一般財団法人　日本科学技術連盟
第35年度ソフトウェア品質管理研究会　成果発表会

対象システムの知識や経験が浅くても有効な指摘を行えるレビュー手法（TPDR法）の提案

-熟練レビューアの勘所「トリガーポイント」から指摘観点を導出-

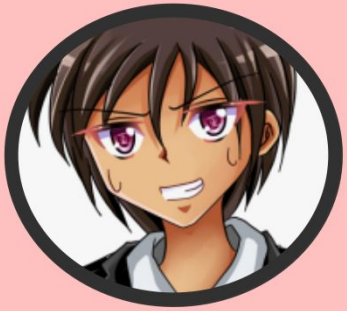
2020年2月21日

研究コース2　ディベロッパーチーム

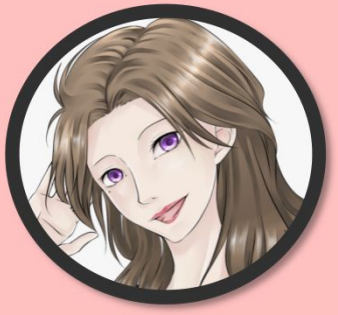
研究員：	○篠瀬　裕太郎	(株式会社リンクレア)
	久保田　卓	(株式会社東光高岳)
	堀内　晨彦	(NTTコミュニケーションズ株式会社)
主査：	中谷　一樹	(TIS株式会社)
副主査：	上田　裕之	(株式会社DTSインサイト)
アドバイザー：	安達　賢二	(株式会社HBA)

解決したい課題

対象システムの**知識や経験が浅くても**
有効な指摘を行えるようにしたい

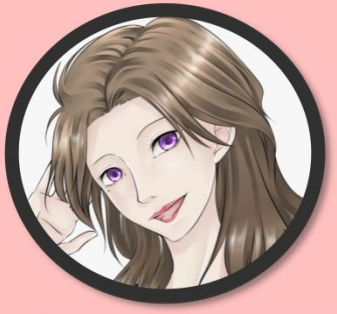


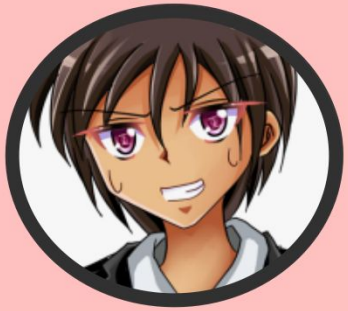
早く成長して
実践で活躍して
欲しいの



現状調査

熟練度高





熟練度低

1
1
3
1
2

（計8件）

3

⑥質問や確認
⑤誤字脱字等の軽微指摘
④対応不要となった改善提案
③対応不要となった欠陥指摘
②重要な改善提案
①重大欠陥の指摘

有効な指摘＝①②

0

5
3
4

（計12件）

研究員の組織における

現状の取り組みと課題

取り組み	課題
設計ガイドに基づく**チェックリスト作成**	・チェック項目が多く**時間がかかる** ・**判断が難しい**チェック項目もある
不具合管理表の作成と**事例共有**	・**更新の継続**ができない ・**記載粒度**が悩ましい

結局、レビューアの熟練度に依存

提案手法

対象システムの知識や経験が浅くても
有効な指摘を行えるレビュー手法

TPDR法（ティーピーディーアール）

(**T**rigger **P**oint **D**riven **R**eview Method)
トリガーポイントレビュー手法

手法のコンセプト

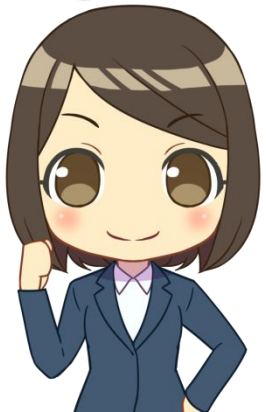
① 熟練レビューアの**暗黙知を形式知化**

② **特定のシステムに特化**した勘所集作成

③ 類似システムから**横展開しない**

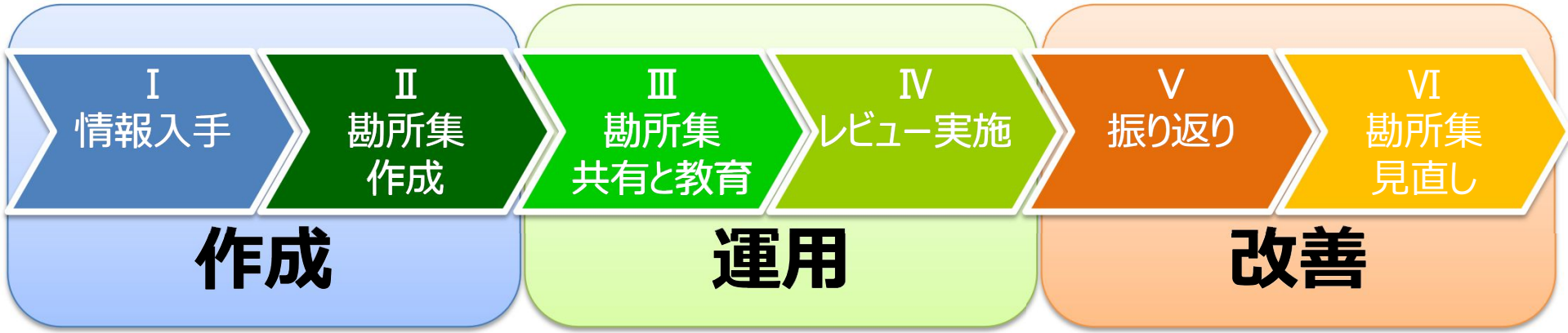
④ **作成-運用-改善**の３フェーズ

⑤ 勘所集を見ずに指摘できるまで**繰り返し教育**

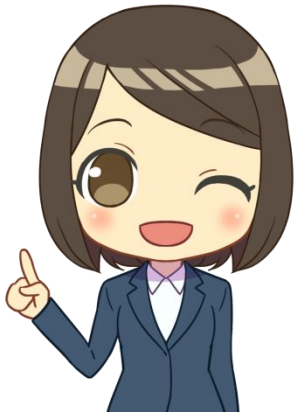


熟練レビューアのやり方を理解し、
無駄なく効率的に、
必ず結果を出せる！

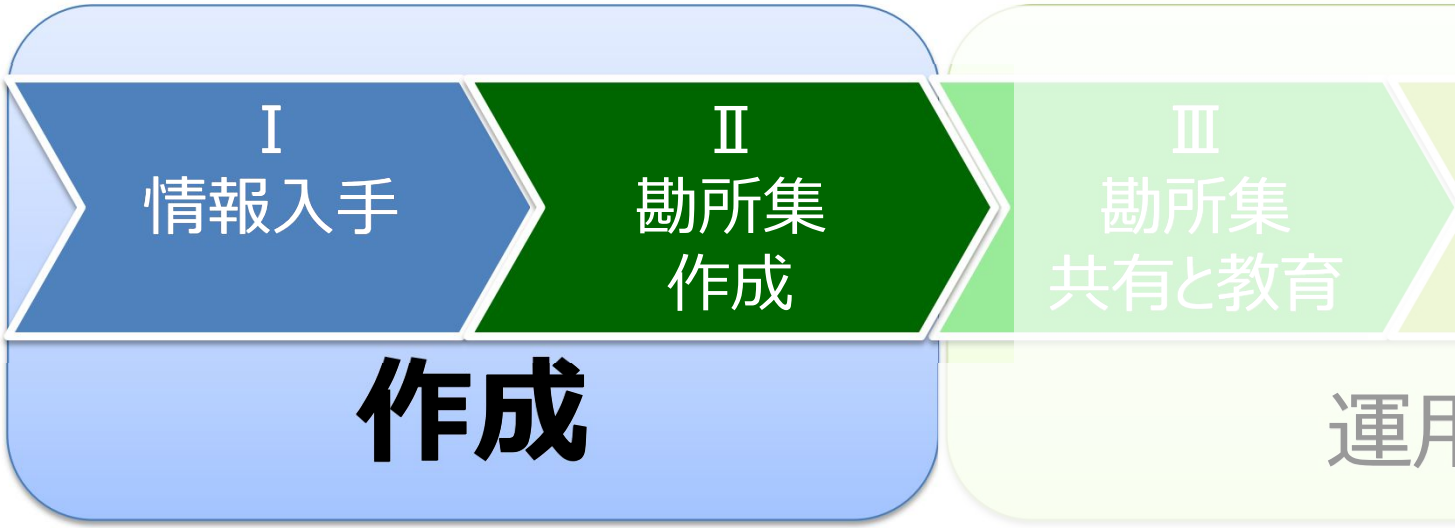
提案手法の実施手順



３つのフェーズ



①作成フェーズ

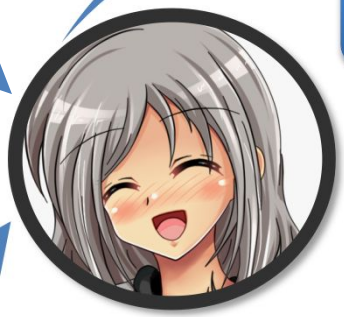


作成フェーズ

過去の不具合情報

過去のレビュー指摘情報

熟練者へのヒアリング



勘所集

過去の情報収集
熟練レビューアへの
ヒアリングを実施

勘所集に記載➡
別の熟練レビューアへヒアリング
情報を追加

「過去の不具合情報」を基に情報入手する流れ

【過去の不具合情報】

Input

共有メモリ
タスク間
排他制御不備
(動作異常)

どんな指摘？

【指摘観点】

共有メモリ
アクセス時
排他制御
考慮している？

その指摘はどこが
気になったから？

他に考えられる
不具合は？

他に考えられる
指摘は？

【トリガーポイント】

メモリ

トリガーポイントとは

解説

レビュー指摘観点を想起させる
「気になる言葉」や「気になる点」

【トリガーポイント】

メモリ

熟練レビューアは
トリガーポイントを
最初に見つける

【指摘観点】

共有メモリにアクセス時、
排他制御を考慮している？

そのトリガーポイントを見つけたら
この観点で見る

関連

【不具合事例】

共有メモリのタスク間で排他制御
の不備(動作異常)

なぜなら、過去にこんな不具合が

情報入手～勘所集作成の例①

＜過去の不具合情報＞

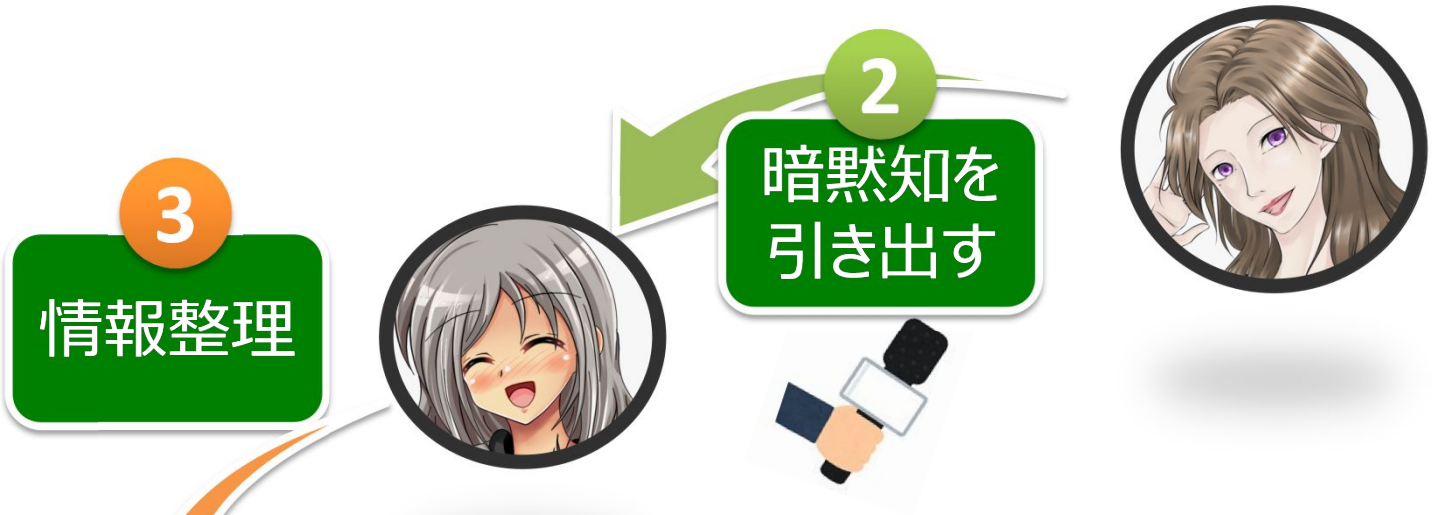
**共有メモリのタスク間で
排他制御不備(動作異常)**

1 情報入手

勘所集

トリガーポイント	サブワード	過去の不具合事例	参照先	影響	指摘観点
不具合を想起させる気になる点	不具合の絞込を行うワード・条件	トリガーポイントに紐付く	発生年度、製品、管理番号	不具合発生工程、改修費用	不具合防止の指摘観点
		★ 共有メモリのタスク間で排他制御不備(動作異常)	★ 2020.製品A.0001	★ 結合試験.10人日	

情報入手～勘所集作成の例②



勘所集

トリガーポイント 不具合を想起させる気になる点	サブワード 不具合の絞込を行うワード・条件	過去の不具合事例 トリガーポイントに紐付く	参照先 発生年度、製品、管理番号	影響 不具合発生工程、改修費用	指摘観点 不具合防止の指摘観点
メモリ	共有メモリ	共有メモリのタスク間で排他制御不備(動作異常)	2020.製品A.0001	結合試験.10人日	共有メモリにアクセス時排他制御を考慮しているか

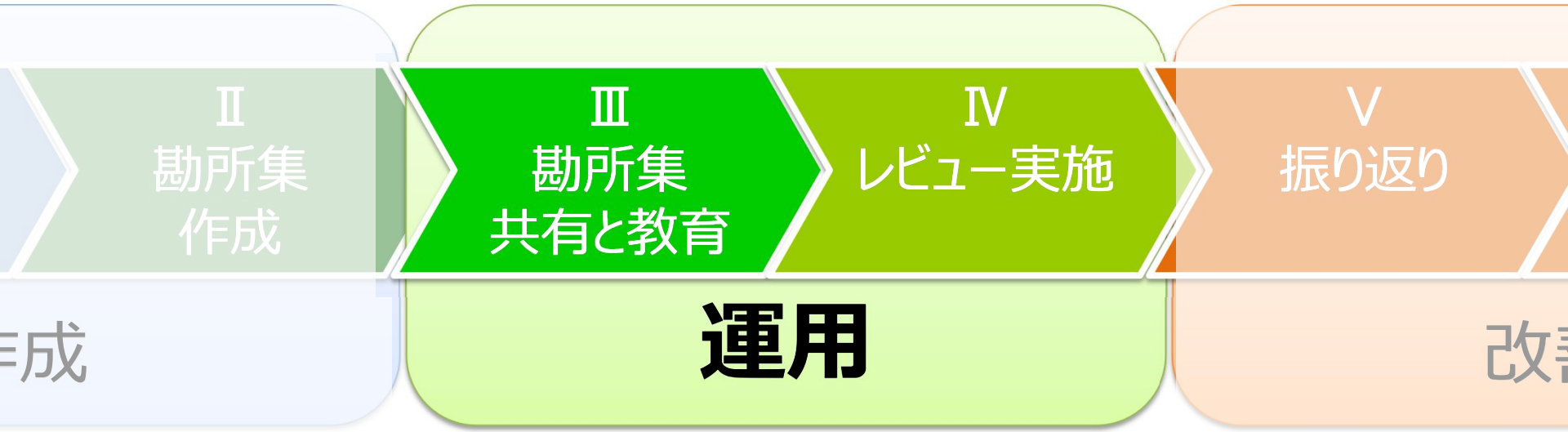
情報入手～勘所集作成の例③

参考

組込系ソフトウェア

トリガーポイント	サブワード	過去の不具合事例	参照先（年度，製品，不具合管理No）	影響（発覚工程，改修工数）	指摘観点（レビューポイント）
メモリ	共有メモリ	・共有メモリを使用する2つのタスクの間で，メモリアクセスの排他制御がうまくできておらず，動作異常となった	2020，製品A-v1.3，A2020-AB0008	結合試験，10人H	・共有メモリにアクセスする時の排他制御を考慮しているか
	外部メモリ（EEPROM）	・I2Cアクセス中にリセットが発生すると，以後EEPROMにアクセスできなくなった	2020，製品A-v1.2，A2020-AB0003	リリース後，50人H	・I2C通信フリーズを考慮してリセットシーケンス処理を入れているか
	→	・メモリのアクセスに時間が長く，連続処理失敗や異常監視リセットが発生した	2020，製品A-v1.0，A2020-AB0001	結合試験，20人H	・異常監視有の場合，アクセス時間が監視時間を超えないか ・監視時間の設計で，ハードの制約で最大値にできない場合、試験で担保が取れているか
:	:	:	:	:	:

②運用フェーズ



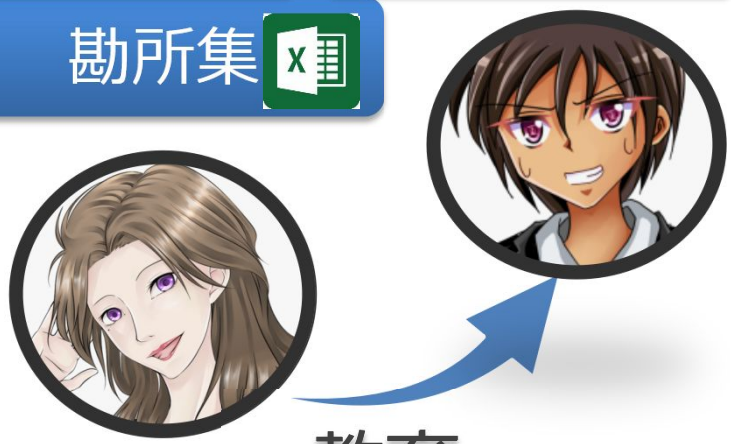
運用フェーズ

勘所集
共有

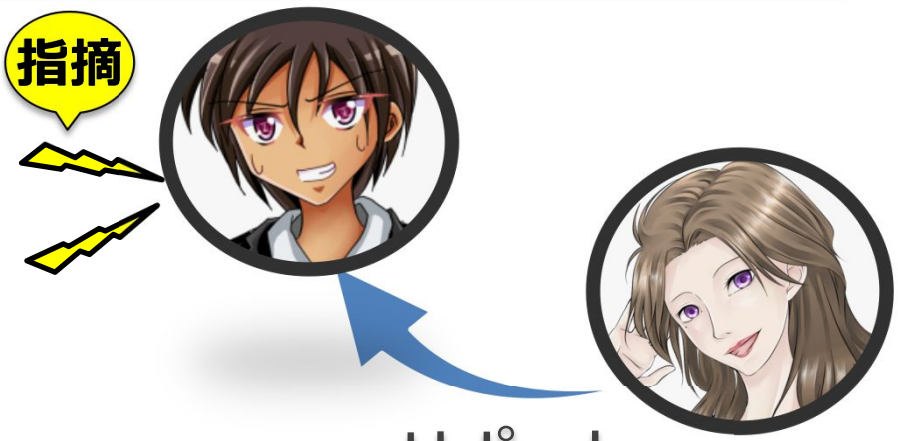
教育

レビュー実施

勘所集



教育

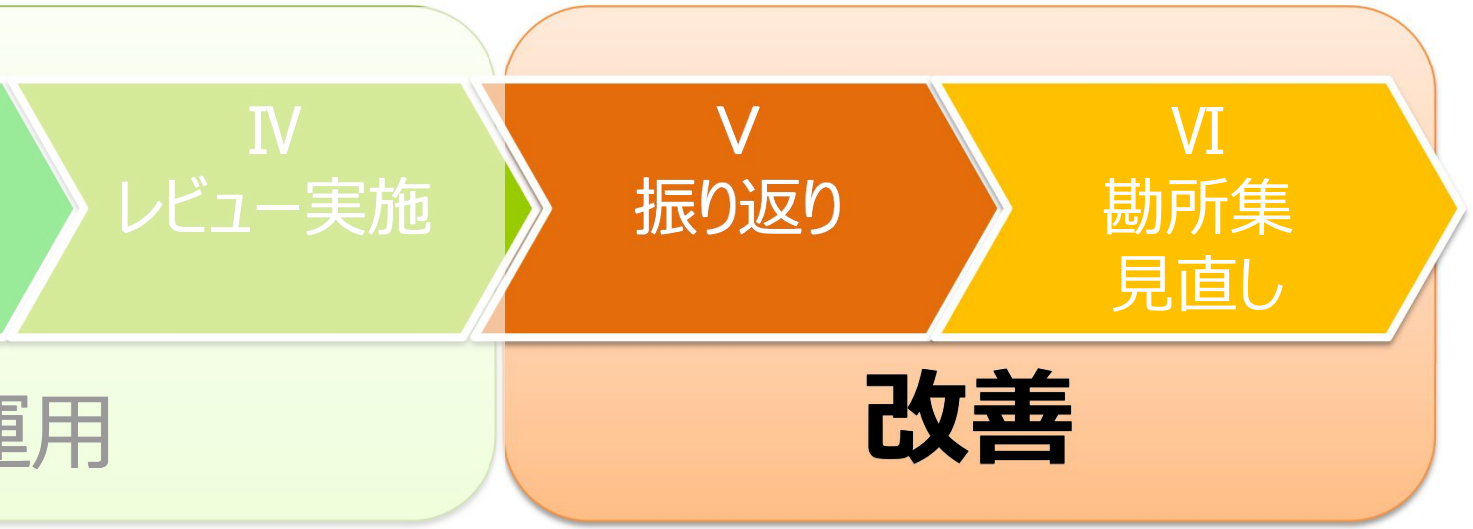


サポート

勘所集でレビューポイントを
学習（１個からでもOK）
模擬レビューを実施
実践で使えるように訓練

トリガーポイントを最初に
見つけて指摘につなげる
※ここで勘所集は見ない

③改善フェーズ

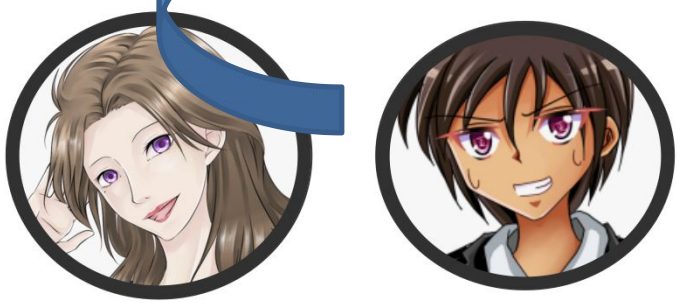


改善フェーズ

振り返り　　**勘所集見直し**

勘所集

勘所集



・レビュー指摘を振り返る
・出来た所は、ほめる
・出来なかった所は、熟練レビューアの指摘内容が勘所集のどの部分か確認

熟練レビューアと一緒に新しい指摘を勘所集に取り込む
※見直しを計画に含める
※勘所集の管理者を置く

実験

目的

TPDR法を適用したレビューを実施することで、
熟練度が低いレビューアが
有効な指摘をどれだけ検出できるかを検証する

手順

1. 被験者が機能変更に関する**設計レビュー**を実施する
   被験者：現場の開発メンバー10名(熟練度 高3名、低7名)
   対象物：難易度・品質が同レベルの実プロジェクトの設計書
2. **TPDR法を適用したレビュー**を
   実施した場合と、実施しなかった場合との、
   有効な指摘の件数を比較する

実験結果

グループ	手法の適用	レビュー対象	有効な指摘		それ以外
			①重大欠陥の指摘	②重要な改善提案	
熟練度 **低**	なし (1回目)	仕様書X	0	0	13
	あり (2回目)	仕様書Y	**3**	0	19
熟練度 **高**	– (1回目)	仕様書X	2	1	5
	– (2回目)	仕様書Y	2	1	6

+3

本手法の適用で、熟練度低の有効な指摘が
0件から3件になった

まとめ

解決したい課題

対象システムの**知識や経験が浅くても**
有効な指摘を行えるようにしたい

熟練度低

⑥質問や確認
⑤誤字脱字等の軽微指摘
④対応不要となった改善提案
③対応不要となった欠陥指摘
②重要な改善提案
①重大欠陥の指摘

課題あり

有効な指摘 0

TPDR法を考案

①暗黙知を形式知化
②特定のシステムに特化
③横展開しない
④作成-運用-改善
⑤繰り返し教育

必ず結果が出る

効果も高い

「**トリガーポイント**」を見つけて、
知識と照らし合わせることで
有効な指摘ができる！