DENSO
Crafting the Core

# Agile開発フレームワークを利用した効率的な開発ナレッジの獲得

## 自動車ソフトウェアにおけるプロダクトライン開発への適用事例

林　健吾*
㈱デンソー　先進安全技術4部
kengo_hayashi@denso.co.jp

ナレッジの獲得は
狙いを定めて
観察できるケースを構築して
小さく反復的に学習する

ナレッジ＝知識＋経験

≒形式知＋暗黙知

# Agenda

1.イントロダクション

2.開発ナレッジ獲得への課題

3.アプローチ

4.Agile開発フレームワークを利用した開発ナレッジの獲得法

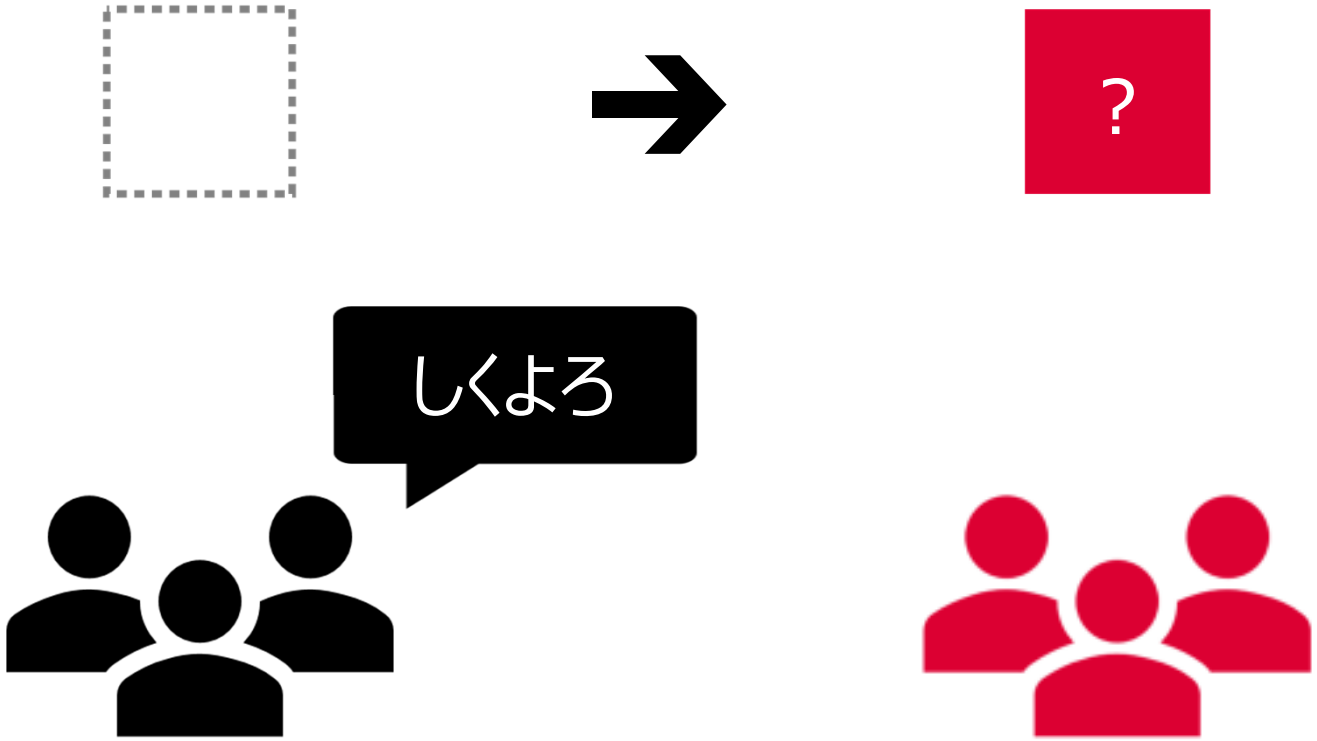
5.プロダクトライン開発への適用（ケーススタディ）

6.考察

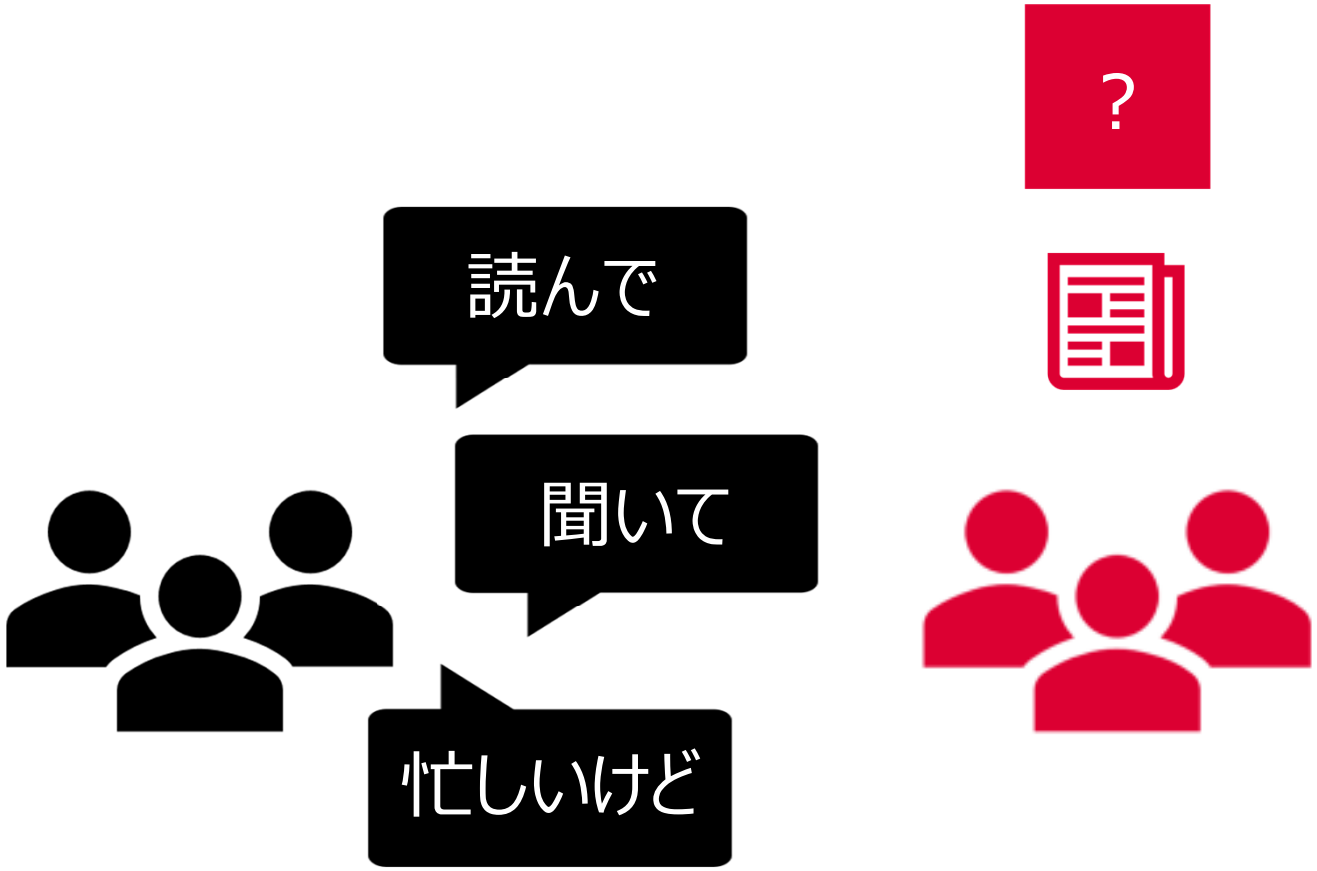
7.まとめ

Appendix. 参考文献

# １．ナレッジのない開発ドメインの開発の移譲



# １．ナレッジ獲得戦略のない場当たり的な進行



# １．当然の帰結としてのプロジェクト失敗

分析不足による修正モレ

- 機能仕様・設計上でしか相関がない修正点の存在
- 実は品質保証されていない要素の存在

⇒ 有識者がいても聞こうという気づきが起きない

設計資料の解釈誤りによる修正ミス

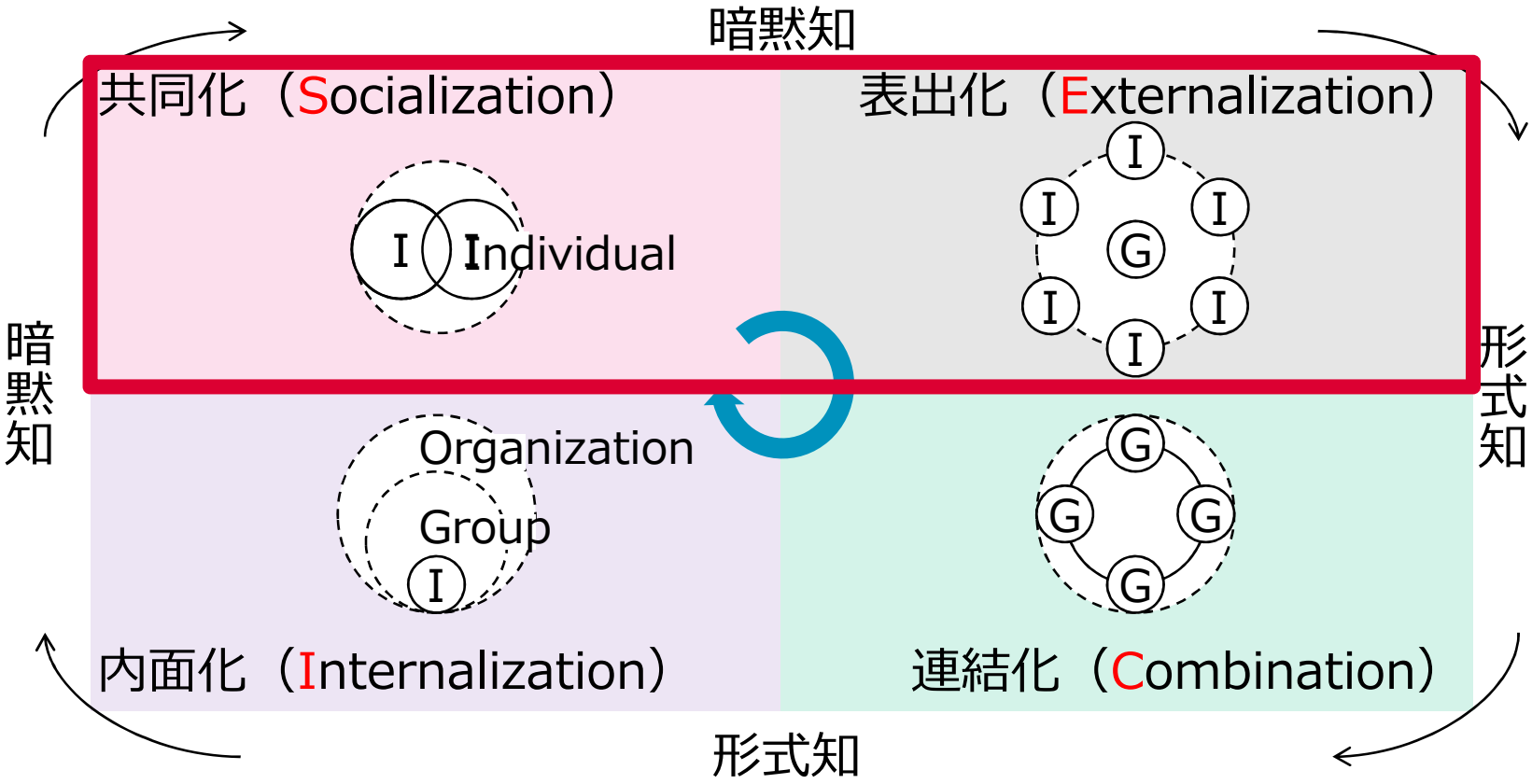
- 見えている範囲での判断・思い込みによる失敗
- 設計資料がそもそも間違っている

⇒ 間違っていると思わなければ聞こうと思わない

**分析スキルの不足？**

# 2．ナレッジの観点で起きていること

## SECIモデル




**共同化の促進阻害と表出化の不足・エラー混入**

# ２．開発ナレッジ獲得の課題

１．共同化が断絶されて暗黙知が継承されない
　　効率のよい共同化は実現できるか？

２．表出化された形式知が誤っている
　　誤りを訂正する手段はあるか？

３．時間制約の中でやり切れない
　　ナレッジを獲得する適切な戦略はあるか？

**ナレッジを獲得する課題は何かに似ていないか？**

# ３．開発ナレッジ獲得の課題と要求獲得の課題

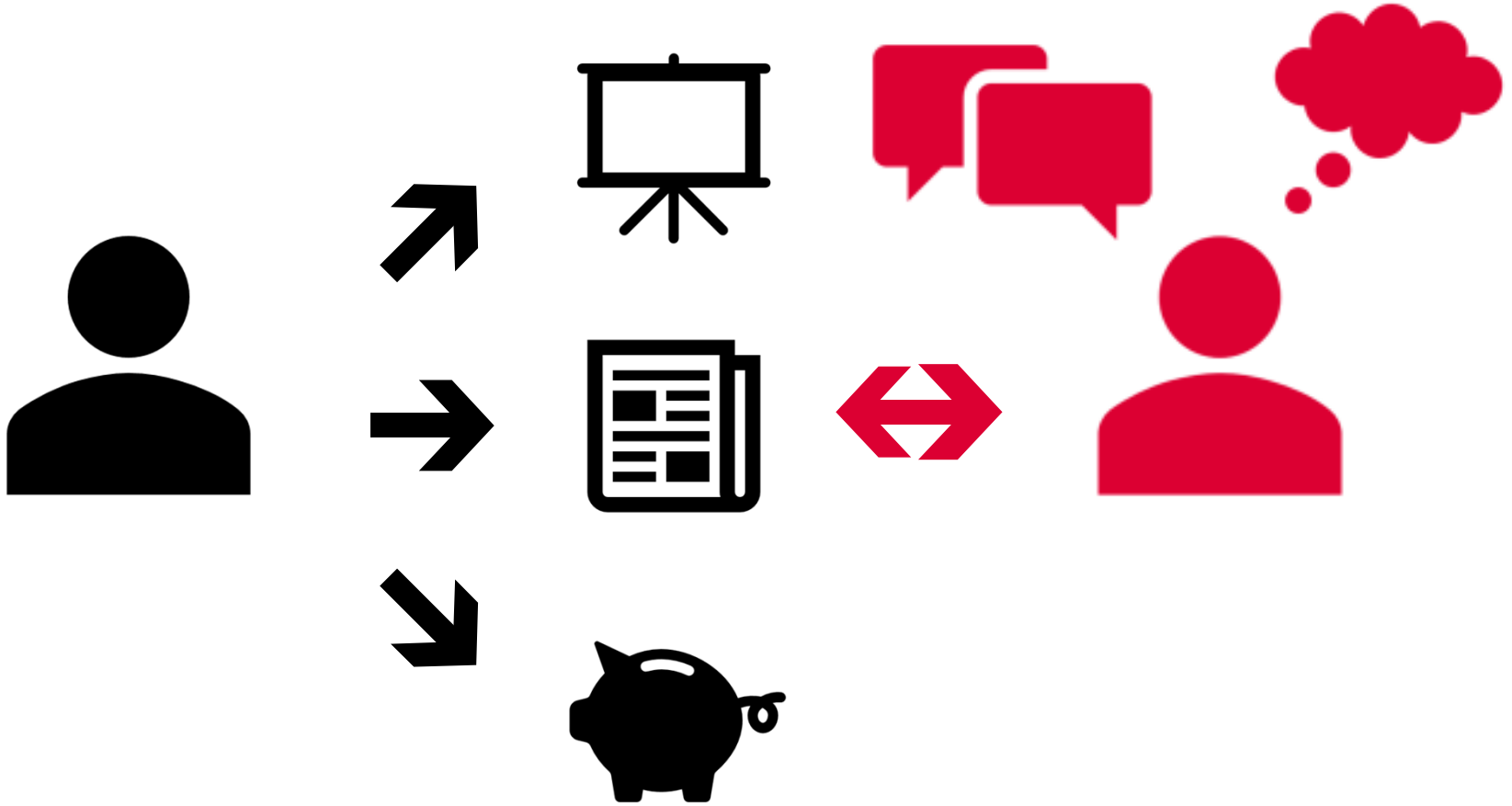
１．共同化が断絶されて暗黙知が継承されない
　　⇒要求者のニーズをうまく引き出せない

２．表出化された形式知が誤っている
　　⇒要求者の真のニーズが捉えられない

３．時間制約の中でやり切れない
　　⇒要求が明確になるとさらに深い要求が出てくる

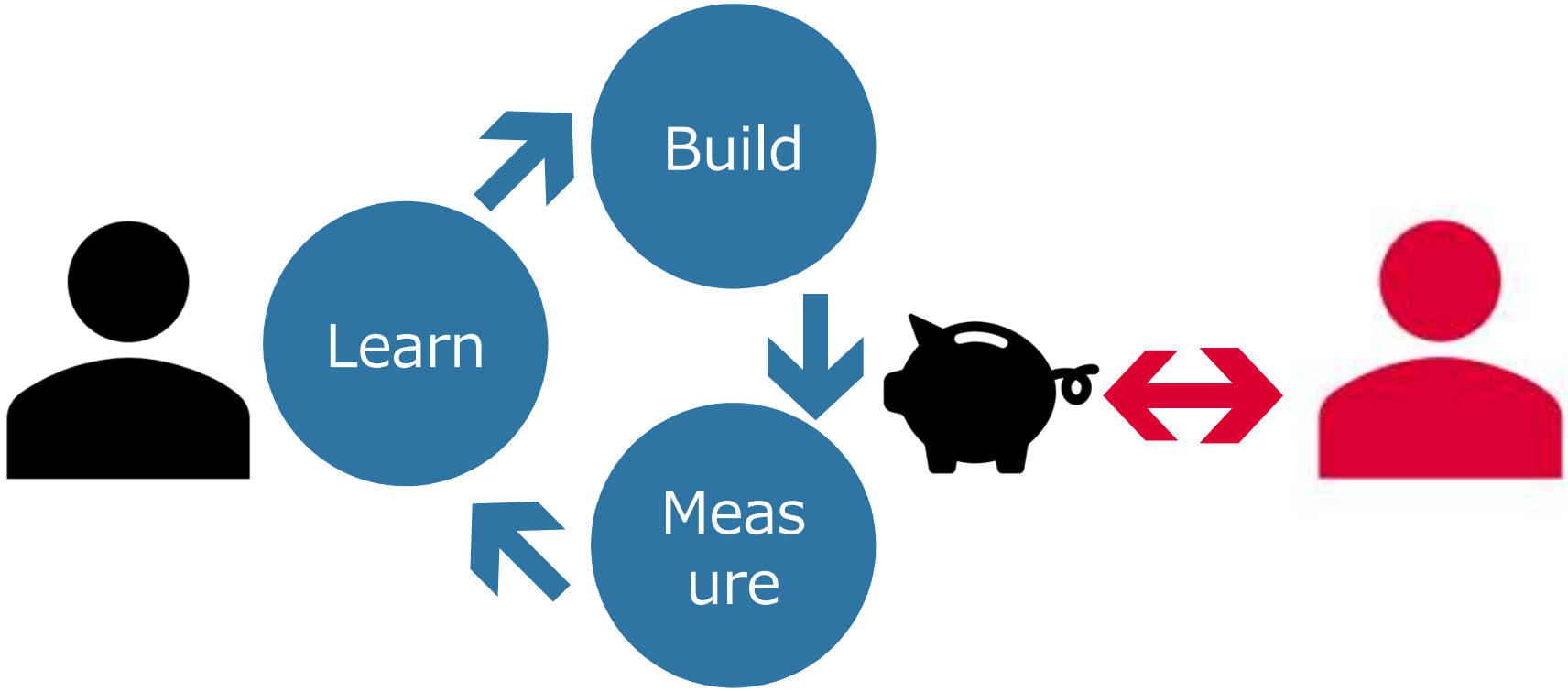
**要求獲得しながらソフトウェア開発する方法の応用**

# 3．プロトタイピング



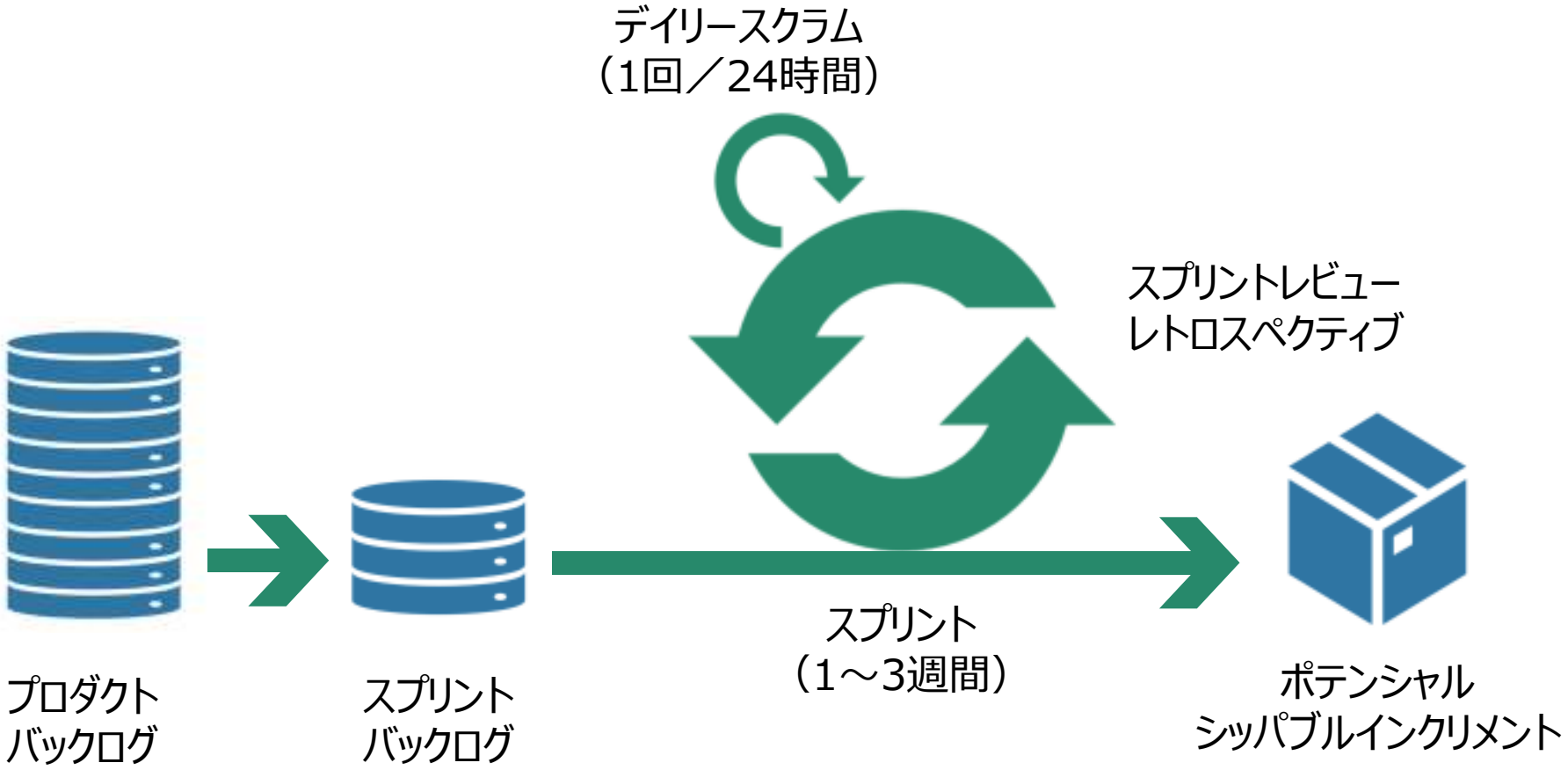
**手に触れるモノを元に顧客と対話的に要求を獲得する**

# 3．BMLループ




**簡単な実物から世の中の反応をみて膨らませていく**

# 3．SCRUM




**透明性・検査・適応を重ねてインクリメンタルに開発する**

# ４．開発ナレッジ獲得のポイント

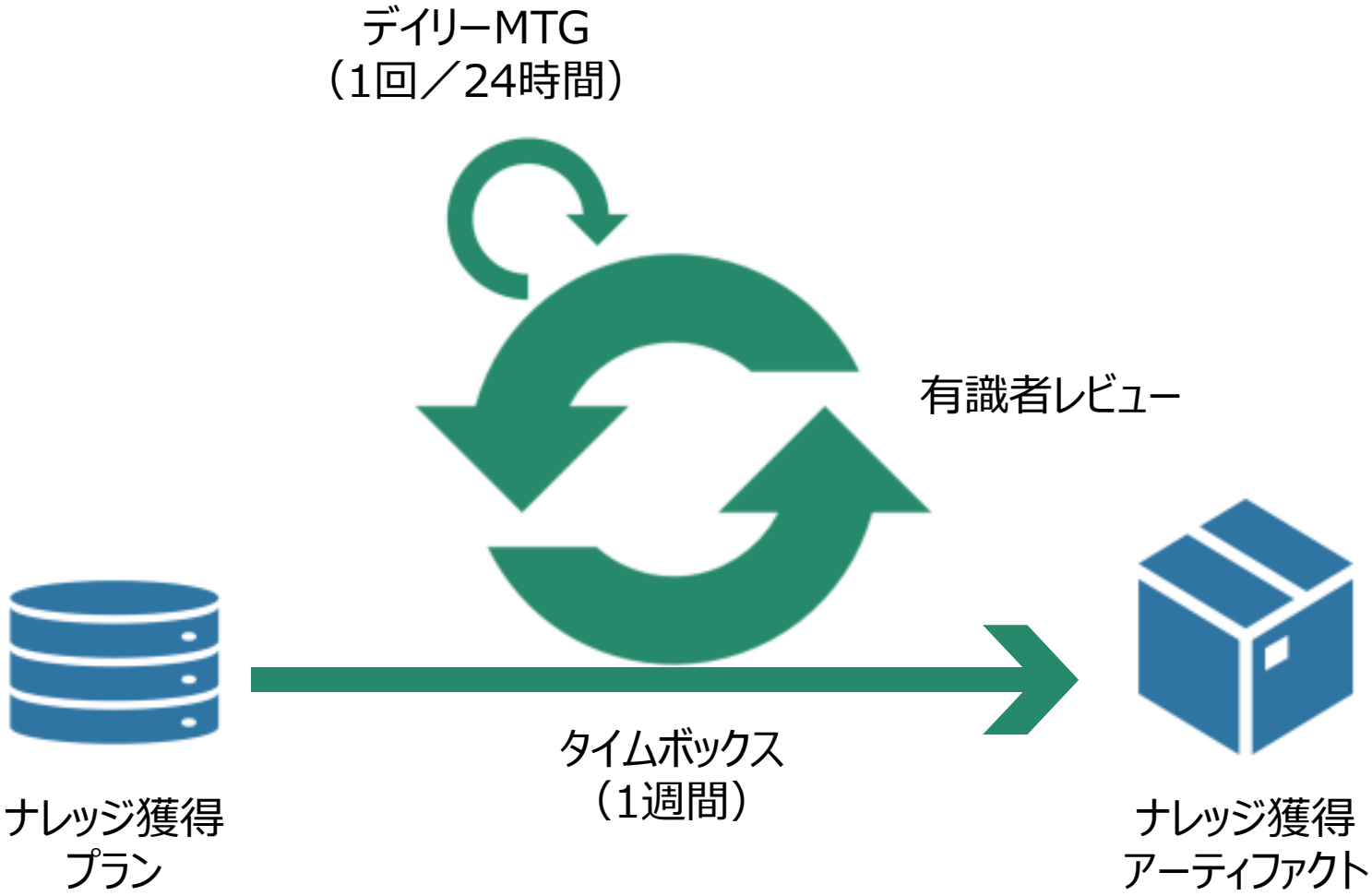
１．何か手に取れる、目に見えるものを構築する
⇒机上の検討だけでは確かなナレッジは得られない

２．得るべきナレッジのスコープは計画的に
⇒行き当たりばったりでは非効率なナレッジ獲得

３．小さく反復的にナレッジを獲得する
⇒探索するスコープと方向性を常に調整する

４．有識者とのコミュニケーションは定期的に配置する
⇒思い込みの排除と暗黙知の掘り出し

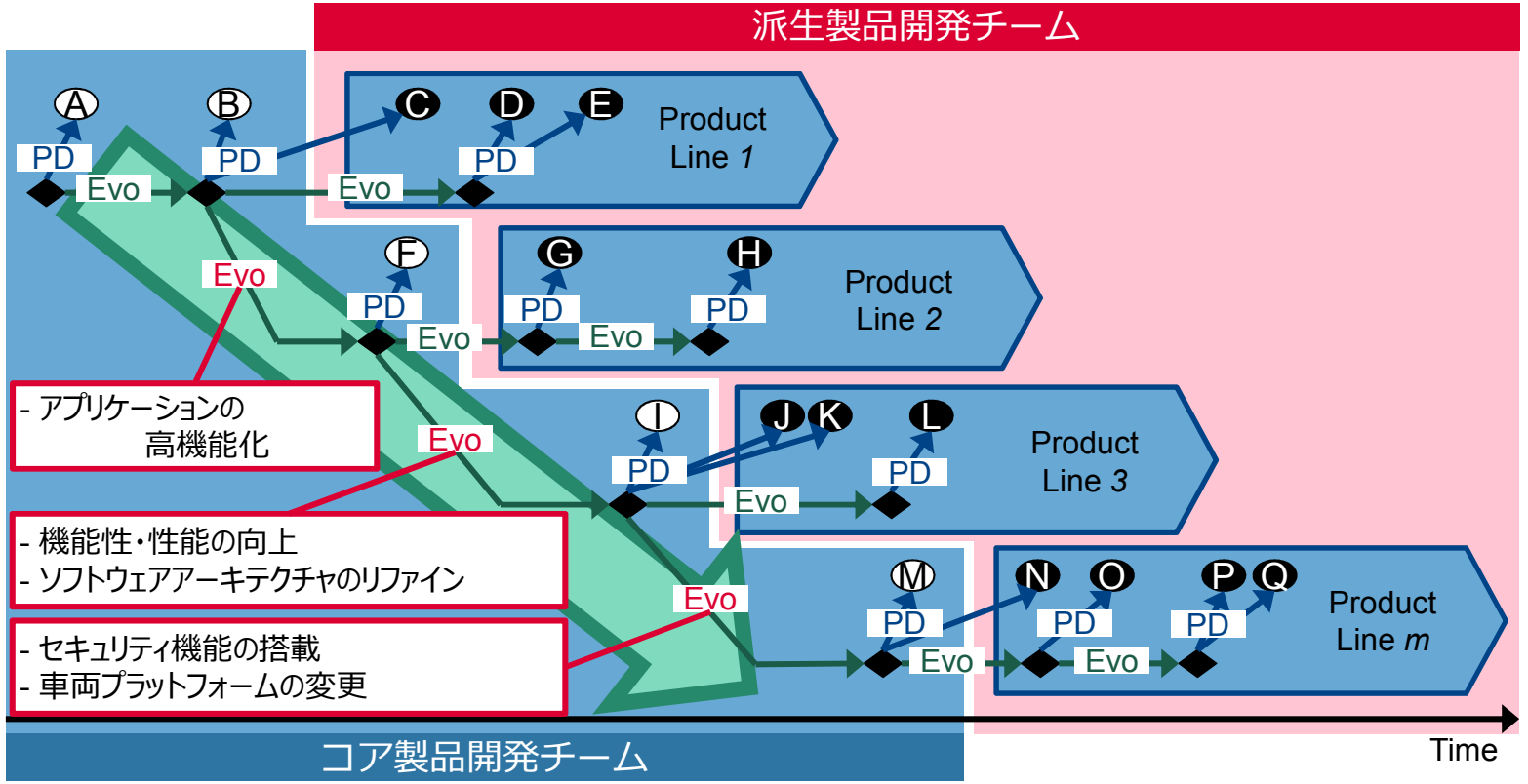
# ４．ナレッジ獲得Agileフレームワーク




**実物を構築しながらインクリメンタルに獲得する**

# 5．プロダクトライン開発事例の開発環境

## コア製品開発チームと派生製品開発チームで開発分担




凡例　PD : Product Derivation　Evo : Product Line Evolution
◆ : コア資産のバージョン　○ : コア製品　● : 派生製品

**派生チームでは頻繁にナレッジを獲得する必要がある**

# Case 1.
# コア開発で未保証機能のナレッジ獲得

# 5．コア開発で未保証機能のナレッジ獲得

| | |
|---|---|
| シーン | コア資産に組み込まれているものの、製品としてリリースされたことのない機能を検証してリリースする。 |
| 適用期間 | 4週間 |
| 獲得ナレッジ | 該当機能の振る舞い<br>改造方法 |
| アーティファクト | 機能検査仕様書<br>実環境での検査結果 |
| レビュー頻度 | 1回／2日 |

# 5．Case 1での活動履歴

| | |
|---|---|
| Time-Box 1 | 検査仕様書作成で生じる疑問を解消し、機能の振る舞いの理解が促進。 |
| Time-Box 2 | 検査実施で８件の欠陥を抽出。<br>2件：実欠陥⇒処置方針を協議<br>6件：振る舞いの理解誤り |
| Time-Box 3 | 欠陥修正の効果確認OK。<br>連動して影響を受ける仕様を抽出。<br>仕様の処置方針を協議の上、決定。 |
| Time-Box 4 | 検査全件完了。<br>仕様修正方法も顧客と調整完了。 |

# 5．Case 1での成果

- 有識者からのナレッジの引き出し

  質問されなければ思い当たらなかったナレッジあり

  仕様書で誤解を与える誤記の発見

- 実施しなかった場合に支払ったコスト予測

  開発終盤に仕様の不整合が発覚し、

  緊急対応でコストと開発遅延のリスク発症の見込み

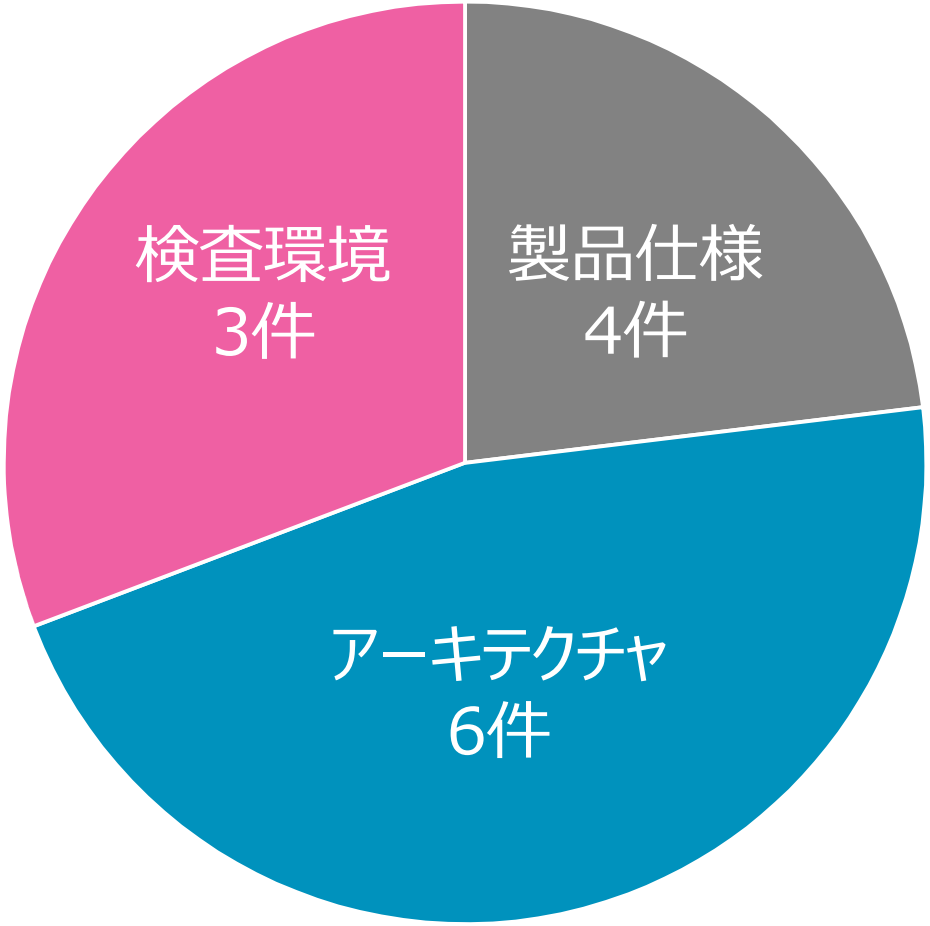
**両チームでナレッジを共有し合うことに成功**

# Case 2.
# コア資産のアーキテクチャのナレッジ獲得

# 5．コア資産のアーキテクチャのナレッジ獲得

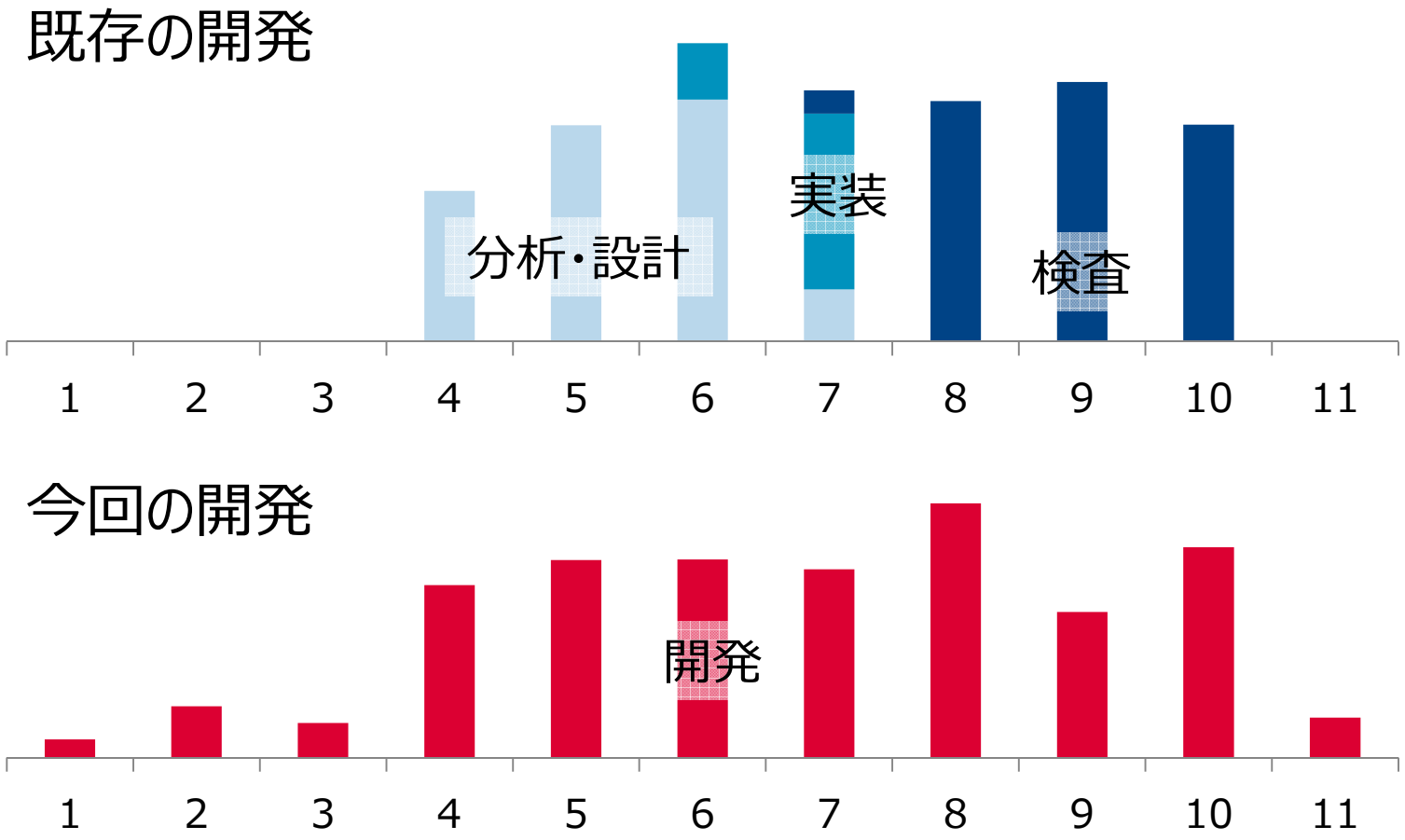
| | |
|---|---|
| シーン | 新たに進化したコア資産から最初の派生製品を開発する。 |
| 適用期間 | 11週間 |
| 獲得ナレッジ | アーキテクチャ<br>製品仕様／検査環境使用方法 |
| アーティファクト | 実ソフトウェア開発 |
| レビュー頻度 | 都度　※各分野で分けて計画 |

# 5．Case 2での獲得ナレッジ数




**どのナレッジも開発終盤で判明すると致命的な内容**

# 5．Case 2での開発工数の比較




**開発総工数と期間は同レベルで開発完了できた**

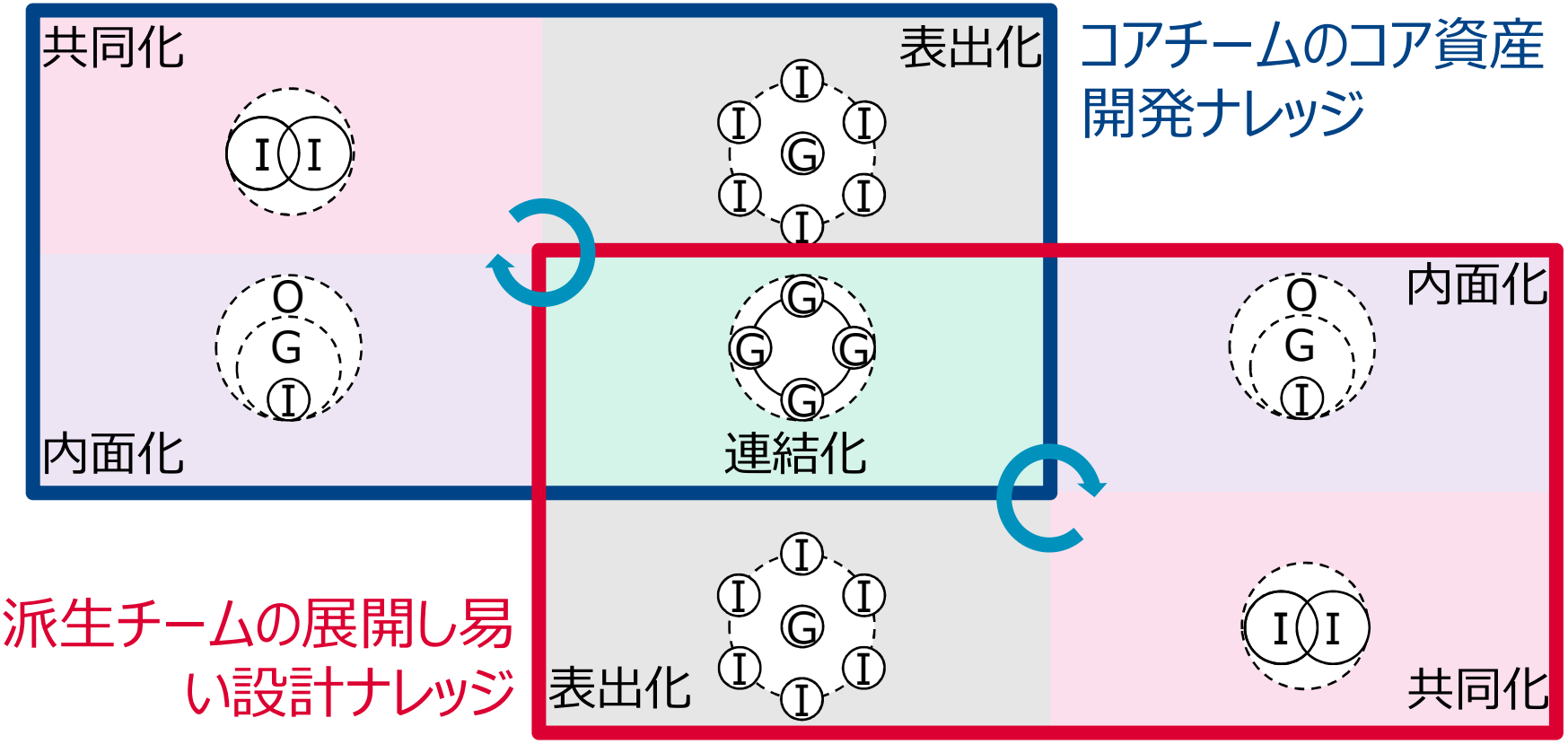
# Case 3.
# コア資産リファクタリングのナレッジ交流
# （応用編）

# 5．コア資産リファクタリングのナレッジ交流

| | |
|---|---|
| シーン | コア資産の派生製品を開発し易くするためのリファクタリングを派生チームで実行する。 |
| 適用期間 | 5週間 |
| 獲得ナレッジ | コンポーネントのアーキテクチャ<br>展開しやすい構造設計 |
| アーティファクト | コンポーネントソフトウェア<br>リファクタリング後の設計モデル |
| レビュー頻度 | 1回／1週間 with コアチーム<br>1回／２週間 with コアマネージャ |

# 5．Case 3で生じた化学反応

リファクタリングを通じて相互のナレッジが交流した




**各組織のナレッジが連結化する場が生まれた**

# 6．考察：課題は解決できたか？

- 効率のよい共同化は実現できたか？
  ⇒ リズミカルなレビューの設定と、
  　実物を元にした議論で効率よく共同化できた

- 表出化された形式知の誤りは訂正できたか？
  ⇒ レビューの場に有識者を招くことで訂正できた
  　但し、充分とは言い切れない

- 時間制約の中でナレッジ獲得戦略は立てられたか？
  ⇒ 見える化を主動とした獲得戦略は立案できた
  　但し、探索的であるためのリスクは残る

# 6．考察：問題は解消できたか？

- 分析不足による修正モレ

  ⇒ 有識者との場が定期的に設けられることで、
  質問不足による修正モレは低減できた
  但し、限られた時間で見過ごすリスクは残る

- 設計資料の解釈誤りによる修正ミス

  ⇒ 実物を使いながらレビューすることで、
  設計資料の解釈誤りのリスクは大きく低減できた

# ７．まとめ

Agile開発フレームワークを利用して、
効率的な開発ナレッジの獲得を目指した

いくつかのケースでナレッジを効率的に獲得し、
ナレッジ不足による後戻りリスクを軽減することに成功した

表出化と共同化を設計することで、
組織間で連結化を促進する場が生じることが経験できた

**Agileのフレームワークはナレッジ獲得にも有効**

ナレッジの獲得は
狙いを定めて
観察できるケースを構築して
小さく反復的に学習する

DENSO

Crafting the Core

# Appendix. 参考文献


[Boeh88] B. W. Boehm, A spiral model of software development and enhancement, Computer, Vol. 21, No. 5, 1988, pp. 61-72.

[Coh05] M. Cohn, Agile Estimating and Planning, Prentice Hall, 2005.

[Nona96] 野中郁次郎, 知識創造企業, 東洋経済新報社, 1996.

[Rie12] エリック・リース, リーン・スタートアップ, 日経BP社, 2012.