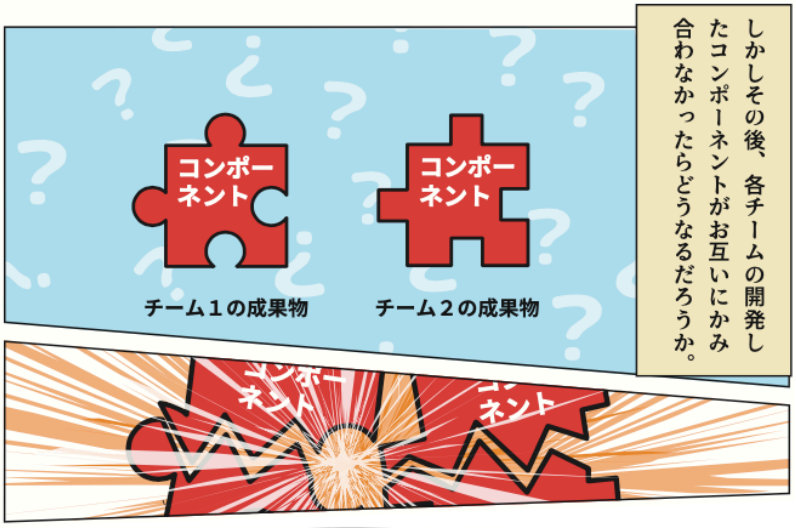
大規模組織におけるソフトウェア開発の誤解その４：各チームが作ったパーツは、魔法の力で組み合わさるのか？

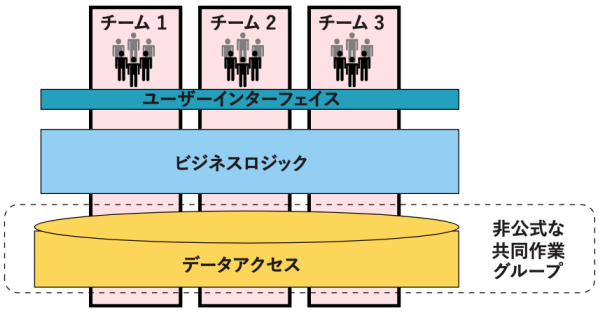
誤解4.1：各チームが開発したコンポーネントを、チームの連携なしで一つのフィーチャーに統合できるか？

私が「コンポーネント」と言う場合は、通常、フロントエンド、バックエンド、「プラットフォーム」、デバイスドライバ等のソフトウェア内のアーキテクチャ層を意味しています。コンポーネントの作業しか行えないチームを有する大規模組織があまりにもたくさん存在します。



コンポーネントチームを無くしてインテグレーションの遅延を回避

アジリティのために通常好ましいのは、複数のコンポーネントにまたがり、エンドツーエンドかつ顧客中心のフィーチャーを共有コードベースで開発できるフィーチャーチーム(https://less.works/jp/less/structure/feature-teams.html)でしょう。コンポーネントチームではこのようなことは行いませんし、行う能力もありません。



フューチャーチームの例

現代的なプログラミング手法（TDD、継続的インテグレーション、トランクベース開発など）を用いるのであれば、フィーチャーチームへ移行すればチーム間の切っても切れない繋がりを少しは減らせる**かも**しれませんが、ゼロにはできません。チームの「生産性」向上ではなく、プロダクト開発のアジリティを目指すのであれば、チームの繋がりをゼロにするべきでは**ない**のです。

誤解4.1.1：○○○の技術アプローチを用いれば、チームが連携する必要は無くなるのでは？

いいえ。

人々がこのような誤解に陥ってしまうのは、マイクロサービスアプローチの売り込みで時折見られる説明の仕方や、SpotifyがChronium Embedded Frameworkを用いてチームの相互依存を減らした方法に関するHenrik Knibergの動画のせいであろうと推測します。本記事の公開からわずか二ヶ月後、とある人物が、チーム間での対話の必要性を無くす手段として「DDD、CQRS、イベントソーシング、リアクティブシステム、アクターモデル」を提案しているのを耳にしました。しかし、PRC、XML、SOAP等によって開発者が協働する必要性が無くなるだろうと誰もが語っていた時代を覚えているでしょうか。どれも素晴らしい発明ではありますが、成功するアジャイルな組織とは、プロセスやツールよりも、個人や対話 (https://agilemanifesto.org/iso/ja/manifesto.html)に価値を置くものです。

誤解4.1.2：きちんと定義されたインターフェースやオープン標準等があれば、チームが連携する必要は無くなるのでは？

既存のAPIは、これまでに共有されてきた種類の情報を共有します。例えば、トラフィックメッセージチャネル (https://en.wikipedia.org/wiki/Traffic_message_channel)を使えば、自動車の交通量を認識できるデバイスを誰でも作成できます。既存のAPIによって、古いものを新しい方法で利用できるのです。

しかし企業は、プロダクトにもっと新しい機能を付けたいとも考えており、そこでは、インタラクションの**両サイド**が未だ変化し続けているため古いAPIでは不十分かもしれません。新たな機能を開発するために新しい種類の情報をソフトウェアの異なるパーツ間で共有する必要があるとすれば、開発者達は、それをどうやって行うか、そして新たな種類の情報とは何かについて、共通認識を作りあげねばなりません。

チームはお互いに対話する必要があるでしょう。アジャイルマニフェスト (https://agilemanifesto.org/iso/ja/principles.html)において、最も効率的なコミュニケーションの取り方とは何だったでしょうか？

誤解4.2：異なるチームが作った複数のフィーチャーを、チームの連携なしで1つのプロダクトに統合できるか？

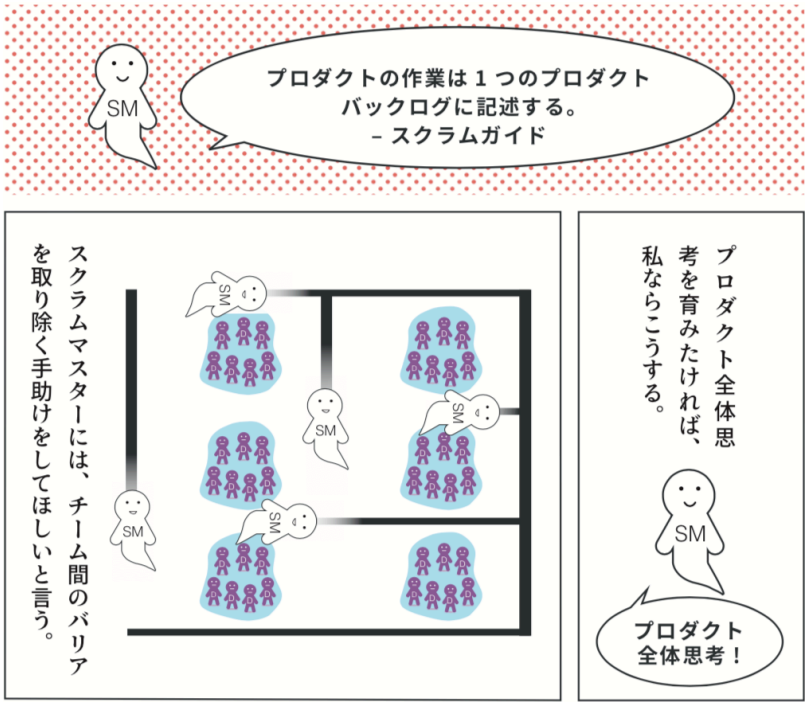
素晴らしいことに、あなたの会社がコンポーネントチームからフィーチャーチーム (https://less.works/jp/less/structure/feature-teams.html)へと移行したとしましょう。**それでもなお**、内外において一貫性を保ったプロダクトを開発するために、フィーチャーチームは互いに連携する必要があります。

主要部分において共有のコードベースへの統合を継続的に行う場合、内部アーキテクチャ及び共有データを乱雑にしないよう保つにはどうすればよいでしょうか。 (https://less.works/jp/less/technical-excellence/architecture-design.html)（もし頭に浮かんだ答えに「マイクロサービス」という言

葉が含まれていたなら、どうぞ前のセクションを読み返してみてください。）それぞれのチームが得た学びを掛け合わせて実りを得るにはどうすればいいでしょうか。どうすれば全員が**プロダクト全体思考**を育むことができるでしょうか。

プログラマーではない人にとって最も分かりやすい例は、ユーザーインターフェースとユーザーエクスペリエンス（UI/UX）かもしれません。みなさんは、パーツ毎に別のデザイナーが設計したように感じるプロダクトを使ったことはありませんか？一貫性のあるプロダクトを作るには、UI/UXに携わるチームメンバーにはチームの垣根を越えて協働してもらった方がよいでしょう。

したがってスクラムマスターは、チーム内の連携を促進するのと同様に、チーム間の連携も促すべきです。幸運なことに、これを成し遂げるための「スクラムオブスクラム」より優れた方法が数多く存在します。



誤解4.3：個別に、もしくはパッケージソフトの一部として販売されているプロダクトは、チームの連携なしに統合できるか？

仮に、10チームを有するプロダクト会社が相互に無関係なプロダクトを同時に10個、いずれも同じ集中力を注いで開発するとしたら、それは奇妙なビジネス戦略でしょう。（ここでは真の**プロダクト会社**について述べています。様々なクライアントに特注のソフトウェアを作成するプロジェクトサービス会社ではありません。）連携しないプロダクトを作っても、企業が競争の優位性を得ることはできないからです。（過去に開発した2，3個のプロダクトを時折保守管理する場合があるかもしれません。大規模組織におけるソフトウェア開発の誤解その２:すべてのチームが等しい価値の業務に取り組んでいるか？で示したとおり、どれも同等の注意を惹く保証はありません。）

新規開発を全く行っていないという些末な場合を除いて、プロダクトを適切に連携させるにはエンジニアリングの努力が必要です。アドビ製品を例に取ってみましょう。様々なアドビ製品を用いる場合、それらの価値は、プロダクト間のインテグレーションがどれだけ上手く機能するか（あるいは機能しないか）に左右されます。

共通項だけを持ち合わせた無難なインターフェイスを用いることにより、業務フローで互換性のないプロダクトを代替できる場合もあります。ただしこのやり方は手間がかかり、機能が一部失われることもあります。

例えば、ある画像編集ソフトで作成したアニメキャラクターを、互換性のないeラーニング編集ソフトに移動させることができるとします。ただし、最初の画像編集ソフトからeラーニング編集ソフトにPNGファイルをエクスポートするには、余分な段階を踏まなくてはなりません。エクスポートの過程でPNGファイルフォーマットからレイヤー情報が失われたことにより、eラーニング編集ソフトではそのキャラクターの表情を変更することができません。

一連のプロダクトの品質を高めるには、自社プロダクトとのインテグレーションを無視してはいけません。

私の顧客の中に、従来の標準やAPIがほとんど存在しない領域に属する企業があります。（アドビと異なり）先駆者のいない事業を行っているためです。その企業では、数百人の開発者が、以前は数十の異なるプロダクトであったものに取り組んでいます。顧客がその全体を一つの統合されたパッケージであると感じたときに、競争における真の優位性を得られることを彼らは理解しています。これらのプロダクトは、元は、連携を予定して開発されたものではありませんでした。

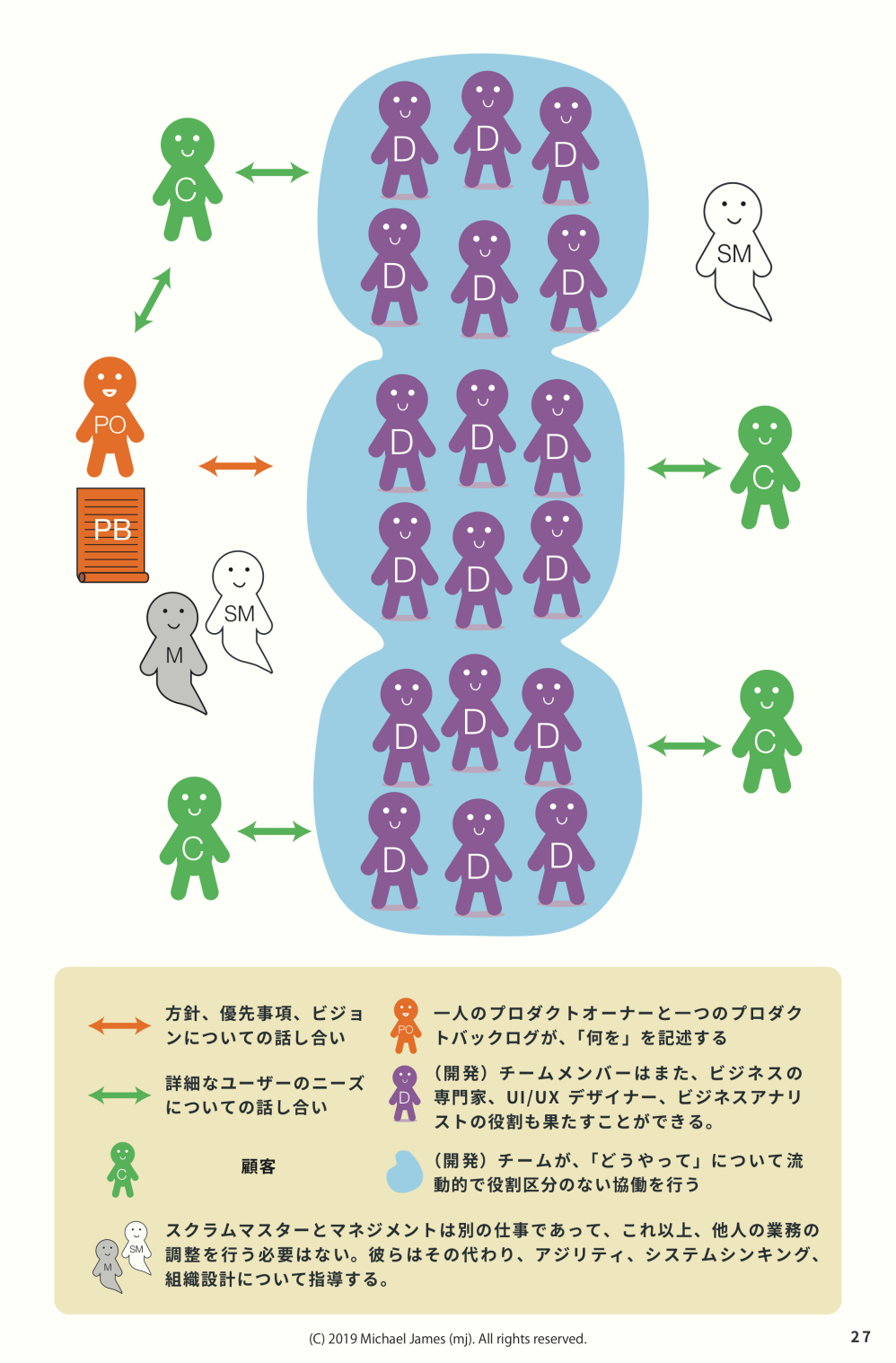
彼らは何百万ドルもの労力を費やしていますが、プロダクトが内外で連携することは、彼らにとって、その価値に見合うものなのです。もしもその企業に派遣されるコーチが、プロダクト全体思考ではなく、私が15年前にやっていたような個別チームの生産性にフォーカスさせる典型的なアジャイルコーチやスクラムトレーナーであったら、有害となるでしょう。

この誤解を解こうとする試みの数々

これは、次のような考えがいかに馬鹿げているかを説明する 1968年のNATO会議録 (http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF)です。

> このシステムをソフトウェアの残りの部分に接続するのは些細なことで、後で簡単に行える。

プロダクト会社に12人以上いるのであれば、各チームは、他のチーム（及び顧客）と協働することが望ましいと考えます。



English version (https://seattlescrum.com/misconception-4-will-parts-made-by-different-teams-fit-together-by-magic/)