ソフトウェアメンテナンスプロジェクトチームの心理的安全確保に向けたCAST適用の事例

日下部　茂[1], 三輪　東[2]

[1] 長崎県立大学, [2] SCSK

2018年12月4日

アジェンダ

- はじめに
- プロジェクト情報
- CAST
- プロセスモデルと相互作用
- おわりに

アジェンダ

- はじめに
- プロジェクト情報
- CAST
- プロセスモデルと相互作用
- おわりに

発生イベント

移行フェーズ(運用直前)に金融系システムで欠陥

- エンドユーザに損害はない
- メンバの心理的安全が損なわれた

当初のレポート

- 委託先
  - 開発者：対象範囲外のコード修正, 欠陥混入, テスト漏れ
  - サブリーダ：プロセス逸脱防止できず,テスト漏れを許す

  個人非難的，心的安全の喪失
- 元請け側
  - 特になし

安全(心理的安全)

- 他者からの反応に怯えたり，羞恥心を感じたりすることなく，自然体の自分自身を晒せる環境

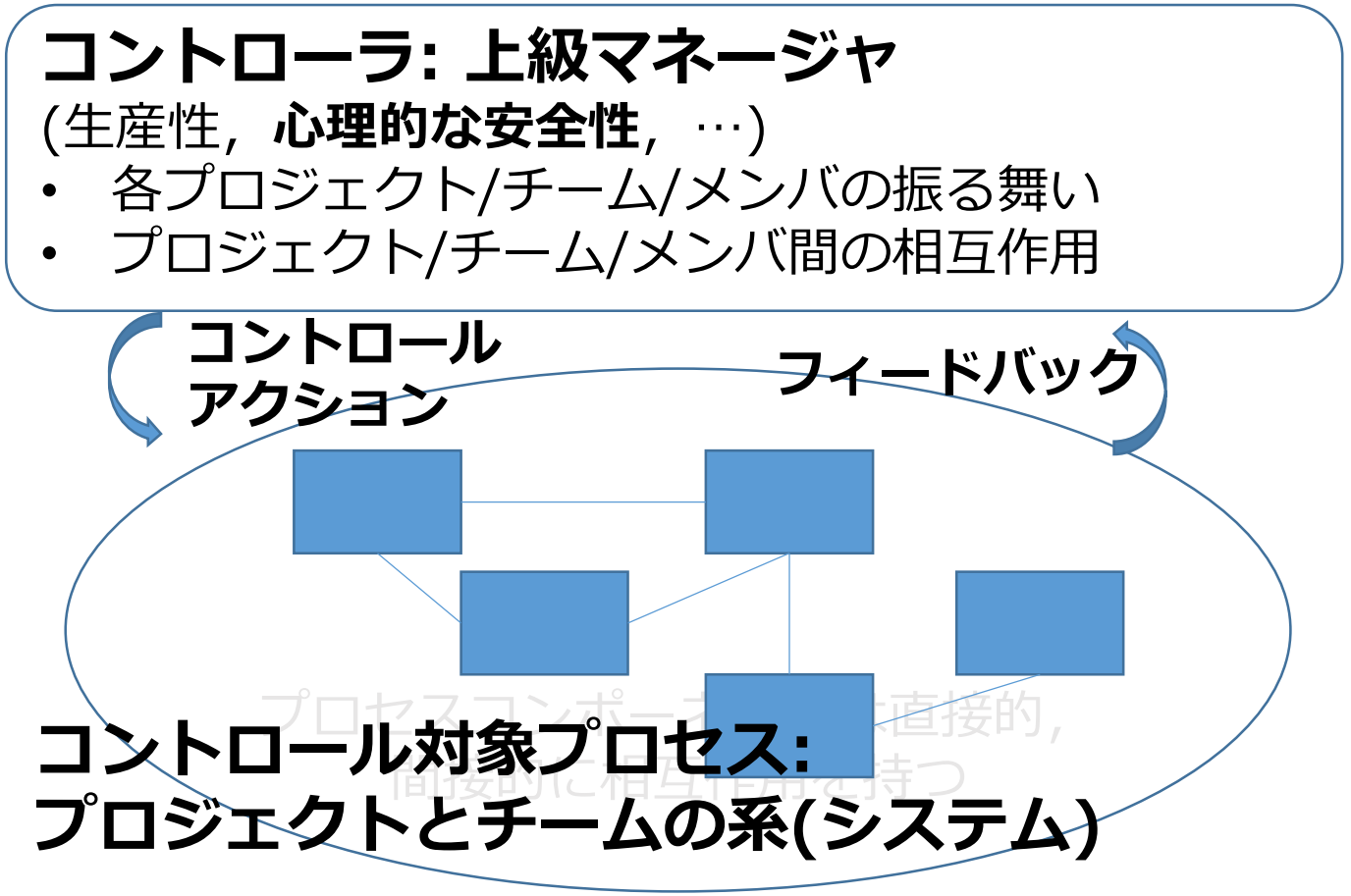
例：ソフトウエア開発関連

- Project Aristotle by Google
- Agile
  - Anzeneering

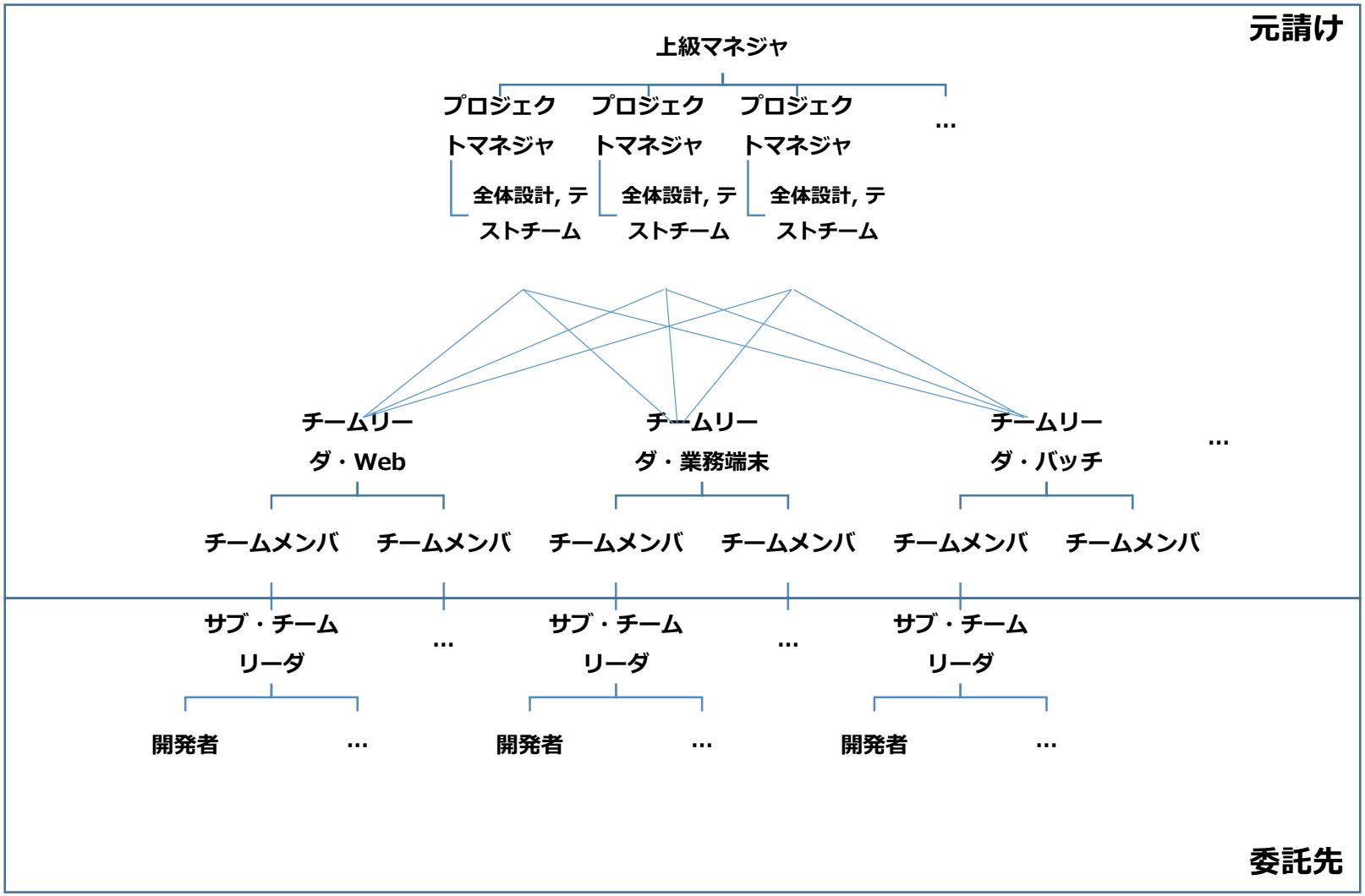
→モダンアジャイル四つの原則

- [最高]人々を最高に輝かせる
- **[安全]安全を必須条件にする**
- [高速]高速に実験＆学習する
- [継続]継続的に価値を届ける

創発的特性の制御


コントローラ: 上級マネージャ
(生産性，心理的な安全性，…)
• 各プロジェクト/チーム/メンバの振る舞い
• プロジェクト/チーム/メンバ間の相互作用
コントロール
アクション
フィードバック
コントロール対象プロセス:
プロジェクトとチームの系(システム)

委託先と元請けの構造概要



元請け業務の特徴（その１）

- 担当領域が広く様々な知識と技術を必要
  - 基幹系システム全体，アプリ・インフラ・運用の開発（立ち上げから導入まで）と保守（24時間365日）
  - アプリケーションごとに動くだけでなく，他システムとの連携や性能なども考慮した高難易度の設計が必要
- ITの前に法律や業務が複雑，用語も固有
  - 業務特有の
  - 取り扱い商品も複数あり，その分の知識が必要

ミッションクリティカルなシステム

業務知識の方が一般的なIT知識よりも希少価値がある

育てるので長く働いてほしい

委託先の背景

- ニアショア拠点
  - 安定した仕事量が欲しい

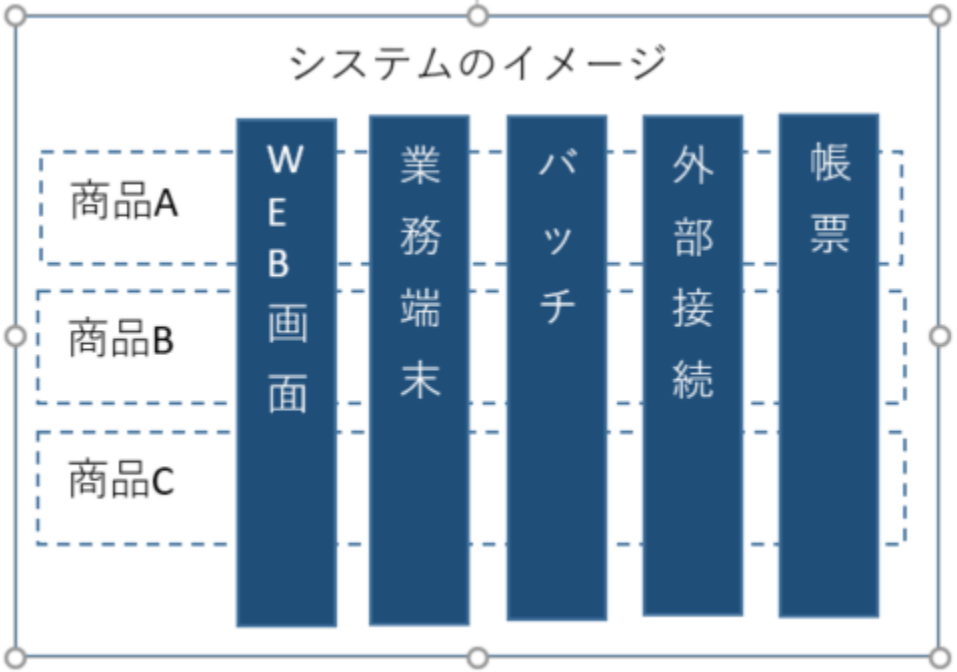
約10年前，リーダーが東京に出てきて，
一緒に仕事を覚えた経験を持ち帰り

ミッションクリティカルへの理解，業務知識をベースとし，かつ元請けで行っていた作業の一部を委託先で巻き取れることを武器に

影響調査・モジュール変更
管理の一部は委託先管理に

元請け業務の特徴（その２）

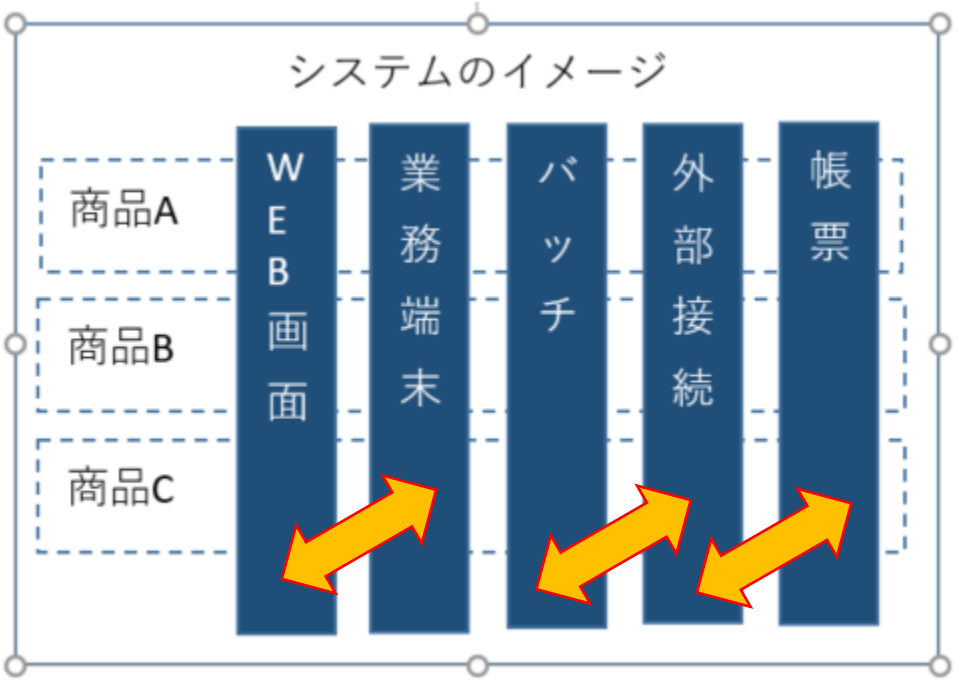
- 多様な頻度と仕事量
  - 毎月なんらかの改修を行っている機能もあれば，年に１回改修が入るか入らないかの機能がある
  - 法令改正や新サービス追加で，ある時期だけ仕事量が急激に増えることもある



柔軟に
配置できる
要員が
望ましい

元請けと委託先で合意

多能工



異なる
チームの
業務が
遂行できる
人財育成

アジェンダ

- はじめに
- プロジェクト情報
- CAST
- プロセスモデルと相互作用
- おわりに

知識・スキルとプロジェクトの組合せ
マトリクス型組織を採用

	Web	業務端末	バッチ	外部接続	帳票
PM プロジェクトA	✓	✓	✓		✓
PM プロジェクトB	✓		✓	✓	
PM プロジェクトC	✓		✓		✓
…					

PMはプロジェクト全体をコントロール

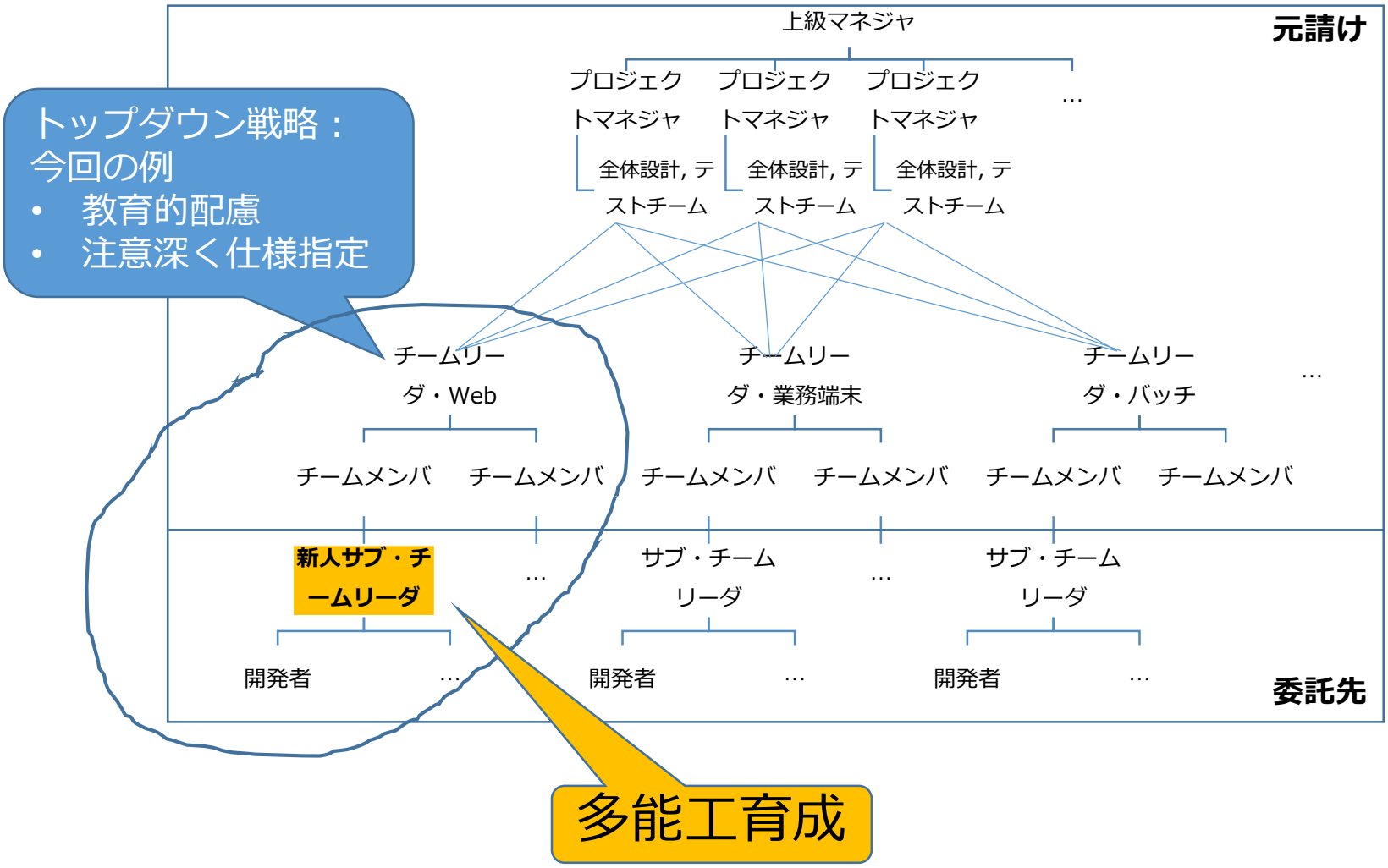
チームリーダー

チームメンバー

委託先
リーダー
メンバー

成果物のQCDコントロールはチーム単位で行われる

組織をまたがるプロジェクトの構造概要



アジェンダ

- はじめに
- プロジェクト情報
- CAST
- プロセスモデルと相互作用
- おわりに

CAST ステップ

1. 損失に関与したシステムレベルのハザードを識別
2. そのハザードを制御するために存在した制御構造を識別
3. その損失につながった近接イベントを究明
4. 損失を物理システムレベル(注)で分析
   a. 関与した物理的な制御や設備を識別
   b. この事故の防止のための全ての物理的な安全要件と制約を識別
   c. 物理的な装置のあらゆるfailuresもしくは不適切な制御を識別
   d. 物理的なfailureもしくは不適切な制御を説明するコンテキスト要因を識別
5. 引き続く各高位のレベルが，どう，なぜ，現レベルの不適切な制御に寄与したか解明するため，高位のレベルを分析
   a. 次の高位レベルの制御に対する，安全関連の責務を識別
   b. 非安全な決定と制御アクションを識別
   c. 非安全な決定と制御アクションを説明するプロセスモデルの欠陥(誤信)を識別
   d. その時点でなぜその振舞いが適切に思えたか説明するコンテクスト要因を識別
6. 損失に寄与した全体的な調整ややりとりを確認
7. 損失に関連したシステムと安全制御構造のダイナミクスや変化だったり，安全制御構造の継時的な弱化を割り出す
8. 改善勧告を出す

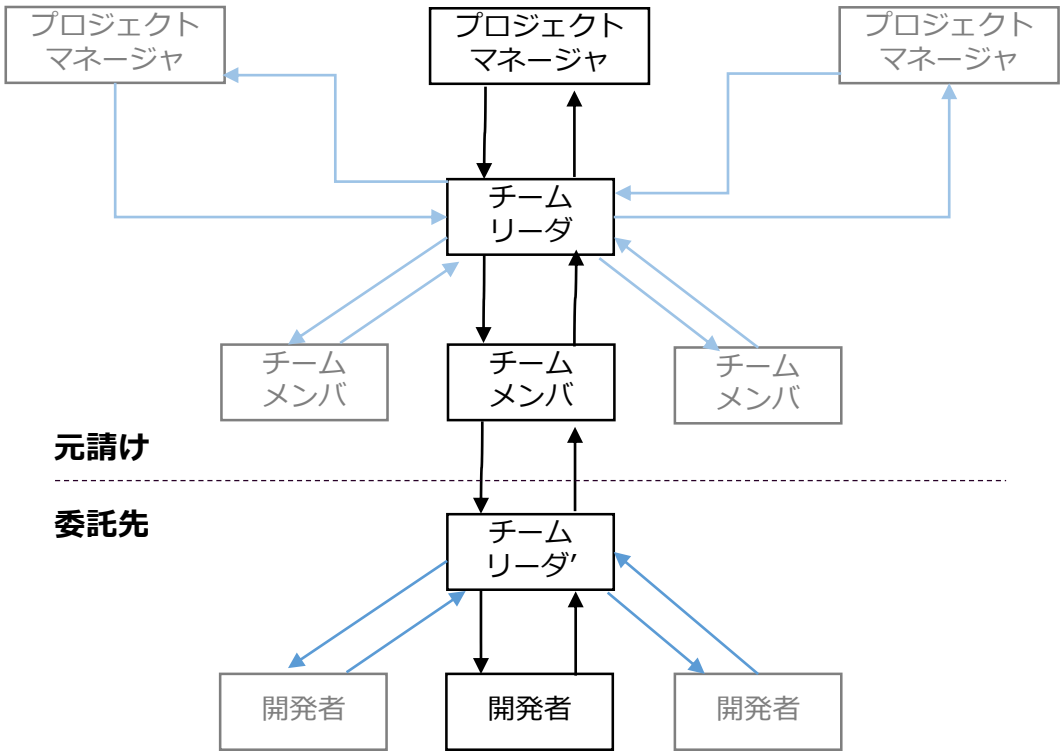
注： 適宜読み替え，例えば組織なら末端の個人行動等

Step1 ハザードの識別

- 失敗したスコープマネージメント
  - 指定範囲外のソフトウェアを修正
- 失敗した検証
  - 修正部分に存在した欠陥を見逃す(テスト対象外に)

Step2 存在した制御構造を識別

- レビュー
- テスト



Step3 損失につながったイベント

- 開発者が指定範囲外のコードを修正
- 修正部分に欠陥を混入
- 修正部分をテスト範囲外とするミス

Step4 末端レベルの損失の分析

開発エンジニア

- 指示範囲外の修正
- 修正部分にミスの混入と残留

安全に関する責務:
- 対象範囲関連も含め指示・仕様に従う
- 欠陥の混入回避に努める
- 混入欠陥の発見に努める

コンテクスト:
- バックログ(プロジェクト非固有)を修正した経験あり
- 新人のサブリーダ

非安全な意思決定やコントロールアクション:
- バックログ修正部分をテスト対象に含めず(勘違い)
- バックログ処理時に適切な扱いが出来ていなかった

プロセスモデルの欠陥:
- テストのカバー範囲に対する誤った認識
- バックログ処理に関して共有すべきことを未把握

Step5 高位レベルの制御を分析

サブチームリーダ

- 修正範囲について一旦指摘後に承認
- テスト範囲漏れについて一旦指摘後に承認

安全に関する責務:
- 対象範囲関連も含め指示・仕様に従う
- 欠陥の混入回避に努める
- 混入欠陥の発見に努める

コンテクスト:
- この領域の新人なのでこの領域のバックログ処理の経験なし

非安全な意思決定やコントロールアクション:
- バックログ処理着手を不適切に承認
- バックログ修正部をテストに含めないことを不適切に承認

プロセスモデルの欠陥:
- 巻き取り文化を背景に，年長者という理由で意見を尊重
- バックログ処理に関する情報共有必要性の認識欠如

Step5 高位レベルの制御を分析

元請けチーム

- 定義されたプロセスの範囲では配慮

安全に関する責務:
- 対象範囲関連も含め仕様を指示
- 欠陥の混入回避に努める
- 混入欠陥の発見に努める

コンテクスト:
- 特別の配慮(委託先サブリーダは新人)．詳細調査(本来委託先のタスク)まで実施し「これ（だけ）を…」と依頼

非安全な意思決定やコントロールアクション:
- バックログのようなチーム固有の課題関連の制御の不足・欠落
- 「これ（だけ）をやって」に対するチェックが欠落したプロセス

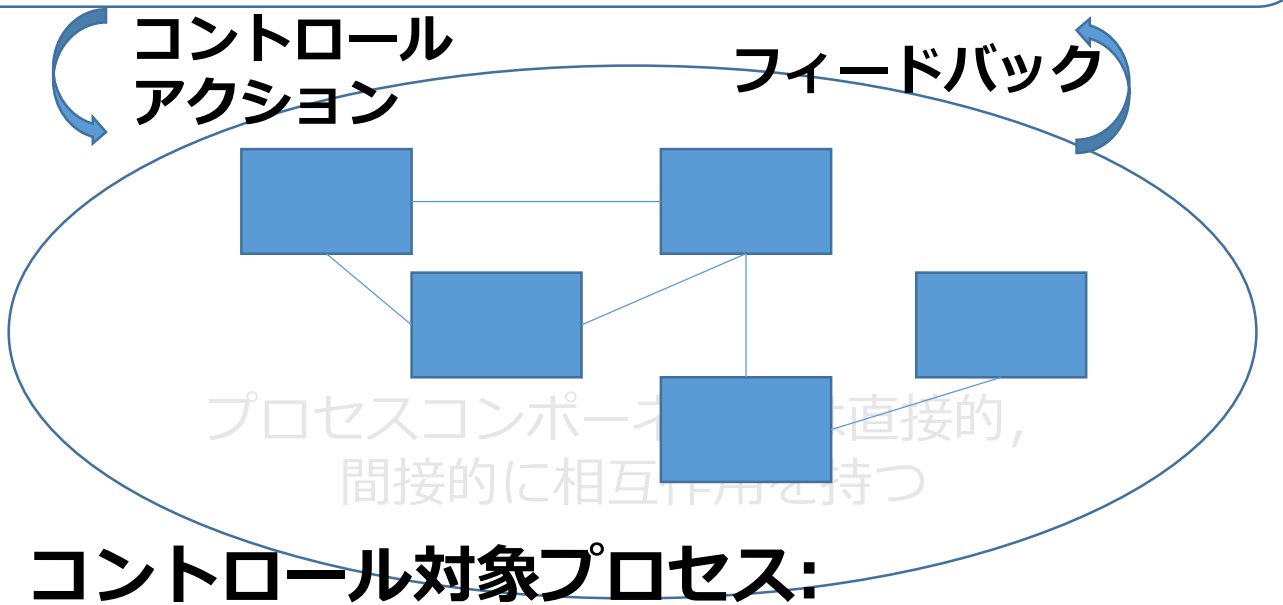
プロセスモデルの欠陥:
- 不明点は何でも質問するはず⇒知らない・分かってないことを本人が自ら把握できるという前提
- 「これ(だけ)やって」の不成立を想定せず(担当が新人でも)

アジェンダ

- はじめに
- プロジェクト情報
- CAST
- プロセスモデルと相互作用
- おわりに

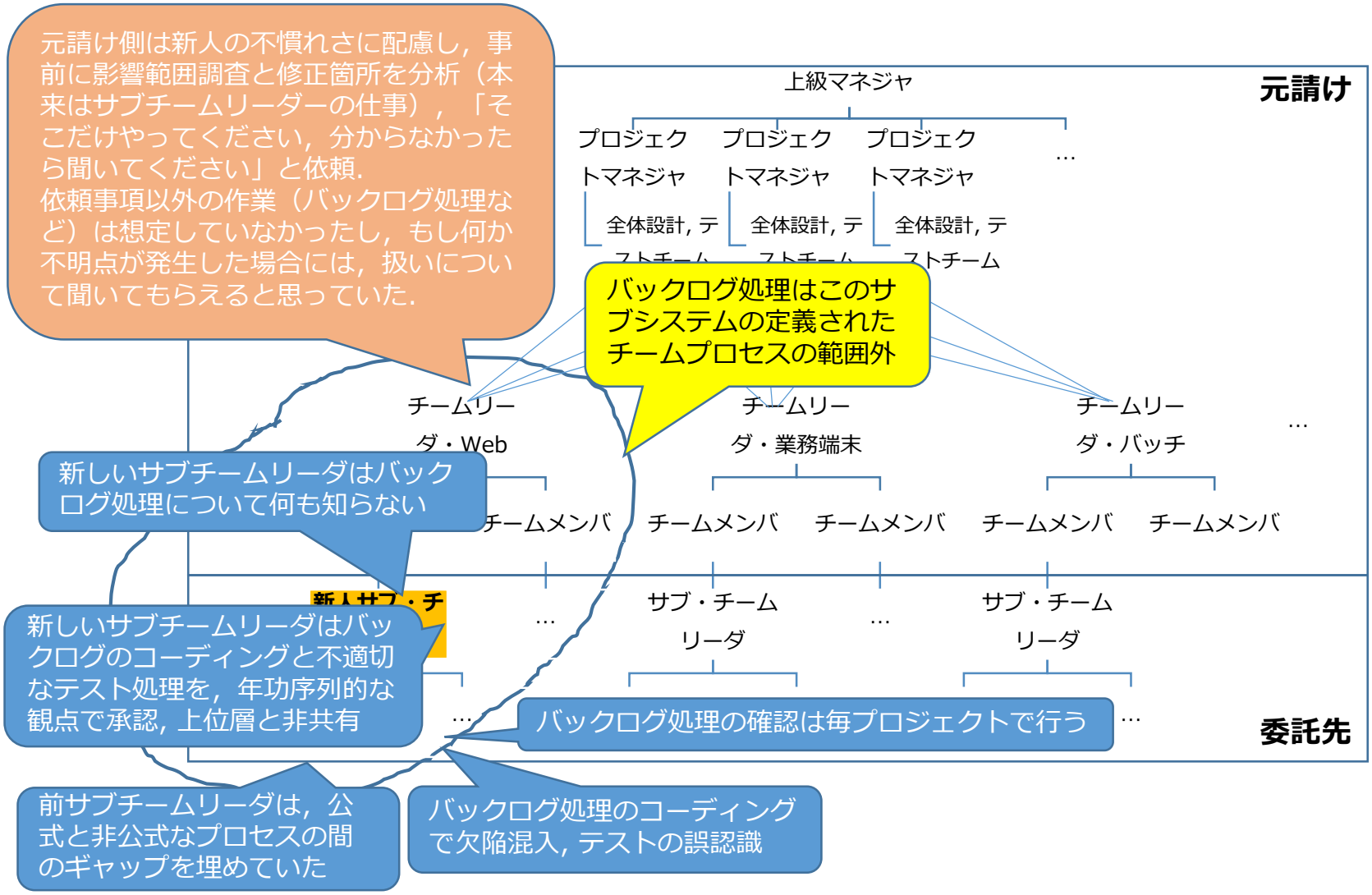
相互作用とプロセスモデル

コントローラ: マネージャ・リーダー
(品質，生産性，納期，**心理的な安全性**，育成…)
- 各プロジェクト/チーム/メンバの振る舞い
- プロジェクト/チーム/メンバ間の相互作用



コントロール対象プロセス:
プロジェクトとチームの系(システム)

Step6 損失につながる調整,やりとり



Step7 経時的な変化

元請けチーム

- 定義されたプロセスの範囲では配慮

安全に関する責務:
- 対象範囲関連も含め仕様を指示
- 欠陥の混入回避に努める
- 混入欠陥の発見に努める

コンテクスト:
- 特別の配慮(委託先サブリーダは新人)．詳細調査(本来委託先のタスク)まで実施し「これ（だけ）を…」と依頼⇒後述の感度低下と結合

非安全な意思決定やコントロールアクション:
- 巻取の下,チーム固有の課題(例:バックログ処理)の暗黙知化と感度低下
- 前任者の暗黙知の属人的能力によりプロセス上のギャップ顕在化せず
- 「これ（だけ）をやって」の成立チェックが欠落したプロセス

プロセスモデルの欠陥:
- 不明点は何でも質問するはず⇒知らない・分かってないことを本人が自ら把握できるという前提
- 「これ(だけ)やって」の不成立を想定せず(巻取あり,担当が新人でも)

Step8 勧告

当初のレポート

- 受託先
  - 開発者：対象範囲外のコード修正, 欠陥混入, テスト漏れ
  - サブリーダ：プロセス逸脱防止できず,テスト漏れを許す

  個人非難的，心的安全の喪失
- 元請け側
  - 特になし

レポート

- **受託先**

  開発者
  - **コーディング,テスティングスキル改善**

  サブリーダ
  - **不明確な事項は上位層に確認**
- **元請け側**
  - **スキル不足も想定した確認プロセスの追加**
    - ✓ **多能工育成時**
  - **安全制約を保つ制御ループの経時劣化**
    - ✓ **巻取下で進むチーム内暗黙知(例えばバックログ)への対処**

アジェンダ

- はじめに
- プロジェクト情報
- CAST
- プロセスモデルと相互作用
- おわりに

おわりに

- CASTにより望まない事象の背景について，その要素と相互関係を可視化することが出来た
- 効率性を追求する過程で，組織の暗黙知が大きくなり，新規参入者のハードルや，ハザードの危険性を上げていることが分かった.
- これらにより
  - 状況の共有が可能となった
  - 組織の心理的安全性が保たれた
  - 建設的な再発防止策の検討につながった

- ただし誰もができるとは…