計画通りに開発が進まないスクラムチームに関する研究
～チーム内の共通理解って大事～

一般財団法人日本科学技術連盟
第37年度（2021年度）ソフトウェア品質管理研究会成果発表会

研究コース4 アジャイルと品質 プロセスチーム
2022年2月25日（金）

アジェンダ

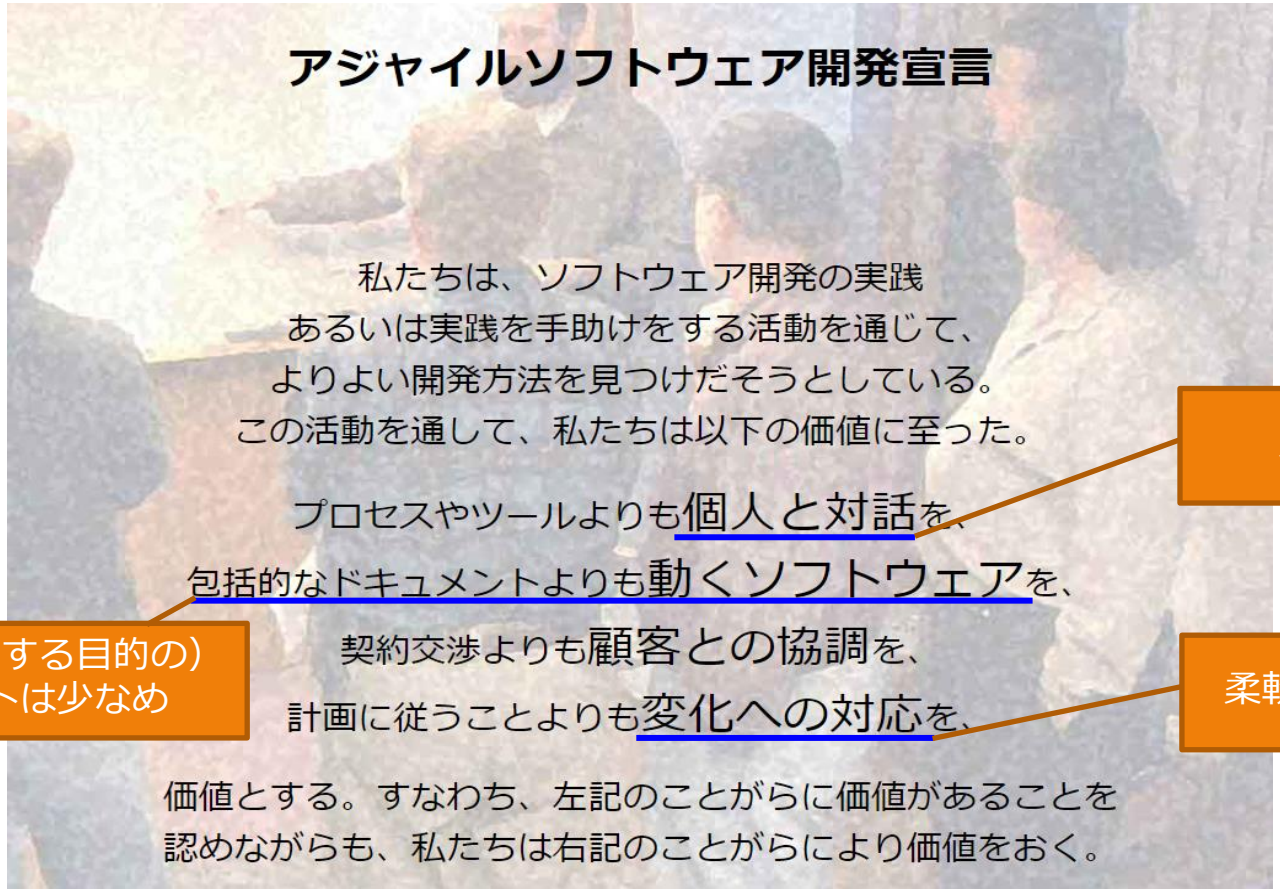
- **発表内容**
  - はじめに
  - 研究課題
  - 実験１（Aチーム）
  - 実験２（Bチーム）
  - まとめ

研究メンバー		
研究員	西村 優	東京エレクトロン株式会社
	澁澤 良	株式会社ｆｅａｔ
	村上 拓也	テックスエンジソリューションズ株式会社
	星野 友基	ビー・ユー・ジーDMG森精機株式会社
	澤口 勲	株式会社ビズリーチ
主査	永田 敦	サイボウズ株式会社
副主査	山口 鉄平	freee株式会社
	荻野 恒太郎	株式会社カカクコム
アドバイザー	細谷 泰夫	三菱電機株式会社

はじめに

アジャイルソフトウェア開発宣言

- **2001年に「アジャイルソフトウェア開発宣言」が発表**
- **アジャイル開発の採用率は増加傾向**



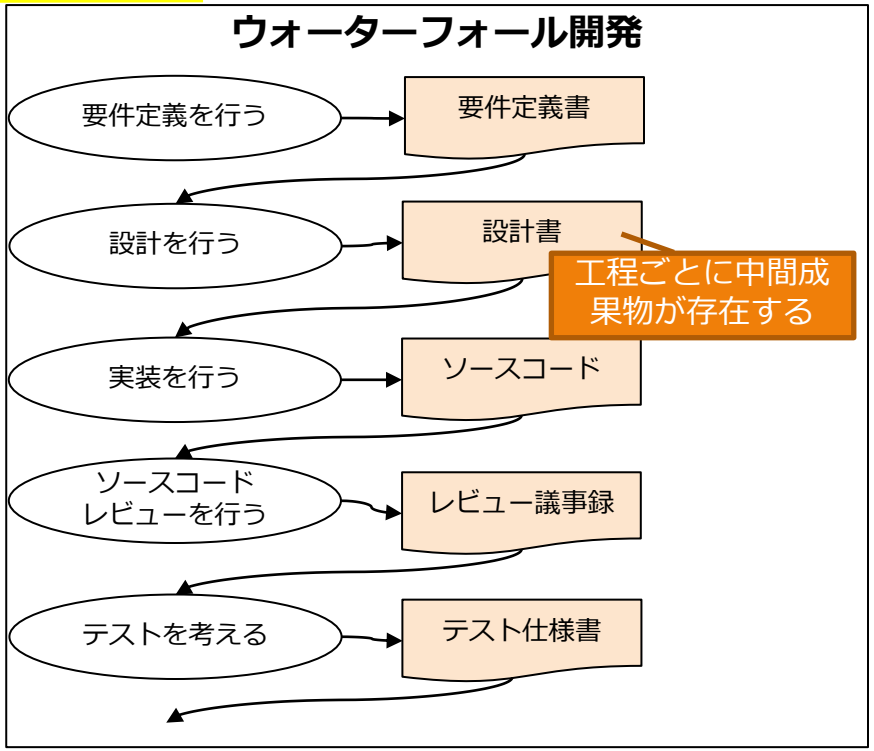
対話重視型

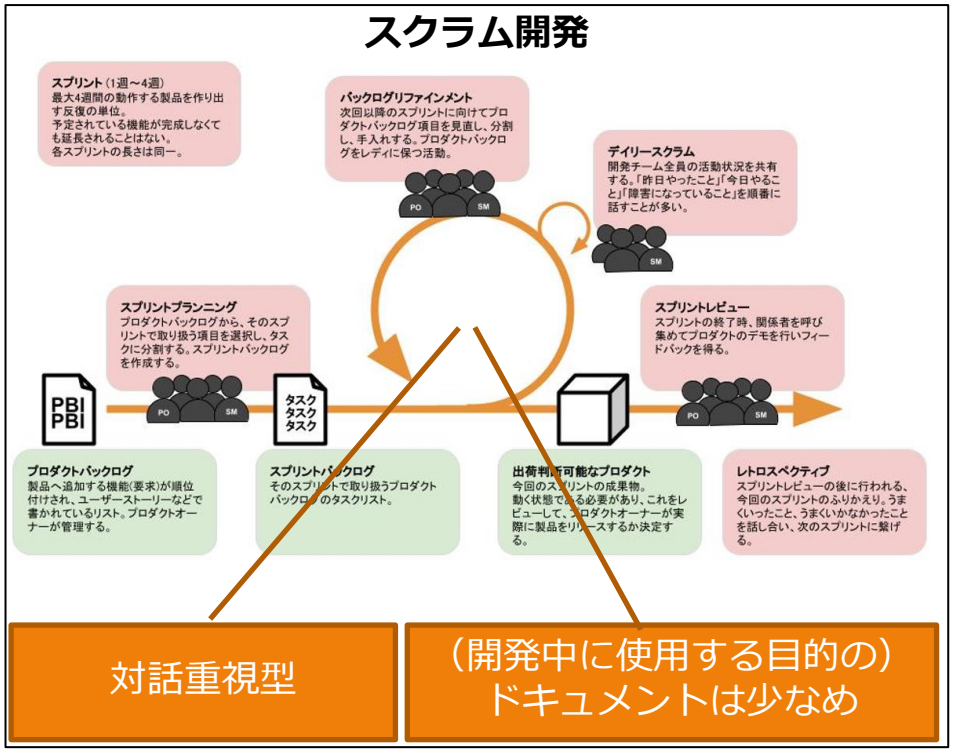
（開発中に使用する目的の）
ドキュメントは少なめ

柔軟、迅速な対応

https://agilemanifesto.org/iso/ja/manifesto.html

研究コンセプト





- **我々のチームの考え方**
  - 外部監査的なチェックでの品質保証ではない形を目指す
  - スクラム開発の場合、開発時点で品質を作り込むことが重要
  - **有形なモノ（成果物）より、無形な行動（プロセス）に着目**

⇒開発を行うプロセスに着目し、作り込み品質向上を目指した

研究概要

研究概要

- **スクラム開発に精通していない組織がスクラム開発を行った場合，計画通りに開発が進まなくなるケースが散見される.**

- **開発作業が計画通りに進まないと遅延が生じ，納期に間に合わせるように無理に開発を行う事で検討不足やテスト不足などによる品質低下に繋がる可能性がある.**

- **本研究では,タスク漏れや手戻りにより,計画通りに開発が進まない状態のチームの改善のための研究を行う.**

研究対象チーム状況

- **Aチーム**
  - 開発中に予定外の作業が増えることが多い
  - 作業を進めていくうちに作業が増え、計画通りに開発が完了していない
- **Bチーム**
  - 期間内に開発は完了している
  - 手戻り作業が多発している
  - 納期を優先して一部工程を省略／短縮し、品質問題が多発している
- **Cチーム**
  - 計画通りに円滑に開発を行っている

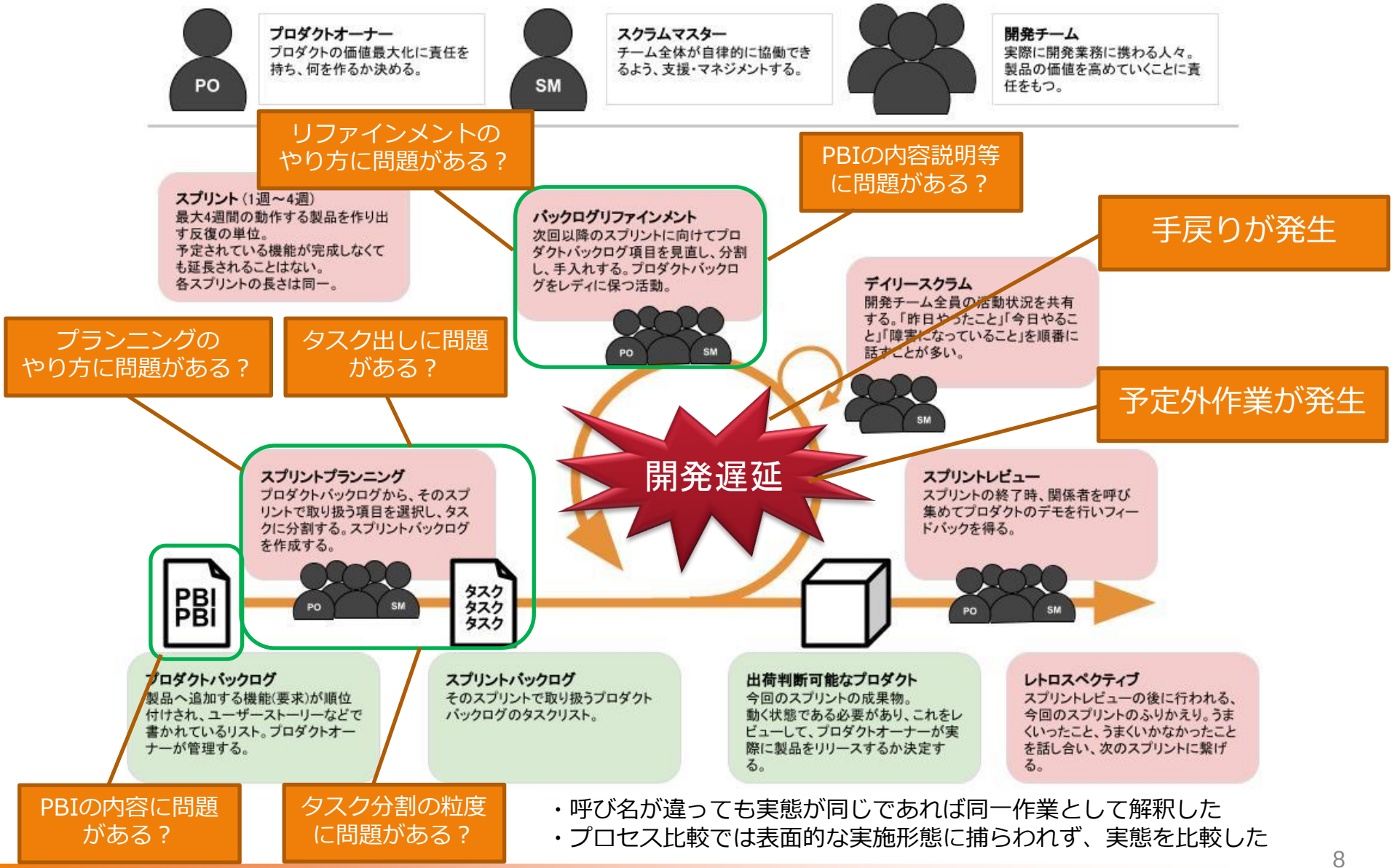
チーム構成

		Aチーム	Bチーム	Cチーム
PO（Product Owner：プロダクトオーナー）		1名	1名	1名
SM（Scrum Master：スクラムマスター）		無し	1名	1名
開発チーム		15名	3名	7名
内訳	開発者	14名	2名	6名
	QA	1名	1名	1名

チーム開発状況

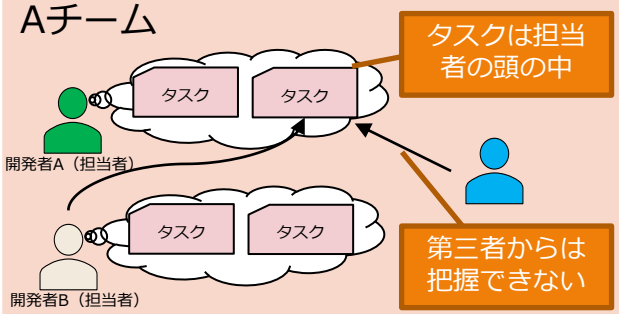
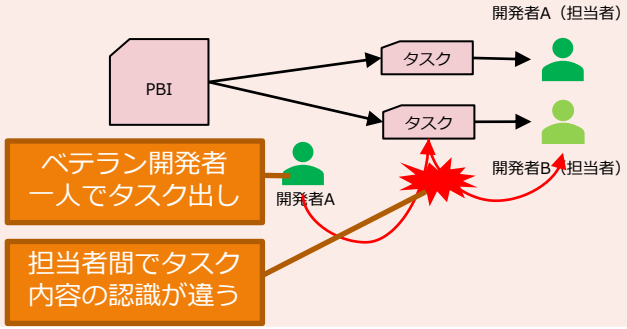
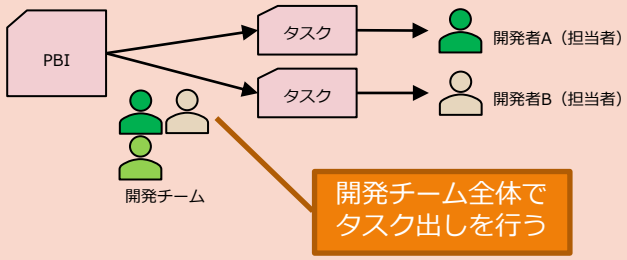
	Aチーム	Bチーム	Cチーム
スプリント期間	2週間	2週間	1週間
1スプリントのPBI数	110 PBI/Sprint	2～3 PBI/Sprint	5 PBI/Sprint
計画遅延発生率（未完了率）	66.4%	0.8%	3.4%
手戻り発生確率	ー	90%	0%

開発遅延要因検討



- 呼び名が違っても実態が同じであれば同一作業として解釈した
- プロセス比較では表面的な実施形態に捕らわれず、実態を比較した

タスク出し作業比較

Aチーム  	・タスク出しをチームで行っていない →タスクは担当者の頭の中に漠然と存在する状態 →担当者以外がタスク内容を把握するのが難しい ・開発作業中に予定外の作業が増えていく状況 **複数人でのタスク出しを行う事で改善するのではないか？(実験1)**
Bチーム  	・タスク出しはベテラン開発者が一人で実施 ・洗い出されたタスクはタイトルのみで情報量が少ない ・担当者に対する口頭説明が不十分で、担当者は理解不足 ・担当者間の認識ずれにより、手戻り（修正や追加作業）が発生 **タスク出し方法を変えることで改善するのではないか？(実験2)**
Cチーム  	・タスク出しは全員参加 ・実担当者もタスク内容は十分に理解できている状況 →タスク作業に関する認識ずれなどは発生していない

複数人でのタスクの洗い出し

実験１（Aチーム）

実験1（Aチーム）

- **Aチームの特長**
  - 計画遅延が多く発生
  - 開発後半に予定外の作業が増えていく状況

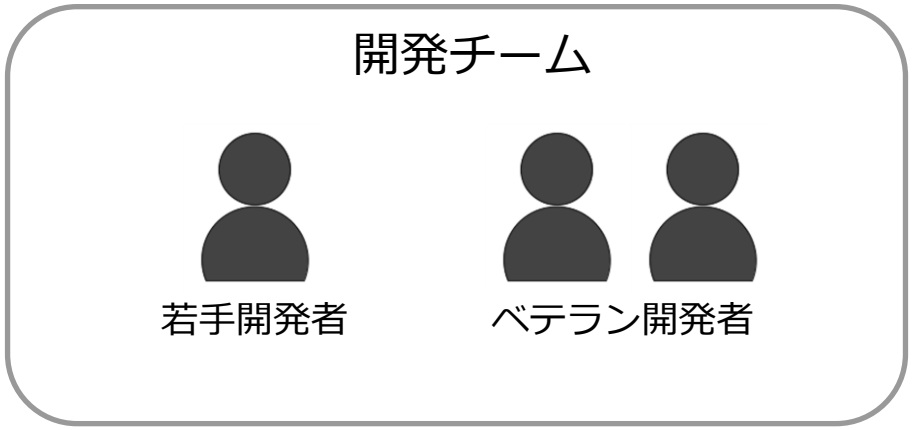
	Aチーム	Cチーム
スプリント期間	2週間	1週間
1スプリントのPBI数	110 PBI/Sprint	5 PBI/Sprint
計画遅延発生率（未完了率）	66.4%	3.4%
手戻り発生件数	ー	0 件/PBI

- **スプリントプランニング**
  - スプリント内で実施するPBIの確認と担当決めを実施
  - タスク出しをチームで行っていない
    - タスクは担当者の頭の中に漠然と存在する状態
    - 担当者以外がタスク内容を把握するのが難しい

⇒タスク漏れが計画遅延の大きな要因と考えられる

実験1（Aチーム）

- **実験内容**
  - 開発チームメンバ複数人でタスク抽出を実施
    （しっかりとプランニングしよう！！）
- **具体的な手順**
  - 担当者(若手開発者) タスク事前抽出
  - プランニングで担当者が事前に洗い出したタスクを説明
  - 参加者全員でタスク抽出
  - 必要と合意したタスクを追加



実験1（Aチーム）

- **実験結果**

若手開発者

こんなたくさん気づいてないタスクがあるなんて！

ベテラン開発者

ソフト開発自体のタスク漏れは無いね

工場やお客様への提供までを考慮できるとよいね！

A君溢れたらB君に渡すのも楽だね！

Aチーム（実験前）	Aチーム（実験後）
パラメータ変更の実装	検証日程の外注とのすり合わせ
パッチの作成	工場への指示票提出
インストーラの作成	工場での確認項目連絡
変更箇所のテスト項目の作成	対象機械が2台、もう1台への提供
検証環境準備	工場での組み立て日程確認
	出荷日と顧客への日程連絡

実験1（Aチーム）

- **タスク漏れが抑制できた**
  - 若手開発者が経験不足のため見落としがちな作業が追加
    - 部外との調整
    - ソフト開発後の現場への展開

    といった開発作業以外のタスク
  - 追加されたタスク数
    - 5タスク→12タスク（7つ追加）

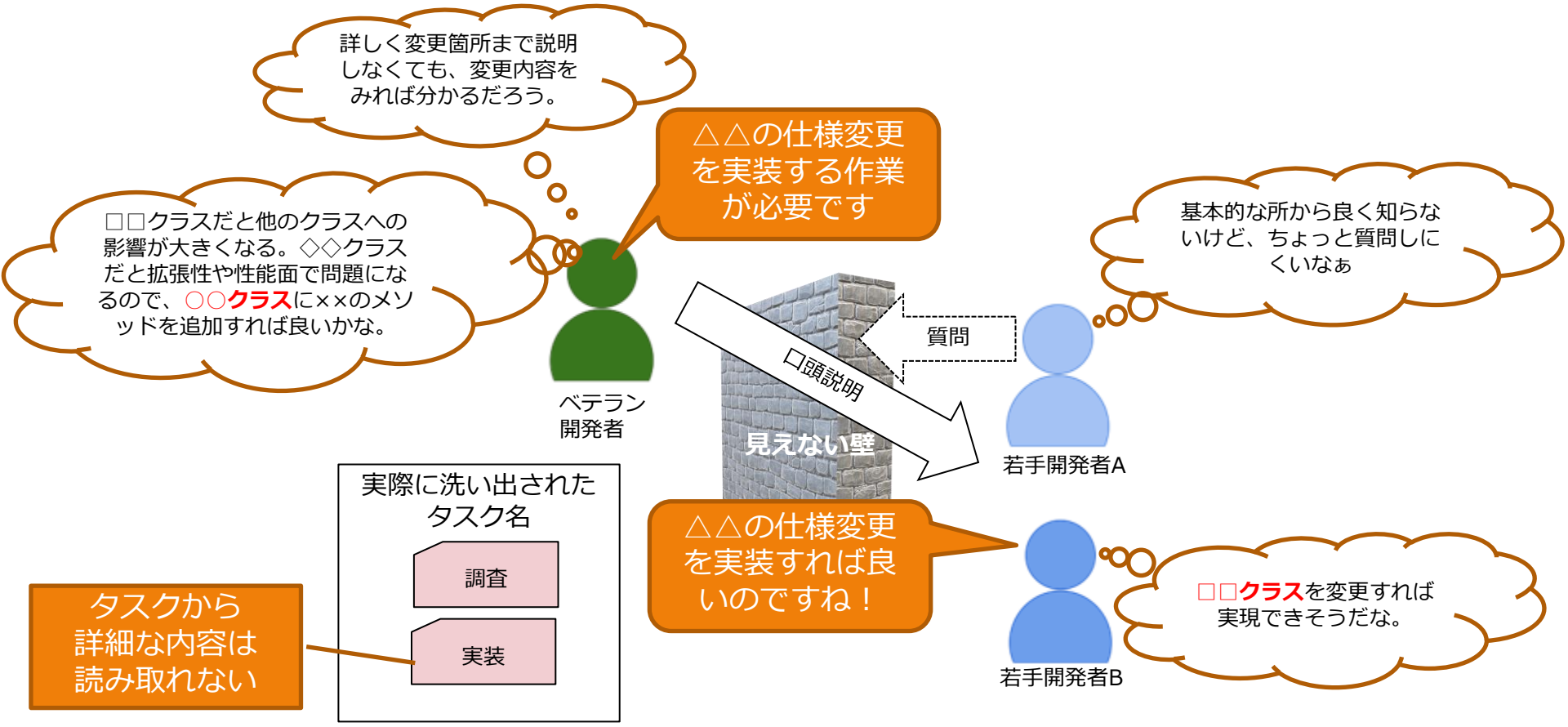
- **実験結果**
  - タスク出しを行う事で、タスク漏れがある事に気が付ける
  - ベテラン開発者を交えてタスク出しを行う事で、経験不足を補える

- **その他の効果**
  - 「状況に応じたタスクの入替」が容易

タスク出し方法の改善

実験２（Bチーム）

実験2（Bチーム）実験前の状況



ベテラン開発者と実担当者で作業内容の認識にずれがある

※イメージ図です

実験2（Bチーム）実験内容

- **実験内容**
  - タスク出しを開発チーム全員参加で行う
  - 開発チーム全員の**共通理解になるように**タスク内容の共有を行う

- **実験手順**
  - チーム全員参加でPBIを議論し，ベテラン開発者が必要タスクを洗い出す
  - ファシリテーターがベテラン開発者にタスク内容を質問し，説明を求める
  - ファシリテーターが参加者全員の理解状況を確認する
  - 参加者全員で合意したタスクの内容をチケット化する

- **実験時のポイント（タスク出しの雰囲気作り）**
  - 場の雰囲気を変える
    - ファシリテーターが先ずベテラン開発者に質問を投げかける
    - 初歩的な質問を交えることで、発言しやすい雰囲気を作る
  - 観察する
    - 若手開発者の顔色や反応を観察し、理解度合いを推測する
  - 問いかける
    - 理解不足が推測される場合は、若手開発者に問いかけを行う
    - 若手開発者が質問しやすい雰囲気を作る

経験や立場の違いも考慮し、タスク出しの雰囲気作りに気を使った

実験2（Bチーム）実験時の状況

今回の実現方法は３個の選択肢があります。
□□クラスだと他のクラスへの影響が大きくなります。
◇◇クラスだと拡張性や性能面で問題になります。
○○**クラス**に××のメソッドを追加するのが良いと思います。

どのあたりが分からないのか、分かってきた

ベテラン開発者

質問

観察・確認

若手開発者

なるほど。□□クラスだと、沢山の参照しているクラスを変更しないと行けなくなるんだな。

影響範囲だけじゃなく、拡張性や性能面も考慮すると○○**クラス**を変更するのが良いのか。

このタスクで想定している修正箇所はどの部分ですか？

○○クラスを変更する理由は？

他の実現手段はどういう物がありますか？

ファシリテーター

○○クラスの変更イメージは理解できた？

理解不足が推測される場合は若手開発者に問いかける

若手開発者の雰囲気を観察する

参加者全員の共通理解になったかな

初歩的な質問をする

若手開発者の代わりに質問する

タスク出し方法の改善 実験結果

- **開発結果**

実験前後のタスク数と手戻り発生数

	Bチーム（実験前）	Bチーム（実験後）
1スプリントのPBI数	2～3 PBI/Sprint	1～3 PBI/Sprint
PBI当たりのタスク数	4 タスク/PBI	10 タスク/PBI
手戻り発生件数	0.9 件/PBI	0 件/PBI

  - 90%の割合で発生していた手戻りは発生しなかった
  - 納期も前倒して完了できた
    - １スプリント目　２日前倒し
    - ２スプリント目　１日前倒し
  - 品質問題も発生していない

- **洗い出されたタスク**

実験前後のタスク名（一例）

Bチーム（実験前）	Bチーム（実験後）
調査	使用できるOSSの調査
	○○仕様書の調査
実装	○○クラスの実装
	○○のエラーケースの実装
評価	連動確認評価
	デグレード確認評価

  - タスク数は4個から10個に増えた
  - タスク内容は詳細化、具体化された

タスク出し方法の改善 まとめ

- **作業分担が可能になった**
  - 今までは一人の優秀なベテラン開発者中心で作業していた
  - 実験では**若手中心の開発チーム全員で分担して作業**を行う事が出来た
- **手戻り抑制**
  - 今までは若手開発者の作業で手戻りが発生していた
  - 若手開発者の作業タスクは増えたが、**手戻りは発生していない**
- **納期短縮が短縮された**
  - **納期遅延がなく、予定より前倒して作業完了**する事ができた
- **若手開発者の関わり方が変わった**
  - 受け身だった若手開発者が実験後は**積極的に質問を行う**ようになった

タスク出し方法の改善 まとめ

- **質問を行う事で、実装箇所が明確になったタスクがあった**
  - 「実装」が「○○クラスの実装」に変わった
  - 複数の実現手段の中でベテラン開発者は**暗黙知**として選択していた
  - 質問により、**なぜそのクラスを選択したのか**、理由を聞き出せた
    - クラス毎のインターフェイスや挙動の違い
    - 影響範囲や考慮すべきこと等

- **ポイント**
  - 若手開発者が実装時には詳細まで配慮して手戻りなく開発できた
  - タスクに書かれている内容以上の事が共通理解となった
  - **ベテラン開発者が気にかけているポイントも共通理解**となった

まとめ

まとめ

- **重要なポイント**
  - チームが全員参加する形で実施する
  - 参加者の**共通理解になるまで対話**を行い
  - **参加者の合意を形成**する

ドキュメントより対話を
重視するアジャイル開発

- **実施のためのポイント**
  - 全員参加の雰囲気作り
    - 初歩的な質問を行う
    - 代わりに質問を行う
  - 参加者の共通理解度の確認
    - 参加者の雰囲気を観察する
    - 参加者に問いかけを行う

ご清聴ありがとうございました。

補足資料

スクラム開発概要