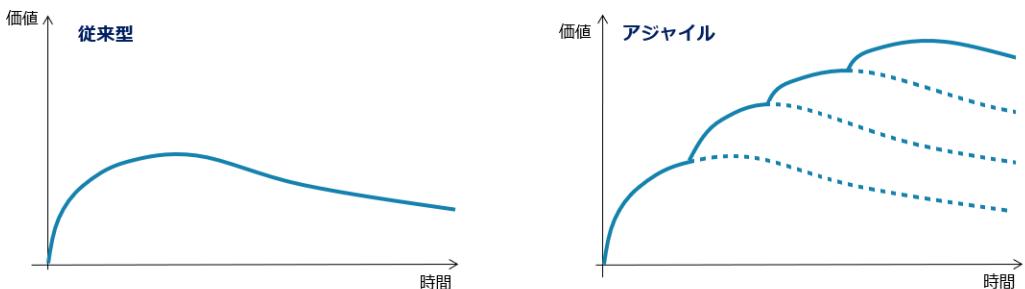
「顧客満足を最優先し、価値のあるソフトウェアを早く継続的に提供します」「動くソフトウェアを、2-3 週間から 2-3 ヶ月というできるだけ短い時間間隔でリリースします」

今や 2-3 ヶ月を「短い時間間隔」と言ってしまうのは議論ありそうですが、そこは 20 年も前の話ですので置いとくとして、つまりここに書かれていることは、図に示すと下記のような感じです。



▲図 10.1 アジャイルの代表的なプロセス

つまり、何が言いたいかというと、価値を見定めつつ、定期的にインクリメントをリリースしてフィードバックを獲得し、それを反映してさらなる価値を付加したインクリメントをリリースし、これを繰り返す、つまり反復することでプロダクトの価値を高め続ける（図 2）、これがアジャイルの価値であり、原則であると言っているわけです。



▲図 10.2 従来開発型とアジャイル開発における価値の推移

10.2 「反復」とは何か

人間は「反復」によって多くの技能を獲得します。スポーツでも習字でも音楽でもそうですが、反復練習なくして上手になることはありません。そして、反復周期は短いほど、つまり頻度が高いほど一般に練習の効果は高いです。年に1回ピアノを弾いたってなかなか上達はしません。が、週に1回、できれば毎日練習すれば上達はずいぶん早くなるでしょう。

そして、アジャイル開発においては、反復がうまく行っていることを示す最も重要な指標は、毎回のイテレーション（スプリント）で「必ず」チームがなんらかの価値あるインクリメントを完成させていることです。いいですか、「必ず」です。ソフトウェア開発においては、価値あるインクリメントとはデモができるインクリメントのことです。逆に、顧客への実動作デモができないなら、それは顧客価値があるとは言えません。なぜなら、実際に動作するところを確認できないと、顧客はそのソフトウェアに価値があるかどうかの判断ができないからです。また、価値判断が出来ないということはフィードバックを返すことができないということでもあり、つまりここでフィードバックループは途切れてしまい、図1や図2のような、アジャイルの重要な特徴は実現出来なくなってしまうのです。

逆に言うと、イテレーションを何回も繰り返さないとデモ可能なアウトプットを出せないチームは、反復がうまく行っていない可能性が高く、これはすなわちアジャイル開発がうまくいっていない可能性が非常に高いと言えます。

10.3 「反復」なきアジャイル

ところがです。最近よく見る自称「アジャイル」はこんなのが多いです。

一応イテレーション（スプリント）は一定周期で区切っているし、スクラムイベントも定期的にやってる（たぶん）。のですが、聰明なこの本の読者ならすぐにお気づきなのではと思いますが、これはアジャイルではないですよね？ はい、おっしゃる通りです。これは「アジャイルに見える」だけの単なるウォーターフォールですね。プロダクトのリリースは最後の1回だけですし、フィードバックなんてどこにも取り込んでないし、価値を高めてもいない。でも残念ながらこういう似非アジャイルが巷にあふれているのは事実です。



▲図 10.3 残念なアジャイル（アジャイルではない）

この方法では反復は全く機能していません。例えばゴルフのスイングを練習するには、自分のスイングを動画に撮って、1回～数回振るごとにビデオをチェックし、良くないところを直してまたクラブを振る、というフィードバックループをくりかえすことが効果的なのですが、図3の方法ではそのループが回っていません。つまり、図3のような状態は、何かを習得するには非常に不利な状態と言えます。

この状態に陥っているチームは、プロダクト開発そのもの以外もうまく行きません。なぜなら、上述したとおり人間は反復によって技能を習得するからです。例えば心理的安全性にしても、自己組織化にしても、T字型スキルにしても、「反復（イテレーション）」をする中で徐々に習得されていくものだからです。

つまり、このままの状態で何回スプリントをやろうと、何年チームを維持しようと改善が進むことはないということです（ちょっと言い過ぎかも）。改善のためには根本原因にリーチする必要があります。

ではなぜこうなるのか？ 原因はプロダクトバックログづくりにあります。

10.4 プロダクトバックログは「プロダクト」のバックログ

プロダクトバックログは、その名の通り「プロダクト」の「バックログ」です。何が言いたいかというと、プロダクトバックログアイテムは、プロダクトを分解したものでなくてはならないということです。プロダクトバックログアイテムは、そのひとつずつが何らかの「価値」を顧客やユーザーに提供できなくてはなりません。「価値」というのは、例えばユーザーが意識的あるいは無意識的に抱えている何らかの問題を解決するとか、新しいことができるようになるとか、今まで以上にお金が儲かるようになるとか、そういうことです。

この分割方法は、よくケーキにもたとえられます。よく知られたメタファなのでここでは解説しません（文字数）。ご存知ない方はググってみてください。たぶん Ryuzee さんあたりが上手に解説していると思います。

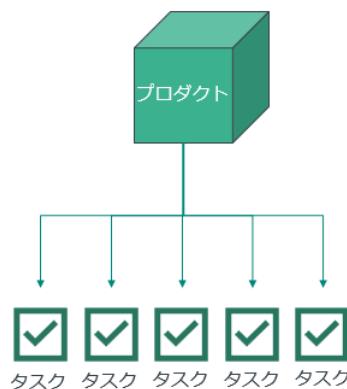
そうやって分割されたプロダクトバックログが実際に作られ、リリースされることで顧客やユーザーは実際にそれを使ってみることができるようになり、使ってみて初めてフィードバックを返すことができるようになるわけです。図3のようなやり方だと、開発序盤から中盤までは、例えば要件定義書や設計書を見せるのが毘の山で、これでは顧客からのフィードバックを得るのはほとんど無理です。そりやそうですよね、例えばカメラの要件定義書を見せて、このカメラ使いやすいですか？ 画質はどうですか？ などと聞かれたって答えようがありません。

10.5 プロダクトバックログは ToDo リストではない

上記のような似非アジャイルになっているチームのバックログをみてみると、バックログアイテムは、たとえば下記のようになっていたりします。

- RMX システムのデータフローをレビューする。
- 撮影画像確認画面の設計をする。
- ユーザープロファイル確認画面についてデザイン部と合意する。

どうでしょうか？ これらはそれ単体でユーザーに価値をもたらすでしょうか。「データフローのレビュー」はユーザーにどんな価値がありますか？ 画面デザインへの合意は、顧客のどんな問題を解決するでしょうか。これらは、それ単体では価値とはならず、その他の多くの作業がつながって初めて何かの価値を生むようなつくりになっています。



▲図 10.4 プロダクトをタスクに分解している

これを図に示すと上図のようになります。つまり、プロダクトをいきなりタスクに分解しているわけです。上述の通り、タスクはそれ単体では価値とはならないため、顧客にリリースすることはできません。これは単なる ToDo リストであり、プロダクトバックログではありません。このような作り方をすると、すべてのタスクが完了するまでプロダクトはリリース出来ません。こういう開発のやり方をなんと呼ぶかというと、そうです、ウォーターフォールと呼ぶわけですね。

Product backlog ≠ To Do list

▲図 10.5 プロダクトバックログは ToDo リストではない

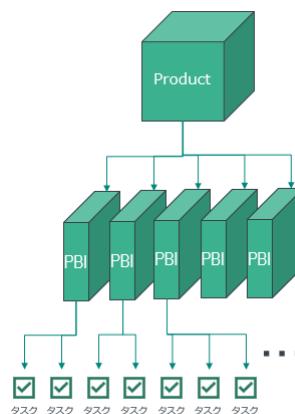
10.6 小さな価値とは

一方で下記はどうでしょう？

- 通知ボタンを押下すると「通知ON」ランプが点灯する。
- チャットボタンを押下するとチャットウインドウが開く。

これらは、機能としてはとても小さなものに過ぎませんが、それ単体で十分な価値があります。通知がONになっているかOFFになっているかは、ユーザーにとって重要な情報です。プロダクトバックログアイテムは単体ではこんな小さな価値が提供できれば十分なのです。

これを図にすると下図のようになります。



▲図 10.6 プロダクトをより小さなプロダクトに分割している

この図ではプロダクトを一旦小さなプロダクト（=価値のあるもの）に分解しています。それは小さい方が故にシンプルで、すなわち早く完成でき、早くリリースでき、その価値はすぐにフィードバックを得ることができます。必要があれば、これらの小さなプロダクト（バックログアイテム）をさらに小さなタスクに分解しても良いでしょう。

つまり、プロダクトバックログさえうまく作れれば反復はうまく回るようになるのがわかると思います。反復がうまく回り始めれば、本線のプロダクト開発だけでなく、チームの改善もまたいくようになり、さらにその効果でプロダクト開発がよりうまくいくというループが回るようになります。

実は、プロダクトバックログづくりは初めてアジャイルに取り組むチームが最初にぶつかる壁のひとつです。図4に示したアンチパターン以外にも、プロダクトバックログアイテムのサイズが大きすぎて、完成までに1ヶ月～数ヶ月かかるというのもよく見かけます。インクリメントが付加する価値は、上述したように「ボタンを押すとランプが点く」ぐらいの、本当に小さなものです。試してみてください。

10.7 まとめ

- 人間は反復によって技能を獲得する。
- 反復がうまく回らないと開発そのものもプロセスの改善もうまくいかない。
- 反復するためには、プロダクトバックログアイテムは、「プロダクト」を分割したものである必要がある。
- プロダクトバックログアイテムにはToDoリストではない。

- プロダクトバックログアイテムは本当に小さな価値で良い。
- アジャイル開発において、反復がうまく行っている最も重要な指標は、毎回のイテレーション（スプリント）でデモができることがある。

上手にプロダクトバックログを作って、反復を回していきましょう！



松永 広明 @qoragile <https://twitter.com/quoragile>

屋号：LSA CONSULTiNG

なんちゃってアジャイルをこよなく憎むフリーランスのアジャイルコーチ。LiSA っ子。

<https://www.facebook.com/lsaconsul>

<https://www.linkedin.com/in/lsaconsulting/>

第 11 章

インセプションデッキからはじめる アジャイル入門

おたけ@otkshol

11.1 はじめに

はじめまして。おたけ@otkshol と申します。開発メンバーの一員として日々奮闘しております。本章では、朝会などの定期的に開催されている開発イベントの目的を答えることができなかつたことをきっかけにアジャイル開発と初めて向き合った話を共有します。

インセプションデッキという言葉も知らなかった状態から、チームに必要なことを考え、カイゼンをスタートさせることができました。私と同じようにアジャイルに初めて向き合いはじめて悩んでいる方にとっての何かの参考になれば幸いです。

11.2 開発チームに新規着任

昨年、私はプロダクトのバッチ開発担当から別の新しい開発チームにアサインされました。5名のチームに対して、自分を含めた3名の新規着任、サブリーダー相当の方が1名離任するという規模の大きいメンバー入れ替えが発生したタイミングでの着任になります。

新しい開発チームではスクラムガイド^{*1}にあるスクラムイベントに相当する開発イベントが開催されており、しっかりアジャイルっぽいことをやっているチームすごいな。というのが正直な感想でした。

初めての案件実施時には、疑問に思ったことを Slack に書き込むと答えられる人に回答をすぐ頂けたり、必要に応じて通話を繋いでサクッと解決したり、しかもその動きをチーム全員で補い合っており、なんて良いチームに来たんだ！ と感じていました。

しかし、上司は「メンバーの大量入れ替えに伴い、チームの文化とスタンス部分を心配している」と言っていました。チームに特に問題を感じていない自分は、その言葉の真意を理解することができませんでした。

^{*1} <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Japanese.pdf>

▼表 11.1 スクラムイベントと開発イベントの対応

スクラムイベント	私の開発チームでの呼び方
スプリントプランニング	開発見積もり会
デイリースクラム	朝会
スプリントレビュー	レビュー会
スプリントレトロスペクティブ	開発ふりかえり

11.3 朝会の目的...?

ある日、いつも通り開催されている朝会に上司がふらっとやってきて、一通り終わった後に「朝会ってなんでやっているんだっけ？ もしかしたらもう少し短くできるかも」と自分とチームリーダーに質問しました。しかし、2人ともその質問に明確に答えることはできませんでした。

この質問に答えられないということは、毎日行われている朝会は実施されているが、何のために行っているのか誰も分からぬ恐ろしい状況であることをこのタイミングで初めて認識しました。

朝会は日々の作業内容を報告すれば良いのかなとぼんやり思っていましたが、進捗状況を共有する理由、朝会では話すべき話題など、会議体として重要な項目が明確でないまま運用されている現状がそこにはありました。

目的や内容がはっきりしてないので、作業進捗共有の流れから話題が逸れて議論が発散してしまい朝会そのものに時間がかかってしまうことも多々ありました。朝会は開発メンバー全員が出席しており、かなり多くの工数を割いています。本当にかけた工数の価値がある時間になっているか自信を持って YES と答えることが自分はできませんでした。

11.4 インセプションデッキとの出会い

このままでは良くないことは理解でしたが、どうやって現状を打破できるか途方に暮れていきました。そんな自分に対して、上司は「インセプションデッキ的な何かをチームで取り組んだ方が良いかもね」という助言をくれました。その助言を聞いた自分は「なるほど、わからん」状態でしたので、その言葉の意味を調べ、書籍「アジャイルサムライ」*2にたどり着きました。

インセプションデッキ*3とは 10 個の質問から構成されるプロジェクト開始前に認識を合わせておきたい内容で 5 つの Why の質問と 5 つの How の質問の合計 10 個の質問から構成されるものです。主にプロジェクトチーム発足時に使用されるのですが、多くのチームメンバーが入れ替わったチームにも適用可能だと思いました。

しかし 10 個の質問すべてに取り組もうすると膨大な時間がかかり、今のチームには必ずしも取り組む必要がない質問もあると感じました。そこで、チームリーダーともインセプションデッキの実施とその内容を議論を行いました。その結果、開発チームのキックオフを対面開催し、その中でインセプションデッキの 1 つである「我々はなぜここにいるのか？」に取り組むことを決めました。

*2 <https://shop.ohmsha.co.jp/shopdetail/000000001901/>

*3 <https://github.com/agile-samurai-ja/support/tree/master/blank-inception-deck>

▼表 11.2 インセプションデッキ 10 個の質問

1. 我々はなぜここにいるのか?
2. エレベーターピッチ
3. パッケージデザイン
4. やらないことリスト
5. 「ご近所さん」を探せ
6. 技術的な解決案を描く
7. 夜も眠れない問題
8. 期間を見極める
9. トレードオフスライダー
10. 何がどれだけ必要か

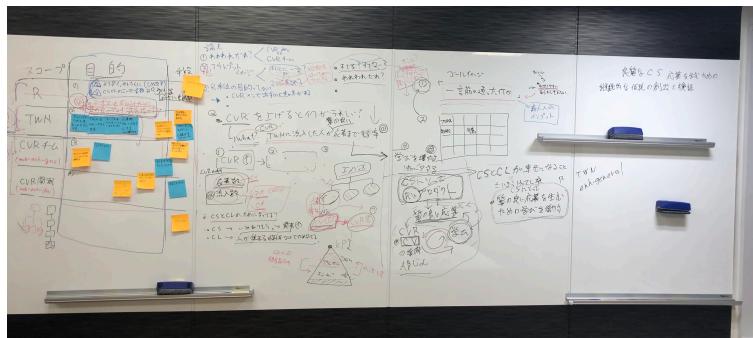
11.5 チーム対面キックオフと得られたもの

事前に出社日を調整して開発チームキックオフを対面開催しました。午前中はキックオフを行い、午後から「我々はなぜここにいるのか」を開発チームで実施しました。議論は白熱して、半日以上の時間をかけてメンバー全員で合意した回答とまとめることができました。

「我々はなぜここにいるのか」に答えることによって、開発メンバー全員で合意したチーム目標が出来上がり、新しい施策や既存の施策の目的を議論するときのチームが立ち返る場所ができました。

個人的な収穫もありました。「我々はなぜここにいるのか」を議論する際、やったことのないファシリテーション役を無茶振りをして頂いたのですが、意外とやってみれば何とかなることを学びました。同時に、悩んでいる様子や喜んでいる様子が表に出過ぎる性質があるので、ファシリテーションをするときは議論の進める邪魔にならない程度にもう少し感情コントロールができるとより良いという課題も自覚することができました。普段の業務でファシリテーションスキルと向き合う機会があまりなかったので、貴重な経験になりました。

そして、初めて対面したチームメンバーとお昼に美味しいラーメンと一緒に食べた紹も手に入れることができました。(嬉しそうに卵をためらうことなくトッピングしていくよりも豪華にした。)



▲図 11.1 キックオフ時のホワイトボードの様子

11.6 キックオフのその後

インセプションデッキのおかげで開発メンバー全員でチーム目標を決めることができました。しかし、これはスタート地点に立ったに過ぎません。

現在は、いつの間にか目的が明確でなくなってしまっていた定例イベントの目的を開発チーム振り返りの時間を使って少しずつ見直しを進めております。今まで当たり前のように過ごしていたイベントの目的を改めて考えるので、そう簡単に議論が進まないのが正直なところです。

しかし、今の僕たちには立ち返るチーム目標があるので、議論が発散しそうになったり、判断に迷った時は立ち返り着実に議論を進めることができるようになりました。これは大きな一歩なのではないかと思います。

チーム全員が定例開発イベントの目的が言えること、その目的がメンバー間で相違がないこと、必要に応じて柔軟にアップデートできること、を目指して今日もカイゼンに取り組んで参ります。

11.7 まとめ

今回の出来事で初めてアジャイル開発と向き合い、目の前のチームに必要なものは何かを考え、もがきながら実践する貴重な経験をすることができました。次は自分が文化を吹き込んで開発チームのアウトカムを最大化させて、プロダクトを使って頂いている方々、チームメイトにお役に立てるようになりたいです。

もし、自分と同じようになんとなく朝会など定期イベントは実施しているけど何でやっている？という質問に答えられない、あるいはチームメンバー全員が同じ回答が出来なさそうな状況であれば、まずはインセプションデッキの「我々はなぜここにいるのか」だけでも時間をかけて考えることをオススメします。チームの拠り所ができて、以降の議論がスムーズになると思います。



竹内尊紀 @otkshol <https://twitter.com/otkshol>

株式会社リクルートに新卒入社。HR領域の開発を主に担当し、現在はタウンワークの開発に携わっている。

第 12 章

独習アジャイル

渡部 啓太 @sobarecord

12.1 独りで学ぶ

そう、独りで学ぶのが好きなんです。受験の時もそうでした。塾講師に質問をした記憶がありません。大学でプログラミングを身につけるのに役にたった本。もちろん、『独習 C』と『独習 Java』です。

社内でアジャイルに関する研修の講師をしていると、こんな声をよく聞きます。「アジャイルで開発をやってみたいんです。だけど、今いるプロジェクトでは既存の開発手法から変わらなさそうなんですよね…」。そうした、もどかしさや機会のなさに嘆いている方は多いのではないでしょうか？

残念。では、独りで学びましょう。独りで、「スクラム」をするのです。

あたりまえですが、独りだと、チームで取り組めません。なので、正確には「スクラム」とは言えません。しかし、独りでできる範囲であれば、経験を積めます。スクラムに関する本を読んだとき、あるいはカンファレンスで他社事例を聞いたときにその効果はあらわれます。自身の経験をふまえた理解、解釈ができるようになります。受験勉強でも自分で問題を解いてから答え合わせをし、解説を読みましたよね？ 独りでスクラムをしても、アジャイルに関する経験値はあがります。

講師だけでなく、ぼくは日々、アジャイル開発チームの立ち上げの支援や、スクラムマスターとしての仕事をもっています。成長していくチームメンバーを見ていて感じるのは、「アジャイルは体験してみないと、真の理解ができない」ということです。アジャイルソフトウェア開発の定義である「アジャイルソフトウェア開発宣言」には具体的なやり方は示されていません。自らでうまいやり方を見つけていく必要があるのです。

独習アジャイルをしましょう。

12.2 独習アジャイルの進め方

ここから具体的な進め方の例を紹介します。スクラムに関する用語は知っているものとしているため、わからない単語があればスクラムガイドを見てください。「スクラムガイド」とネット検索すれば、最初にでてきます。

スクラムをするといつても、独りなのでそこまで構える必要はありません。誰かの承認も、報告義務もありません。内容も簡単です。もしかすると、すでに仕事のやり方として身についているかもしれません。ただ、そこに「スクラムをやっているんだ」という意識をのせていきましょう。

12.2.1 準備をする

まずは準備です。自分が持っているタスクをひたすら洗い出しましょう。これは仕事のことのみでもよいですが、プライベートなタスクも含めると、より効果を実感できると思います。注意点は1つ。全てのタスクを1日以下の大きさにしましょう。それ以上の大きなタスクだと、日々の状況を検査することができませんからね。

洗い出す先は紙のノートでもいいですし、Trello や Miro、Notion といった Web サービスを駆使してもいいです。あるいはパソコン上のテキストや Excel も OK。自分の使いやすいツールをつかいましょう。

タスクを洗い出せたら、それらを優先順位で並び替えましょう。期日や重要度、価値、リスクを基準として順番を入れ替えていくとよいでしょう。このリストが、プロダクトバックログです。

プロダクトバックログができたら、次はスプリントの期間を決めましょう。特別な事情がない限り、1週間を一区切りとしたスプリントを回していくのがわかりやすいです。たとえば、月曜始まり日曜終わりのような、くり返しやすく設定しましょう。

12.2.2 スプリントを回す

さて、準備が整ったのなら、いよいよ独りスクラムの始まりです。次の1週間でやることをプロダクトバックログから取り出しましょう。もちろん、優先順位の高いタスクからです。この際に1週間でギリギリおさまるくらいの量を見極めようとする気持ちが重要です。結果として過不足があったとしても、ギリギリの量を見積もることを繰り返すうちに、自分が1週間で消化できる限界値が分かってきます。これがスプリントプランニングです。

スプリントプランニングで週の計画を立てることができました。つぎは、タスクをこなしていきましょう。そして毎日、決まった時間に、立てた計画通りに進んでいるかチェックします。昨日は何をやったか、今日は何をやるか、何かリスクや問題はないか。順調に進んでいるのなら問題ないですが、予定とズレてしまうこともあります。特に、遅れている場合はリカバリープランを考えます。タスク内容によってはチーム内で相談したり、誰かの助けが必要な事もあるでしょう。このように、日々状況を確認し、必要なら計画を立て直していきます。これがデイリースクラムです。

そして、1週間のおわりに、成果物を他の人に見てもらいます。できればチームメンバーやマネージャからコメントをもらいましょう。得られたフィードバックから新たなプロダクトバックログアイテムを追加したり、方針の再検討をします。人に見てもらうのが難しい場合は、自己評価でも問題ないです。その場合は、できるだけ客観的に確認するようにしましょう。これがスプリントレビューです。

さいごに、1週間のふりかえりをします。うまくいったことと、いかなかったこと。それはなぜなのか。次の1週間をより良くするためのアクションはなにか。KPT（良かったこと、困ったこと、次に試すこと）や YWT（やったこと、わかったこと、次にやること）といったふりかえりのフレームワークを使うのも手です。決めたアクションアイテムは、必要ならばプロダクトバック

ログ追加しましょう。タスクではなく、ルールや心構えであれば、壁やモニターに貼りだしておきましょう。これがスプリントレトロスペクティブです。

12.2.3 くり返し、進化・深化させる

ふりかえりが終われば、新しい1週間のはじまりです。再びスプリントプランニングから始めていきましょう。スプリントを回していると、タスクの増減や優先順位の変化は日々起こるはずです。プロダクトバックログは常にアップデートし、意味のあるリストにしましょう。

ここまで的基本の流れに加え、自分がやりたいと思っている、アジャイルに関するプラクティスを入れていくのももちろんOKです。もし、テスト駆動開発がやりたいのであれば、ローカルの開発環境だけで動くようなテストやCI環境を用意すれば誰にも文句は言われません。

何スプリントかこなすと、リズム良く仕事をする大切さがわかってきます。毎日、毎週計画を立てる。ふりかえりをする。学習のループが回る。まさしく、スクラムの三本柱「透明性」「検査」「適応」ですね。

12.3 つまり、タスク管理なんです

ここは、持論です。つまりは、タスク管理なのではと思うのです。個人なのかチームなのか、向かう先がプロダクトなのか自分なのか。それが違うだけです。

タスク管理の本には、アジャイルに関する話だと思えるような内容が出てきます。

豊かではあるものの、変化の激しい現代において、ゆとりをもって無理なく結果を出していくには、今までの考え方や仕事のやり方では歯が立たない。今までとはまったく違う、新しい手法やテクニック、新しい習慣が求められているのだ（書籍『ストレスフリーの整理術』より）

「1つのことに集中し、それが完了してから次に進む」これが成功の王道（書籍『仕事に追われない仕事術 マニヤーナの法則』より）

小さな失敗を重ねることで、何がうまくいくか、うまくいかないかを知ることができます（書籍『「やること地獄」を終わらせるタスク管理「超」入門』より）

もっと簡潔にいうと、仕事をうまくやりましょうということ。独習アジャイルを進めていけば、わかるはずです。

12.4 もうひとつの重要なポイント

あらためて、独習アジャイルのメリットとデメリットをおさらいします。

* メリット* 独りでもできる* アジャイルの解像度が上がる* デメリット* コミュニケーションがないのでその部分はあまり学べない

さいごに、もうひとつ重要なポイントを紹介します。それは「アジャイルの良い点を他人に説明できるようになる」です。

じつは、ぼくがチームに初めてのスクラムを導入できたきっかけは、この独りで行うスクラムだったのです。独習アジャイルを通じ、アジャイルのメリットを理解しました。そして、「アジャ

イルをやりたいという熱量」にプラスして、「スクラムで生産性があがった、働きやすくなった」という話をマネージャーに説きました。その場で「やってみよう」という返事をもらいました。

チームのスクラムマスターとして自分の学びをチームに注入。スクラムマスターとしての一歩を踏み出しました。さらに、そのスクラムチーム立ち上げの経験をもとに、全社にアジャイルを推進する組織を立ち上げることもできました。さらにさらに、今ではアジャイルコーチとして様々な社内外のチームを支援しています。「アジャイルをしたいけどできない」とくすぶっていた時はえらい違いです。

だから、独りで学ぶのが好きなんです。あなたにも好きになってほしいなあ。



渡部啓太 @sobarecord <https://twitter.com/sobarecord>

チーム作り支援士／意思決定コーチ

チーム作りの専門家、アジャイルコーチ。チーム作りを通じ、誰もが楽しく働く社会を目指す。ソフトウェア開発者としてキャリアをスタート。仕事をしている中でチーム作りの大切さに目覚め、スクラムの導入や全社アジャイル推進活動を経験し、現在は NRI bit Labs に所属。アジャイル導入支援・コーチを行っている。チームの立ち上げや、軌道修正が得意。

第 13 章

アジャイルの始め方～化石エンジニア がアジャイルの「本の探し方」（自己 流）を共有します～

ammonite_404

13.1 テンケー！（天啓 or 典型）

4 年くらい前のとあるイベントで「アジャイル」の精神に触れた（と思った）化石エンジニア・アンモナイトは、唐突に「自分はもはや化石的なエンジニア=生きるレガシー=驅逐される」ことを悟った。まだローンあるのに。

13.2 無知の知

自分が化石エンジニアであることは理解した。「アジャイル」が好きなことも自覚した。だから、思い切って、アジャイルイベントに参加してみたり、したのだが - -。

現実：「アジャイラー^{*1}と会話が成立するには圧倒的にアンモナイトの知識量が足りない」

アジャイラーはとても勉強熱心なのである。WF (Waterfall) でも勉強熱心な方はいるが、アジャイラーは化石から見ると、勉強が「アクティブすぎる・守備範囲が広すぎる」。

アジャイルの理論に関する本は必要性がわかる。XP、スクラム、リーン等は知見を広げ、深めるためにインプットが絶対に必要。他のアジャイルの実現手法もきっと同じだと想像する。

しかし、なぜアジャイルイベントの会話で「哲学」「組織論」「マーケティング論」「心理学」「文化人類学」「動物行動学」等が出てくるのだろうか？

「組織論」「心理学」「文化人類学」「動物行動学」はまだわかる。アジャイルは結局、ひとがやることなので、そちら方面の知識は必要だと感じた。「プロダクトを作る方法」 = 「アジャイル開発」と考えれば、「作る」の先、もしくは前提として「マーケティング論」も必要かもしれない。でも「哲学」は？ 未熟なアンモナイトには接点すらわからない。

いきなり全方面カバーは「1 日 = 24 時間の世界線&資金に限界あり」では不可能と判断、アン

^{*1} 仮称。アジャイル開発の実践者・関係者

モナイトはいったん、アジャイル・スクラム・ファシリテーション関係の書籍を読み漁ることにした。

13.3 きりがない

……どういう、ことだ……。

いったいどうやってアジャイルコミュニティの方々は書籍情報をゲットしているのだ？ 昨夜アンモナイトが密林で探し回り、ようやくポチった本の話題を普通に今夜している。

しかもすでにしっかり読み込まれていて、「あそこの解釈はこうだと思う」「それはあの本にもあったよね」「(本の略称)は読んでる」という台詞が頭上を飛び交う。

アジャイラーのアンテナ、どこに立ってるの？ どういう感度で、どういう範囲なの？

長らくアジャイルをされていた方なら蓄積があるのかなと推測できるが、「ベテランじゃないです」と自己紹介した方も、ものすごい読書してる。間違いなく密林ハイ課金者。「ベテランじゃないです」＝「初心者」ではない。「アンモナイトよりも高度な知識レベルの実践者」なのだ。

13.4 やってみた

では、どうしたらいいのか？ 自分のアンテナを張るしかない。感度を上げて、リサーチするしかない。アンテナの張り方は下記4つがあると考えた。アンモナイトはこの中で2-4を実行した。(1はURLだったり、英語論文だったりして難易度が高いものもあるため)

- 1冊目の巻末の関連書籍を読み漁る
- コミュニティやイベントに参加する
- 密林で表示されるおススメ本を買いあさる
- 1人の著者の本を徹底的に追う

なお、アンモナイトは化石であり、かつ、かなりポンコツである。正直、書籍集めはできるだけショートカットしたかった。「これ！」という本だけをチョイスしたかった。←我慢弱い

しかし、そう簡単にいくわけがない。「自分のレベルに合った・自分のレベルを上げてくれる本」の探し方を見つけなければならない。Go ● gle先生の使い方やショートカットキーの使い方同様、「そのひとが使いやすく・成果を上げやすい方法」でなければ、意味がない。ゆえにアンモナイトの方法を共有しても、「使えない！」というご感想もあると思います。というか、必然的にそういうなります。

先にお詫びします。「申し訳ありません」

13.5 「本の探し方」 説明

では、これから1-4の内容を順番に紹介します。※自己流ゆえに、成果は保証できません！

1：関連書籍から攻める =1冊目を読んだ際に、巻末の関連書籍を読み漁る

メリット：1冊目から得た知識を補強できる：水平方向への展開

デメリット：たくさんの関連書籍の中でどれが自分のレベルにあってるか判定がムズイ＆英語論文があったりする

2：イベントから攻める =コミュニティやイベントに参加、LTなどでおススメ本の情報を得る

メリット：匂がわかる：未知の別軸への展開・おススメ本コーナーを持ってるコミュニティがあつたりする

デメリット：自分のレベルに合っているか判定がムズイ・話してるのはバックボーンありきで会話を出している

（※本来の趣旨であるコミュニティ・イベントにちゃんと参加すること）

3：密林を信じる =密林で表示される関連のおススメ本を買いあさる

メリット：広めに探れる。密林が勝手に関連書籍を勧めてくれる

デメリット：ハイ課金。廃人課金。カートに入れた後、請求額に震える

4：粘着する =1人の著者の本を徹底的に追う

メリット：その著者への理解が深まる・その著者の出版予定が分かる（垂直方向への展開）

デメリット：著者の研究内容・実践内容に情報が偏る可能性あり

あまりにも粘着するとファン→ストーカーへジョブチェンジする

上記4つが初心者でも可能な攻略ルートで、個人的に実行する順番は3→2→4→1です。

まず会話の成立のために、3の「当たり前品質」の本を知らないといけない。

それから2へ突撃、他の方の意見をうかがったり、自分の解釈を発言、質疑して理解を深める。

4は2-3の間に「この著者のことをもっと知りたい・どういう方だ？」と思った時にやる。

一気に2、4が解決するのは「読書会イベントに参加」。強制的に課題本を読み、イベントで意見・知見を共有、イベント後は密林で関連本を探しつつふりかえりになるので、ありがたい。

また、2、3は、相乗効果で、イベントでよく聞く本、密林でよく運動して表示される本は「あ、またこの本だ。外せない本なんだな」とわかるようになる。

2のイベント参加にはもう1つ大きなメリットがあって。オンラインイベントだと「社内では使っていないツールに触れることができる」ため、新しいツールへの心理的ハードルがめっちゃ下がります。※セキュリティとコンプライアンスにはくれぐれも注意！

たとえばzoom、Slack、GoogleMeet、Discord、Miro、Mural、Notionは会社によっては使わない。でもイベントで使うと確実に経験値上がります。環境が変わった際に役立ちます。

「あー、これ、Miroなら瞬殺なのに」→社内でもMiro使えないか訊いてみようか。「オンラインMTGなら背景画像があったほうがいいよな」→公式画像があるか確認してみよう。もししくは探そう。コミュニケーションツールは、だいたいzoomのブレイクアウトルーム、録画等に似た機能があるので、ほぼチュートリアルなしで即日から機能使用可能。

13.6 まとめ

というわけで「本の探し方」について書いてみました。

「魚が欲しいひとに、魚をあげるのではなく、魚の釣り方を教える」（By老子）

このトピックを読まれたということは、「釣りたい気持ち」（Why）はもうありますよね。「アジャイル用の釣り竿がわからないよ！（渓流釣り用のやつ？ それともカジキ釣るやつ？ え、

リールには自動がある？ 疑似餌？ なにそれ」という問題（How）だと思います。

この投稿があなたの釣り竿になれば幸いです。

ご参考まで、アンモナイトが作った『アジャイルサムライ』を0地点とした場合の本の分布図（2021年版）をオマケに添付します。



▲図 13.1 bookmap_20220123_2

これは忘れっぽいアンモナイトが「どんな本だっけ？」を把握するために作ったものです。脳内の整理にもなるので、「どんな本を読んだか」というアウトプットも大事かなと思います。

ここまで長文にお付き合いいただき、ありがとうございました！



アンモナイト (ammonite_404)

ほぼブログの Qiita : https://qiita.com/ammonite_404

ものすごく静かな Twitter : <https://twitter.com/4Ammonite>

長らくWFの沼にたゆたっていたが、ある日アジャイルと出会い、自分が化石エンジニア=生きるレガシーであることを知った。気づかぬうちにIT系としてのレベルがLv3（推定）くらいになっている。このままではヤバイ、レガシーはいずれ駆逐される。生き残りを目指して、唐突に化石→人類を目指して進化を始めました。社内プロジェクトでスクラムマスターを経験後、認定スクラムマスター（CSM）取得。Qiitaでアジャイル本の地図を作ったり、技術力Upのため、Kaggle挑戦や他社研修の視聴等をやっています。

第 14 章

コミュニティの探し方

おやかた@oyakata2438

あなたが所属しているコミュニティはありますか？

どうやって探す？ 人見知りなんだけどどうしたら？ 何を話したらいいの？ たいていは杞憂です。まずは飛び込んでみましょ。

本章では、コミュニティの探し方、日ごろの立ち居振る舞いについて述べてみましょう。

14.1 コミュニティに参加する

参加するといっても、Slack や Discord といったオンラインスペースに登録し、会話を眺めたり、コメントしたり、スタンプを押したりするだけです。招待リンクをクリックしたり、登録メールを送るくらいのもので、お金もたいていの場合かかりません。

会話への参加、コメントも義務ではありません。アクティブな人より、ROM な人、何なら見てすらいない人の方が多いくらいでしょう。イベントで一度参加したっかり、とか。

14.2 コミュニティに入るメリット

コミュニティに所属することでいくつもメリットがあります。

繋がりができる。単純に楽しい。次の /類似のイベントの情報が手に入る。雑談だって楽しい。相談事もできる。

ある（共通の）トピックスに興味を持つ人が集まる場所というのはとても貴重です。バックグラウンドが近しいということですから、共通の話題で盛り上がることも頻繁にあります。あるあるネタで盛り上がる。今困っていることを相談する。相談してみるだけで気が楽になることもありますし、隣の人が同じ経験、さらには解決策を持っているかもしれません。

オフラインの勉強会、カンファレンスでは、そこであった人と出会えるかもしれません。「いつも Slack でお世話になってますー」といった感じで会話のとっかかりにもなります。

14.2.1 デメリットはあるか？

デメリットはないでしょうか？

コミュニティはたいていの場合、参加も脱退も任意です。合わないなー、と思ったら抜ければ

よいのです。四六時中コミュニティの Slack をチェックするとか、入りびたるようになってしまふと、むしろ依存症的な意味で黄色信号を気にした方がいいかもしれません。居心地がよいのでつい入り浸ってしまうのはとってもよくわかるところですが・・・。そういう意味では、コミュニティに関わることで、時間がどんどんなくなっていくのは困った点です。面白いオンライン/オフラインイベントがある。面白い本が紹介されてた。雑談に盛り上がった。ああ・・・時間がどんどんなくなってしまいます。困った。

少し活動に疲れたら、しばらく ROM (Read only member: 読んでるだけの人) だけ、あるいはしばらく離れてみるとよいでしょう。参加、コメントは義務ではありません。離れているときは読む必要もありません。離れること（あるいは戻ること）を公言する^{*1}必要もありません。ゆるく繋がれるというのは、技術/オンラインコミュニティの大きなメリットです。

14.3 どうやって探す？

一番手っ取り早い参加のしかたは、勉強会やカンファレンスに参加することです。最近のカンファレンス・勉強会はたいていの場合、オンライン開催です。2019年までのようにオンサイト（オンライン）開催ばかりではありません。その結果、日本中どこでも、あるいはリアルタイムで参加する^{*2}必要すらありません

さて、主催コミュニティのベース基地となる Slack や Discord があるときは、カンファレンスの参加者へ誘導がなされます。すでにたくさん的人がいるでしょう。雑談チャンネルやイベントのチャンネルで盛り上がっているといいですね。ここに入ることで、自動的にコミュニティに参加することができます。もちろん入ったからといって、そこにどこまで関わるかは任意です。ROM でも、雑談だけでも、あるいは運営にかかわっていくのでも、自由です。

14.3.1 人づてに探す

今いるコミュニティに参加している人、あるいはフォローしている人が参加しているコミュニティを探してみましょう。なんなら直接聞いてみるのが手っ取り早いでしょう。良さげなところを紹介してもらえるかもしれません。さらにその先で別のコミュニティに出会えるかもしれません。

気軽に入ってみて、ちょっと違うかも？ と思ったら抜けても、放置してもよいのです。

14.3.2 なければ作る

入りたいコミュニティがなかったら、いっそ作ってしまいましょう。

無料の Slack または Discord などに適当な名称でチームをつくり、そこを拠点に活動を始めるという手があります。幸い Slack も最初は無料です。無料だと 1 万ポストまでといった制約はありますが、1 万まで行くには結構な時間がかかります。Discord は無料ですし、メンバーの権限付与などの制御が Slack よりやりやすい面があります。他のプラットフォームでも大きな問題にはならないでしょう。

^{*1} コミュニティのボード（運営）に関わっているようになっているならば、他の運営メンバーに「疲れたからしばらく休む」くらいは伝えるとよいでしょうが、普通のメンバーならそれすら必要ありません。

^{*2}もちろんリアルタイムで参加するメリットもたくさんあります。登壇者に Zoom や Twitter などで直接質問を投げかける、（オンライン）懇親会で会話する、Twitter 実況をする、などなど。

いずれにせよ、**コミュニティがない場合には、新しく作る**という選択肢があります。

ベース基地があるということが重要で、集まれる場所があると、人が集まってきたり、会話が始またりします。最初は過疎っていてあまり楽しくないかもしれませんぐ・・・。

またこのとき、**コミュニティの位置づけを明確にしておくことで参加しやすくなる**面があります。Not for Me と思われても参加者は増えませんが、何をやっているところかわからないというのも参加をためらう要因になります。

例えば、「IT 技術者（全員ウェルカム）」というコミュニティを作ろうとしたと考えましょう。あなたはそこに参加しようと思いますか？ IT 技術者ってなんだろう、自分が合致するだろうか？ 実は全然違う感じだったらどうしよう？ と思いませんか？

それよりも、「Web 系フロントの初心者」といったように、ある程度限定されていた方が入りやすくないですか？ これらはあくまで例ですが、ある程度具体的な参加者を想定する方がよいでしょう。ペルソナを設定すると言い換えてもいいかもしれません。ターゲットを明確にすることで、検索の精度も向上します。すでにコミュニティがあるかもしれません。

おっと、すでにコミュニティがあるから新しい類似のコミュニティを作ってはいけないという意味ではありません。同じ内容に見えても、別のコミュニティは別のコミュニティです。進次郎構文にも見えますが、別物なので、参加者も異なり、雰囲気も異なります。気軽に作ってみて、うまくいかなければ閉鎖/放置すればよいのです。繰り返しになりますが、最初は無料で始められます。

14.4 コミュニティの居心地をよくする

居心地のよいコミュニティにはだんだん人が集まっています。少しづつ人が増えてくると、いつもいる人が出てきてゆるく雑談していたり、イベントの相談をしてしたりといった雰囲気になります。今困っている問題を相談した（半分愚痴った）ところ、アドバイスがもらえたり、口にすることで整理されて自己解決したり、そういう経験はありませんか？

居心地のいいコミュニティは活動も盛んです。逆かもしれませんね。活動が盛んなところは、みんなが Gentle なので居心地がいいのかもしれません。

14.4.1 雜談する

誰かがいて、適当に反応が返ってくると嬉しいですよね。同じコミュニティにいるということは、少なからず興味やバックグラウンドが近しいということ。共通の話題もありますね。

ちょっとした雑談であっても、会話が進んでいるということはそれだけコミュニティの活発さの指標になります。すべての雑談に絡む必要はもちろんありませんが、面白いと思ったら反応しましょう。スタンプを付けるだけでも OK です。

雑談ですからちょっとしたことでも OK。人がいて反応をしてくれるというだけでうれしくなり、さらに活発になります。もしあなたが主催者であれば、ぜひ（疲れない程度に）反応をしてあげてください。初めて参加するような人にとっては、最初の投稿は案外ハードルの高いもの。そして投稿しようかどうしようか、関係ないって怒られないかしら、無視されたらどうしよう、など不安に思っている場合もあるでしょう。そういうときに優しく反応してあげられるといいですね。

Slack や Discord によって、非同期のコミュニケーションが成立しやすくなりました。電話ほどリアルタイムでなく、メールほど宛先が明確ではなく、ゆるく返事したりスタンプ押したり、きがむいたら返事したりとすることができます。

14.4.2 アンチハラスメントポリシーを定める

コミュニティが大きくなってくると、一定の確率でトラブルが生じる可能性が出てきます。そういうときのために、アンチハラスメントポリシーを定めておきましょう。行動規範という言い方をする場合もあります。

ごくごく簡単に言えば、他者に敬意を持ち、攻撃的な言動や様々なハラスメントはやめましょう、という当たり前の宣言です。様々な勉強会やカンファレンスに表示されることが増えました。内容としては当たり前ですし、アンチハラスメントポリシーがなくとも、たいていの場合特に問題は生じません。関係者、参加者のほとんどはそういったトラブルを起こすような人ではありません。

セクハラやパワハラ、その他のハラスメント、攻撃的な言動が横行するようなコミュニティに参加したいと思う人はいませんよね。

しかし、アンチハラスメントポリシーがあったからといって、トラブルが起こらないというものではありません。残念なことですが。

それでも、アンチハラスメントポリシーにて禁止事項を謳っておくことで、そういう行動をする人（たち）に対して、警告や排除といった対処を取りやすくなります。また、被害者にとっても行動規範に違反する行為に遭遇した場合に通報していいのだ、見かけたという人にとっても通報していいのだ、と思ってもらう効果もあります。

アンチハラスメントポリシー/行動規範は自分で作ってもよいですが、他のコミュニティの行動規範を参考にするのもよいでしょう。

エンジニアの登壇を応援する会では、行動規範を公開しています。

<https://portal.engineers-lt.info/guideline>

こちらを参考にして見てください。

14.5 まとめ

コミュニティに参加する意義、そして作ってみることについて述べました。コミュニティに参加して得られるものもあります。単純に楽しい！ という点だけでも参加する価値はあります。ベース基地を見つけに行きましょう！



親方 @oyakata2438 <https://twitter.com/oyakata2438>
サークル名：親方 Project

ワンストップ本シリーズ企画・編集（一部執筆）しています。コミケと技術書典に出没。ついには技術書同人誌博覧会（技書博）のコアスタッフとして運営側に参加しています。

第 15 章

Agile イベントまとめ

J.K(@project_j_k)

現在アジャイルに関連するさまざまなイベントが日本中で開催されています。

コロナ禍の状況下でオンライン開催が難しい情勢にある昨今ではありますが、新しいオンラインカンファレンスが生まれ続けており、変化の中で新しい機会を生み出しているのも、Agile な人たちの特長だと感じます。

以下に、20 年前から続いているものからこれから誕生予定のものまで、カンファレンスの名称と公式ページへのリンクを並べてみました。知っているものがいくつあるか確認してみたり、イベントの雰囲気を見たり、次回イベントの情報を確認したりと、これからのご参考にと活用してみてください。

XP 祭り

<http://xpjug.com/>

2002 年に誕生。日本のレジェンドさんたちもニューカマーもわいわいするお祭り。いろんな人が集結していてとっても賑やか！ プロポーザルの登竜門のような存在でもあるとか。

Agile Japan

<https://agilejapan.jp/>

2009 年に誕生。アジャイルマニフェストを翻訳された平鍋さんに立ち上げられた、おそらく国内初めてのアジャイルカンファレンス。

歴代実行委員の方々が築いてきたからこそ今がある、伝統に支えられながら変化を続けている。

Regional Scrum Gathering Tokyo

<https://www.scrumgatheringtokyo.org/>

2010 年に誕生。日本のスクラムの熱量を牽引し続けるカンファレンス。

ここで登壇するために、一年間何を積み重ねていくのかを考えている人が多く、チケットやスポーツバー枠の完売の早さも風物詩となっています。

Innovation Sprint 2011

<http://innovationsprint.com/>

2011 年に開催。

『スクラム』の生みの親である野中郁次郎氏と育ての親とも言えるジェフ・サザーランド博士が集まった伝説のイベント。グローバルやイノベーションがキーワードとなっていた場のようです。

DevOpsDays Tokyo

<https://www.devopsdaystokyo.org/>

2012 年に誕生。

世界中で開催されている DevOpsDays の日本リージョン。桜の描かれた素敵なロゴは海外からもスピーカーさんにきて欲しいという想いからで、毎年 4 月に開催されている。

Scrum Fest Osaka

<https://www.scrumosaka.org/>

2019 年に誕生。

Regional Scrum Gathering Tokyo から生まれたスクフェス第 1 号。2 年目からはオンラインで全国の地域コミュニティが集うフェス形式に。

Scrum Interaction

<https://scruminc.jp/event/jp/>

2019 年に開催。スクラムの父、ジェフ・サザーランド博士、そして、スクラムの祖父、野中郁次郎先生らを招いたカンファレンス。ワークショップ型の内容となっており、Scrum@Scale を体験して帰れる仕組みが印象的でした。

Agile Tech EXPO

<https://agiletexchexpo.com/>

2020 年に誕生。パンデミックにより対面で集まりたくても集まれなくなってしまった自分たちにオンラインで集まる機会を切り拓くべく発足し、始動。学生も社会人も国籍もないカンファレンスを目指し日々活動中。

Scrum Fest Mikawa

<https://www.scrumfestmikawa.org/>

2020 年に誕生。スクフェス第 2 号。初回から現地会場 x オンラインのハイブリッド開催で賑わう。Earlybird チケットが 3,111 円などろに素敵なこだわりを感じます。

Scrum Fest Sapporo

<https://www.scrumfestsapporo.org/>

2020 年に誕生。歴史と伝統のあるアジャイル札幌さんが主催で、暖かい雰囲気が大好き。お手製のお土産ボックスが最高に美味しい体験を提供してくれて、北海道に思い馳せてしまう日々でした。

ふりかえりカンファレンス

<https://www.facebook.com/hurikaerijissenkai/>

2021 年に誕生。ふりかえり実践者さんたちが集う、学びの多いカンファレンス。録画を見るだ

けでも参加する価値があるかも。ふりカエルが可愛い！

アジャイル経営カンファレンス

<https://agile-keiei-conf.jp/>

2022年に開催。ビジネスアジャリティを高めてスピーディな経営判断を実現する「アジャイル経営」という考え方を実現・推進するマネジメント層向けのイベント。

Lean Conference Japan

<https://lean-conference.com/>

2022年に誕生。「日本"式"の理想的経営を考える」トヨタ生産方式から生まれ世界中に影響を与えていたる Lean を日本からどう向き合っていくかを考える場として誕生しました。

Scrum Fest Niigata

<https://www.scrumfestniigata.org/>

2022年に誕生。オーガナイザーさんの愛と情熱のおもてなしが現地参加者を歓喜の渦に。日本酒が好きな人にはたまらないフェス&地域です。

Scrum Fest Sendai

<https://www.scrumfestsendai.org/>

2022年に誕生。

早くもサメのようなマスコットキャラクターが誕生したようです！ ハイブリッド開催予定。

Scrum Fest Fukuoka

2023年開催予定。

スクフェスが遂に九州に上陸。期待して待ちます！ こちらもハイブリッド開催の予定とのことです。

15.1 良いカンファレンスに出会う、ということ

良い、の定義はさまざまですが、私自身が良いと感じたカンファレンスは、様々な出会いがありました。例えばサーバントリーダーシップという言葉に初めて出会ったのも、日本にいたら絶対に会えないような海外のスピーカーさんのお話を聞けたのも、社内で悩んでいることを相談できる社外の仲間と出会えたのも、一緒にカンファレンスを運営してくれる・したいと思える仲間と出会えたのも、新しい職場を見つけたのも、全てカンファレンスをきっかけとして起こった出会いでした。

良い出会いは、必ず自分の人生を教えてくれました。一つ一つの出会いがなければ、間違いなく今の自分はありませんでした。この本に携わることができたのも、コミュニティからの出会いのおかげでした。

同じカンファレンスに通うと、一年越しの再開に感動することもしばしば。お互いがお互いの場所で取り組んできたことを持ち寄って、また新しい元気や勇気をもらえるのもカンファレンスの魅力です。

15.2 他にもたくさんあります

カンファレンス以外にも、様々なイベントやコミュニティなどもあります。初めはみんな知り合いがいない状態からスタートですが、その一步を知っている人たちだからこそ、暖かく歓迎してくれる人たちがそこにはいます。情報は各イベントのプラットフォームや Twitter アカウントなどをフォローするのがオススメです。ステキなイベント、コミュニティ、そして人との出会いに巡り合えますように・・・！



J.K @project_J_K https://twitter.com/project_J_K

Agile Tech EXPO Organizer / Agile Japan 実行委員

アジャイルで日本から世界を楽しく！ アジャイルコーチや組織開発に従事。カンファレンス運営などを通じ、Agile が楽しく広まることを夢見て日々活動中。

第 16 章

スクラムマスターの資格の選び方

増田謙太郎@scrummasudar

こんにちは。増田謙太郎 (@scrummasudar) です。現在、フリーランスのスクラムマスターとして、スクラムチームおよび組織の支援や、アジャイル開発の研修を提供しています。

仕事や技術コミュニティの活動をしている中で、スクラムマスターの資格について質問をいたることがあります。例えば、「認定スクラムマスター」という同じ名前の資格があるけれど、どの資格を取得すればよいのか?」、「研修を実施している会社が複数あるけれども、選び方がわからない。」といった内容です。

本章では、複数あるスクラムマスターの資格に関する詳細および、私の考える資格と研修の選び方について、お伝えします。

16.1 そもそもスクラムマスターに資格は必要?

スクラムマスターになるために、資格は必要ありません。医師のような業務独占資格、キャリアコンサルタントのような名称独占資格ではありません。そのため、ソフトウェア開発を含め、ものごとをスクラムで進めていこうと決め、スクラムマスターを担当することになれば、その日からスクラムマスターを名乗ることになります。

しかし、スクラムマスターを担当することになっても、スクラム初心者の場合であれば、初日からスクラムマスターの責任を果たすことは難しいです。スクラムとはなにか、スクラムマスターとはなにかを理解した上で、スクラムチームやステークホルダーに対して、スクラムマスターの責任を果たす必要があります。スクラムやスクラムマスターを理解するための方法として、スクラムガイドや関連する技術書を読む、会社内や技術コミュニティなどにいる先輩スクラムマスターやアジャイルコーチに相談する、カンファレンスに参加するなどがあります。数ある方法の中で、1つの有用な方法として、「スクラムマスターの研修を受け、資格を取得する」があります。

16.2 スクラムマスターの資格について

2024 年 7 月現在、スクラムマスターの資格は、大きく 3 つあります。

それぞれ提供する団体・企業が異なるため、日本語で「認定スクラムマスター」と表現されても、別の資格です。

- Scrum Alliance®が提供する認定スクラムマスター (Certified ScrumMaster® /CSM®)

- Scrum Inc. が提供する認定スクラムマスター (Registered Scrum Master[®])
- Scrum.org が提供する Professional Scrum Master[™]

16.2.1 認定スクラムマスター (Certified ScrumMaster[®] /CSM[®])

日本において「認定スクラムマスター」の資格で真っ先に想像するのが、Scrum Alliance[®] の提供する Certified ScrumMaster[®] (以下 CSM) です。2日間から4日間の研修を受け、その後テストに合格することで、資格を得ることができます。

2024年7月時点で、日本で研修を実施している企業は、株式会社 Odd-e Japan、株式会社アトラクタ、アギレルゴコンサルティング株式会社、アジャイルビジネスインスティテュート株式会社、TIS 株式会社の5社です。各社が提供している研修日程は、Scrum Alliance[®] の Web サイト (<https://www.scrumalliance.org/>)、もしくは各社の Web サイトから確認することができます。

研修を受けた後、講師からテストを受ける資格があると認められると、テストに関する案内が送られてきます。テストはオンラインで実施され、テストに合格すると資格を得ることができます。

資格取得のための費用は、20万円から30万円程度です。

16.2.2 認定スクラムマスター (Registered Scrum Master[®])

日本語では同じ「認定スクラムマスター」ではありますが、Registered Scrum Master[®] (以下 RSM) は、Scrum Inc. が提供する資格です。2022年7月に、Licensed Scrum Master から Registered Scrum Master[®] に名称が変更されました。

2日間の研修を受け、その後テストに合格することで、資格を得ることができます。

2024年7月時点で、日本で研修を実施している企業は、Scrum Inc. Japan、株式会社永和システムマネジメント、LSA CONSULTiNG 株式会社の3社です。研修日程は、各社の Web サイトから確認することができます。

資格取得のための費用は、20万円程度です。

16.2.3 Professional Scrum Master[™]

Professional Scrum Master[™] (以下 PSM) は、Scrum.org が提供する資格です。PSM は、I からIIIまでの段階が設定されており、I が基礎コースになっています。CSM や RSM と異なり、研修への参加が必須ではなく、テストに合格するのみで、資格を得ることができます。

また、日本語への対応が他の2資格と比較すると遅れていますが、テス

トはオンラインで実施されており、ブラウザ上の Google 翻訳拡張機能を使用することが推奨されています。そのため、英語に不安がある方でも、問題ありません。

テス

トを受けるための研修は不要ですが、Professional Scrum Master[™] I への試験料込の研修が実施されています。2024年7月時点で、日本で研修を実施している企業は、株式会社 IT プレナーズジャパン・アジアパシフィック、サーバントワークス株式会社の2社です。研修日程は、Scrum.org の Web サイト (<https://www.scrum.org/>)、もしくは各社の Web サイトから確認することができます。

資格取得のための費用は、テストのみであれば、2万円程度です。研修を受ける場合は、20万円程度です。

16.3 資格の選び方

資格が3つあり、それぞれの差について説明をしました。その中で、どの資格を選択するべきかということになります。しかし、資格そのものに差はない、と私は考えています。むしろ、どの講師の研修を受けたかが、大事だと考えています。同じ資格であっても、講師によって研修内容が大きく異なります。研修を受ける前には、アジャイル開発やスクラムに、多少の差はあれど、すでに触れた状態だと思います。その状態で、どの研修が、自分にとってよいかを考えることが、重要です。

研修の選び方は、以下4つの軸があると、私は考えています。

- 講師の考え方や理念で選ぶ
- 講師の得意分野で選ぶ
- 日本語で講師と対話できるかで選ぶ
- 研修方式で選ぶ

16.3.1 講師の考え方や理念で選ぶ

講師は、書籍の執筆・翻訳に携わっている、カンファレンスで発表している場合があります。そのため、書籍や発表で感銘を受け、講師の考え方や理念を深く知りたいという場合があると考えています。

研修に参加するのは、あなたご自身なので、その講師の研修を受けたいという気持ちが非常に大事です。

例えば、『SCRUMMASTER THE BOOK 優れたスクラムマスターになるための極意——スキル、学習、心理、リーダーシップ』のZuzana Sochovaさん、『組織パターン』のJames O. Coplienさん、『SCRUM BOOT CAMP THE BOOK【増補改訂版】』スクラムチームではじめるアジャイル開発』の吉羽 龍太郎さん、『これだけ! KPT』の天野 勝さんのように、書籍を執筆されている講師もいます。彼、彼らの書籍を読んで、既に仕事で生かしている場合に、おすすめの選び方です。

研修を受ける前に、講師について知るため、カンファレンスに参加し、コミュニケーションを取ってみることも、研修をより良くする事前準備になるでしょう。日本では、Regional Scrum Gathering Tokyo、日本各地で開催されるスクラムフェス、アジャイルジャパンなどのカンファレンスがあり、参加してみるとよいでしょう。

16.3.2 講師の得意分野で選ぶ

講師はスクラムマスターの研修ができるだけの、全般的なスキルを当然持っていますが、それでも講師それが得意とする分野があります。ソフトウェア開発に関する方が得意分野の方もいらっしゃれば、プロダクトマネージメントを得意とする方もいらっしゃいます。

研修に参加するあなたご自身や、関わっているチーム・組織の課題に合わせて、講師を選ぶことで、研修の体験がよりよくなると思います。

例えば、アジャイルビジネスインスティテュート株式会社のJoe Justiceさんは、Management

3.0に精通しています。スクラムマスターに関する事柄だけではなく、Management 3.0に関する事柄を、研修を通して、体験し、学習することができると思います。LSA CONSULTiNG 株式会社の松永 広明さんは、エンベデッド(組み込み開発)領域や非IT領域へのアジャイル開発の経験があります。ソフトウェア開発とは離れた分野でスクラムを実践される方にとっては、適切な相談先となるでしょう。スクラムについて、通常よりも深く知りたい場合には、スクラムをパターンランゲージ形式で記述した書籍『A Scrum Book』の著者である原田 駒郎さんがおすすめです。いくつかのチームや組織でスクラムマスターを経験された方は、過去うまくいった事象について、高度な言語化を体験できると思います。

16.3.3 日本語で講師と対話できるかで選ぶ

講師が英語話者の場合、通訳の方を通して研修内容を理解することになります。通訳の方はアジャイルコーチをされているなど、スクラムに精通している方が多いですが、せっかく研修に時間を使うのであれば、母国語である日本語で研修を受け、質疑応答を多くしたいという方もいらっしゃると思います。

講師と数多くのコミュニケーションを取りたいという場合は、日本語で研修をする講師を選ぶといいです。

16.3.4 研修方式で選ぶ

コロナ禍以前は、会議室で実施されるオンサイト研修が一般的でした。研修の多くが東京で実施されており、地方で開催される研修の数は少なかったです。しかし、コロナ禍では、オンライン研修が主流になりました。コロナ禍が落ち着いた現在では、オンラインとオンラインの研修を受講者が選択できるようになっています。

現在、研修方式は、大きく3つあります。

- 通常のオンライン研修
- 合宿型のオンライン研修
- オンライン研修

通常のオンライン研修

通常のオンライン研修は、コロナ禍以前からある、伝統的なスタイルです。研修が実施される会議室に参加者が集まり、講義を受けたり、ワークショップに取り組みます。実施されるオンライン研修の多くが、このパターンに当てはまります。

オンライン研修かオンライン研修のどちらかで迷った場合には、私は基本的にオンライン研修をおすすめしています。オンライン研修では、研修の間にランチや休憩の時間も講師や他の受講者と過ごすために、直接的な研修の時間以外でも、接点を持つことができます。そのため、講師にオンライン研修よりも質問ができたり、受講者同士で感想を共有したりすることで、研修の内容を深めることができます。また、各日の研修が終わった後、講師も参加する飲み会が深夜まで開かれることもあります。そういった、研修以外の時間を多く過ごすことで、その先も友人として交流を持ち、スクラムについてより学べる環境に繋がります。

合宿型のオンライン研修

合宿型のオンライン研修は、ホテルなどの施設で研修を受けるスタイルで、日本では近年生まれました。現在、合宿型のオンライン研修を実施しているのは、株式会社アトラクタ、株式会社永和システムマネジメントの2社です。

合宿型のオンライン研修は、研修が行われるすべての時間を、講師や受講者と過ごすため、会議室で行われる通常型のオンライン研修よりも、更に濃密な時間を過ごすことができます。普段の仕事場と隔離された環境だからこそ取れるコミュニケーションが活発に行われ、より深い学びにつながります。

オンライン研修

オンライン研修は、コロナ禍に生まれたスタイルで、現在多くの会社が実施しています。オンライン研修は、研修時間と休憩時間が明確に分かれており、資格取得という観点では一番コストパフォーマンスがよいです。研修中に講師への質問時間は十分にあるため、資格取得に向けた学習という観点では不足はありません。

受講者同士でのコミュニケーションは、あくまでワークショップなど研修の一環の中で行われることが中心です。もちろん、受講者同士が活発に研修時間外でもSNSなどを通してコミュニケーションを取っている場合もありますが、オンライン研修と比較すると少ないです。受講者同士での仲を深めたい方にとって、オンライン研修は物足りなく感じことがあるでしょう。

オンライン研修にはないオンライン研修のメリットは、言語の壁を乗り越えることができれば、世界中の研修にアクセスできることです。日本語での研修を必須としないのであれば、本章で記載している日本の会社以外にも調べてみるとよいでしょう。CSMであれば、Scrum Alliance[®]のWebサイトから、世界中のオンライン研修を調べることができます。

16.3.5 他の選び方について

原則、研修は平日に実施されます。しかし、どうしても仕事の都合で、土日に研修を受けたい方もいらっしゃると思います。アジャイルビジネスインスティテュート株式会社では、土日に研修を実施している場合があります。日程の柔軟性を優先したい場合は、アジャイルビジネスインスティテュート株式会社の研修日程を調べてみるとよいでしょう。

16.4 おわりに

本章では、スクラムマスターの資格の詳細と、資格と研修の選び方について、紹介しました。特に、2日間から4日間と長い時間を過ごす研修を、どのように有効活用するとよいかという観点で、研修の選び方を紹介しました。「講師の考え方や理念で選ぶ」、「講師の得意分野で選ぶ」、「日本語で講師と対話できるかで選ぶ」、「研修方式で選ぶ」のいずれの選び方も良い方法だと考えていますし、複数の観点をかけ合わせてより良い回を選ぶことも可能だと思います。

今後、スクラムマスターの資格を取得したいと考えている方の参考になればさいわいです。



増田 謙太郎@scrummasudar <https://twitter.com/scrummasudar>
<https://scrummasudar.hatenablog.com/>

フリーランスのスクラムマスター。スクラム道関西運営メンバー。アジャイルラジオメインパーソナリティ。2014年、セキュリティソフトウェア企業でアジャイル開発に出会い、2015年からスクラムマスターを担当。2021年、フリーランスとなり、ゲーム業界でスクラムマスターとして、LeSSに取り組んでいる。

第 17 章

アジャイルを勉強した後のキャリアの 5 つのロールモデル

roteskleid (https://note.com/roteskleid)

17.1 最初に

こんにちは、某お固めの会社の情シスで、アジャイル推進をしているものです。子供が二人（小学生・中学生）いるママ SE でもあります。

今回、同タイトルのブログ記事^{*1}を再編集してお届けします。

アジャイルについて興味がある方々がこの本を読まれると思います。これは、アジャイルを勉強した先にどんな未来があるのか、気になる方向けの記事です。

自分の考えを整理するために書いた記事ですが、ぜひ読者の方にも、アジャイルを学んだ後にどんなキャリアを築けるかを考えて頂きたいと思います。

17.2 キャリアの 5 つのロールモデルを考えることになったきっかけ

私は「Chikirin の日記」^{*2}というブログのファンなのですが、この方の記事で、「キャリア形成における「5 つのロールモデルメソッド」」^{*3}なるものがあります。この記事では具体的にエンジニアの 5 つのロールモデルが示されており、自分のキャリアというものに無自覚でいた私に、考えるきっかけをくれました。

以下、記事を引用します。

「私はエンジニアの人に、キャリアのロールモデルを 5 つぐらい提示して選んでもらったらいいんじゃないかなと思ってるんです。

たとえば、第一の道として、「この分野に関してはコイツの右に出る奴はない」みたいなオタクエンジニアになる道。狭く深い知識で生きていくことになるから社内での出世は難

^{*1} <https://note.com/roteskleid/n/nef5c5444955b>

^{*2} <https://chikirin.hatenablog.com/about>

^{*3} <https://chikirin.hatenablog.com/entry/20120512>

しいけれど、ノーベル賞を取るぐらいの勢いで頑張る人なら挑戦していい道だよ、と。

2つめに、そこまで高い技術力はもっていないけれど、トレンドに合わせて売れる商品を器用に開発していくという売れっ子エンジニアの道。

3つめは、エンジニアとしては「まあ、ちょっとね」という人でも、マーケティングや営業をやらせると、技術のバックボーンを活かして他の営業マンとはひと味違う営業をります、という道。

(中略)

「俺は5つのうち、どれに向いているエンジニアかな?」と考えながら必要な勉強を積んでいくことは、エンジニアのためにも、会社のためにもなりませんか?」

是非皆さんにも上述の記事を読んで頂ければと思いますが、私がこれを最初に読んだのは、システムエンジニアとしての自分のキャリアについて迷っていたときでした。

当時、子どもがまだ小さく、これまでセンスがないなりに長時間使ってなんとかこなしてきた開発の仕事も、時短勤務になり上手く回らなくなりました。同じ時短勤務でも、時間の使い方の工夫や高い技術で効率的に進めている方もいらっしゃる中で、このままではいけないと焦ってばかりでした。

そんなときにこの記事に出会い、別のキャリアもあり得るのかと感銘を受け、方向転換を考え始めました。時間はかかりましたが、最終的に社内の開発セクションから別のセクションへの異動希望を出し、異動先でアジャイル開発に出会うことになりました。

このような経緯があったことから、元ブログを書くにあたり、キャリアの5つのロールモデルについて思い出しました。アジャイル開発という、より狭い範囲でのキャリアのロールモデルを整理するのは面白いのでは、と考えました。

17.3 アジャイルに関するキャリアの5つのロールモデル

では、アジャイルに関わるキャリアには、どんなロールモデルがあるのか、をあげていきます。

アジャイルは、狭義にはソフトウェア開発手法の一つであり、アジャイルを勉強したからには、アジャイル開発における開発者を目指すのがぱっと思い浮かぶモデルかもしれません。

しかし、周囲のアジャイルに関わる仕事人を見ていると、決して道はそれだけではなく、アジャイルを手持ちの札に加えることで確実にキャリアの幅が広がると感じています。

ちきりんさんの例にならって、アジャイルに関する5つのロールモデルを上げていこうと思います。

1. アジャイル開発チームの中で輝くスーパー開発者

主にスクラムチームにおける開発者をイメージしていますが、開発チームの一員として、常に改善を心がけ、技術的負債やプロダクトのビジネスの価値についてプロダクトオーナーと対等な議論が出来る、「世界を変える」ことの出来る開発者です。

ユーザー企業・SIer・フリーのどこでも積めるキャリアです。

スーパー開発者になりたいと思ったら、まず毎日新しい情報に触れるよう習慣化することだと、見ていて思います。

スーパー開発者は毎日新しい情報に触れ、試し、良いと思ったら取り入れています。毎日の積み重ねがこの人たちを遠いところへ連れて行くのだといつも感じています。

2. スーパースクラムマスター/アジャイルコーチ

アジャイルの深い知識も技術力も持ちつつチームを俯瞰して、チームを軌道に乗せるスクラムマスターです。スキル・経験を積み上げた人はコーチとして様々なチームを指導します。スクラムマスター研修などの講師もその先にあるキャリアです。

ユーザー企業・SIer・フリーのどこでも積めるキャリアです。

注意点として、ユーザー企業の社員として、スクラムマスターとしてやっていくには、組織がスクラムマスターという役割を理解している必要があります。

従来のプロジェクトマネージャーと混同して、開発者の上司がスクラムマスターを担当してしまい開発者が遠慮してしまうケースが散見されます。

3. イケてるプロダクトを世に出すプロダクトオーナー

プロダクトビジョンを考え、チームメンバーに説明し、開発内容に責任を持つ人です。ビジネスセンスとプロダクトへの愛があり、技術にも理解があると鬼に金棒、社内のステークホルダーに話を通せる一定の政治力が必要です。

ユーザー企業で積むキャリアが一般的かと思います。

エンジニアの転職先としてブルーオーシャンなのではないでしょうか。エンジニア出身のプロダクトオーナー2名とお会いしたことがあります、開発にもプロダクトにもお詳しく述べても頼もしかったです。

4. アジャイルのFWを使って社内のDX推進を行う変革者/コンサル

アジャイルは皆さんもご存じの通り、ソフトウェア開発だけではなく、社内変革にも使われ始めています。人事・予算・開発方法を従来とは違ったやり方で進めていくために、従来のやり方の把握・分析、社内の根回し、各種企画、目玉となるプロダクト開発の立ち上げなどが必要です。コンサルとして他社の変革をサポートするという道もあります。

別のロールで一定のキャリアを積んでから、ユーザー企業、SIerなどで変革を担当することが多いと思います。私が知っているのは、現在某外資系クラウド事業会社で組織変革コンサルをしているこばやしさん^{*4}がまず浮かびました。私が目指しているのもこのキャリアです。

5. アジャイルをそれぞれの職種で生かしてキャリアアップする

最後はアジャイルの考え方、工夫を取り入れてそれぞれの職種で生かす道です。あやなるさんのような人事・育成で、WF開発のプロマネとして、営業として、事務員として……様々な道で生かせると思います。

まず小さく試し結果を見て改善していく、立場が違う関係者を巻き込みチームとして成果を出していくなど、アジャイルの考え方を取り入れることで、これまでとは違った手応えを感じて仕事をしていくけるはずです。

^{*4} https://note.com/kobaya_mik/n/n331382984d1c

17.4 最後に

アジャイルを勉強した後のキャリアの5つのロールモデル、いかがでしたでしょうか。つたないところもありますが、まずは考えたものを形にしてみようと思いました。

私が考えたもの以外にも、おそらく色んな道があるのではと思います。ぜひ皆さんも自分が目指したいロールモデルについて考えてみてください。

以上です。今回は、文章を書く機会を頂きありがとうございました。



roteskleid <https://note.com/roteskleid>

某お固めの企業の情シスに出向している、系列IT会社の社員。出向先でアジャイル推進をしている。情シス・系列会社に向けてのアジャイル研修の講師やアジャイル案件への支援などを担当。CSM、SPC(SAFe® Program Consultant)を保有。2021年度、初めて外部カンファレンスにて登壇。

あとがき

この本を手にとっていただきありがとうございます。

2024年5月
編集長 親方@親方 Project 拝

Agile2

0205 年 2 月 1 日 初版第 1 刷 発行

編 集 親方 Project

発行所 親方 Project

印刷所 株式会社?

(C) 2024 親方 Project