JUAS 一般社団法人 日本情報システム・ユーザー協会
Japan Users Association of Information Systems

システム開発・保守QCDs研究会2025

一般社団法人　日本情報システム・ユーザー協会

2026年4月9日

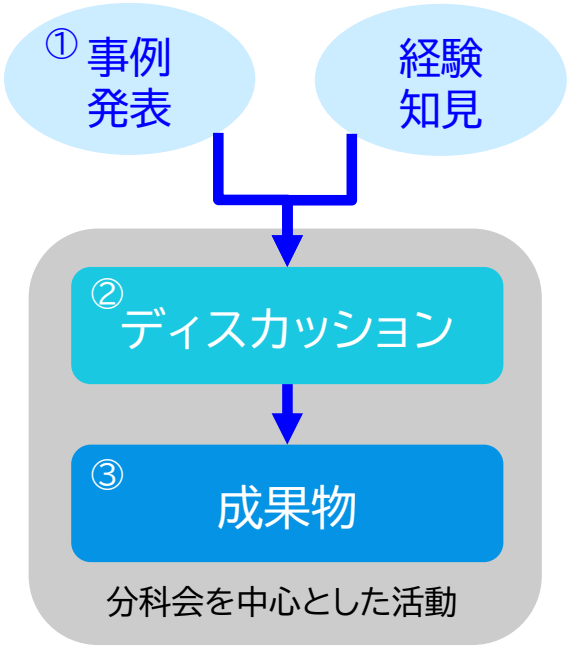
JUAS

目次

1．研究会の取り組み

1. 研究会の取り組み - 分科会活動 -

**分科会ごとに定めた年間の活動ゴールに向けて、
ディスカッションを中心とした課題の深堀りを行い、
活動の成果としてまとめ、発表を実施。**



①各社事例発表

年間の活動テーマに対して各社から事例、または課題を発表

②活動テーマ別ディスカッション

分科会にて年間ゴールに向けた課題解決のアイデア検討や
経験・知見を共有し、ディスカッションにて仮説と検証
※Give＆Takeが前提

③成果物の作成・発表

1. 研究会の取り組み - 活動サイクル -

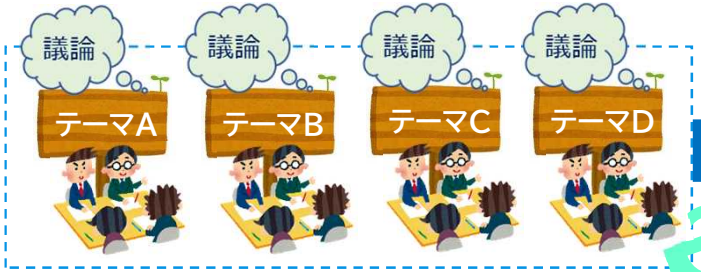
活動テーマ　テーマA　テーマB　テーマC　テーマD

※テーマの選定案として、事前にアンケート回答いただいた内容をもとに事務局で仮設定

各社の事例発表 ＋ テーマ別の議論 ⇒ 成果物

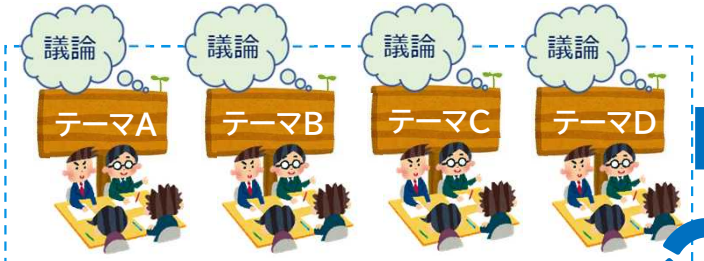
⓪活動テーマの議論へのinputとなるよう活動テーマに対する取組み事例を発表

①合宿までは同じメンバーで議論を行い、中間成果物を作成



合宿
★中間成果物の発表

②合宿後は、他メンバーも加え、議論の視点を広げつつ、深堀・議論を行い、最終成果物を作成



★年間成果物の発表

翌年のinputへ

1. 研究会の取り組み - 事例発表 -

年間を通して成果物を作成していくための支援や学びの時間として、
各参加者が、分科会の検討テーマに関連する事例を発表

(1) 研究会1日あたりの発表数

4社程度

(2) 発表・質疑時間

発表：約15分　質疑応答：約5分

※年間テーマに対するディスカッションに重きをおくため、事例発表時間は15分以内

(3) アンケート

参加者のコメントをアンケート形式で集計し、後日、発表者へフィードバック

1. 研究会の取り組み - QCD「S」の視点 -

従来の品質・コスト・納期の維持向上テーマに加えて
「S」を追加することで、視点を広げて考えてみることに取り組む。

QCD「s」の一例。下記に限らず様々な視点で議論

Scope(スコープ)、Slim(スリム)、Speed(スピード)、
Spirit/soul(精神・魂)、Sustainable(持続可能)、
Stable operation of mind(心の安定稼働) 等

※「s」は固定のものを指すのではなく、視点を広げるという意味合い

1. 研究会の取り組み - 年間計画 -

全体での事例発表と、各グループに分かれての分科会活動を実施。

6月度	7月度	8月度	9月度	合宿
進め方の説明	事例発表			
分科会活動	ディスカッション			
活動ゴール設定	中間成果物の作成			
★活動ゴール共有				★中間成果物の発表

11月度	12月度	1月度	2月度	3月度
事例発表				
ディスカッション				
最終成果物の作成				
				★最終成果物の発表

2. 研究会参加企業

2. 研究会参加企業

No.	役割	会社名	氏名
1	部会長	東京海上日動システムズ株式会社	東山
2	副部会長	株式会社ジャステック	鈴木
3	副部会長	ANAシステムズ株式会社	岩谷
4	副部会長	スミセイ情報システム株式会社	小沢
-	事務局	JUAS	森
-	事務局	JUAS	酒井

No.	会社名	No.	会社名
5	NRIシステムテクノ株式会社	15	東京ガスiネット株式会社
6	株式会社カジマアイシーティ	16	東京電力ホールディングス株式会社
7	コープ情報システム株式会社	17	東芝インフォメーションシステムズ株式会社
8	コニカミノルタ情報システム株式会社	18	独立行政法人住宅金融支援機構
9	JFEシステムズ株式会社	19	ニッセイ情報テクノロジー株式会社
10	JALデジタル株式会社	20	日本製鉄株式会社
11	ストラクチュアルライン株式会社	21	日本ハムシステムソリューションズ株式会社
12	SOMPOひまわり生命保険株式会社	22	東日本電信電話株式会社
13	TDCソフト株式会社	23	明治安田システム・テクノロジー株式会社
14	東京海上日動システムズ株式会社	24	株式会社大和総研インフォメーションシステムズ

～ 幹事団のありたい姿と行動指針 ～

幹事団の「ありたい姿」と「行動指針」は以下のとおりです。

【QCDs研究会・幹事団のありたい姿】

全員の心理的安全性を確保し、運営の立場から寄り添いながら、
全員の探求心をより引き出し、満足のいく成果物作成を支援する。

【ありたい姿の実現に向けた行動指針（大切にすること）】

①会を支えるだけでなく 誰よりもQCDsに関する学びを得る意識を持つ。
②いかなる状況においても、 前向き かつ 建設的なマインドで行動する。
③お互いの立場や価値観を尊重し、忌憚の無い意見交換を推進する。

3. 分科会のテーマ

3. 分科会のテーマ

①プロマネとは見積もりだ！（6名）

経営と現場で認識ギャップがなぜ起きるのか、炎上しないための見積計画を立てるためには？

②障害は必ずゼロにできる！（5名）

トラブルはなぜ起きるのか、テスト漏れが起きるがどこまでテストすればいいのか、〇〇品質とは何か？

③保守・メンテを極めよう！（5名）

属人化はなぜダメなのか、一人親方から卒業するには、顧客が、満足する保守メンテナンスとは？

④チームひろあき（新技術）（8名）

AI活用と品質担保、外部サービス利用と品質担保、AIと外部サービスを失敗せずに、うまく活用するには？

4. 沼津合宿(集中討議)

4. 沼津合宿(集中討議)

- **日時・場所**
  日時:2025年9月26日(金)13:00
  ～ 27日(土)12:00
  場所:沼津・プラザヴェルデ

- **合宿の目的・実施要領**
  分科会毎に分かれてディスカッションを行う。
  各社の施策事例や参加者の経験を積極的に共有しながら進めることで、
  各々が抱える課題・問題の本質を明確にし、解決に繋がる気付きを得る。

【我々が大切にしたいこと】
- 成果物作成を目的にせず、より多くの時間をディスカッションに充てる。
- 全員が活発に意見交換を実施し、より多くの気付き、知見を得る。
- 合宿という非日常の空間・時間を利用し、更なる交流、懇親に繋げる。

4. 沼津合宿(集中討議) -タイムスケジュール-

Start	End	
9月26日（金）		
13:00	13:05	合宿の大まかな流れの説明
13:05	15:00	ディスカッション①
15:00	15:30	中間発表（１チーム５分×4チーム）
15:30	17:30	ディスカッション②
17:30	18:00	ホテルに荷物を移動 ⇒ 懇親会会場への移動
18:00		合同懇親会 ⇒ 二次会へ（全員参加）
9月27日（土）		
08:45		討議会場解錠
09:00	10:30	ディスカッション③
10:30	11:30	最終発表 （１チーム１０分発表、質疑応答５分×4チーム）
11:30	12:00	お片付け・解散

4. 沼津合宿(集中討議) -中間成果物-

まとめ

- 見積もりへの思い
  - 見積もりミスで炎上するプロジェクトはいまだに後をたたないので、少しでも炎上プロジェクトを減らしたい
- 見積もりリスクを減らす大切な点
  - “見積もり＝計画”の視点をもつことで、工程の漏れや無理な計画などを減らす
  - 超概算見積もりで予算が組まれるため、どのフェーズの見積もりでも、機能の洗い出しは仮でもよいので実施して、できる限り精度の高い見積もりをする
  - レビューを必ず実施して、個人の判断ミスを減らす
- 最後に
  - 現時点で、見積もりリスク気づきシートは叩き台。現実的ない点や整理しきれていない点もあるので、皆さんの知見で、後半戦でさらに磨き上げていきましょう。

1. 沼津合宿最終成果物発表

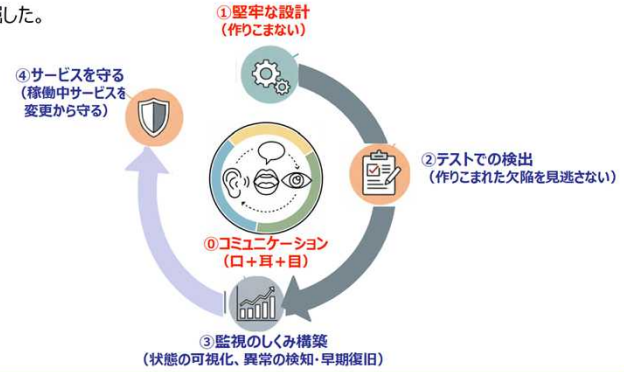
1-1：成果物一覧

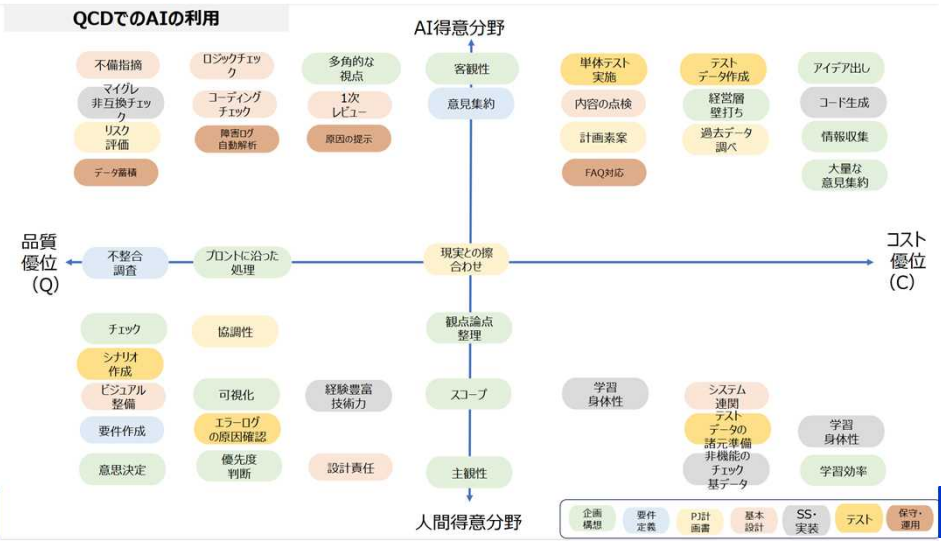
最終成果物名：Operational Legacy Enhancement Compendium（仮）
（オペレーショナル・レガシー・エンハンスメント大全）
…通称OLEC(オレック)

a1.システム・カルテ（1枚サマリ）：基本情報、担当者、稼働環境、依存関係、ロードマップ
a2.保守俯瞰図：保守範囲・レベル・SLA・外注範囲
a3.保守作業一覧：定期作業＋年間イベントカレンダー（実施者・所要時間）
a4.作業フロー図／手順書（Runbook）：手順・障害対応・エスカレーション
a5.設計／要件ドキュメント＋リンク規約：改訂履歴から要件に必ず遡れる仕組み
a6.履歴・改訂管理ルール：バージョン、変更理由、承認フローの統一テンプレ
a7.引継ぎテンプレ（新任教育用）：5W1H、緊急対応チェックリスト
a8.スキルマップ：担当者スキル可視化（ITSS.
a9.ロードマップ雛形：短中長期計画とドキュメント改修範囲
a10.AI利用ガイド：ソース優先での活用方針と検証ルール

1．障害をゼロにする！ための全体像

「障害をゼロにする！」ために重要なポイントは、
⓪関係者とのコミュニケーションをベースに、①堅牢な設計、②テストでの検出、③監視のしくみ構築、④サービスを守るであると考えた。
今回は、以下について深堀した。
⓪コミュニケーション
①堅牢な設計





4. 沼津合宿(集中討議) -参加者の感想-

長い時間をかけて議論することができたため月次の開催よりも大きく進捗させることができた。またそのような環境であったためか他チームメンバーと会話する機会も多くなり交流という意味でも有意義だったと思います。

日常と違う静かな環境(沼津)に行き、一堂に会して集中的に議論すると、幅広い考えが浮かびました。議論の進むスピードも加速度的だったと感じます。成果物の方針を決めイメージを具体化するために、合宿は必要だったと思います。

合宿を通じて自分チームのメンバーだけではなく、他チームの方々と交流が深まったので、とてもよかったです。チーム内でも研究テーマの背景、選定理由と課題等について、みんなで目線合わせできて嬉しいです。二日間は、とても有意義、濃厚な時間でした。

「キューシーディー(QCD)」の掛け声に続けて、みんなで「エース!(S)」と叫ぶところが一番よい。一気に一体感が生まれた。

5. 2025年度 活動まとめ

JUAS 一般社団法人 日本情報システム・ユーザー協会
Japan Users Association of Information Systems

2025年度システム開発・保守QCDs研究会　成果物

見積もりリスクを下げるポイントの検討と整理

2026年4月9日
分科会①「プロマネとは見積もりだ！」

JUAS

目次

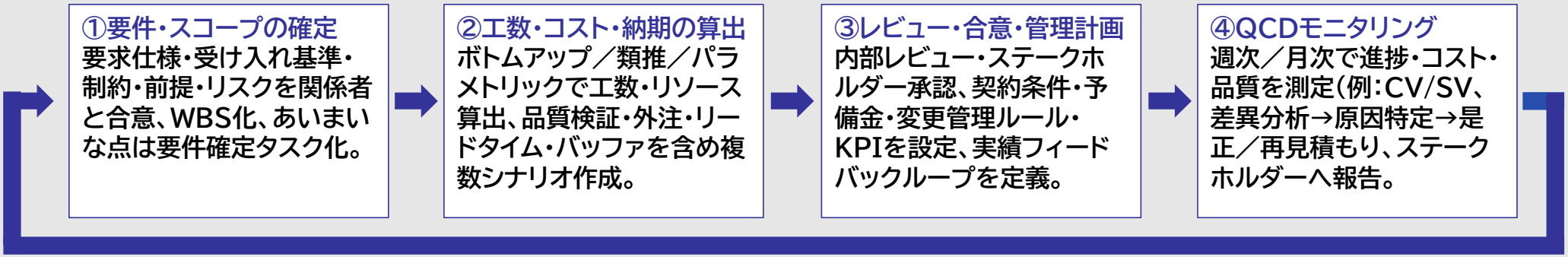
01　前提事項：QCDと見積もり

見積もりがQCDへの最初のアプローチである

以下は見積もりについての一般的な考え方であり、こうした考え方に基づき、見積もりがQCDに対して重要であるとの認識から、当分科会での議論を進めていった。

【見積もりとQCDの関係性】
見積もりはスコープ・工数・コスト・スケジュール・品質基準を数値化し前提とリスクを明示することで、QCDの初期設計を行う最初のアプローチである。品質と納期のトレードオフ、リスク予備や外注判断をこの段階で整理することで後工程の変更コストや遅延を抑制できる。見積もりはまたステークホルダー合意や契約条件の基礎となり、実績フィードバックで精度を高めてQCDを最適化することにつながる。

見積もり⇒QCD最適化のイメージ



02　研究テーマの説明

■研究テーマ 「見積もりリスクを下げるポイントの検討と整理」

前述の通り、見積もりはQCDへの最初のアプローチであり、見積もり結果がプロジェクトの成否を大きく左右する。
その重要性に鑑み、システム開発のプロジェクトにおける「見積もりリスクを下げるポイント」を検討・整理する。

■見積もりリスクとは

見積もりが過大・過小になることにより、QCD等に影響を及ぼすリスク。

	Q:品質面	C:コスト面	D:納期面	Q:経営・対外面
過小見積もりのリスク （実見積額 ＜ 適正見積）	×テスト不足・手戻り増加で品質低下 ×顧客クレーム増加	×追加コスト発生（追加人員・残業・外注） ×利益圧迫、赤字リスク	×納期遅延や納品失敗 ×契約違反によるペナルティ	×信頼喪失 ×リピート受注減少
過大見積もりのリスク （適正見積 ＜ 実見積額）	×過剰品質や不必要なプロセス導入で効率低下（無駄な作業）	×競争力低下による受注喪失 ×顧客からの値引き要求 ×社内予算の非効率配分	×必要以上に長いスケジュール提示で機会損失	×市場競争力の悪化 ×顧客との価格交渉で評価低下

02　研究テーマの説明

■研究テーマの選定理由

要求が固まらない段階での見積もりは困難、リスク・要員体制・スケジュールを織り込む見積もりは経験値が必要であり、対応策を検討・整理し、共有したいというのが主なテーマ選定理由である。

下表に、見積もりが上手くいかない主な原因を分類する。

#	分類	説明	例
1	要件・スコープ関連	何を作るかが不明瞭・変動しやすいことによる影響	仕様が曖昧／受け入れ基準不在、スコープクリープ、非機能要件の把握不足、前提条件の未共有
2	実績データ・見積手法関連	根拠となるデータや標準手法が不十分なための誤差	類似案件実績がない／蓄積不足、WBS分解が粗い、パラメトリックや指標の未整備、見積基準がバラバラ
3	人的要因・スキル・意識	担当者の能力や心理的要因による誤った判断	楽観バイアス／希望見積もり、経験不足、工数見積の属人化、レビュー文化の欠如
4	プロセス・管理体制の問題	見積もりプロセスやレビュー・変更管理が整備されていないこと	WBS作成やレビューの欠如、変更管理ルール未整備、見積もり承認フロー不明確、実績フィードバックがない
5	リスク評価・余裕（バッファ）管理不足	不確実性を定量化せず予備を取らないことによる欠落	リスク洗い出し不足、発生確率×影響の未評価、コンティンジェンシー未設定、リードタイムの見落とし
6	顧客／営業・契約関連	外部要因や商談圧力で無理な見積りが出る状況	営業の短納期圧力、価格競争での過小提示、契約形態（固定価格）とリスク配分の不整合、前提条件が契約で未明記
7	技術・環境依存性	技術的チャレンジや外部環境（サードパーティ／インフラ等）に起因する不確実性	新技術／不慣れなプラットフォーム、外部APIやサードベンダ依存、環境差異（本番と開発の差）

03　検討のアプローチ

見積もりには大きく3つのフェーズがあり、それぞれ目的や手法が異なるため、フェーズ毎に見積もりリスクを下げるための、
「見積もり観点確認シート」を作成し、重要な観点を考慮した見積もりを目指す。

見積もりの3つのフェーズ

企画構想 → 要求定義 → 要件定義 → 基本設計 → 詳細設計 → 製造 → 試験 → 運用保守

①超概算見積もり

主目的	案件の初期判断(Go/No-Go)や概略の資金枠・優先順位決定のために、早期に規模感を示す。
利用想定	・営業が案件獲得の可否判断をする(案件追跡継続／辞退)。 ・経営が予算案や投資優先度を検討する(事業計画やPoC実施可否)。 ・顧客への初期提案やRFI段階で期待コストのレンジを提示する。

②概算見積もり

主目的	予算確保・提案比較・契約候補としての具体的方針決定のために、より精度の高いコスト感とスケジュール案を提示する。
利用想定	・顧客提案(RFP/RFQ)で競合比較をする。見積レンジではなく比較可能な数値やシナリオを提供。 ・社内での予算申請や上長承認(投資判断)を受けるための資料化。 ・複数案(標準・短納期・コスト最適)を提示して顧客とスコープ調整を行う。

③詳細見積もり

主目的	契約締結・プロジェクト立上げのための確定見積もり(予算・スケジュール・リソース割当)の提示。実行計画とQCD基準のベースライン化。
利用想定	・契約金額・納期の最終確定、社内予算承認、プロジェクト開始(キックオフ)判断。 ・リソース配分・採用／外注発注の具体化、調達計画の確定。 ・プロジェクト管理(EVMやKPI設定)のベースライン作成とチェンジコントロール開始。

03　検討のアプローチ

■「見積もり観点確認シート」の構成

見積もり観点確認シートは「目的」および「見積もり前」「見積もり時」「見積もり後」の3つの段階での観点を示す。
それぞれの段階で確認すべき観点は以下の通りとなる。

段階	観点	定義/意義	記載内容
－	目的	その見積もりが何のために作成されるか。意思決定者の利用シーンを明確化することで、精度・提示形式・前提が決まる。	・主目的　(例：Go/No-Go判断、予算申請、契約金額提示) ・利用シーン／関係者(営業、経営、顧客、プロジェクトマネージャー等) ・決定に必要な精度(%幅)と期限　・想定する意思決定(受注／辞退／フェーズ移行 等)
見積もり前	前提条件	見積もりを成立させるための仮定。曖昧だと発注後に乖離が発生するため、明文化が必須。	・スコープの境界(含む／含まない機能)　・技術的前提(ミドルウェア、互換性の前提) ・環境・リソース前提(担当スキル、作業場所、利用APIの安定性) ・契約・支払条件　・リスク前提(想定される障害や外部要因の扱い)
	制限事項	見積もりで明確に含めていない事項や、制約条件(時間・資源・契約等)を示す。前提と混同しないよう区別する。	・リソース制約(最大同時稼働人数、特定技術者の稼働不可等) ・契約的制約(上限金額、支払条件、納期固定の可否) ・法的・規制上の制約(遵守が必要な基準)
見積もり時	手法	どの見積手法で算出したか(根拠)を明示し、再現性と信頼性を担保する。	・手法名(例：類推、パラメトリック、ボトムアップ、FP、三点見積、モンテカルロ) ・使用した係数・生産性値(出典：実績データ、ベンダー提示など) ・計算ロジック(簡潔な式や例)とツール(スプレッドシート、見積ツール名)
	インプット	見積もりに用いた資料やデータ。出所が明確であれば検証や再利用が容易になる。	・仕様書(要件定義書、RFP、画面遷移図等)　・実績データ(類似案件の工数データ) ・外注見積もり・サードパーティ資料　・人員情報(スキルマトリクス、稼働可能日) ・契約書素案、顧客提供条件
見積もり後	レビュー	見積もり精度を担保するためのチェック体制。誰がどの観点で承認・検証するかを明示する。	・レビュー実施者(事業責任者、技術レビュー、PMO、第三者査定) ・レビューの観点(スコープ整合性、工数根拠、リスク反映、契約条件適合) ・レビュー結果と対応履歴(レビューコメントと是正措置)
	アウトプット	見積もりの成果物。提示形式を統一することで意思決定がスムーズになる。	・見積金額(レンジ／固定)と内訳(人件費、外注、ライセンス等) ・スケジュール案(主要マイルストーン、クリティカルパス) ・WBS／工数表(タスク別、人時別)　・前提・制限・リスク一覧、コンティンジェンシー額

04　見積もり観点確認シート

①見積もり観点確認シート（超概算見積もり）

超概算見積もり

超概算見積もりは初期判断用で規模感と予算枠をレンジ提示する。類推やパラメトリックを使い画面数等のハイレベルな機能洗い出しを行い、前提・除外、主要コストドライバーと上位リスク、コンティンジェンシー目安、概算移行条件（要件確定・PoC等）を明記する。精度目安は概ね-50%～+100%、レビューで妥当性を担保。

段階	観点	概要	具体例	理由（目的・リスク）
ー	目的	中長期予算確保のため、経営層承認や社内稟議用に、レンジ付き概算見積を作成	事業計画用、稟議用、10人月粒度	承認が得られない／却下リスクを回避
見積もり前	前提条件	使用技術・人員数・開始時期を仮置き	Java利用、PM1+SE2、7月着手	曖昧だと破綻リスク増大
	制限事項	予算・納期・インフラ環境制約を明記	予算2000万、納期10月末、社内インフラ制約	考慮しないと実行不可
見積もり時	手法	・過去案件比較の類推見積 ・パラメトリック（係数）見積 ※根拠として仮の機能一覧の洗い出し （WBSのL3：機能グループ　～　L4：機能）	過去案件比 機能・非機能粒度の確認レベル：ログイン/ログアウト/ホーム画面などWBS L4（機能単位）	超概算でも大ぶれ回避が必要
	インプット	事業計画や稟議に必要な要求・条件・背景 （予算・体制・開始時期などの仮条件）	予算2000万、着手7月、要員3名	前提を整理しないと誤差拡大
見積もり後	レビュー	・概算の根拠を第三者点検 ・複数担当者で見積を実施し差異を分析、妥当性を担保	WBS・機能一覧と照合	属人化や誤差を防止
	アウトプット	レンジ付き概算と仮置きスケジュール 組織としての判断材料（Go/No-Go）	誤差-50%～+100% （PMBOK、ISO21500を参考） 粗い工程表	意思決定の材料を提供

04　見積もり観点確認シート

②見積もり観点確認シート（概算見積もり）

概算見積もり

概算見積もりは予算確定・提案比較向けに、上位WBSや主要機能で工数・コスト・スケジュールを示す。パラメトリック＋ボトムアップで算出し、三点見積やシナリオ（標準／短納期／低コスト）を用意。前提・除外、外注・テスト・移行を含めてコンティンジェンシーを設定し、技術レビューで根拠とリスクを検証する。精度目安は-25%～+75%。

段階	観点	概要	具体例	理由（目的・リスク）
－	目的	提案価格の根拠を示し顧客に納得感を与える	設計/テスト/移行/管理まで明細化	説得力を増し失注リスクを減らす
見積もり前	前提条件	提案時点のスコープ・非機能要件を整理	性能要件・運用要件・監視要件・可用性	漏れると追加費用・再作業に直結
	制限事項	システム制約、セキュリティ制約、納期、要員、法規制等	クラウド制限、納期厳守	不適合だと失格/事故リスク
見積もり時	手法	パラメトリック精緻化 ボトムアップ見積（ WBSレベル4～5：機能、工程・詳細機能）	画面×単価、帳票×単価、WBS L4（機能）、L5（詳細機能） 外部IF数、外部API、SaaS料金体系	根拠ある積算で信頼性担保
	インプット	機能要件、非機能要件、制約条件、プロジェクトの背景・目的 （例：クラウド制限、セキュリティ要件）	API必須、冗長化要件、AWS不可	曖昧だと追加費用や失注リスク
見積もり後	レビュー	・概算の根拠を第三者点検 ・複数担当者で見積を実施し差異を分析、妥当性を担保	WBS・機能一覧と照合	属人化や誤差を防止
	アウトプット	提案金額とその根拠を示す見積書、制約条件リスト	誤差-25%～+75% （PMBOK、ISO21500を参考） 契約金額3000万以下、非機能要件反映	根拠不足だと失注リスク

04　見積もり観点確認シート

③見積もり観点確認シート（詳細見積もり）

詳細見積もり

詳細見積もりは契約・実行ベースの確定見積で、WBSの細分化に基づくボトムアップが基本。タスク単位で担当を割当て三点見積等で不確実性を定量化し、テスト・移行・運用・ドキュメント等の作業と外注費を含める。クリティカルパス・バッファ・コンティンジェンシーを明記し、技術・財務・PMOレビューで根拠を証跡化する。精度目安は-5%～+15%。

段階	観点	概要	具体例	理由（目的・リスク）
ー	目的	スケジュール・コストのベースラインを確定し、契約・変更管理の基準とする	契約金額、変更管理基準	基準が無いと変更管理が混乱
見積もり前	前提条件	要件定義成果物を前提に精度を高める	非機能要件合意済、外部IF確定	仮置きだと変更リスク残存
	制限事項	契約条件（支払い条件など）・技術的制約を明記	支払一括、クラウド制約	無視すると実行不可/資金リスク
見積もり時	手法	ボトムアップ見積 （WBS L4：機能、L5：詳細機能、工程） 詳細WBS単位に三点見積を適用し、リスク調整	画面×単価、帳票×単価 WBS L4（機能）、L5（詳細機能） 各作業を楽観・悲観・最可能で算出 （※）三点見積もりの計算方法 期待値＝（楽観＋4×再可能＋悲観）÷6	リスクを織り込まないと遅延
	インプット	要件定義成果物・基本設計・仕様・非機能要件・インフラ制約	DB流用、要件定義書、官公庁クラウド,基本設計書	把握不足は再作業・再見積に直結
見積もり後	レビュー	・詳細WBS・工数妥当性を検証 ・複数担当者で見積を実施し差異を分析、妥当性を担保	1画面PGが上級者基準か確認	属人化や誤差を防止
	アウトプット	確定見積・基準スケジュール・変更管理基準・リスク対応費・マネジメントリザーブ	誤差-5%～+15%、支払計画	過信は契約交渉を困難化

05　まとめ

■見積もりへの思い

　見積もりミスが原因でプロジェクトが炎上する事例は依然として多く、人的被害・顧客信頼の毀損・収益悪化といった重大な影響を生む。見積もりは単なる「数字合わせ」ではなく、プロジェクトの設計図であり、QCD（品質・コスト・納期）の最初の決定行為である。だからこそ、見積もりの精度を高め、誤差や想定外を減らすことは、炎上プロジェクトの予防と早期発見に直結する。そのため、個社の経験に依存しない普遍的で再現可能な見積もりプロセスやチェックリストを作り、現場の失敗を減らすことを本分科会の目的とした。

05　まとめ

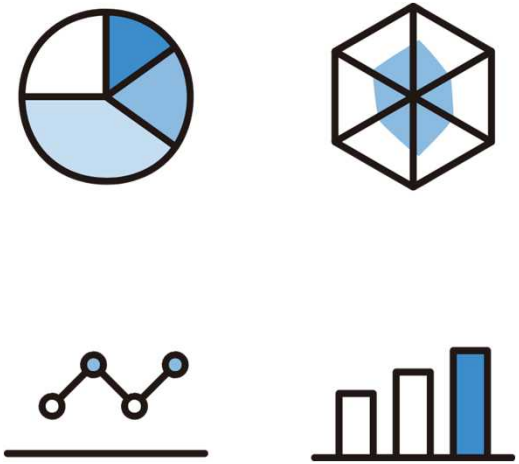
■見積もりリスクを減らす大切な点

①“見積もり＝計画”の視点を持つ

見積もりは単なるコスト提示ではなく、実行計画の根拠。工程の抜けや無理な並行スケジュールは、見積もり段階での設計不足が原因であることが多い。

実務アクション：
WBSや主要マイルストーンを作成し、各工程（設計・実装・テスト・移行・受入・ドキュメント・運用準備）を漏れなく洗い出す。クリティカルパスとリードタイム（承認や検証）を明示し、リソース制約を反映したスケジュールにする。

効果指標：
工程抜けによる追加工数件数、スケジュール偏差（SV）の減少。



05　まとめ

■見積もりリスクを減らす大切な点

②超概算でも機能洗い出しを行う（仮で良いから量を出す）

初期段階の予算は超概算で決まることが多く、この段階での不正確さが後の設計や調達に波及する。大枠でも機能・画面・データ連携・外部依存の数を出すことで、見積もり幅を狭められる。

実務アクション：
ROM段階でも「機能リスト（ハイレベル）」を作成し、機能ごとに代表的な工数係数を適用する（例：画面×○人日、帳票×○人日）。不明点には必ず「不確実性フラグ」を付け、見積もりレンジに反映する。

効果指標：
超概算と詳細見積もりの乖離率、ROMから概算へ移行した際の見積誤差の縮小。

05　まとめ

■見積もりリスクを減らす大切な点

③レビューを必須化し、個人依存を減らす

楽観バイアスや経験者の過信により過小見積もりが発生しやすい。複数視点のレビューで盲点を早期に補完できる。

実務アクション：
見積もりは「作成者→技術レビュアー→PMO/事業責任者」の最低2段階レビューを必須とする。重要案件は第三者（別プロジェクトPMや外部専門家）による監査を行う。レビュー項目には「前提の妥当性」「非機能作業の抜け」「外部依存の反映」「コンティンジェンシーの設定」を含める。レビュー記録は見積書に添付する。

効果指標：
レビューで検出された修正項目数、レビュー未実施による追加請求・手戻り件数の減少。

参考01　見積もり観点確認シートの評価

■Copilotによる評価 

当資料は、複数のシステム開発会社が共同で行っている研究会で、見積もりについての研究成果を発表するためのものです。この資料の見積もり観点確認シートについて評価してください。

■総合評価

3種類の見積もりに対し、目的・手法・前提条件・制限事項・レビュー・インプット・アウトプット という統一観点で整理されており、フェーズ別の“見積もりの深度差”を明確化できている点が大きな強みです。

特に以下が優れています：
・**観点の網羅性が高い**（漏れの原因になりやすい前提／制限／レビューまで記載）
・**見積もり精度レンジ（例：-20%～+75%）が記載されており実務的**
・**WBS粒度（L3～L5）を明示しているため再現性が高い**
・**超概算→概算→詳細と、手法の精度向上が連動している**

一方で、産業界のベストプラクティスやPMBOK・IPA 見積もりガイドとの比較でみると、より実用的にする余地がいくつかあります。以下にページごとの詳細評価を示します。

参考01　見積もり観点確認シートの評価

■Copilotによる評価

■超概算見積もりシートの評価

◎良い点

・“超概算でも仮機能洗い出しを行う”という方針が明示されている
　・実務では超概算ほど「スコープ未定」「根拠なし」になりやすいため、この姿勢は非常に適切。
・手法が類推・パラメトリックに限定されており、現実的
　・過去案件比、係数見積もりを使うという明記は分析可能性を高める。
・前提条件と制限事項を分離して記載
　・他社案件ではここが混同されがちで、後工程トラブルの原因。
・レンジ付き概算（-20%～+75% が例示）
　・レンジ幅を明示すること自体が超概算の信用性を高める。

△改善提案　※取り込み済み

1. レンジの根拠を示すとさらに有用
　・例：PMBOKやISO21500では概算は -25～+75%、超概算は -50～+100% が一般的。
　・本資料では例示だけなので、標準値とどう異なるかを示すとビジネスで使いやすい。
2. “仮の機能一覧” の作成方法が不明確
　・L3/L4と記載はあるが、実際には「機能粒度は最低限どこまで必要か」「非機能はどこまで洗うか」をガイドとして示すとブレが減る。
3. 組織としての判断基準（Go/No-Go）が欠けている
　・企画審査会などで使われるなら、「判断材料として何を最低限提出すべきか」を明記した方が利用価値が高い。

参考01　見積もり観点確認シートの評価

■Copilotによる評価

■概算見積もりシートの評価

◎良い点
・手法の精緻化（パラメトリック → ボトムアップとの併用）
・具体例が画面×単価、帳票×単価と実務に即している
・提案段階で重要な“顧客納得性・失注リスク”の観点が明記されている
・制限事項にセキュリティ・クラウド制限を明記している
　・実際に見積もり漏れの発生しやすい箇所。

△改善提案 ※取り込み済み
1.　非機能要件の扱いをさらに明確化すべき
　・概算見積もりで最も漏れやすいのが非機能。
　　例：性能要件・運用要件・監視要件・可用性
　・表では「冗長化必須」程度の記述に留まっているため、もう少しカテゴリを提示すると実用度が上がる。
2.　依存関係・外部調達コストの明確化が弱い
　・RFP/RFQ応札では外部ベンダ見積もりの不足が誤差原因になるため、「外部IF数」「外部API利用」「SaaS利用の料金体系」
　　などを観点化しても良い。
3.　精度レンジ（-10～+30%）の根拠がない
　・外部提案の標準値と比較するコメントがほしい。

参考01　見積もり観点確認シートの評価

■Copilotによる評価

■詳細見積もりシートの評価

◎良い点

・三点見積法(楽観／悲観／最可能)の導入が非常に優れている
　・このレベルの研究会資料で三点見積を明記しているのは質が高い。
・契約・変更管理まで含めて“ベースライン化”を目的にしている点が明確
・インプットが「要件定義成果物・基本設計書・クラウド制約」まで具体化
　・実務プロジェクトの精度確保に直結。

△改善提案 ※取り込み済み

1. WBSレベルの統一基準の例示がほしい
　・例:「1画面＝何時間が標準なのか」「詳細機能の粒度はどこまでか」
　・これを定義しないと属人化リスクが残る。
2. 三点見積の計算方法(期待値の算出)がない
　・期待値 ＝（楽観 ＋ 4×最可能 ＋ 悲観）÷ 6
　・このような式を明記すると教育資料として強力になる。
3. リスク対応費(コンティンジェンシー)とマネジメントリザーブの区別が必要
　・PMBOKに沿うならここを分けるべき。
4. 精度(-10% ～ +30%)はやや広め
　・このレンジは概算寄りで、詳細見積もりなら -5%～+15% 程度が一般的。

参考02　参加者の所感

見積もりは精度の高いシステム開発の計画と経営観点や営業観点も必要となる。
そのうえで十分に注意して実施する必要があるため、見積もり観点確認シートはその母数感を補うものとしてとても利用価値があると感じた。

見積もりをどのような切り口でとらえるかというものが、明文化することができたと思う。
方法論やチェックポイントでは、手が届かない部分にアプローチできたと思う。

見積もりフェーズごとに具体例とリスクが整理されている点が実務的。
超概算から詳細まで一貫した観点で確認でき、見積もり精度向上に貢献できると考える。

普段の見積作成において経験則、社内のルール等に引っ張られて視野が狭くなってしまうことを感じていたが、研究会を通して新しい気付き、改善点を認識することができた。

①フェーズごとに超概算見積、概算見積、詳細見積を実施することの重要性を改めて認識いたしました。
②見積もり観点確認シートは分かりやすく、実践的なフレームワークであると感じました。
今後のプロジェクト推進においても、積極的に取り込んでいきたいと思います。

見積もりの難しいポイント、勘所を改めて考えるいい機会となった。ここでまとめた観点は社内でも十分に活用できる内容になったと思う。

JUAS 一般社団法人 日本情報システム・ユーザー協会
Japan Users Association of Information Systems

2025年度　システム開発・保守QCDs研究会 成果物

システム障害を抑制するためのポイント整理

コミュニケーション重視の体制構築と
上流～設計フェーズにおける心得・基本行動

2026年4月9日

分科会②：障害は必ずゼロにできる！

JUAS

目次

１．障害をゼロにする！ための全体像

重要なポイント

システム開発全体における重要なポイントは、「精緻な見積もり」、「密なコミュニケーション」を前提として、「堅牢な設計」、「テストでの検出」、「監視のしくみ構築」、「保守メンテによるサービス維持」と考える

上記のうち今期メンバーが特に課題として持っている要素は、「精緻な見積もり」、「密なコミュニケーション」、「堅牢な設計」、「保守メンテによるサービス維持」の４点

「精緻な見積もり」と「保守メンテによるサービス維持」は【分科会①プロマネとは見積もりだ！】と【分科会③保守・メンテを極めよう！】で検討整理されるため、**本分科会では「密なコミュニケーション」、「堅牢な設計」について検討整理**

2．背景・課題

背景課題

・各社事例持ち寄りの結果、システム障害の主因は設計段階での不備埋め込みが多い傾向
・それらの根本原因は「コミュニケーション不足」、「基本行動不遵守」に偏重している

コミュニケーション不足

・認識齟齬、要件の理解不足
・課題、進捗状況の情報共有不足
・信頼と士気の低下　etc…

基本行動不遵守

・影響範囲の確認漏れ
・設計書への理由、根拠未記入　etc…

システム障害を抑制するための重要なポイントとして以下2点を考える
- **『コミュニケーション重視の体制構築』**
- **『遵守するべき基本行動の定義』**

3．コミュニケーション重視の体制構築①（取り組み事例）

ポイント

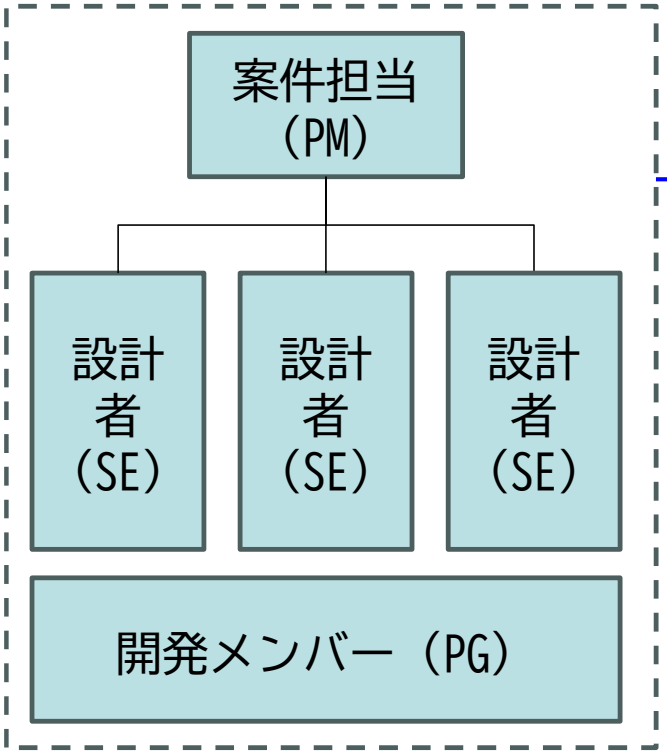
・関係者と**気軽に素早く『報告』『連絡』『相談』が行える状態**の組織
・密にコミュニケーションをとることで、認識ズレやミスを防ぐことができる
・具体的な事例として以下の取り組みをできていることが望ましい

No	望ましい取り組み事例	効果
1	スタンドアップミーティング等の開催	メンバーの抱える課題や進捗遅延の早期発見・対応が可能になる
2	スタンドアップミーティングでのアイスブレイクを用いた良い関係の醸成	メンバー同士の信頼関係や親しみが深まり、相談や意見交換がしやすくなる
3	1on1ミーティングの実施	個々の悩みや課題を気軽に共有でき、早期解決やモチベーション向上につながる
4	グループワークとディスカッション	チーム内の協力関係が強化され、創造的なアイデアや解決策が生まれやすくなる
5	上席からの積極的な声かけ	「何かあったら言ってね」ではなく、積極的に声かけすることで相談しやすい風土を醸成

3．コミュニケーション重視の体制構築②（推進体制）

ポイント

・業務知識・マネジメントに長けた要員をフォロー役としてを配置
・適切なタイミングで関係者とコミュニケーションをとり、『指導』『方向修正』『課題解決』を行い、フォローできる体制を構築



フォロー役

所謂、名誉職的な『評論家ポジション』ではなく、自身のミッションに責任をもって取り組み、プロジェクトマネジメントから作業レベルまで俯瞰した視点で、全局面のフォローに責任も持って取組む役割
※複数案件を並行してフォローする

３．コミュニケーション重視の体制構築③（フォロー役の必要スキル）

ポイント

・プロジェクトマネジメントから作業レベルまでの俯瞰視点とスキルが必要
・担当者(PM)を次のフォロー役へ引き上げる育成視点(スキル)が必要
・具体的なスキルのポイントは、以下４点と考える

状況把握・俯瞰力

・案件全体(進捗・リスク)を一段上の視点で捉えられる
・担当者が気がつきにくい違和感を察知できる

業務知識・技術知識

・業務背景や設計意図を理解し適切な判断ができる
・「納得感のある指導」ができる

コミュニケーション力

・指摘ではく「対話」で軌道修正できる
・相談しやすい雰囲気を醸成できる

育成視点

・すぐに手を出さず成長機会を見極めて支援
・「任せる」「フォロー」のバランスをとる

３．コミュニケーション重視の体制構築④（取り組み結果事例）

事例

・分科会参加企業１社の一部所属にてコミュニケーション重視の体制を用いた運営を実施
・対前年度（２０２４年度：２０２５年度）においてシステム障害の抑制効果を確認

２０２４年度

開発工程	工程	原因
外部設計	コミュニケーション不備	３件
	思い込み	２件
	基本行動不遵守	１件
製造	基本行動不遵守	１件
テスト漏れ	コミュニケーション不備	１件
運用	思い込み	１件
合計	ー	９件

２０２５年度

開発工程	工程	原因
外部設計	基本行動不遵守	１件
テスト漏れ	思い込み	１件
合計	ー	２件

フォロー役を中心とし「密なコミュニケーション」を出来る状態にすることて、多くの障害は抑制できる可能性がある

４．守るべき設計フェーズでの心得、基本行動

ポイント

・「要件定義」「基本設計」の各工程で不備が組み込まれるのを防止するため、各社から意識すべきポイントを持ち寄り、共通点や共感点について掘り下げて議論

・設計工程において重要となる基本的な行動を「ＭＵＳＴ　１０」（仮）」として定義

No	分類	心得・基本行動
1	コミュニケーション	確認は信頼への第一歩
2		相手の業務にも気を配れ
3		仕事を任せた後もフォローしろ
4		隠すより 笑って話そう 解決を
5		信じるな 他人のチェックと 済みマーク
6	堅牢な設計	「なぜやるか」が、価値をつくる
7		書いて伝えよう、設計・設定の理由（ワケ）
8		要件は、「できる」「できない」合意して
9		踏襲は 意外と多いぞ 違うとこ

（心得・基本行動：１）確認は信頼への第一歩

確認は信頼への第一歩
～相手の理解を確かめることで、認識の齟齬を防ぐ～

●人は異なる背景や価値観を持つため、同じ言葉でも受け取り方が異なることがある

●「伝わるはず」と思い込まず、「誤解されるかもしれない」と考えることで、丁寧で正確な伝え方を意識できる

●誤解を防ぎ、信頼関係の構築や効率的な対話につながる

●相手が理解したことを確認して認識齟齬を防ぐことができる

（心得・基本行動：２）相手の業務にも気を配れ

相手の業務にも気を配れ

●複数の組織が関わる業務を進める際は、セクショナリズムに陥らず、関係する組織への配慮が重要。
自分たちが提供するデータがどのように利用されるか、また受け取るデータがどのように作成されたかを、組織間で相互に確認し合うことで、誤解や情報の漏れを防ぐ。

●円滑な連携のためには、コミュニケーションを密にし、各組織の役割や業務内容を理解し合う姿勢が求められる。

（心得・基本行動： 3）仕事を任せた後もフォローしろ

仕事を任せた後もフォローしろ

●仕事を依頼した後、依頼しただけで安心せず、
進捗状況を確認したり、適切なタイミングでフォローを
行うことが重要。

相手が問題なく進めてくれるだろうと放置すると、
思わぬミスが発生することがある。

こまめなコミュニケーションやサポートを心がけることで、
ミスを未然に防ぎ、円滑に業務を進めることができる。
依頼後のフォローは、仕事の質を高めるポイントの一つ。

（心得・基本行動：４）隠すより　笑って話そう　解決を

隠すより　笑って話そう　解決を

●開発プロジェクトでトラブルが発生したときに、
担当者だけで問題を抱え込まず、早期発見・共有・対応することが重要。
担当者が問題を抱え込んで初動対応が遅れると、
トラブルが長引いたり重大化する等のリスクを招く。

そのためにはプロジェクト管理ツールによる業務進捗の共有や、複数メンバーによるチーム制、定例会議での対応協議等により、担当者の孤立化を防ぐ体制構築が有効。

（心得・基本行動： 5 ）信じるな　他人のチェックと　済みマーク

信じるな　他人のチェックと　済みマーク

～他人のチェックに甘えず、責任を持ってチェックしよう～

●複数のチェック者やレビュアーがいる場合、「自分がやらなくても他の誰かがチェックしてくれるだろう」と甘えがち

●他人への依存心や甘えを持ってしまうと、結果として誰も責任を持ってチェックを行わないことに・・・

●他人に頼らず、自分自身が最後の砦であるという意識を持つことが重要

●各自が責任を持つことで、作業ミスやトラブルを未然に防ごう

（心得・基本行動： 6 ）「なぜやるか」が、価値をつくる

「なぜやるか」が、価値をつくる
～意図を持って動くことで、効率的で意味のあるアウトプットに～

●作業や判断を行う際には、「何のためにそれをするのか？」という問いを常に自分に投げかける

●この問いかけにより、無駄な作業や誤った方向性を防ぐことができる

●目的を明確にすることで、手段や優先順位の選定が的確になる

●チーム内の認識のズレや誤解を減らすことができる

（心得・基本行動： 7 ）書いて伝えよう、設計・設定の理由（ワケ）

書いて伝えよう、設計・設定の理由（ワケ）

～設計書には「何を作ったか」だけでなく「なぜその設計・設定を選んだか」という理由と根拠を記載～
この記載は、将来のリスクを防ぐための投資です

【目的と効果】

- **保守・改修の効率化**
  数年後の担当者や改修者が、設計の意図をすぐに理解できるようにする
  無駄な再調査や誤った変更を防止。
  理由が記録されていないと、過去の重要な判断基準が失われ、
  システムが不安定になるリスクが高まる

- **正当性の担保**
  技術的・ビジネス的なトレードオフの結果としての設計の正当性を証明
  レビューや監査に迅速に対応できる

- **知識の継承**
  判断の背景を伝えることで、チームのノウハウを蓄積
  メンバーのスキルアップにつながる

（心得・基本行動： 8）要件は、「できる」「できない」合意して

要件は、「できる」「できない」合意して
～要件定義書は認識のズレを防ぎ高品質な開発を実現するため、「できないこと」も明記すべき～

【目的と効果】

●期待値の適正化
ユーザーの「当然できるだろう」という過度な期待や、暗黙の前提による
コミュニケーションミスを未然に防ぐ

●スコープの明確化
「やらないこと」を明示することで、開発途中の手戻りや追加要求を防ぎ、
プロジェクトの範囲（スコープ）を確定させる

●品質の担保
技術的な限界や非機能要件の制約（例：同時接続数の上限）を共有することで、
設計のブレを防ぎ、テスト範囲を明確化し、一貫した品質を保証する
「できないこと」の明記は、認識のズレを防ぎ、プロジェクトを円滑に進める
最強の防御策となる

（心得・基本行動：９）踏襲は　意外と多いぞ　違うとこ

踏襲は　意外と多いぞ　違うとこ
～前回と同じという思い込みは障害の素～

●定例作業やリプレース等で「前回同様 / 現行踏襲」という言葉に安心し、油断しがち

●作業内容が同じであっても、実施される環境や条件が前回とすべて同じとは限らない

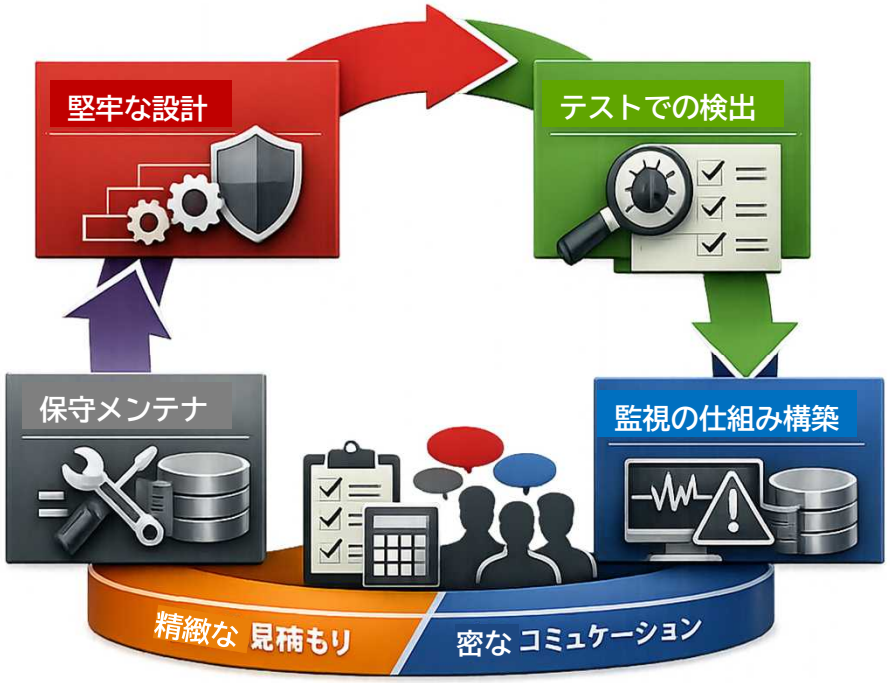
●油断は誤解や手違いにつながる可能性があるため注意が必要

●表面的には「同様」であっても、細部に違いが潜んでいるケースは多く、
どこかに変更や異なる点がないかを確認し、影響を把握することが重要

5．「まとめ」と「次年度以降の検討課題」

まとめ

・今年度は『コミュニケーション』と『設計の心得』について集中的に検討、意見交換
・最重要ポイントは設計時に不備を組み込ませない『密なコミュニケーション』と認識
・その土台としてコミュニケーション重視の体制構築が必要と考える



次年度以降への持ちこし課題

・在宅勤務、リモート会議の活用で生じる品質低下課題の改善検討

・品質意識、文化向上に資する施策検討（品質川柳　etc）

・ＭＵＳＴ１０（仮）の命名

2025年度システム開発・保守QCDs研究会
分科会③「保守・メンテを極める」チーム

最終成果物発表

2026年4月9日

がしやま、やまらぁ、なべさん、もっちゃん、あっちゃん

目次

1. 背景・目的

(1) 課題

・運用保守が属人化しており、ドキュメントが散在・未整備で引継ぎ漏れや対応遅れが発生している。
開発側へ情報移行できていない、細かな仕様やノウハウの継承手法が未定義。

(2) OLEC作成の目的

・運用保守の標準化とドキュメントの一元化・テンプレート化により、最低限の引継ぎを確実にする。
・OLEC（Operational Legacy Enhancement Compendium※運用保守改善体系）をテンプレ／運用ルール集として整備し、保守品質を安定化させる。

⇒ 引継ぎ時間短縮、障害対応（MTTR）短縮、監査・外注管理の容易化。

OLECを活用・推進し、保守・メンテからIT投資全体に好循環を生み出そう!!

2. 最終成果物

a1.システム・カルテ

システムカルテ　Ver.3

記載年月日	yyyy/mm/dd	記載者氏名	JUAS 太郎

システムの目的

目的	効率よくトラックを割り当てるため、運転手およびトラック・ルート・積み荷を一元管理する

基本情報

APコード	ABC001	APコード名称	ABC		
業務オーナー企業名	ABC物流株式会社	業務オーナー部署名	配車管理部	業務オーナー窓口	○○　次郎様
本番稼働開始日	yyyy/mm/dd		扱う情報の種別	個人情報無	
海外からのアクセス有無	無		モバイルアクセス有無	有	
問い合わせ頻度	3回/週		※2回/週、1回/2ヶ月など、おおよその目安をお願いします		

定常業務	※毎月n日に何をする・・・など
1）自車両の棚卸し作業（月初） 2）運転手情報の棚卸し（月初） 3）集荷荷物の区分追加（半年ごと） 4）営業カレンダー追加（月初）	

運用処理一覧	※日次・週次のバッチなど
1）3:00～5:00夜間バッチ起動（日次） 2）1:00～3:00システムメンテナンス（月次） 3）22:00～24:00営業レポート作成（週次）	

集配信コード	ABCDE
他システムとのインタフェース	※関係するシステム全てを記載してください（調整先担当者含む）
1）人事システム（非同期・HULFT集配信） 2）製造管理システム（オンライン・API） 3）物流基幹システム（オンライン・API）	

システムドキュメント類保管フォルダパス	※ドキュメントのパス（¥¥/aaaa/bbbb/cc）を全て記載して下さい
¥¥/aaaa/bbbb/cc	

【成果物説明】
システムの基本情報を1枚に
集約。新規参画者がシステムの概要を
理解するために最初に利用するもの。

【項目例】
・ 業務システムの目的
・SLA…可用性目標、障害対応時間、
復旧目標、優先度分類
・お客さま情報
　┗関係者連絡先（別冊　システム
　担当者/連絡先情報など）
・サポート時間時間
・システム稼働時間
・起動方法(Web、client)
・利用環境
・起動方法

a2. システム構成図

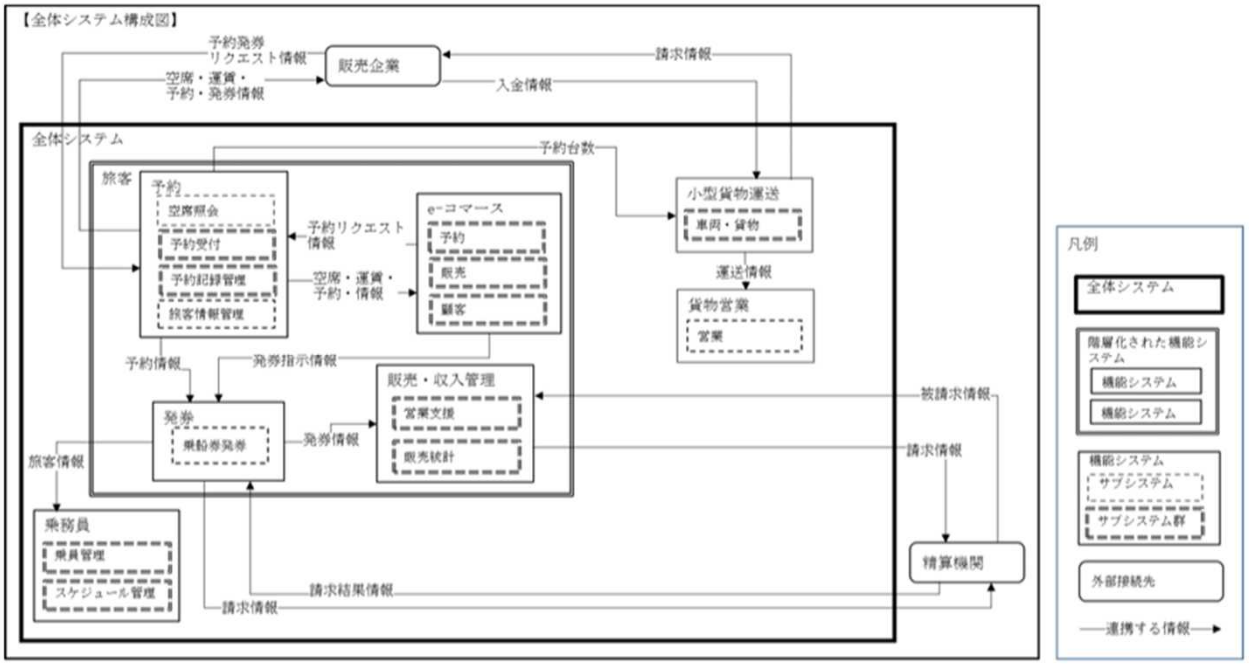


図 2-2 「全体システム構成図」(例)

出典：IPA DX実践手引書 IT システム構築編 レガシーシステム刷新ハンドブック
https://www.ipa.go.jp/digital/dx/hjuojm000000eem6-att/000089583.pdf

【成果物説明】
システム全体の要素と接続を視覚化し、設計・開発・運用の共通認識を作る図。
障害対応や変更影響の把握、セキュリティ/監査対応にも利用するもの。

【項目例】
・コンポーネント（構成要素）
・ネットワーク構成
・ホスティング
・環境
・データフロー
・外部接続・連携

a3.保守作業一覧

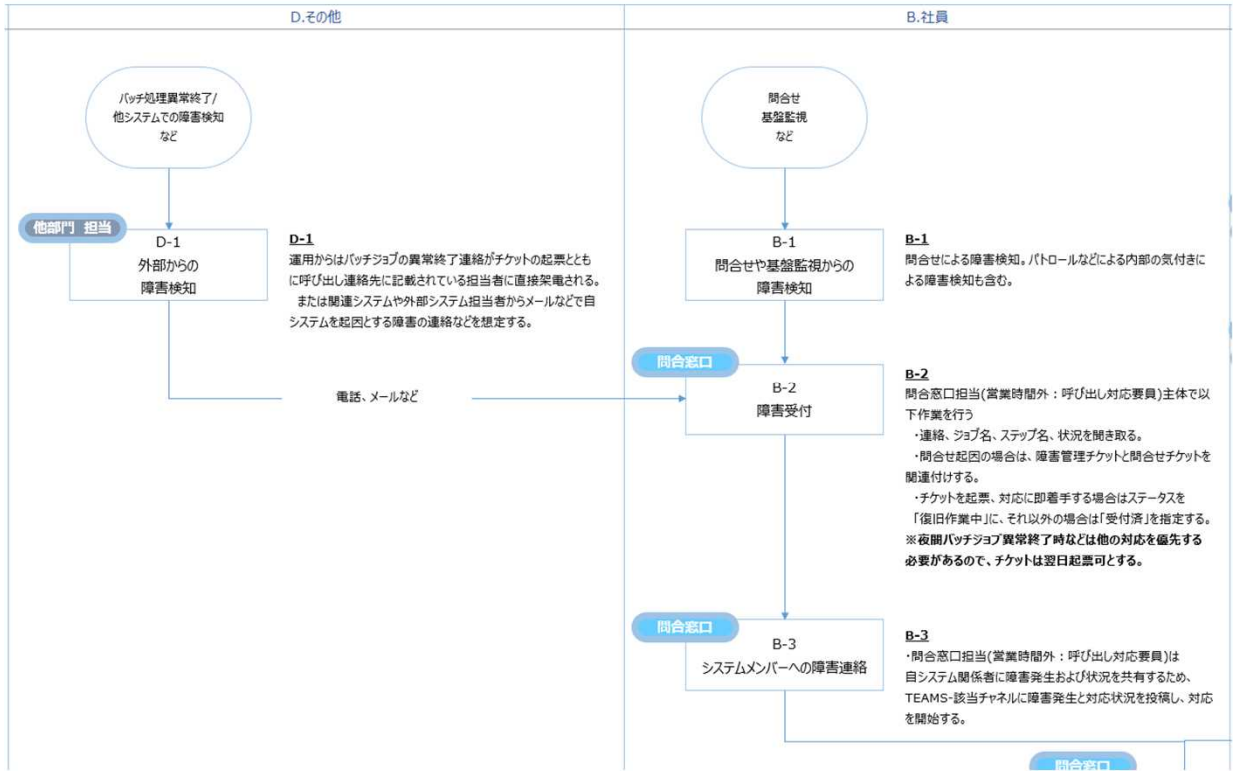
APコード	作業名	年間作業数	1回の作業時間（h）	合計	4月	～	3月	維持費内 or 維持費外
ABC001	自車両の棚卸し作業	12	0.25	3	0.25		0.25	維持内
ABC001	運転手情報の棚卸し	12	2	24	2		2	維持内
ABC001	集荷荷物の区分追加	2	0.4					維持内
ABC001	営業カレンダー追加	12	0.3	3.6	0.3		0.3	維持内

実施時期（随時の場合頻度も記入）	トリガー	手順書パス その他資料	必要な事前準備	備考
月次	毎月1日	¥¥/aaa～	本番データ書換申請	
月次	毎月1日	¥¥/bbb～	人事システム利用申請	
半年	6月1日、12月1日	¥¥/ccc～	本番データ書換申請	
月次	毎月1日	¥¥/ddd～	本番データ書換申請	○○用営業カレンダーを使用

【成果物説明】
保守作業の全体像を可視化し、誰が・いつ・どの作業を・どのくらいの頻度で・どのくらいの時間をかけて実施しているかを明確にすることで、属人化防止・効率化・引継ぎ円滑化を図る。

【項目例】
・実施頻度
・実施者
・所要時間
・実施手順（詳細手順）
・a4.作業フロー図／手順書のリンク

a4.作業フロー図



【成果物説明】
保守作業の流れを可視化し、誰が・どのような手順で・どのタイミングで作業を進めるかを明確にすることで、属人化防止・作業品質の均一化・引継ぎの円滑化を図る。

【項目例】
・作業トリガー
・担当者
・作業内容

a5_設計／要件ドキュメント+リンク規約 兼
a6_履歴・改訂管理ルール

【機能設計】　バッチ仕様書		プロジェクト	ＡＡＡプロジェクト				
Ver.	更新内容	作成日/更新日	作成者/更新者	承認日	承認者	リンク	リンク
0.1	初版作成	2020/04/02	担当者名	2020/06/12	管理者名		
1.0	外部設計完了版	2020/06/12	担当者名	2020/06/12	管理者名	redmine url	svn pass
3.0	稼働後フォローフェーズ完了版（承認）	2022/07/29	担当者名	2022/07/29	管理者名	redmine url	svn pass
3.01	システム変更 #71234シート「機能概要」 ・入出力構成の出力1999999の補足にuuuuu追加 ・入出力構成の出力1888888の補足からiiiiii削除	2023/01/25	担当者名	2023/03/22	管理者名	redmine url	svn pass
3.02	システム変更 #72345 シート「機能概要」、「出力仕様」 ・「あああああ」→「いいいいい」に変更	2023/02/08	担当者名	2023/03/22	管理者名	redmine url	svn pass
3.03	システム変更 #73456シート「機能概要」 ・入出力構成の出力199999の補足にxxxxx追加 ・機能概要、機能内容の「ｙｙｙｙ」の記載に、所定のディレクトリに格納することを追記。 ・xxxのzzz格納は後続バッチ(Bzzzzz_あああああ)の処理のため削除	2024/03/29	担当者名	2024/03/29	管理者名	redmine url	svn pass
3.04	システム変更 #81234シート「機能概要」 ・「あああああ」ファイルから廃止された連携先（gggggggg、zzzzzzz）を削除	2024/08/28	担当者名	2024/08/28	管理者名	redmine url	svn pass

【成果物説明】
設計・要件ドキュメントの改訂履歴から、必ず要件まで遡れる仕組みを構築し、5W1H（いつ、どこで、誰が、何を、なぜ、どのように）を明確にすることで、トレーサビリティと品質保証を強化する。
リンク欄にredmineやsvnなど構成管理ツールのパスを記載しトレーサビリティを確保する。

【項目例】
・バージョン情報
・更新内容
・作成/更新日
・作成/更新者

a7.新任教育用引継ぎテンプレート

当該システム【業務領域】において習得すべき事項						配備すべき資料	目的や内容	重要度 ※下欄参照	着手予定	完了予定	着手実績
コード	大分類	コード	中分類	コード	小分類						
01	基本学習	010	用語	010	各種用語の意味	用語集	対象システムで使用する用語とその意味を理解	中	8/8	8/8	8/14
		020	業務理解	010	顧客業務の理解	顧客業務説明資料	顧客業務について理解する。	低	8/8		
		030	事前準備	010	自端末の整備	開発環境セットアッ	保守開発に向けての必要なアプリの導入手順	高	8/5	8/13	
						ツールフォルダ	保守開発に向けての必要なアプリ（ツール）フォルダ	高	8/5	8/13	8/14
				020	システムの基礎学習	Linux基礎	Linuxの概要、基本的なコマンド等を理解する	中	8/5	8/13	8/14
						Javaの基礎	Java言語について理解する。	低	8/5	8/13	
						データベース基礎（S	データベースについて理解する。	低	-	-	-
		020	システム概要	010	目的・利用ユーザー・概	システム概要	システム機能とシステム構成を理解する	高	8/8		
						XXXシステムを操作し	結合テスト時に普段行っている打鍵準備や、オンライン画面からのログイン、ツールにより画面打鍵を一通り体験し、オンラインの具体的なイメージを	高	8/22	8/22	8/22
				020	周辺システム	システム関連図	システム全体像の概念とその中にある当該システムの位置づけ、環境、連動するシステムについて	高	8/8		
						各システム説明資料	新入社員向けプロダクト説明資料	高	8/8		
				030	H/W構成	-	システム全体の構成を踏まえたうえで、当該システムを構成しているH/W構成（機種／スペッ	低			
				040	S/W構成	-	システム全体の構成を踏まえたうえで、当該システムを構成しているS/W構成（プロダクト、バー	低			
				050	機能配置と機能概要	ディレクトリ構成	当該システムの中で実現している機能の概要と	中			8/15
				060	業務フロー/作業フロー	定型作業フロー	定型保守作業の作業フローを理解する。	高			8/15
				070	データフロー	-	業務フローを理解したえで、当該システムの機能を構成する主要データの流れと種類を、外部連携に関してはそのデータのレイアウト、内容、ライフ	中	8/21	8/21	
				080	構成管理	ライブラリ管理	ライブラリ管理の構成と運用の概要を理解する	高	8/21		8/21
						ライブラリ管理運用手	ライブラリ管理運用の詳細手順。最新版を参照	高	8/21		8/21
		030	個別機能（仕	010	画面機能	-	各画面で実現している機能と、その理由（業務要件）を理解する。また、それを実装している方	中	8/21	8/21	

【成果物説明】
新任担当者が業務を円滑に引き継ぎ、早期に戦力化できるよう、必要な情報・スキル・資料の所在を体系的に整理する。

【項目例】
・大・中・小分類
・配慮すべき資料
・目的や内容
・重要度

a8.スキルマップ

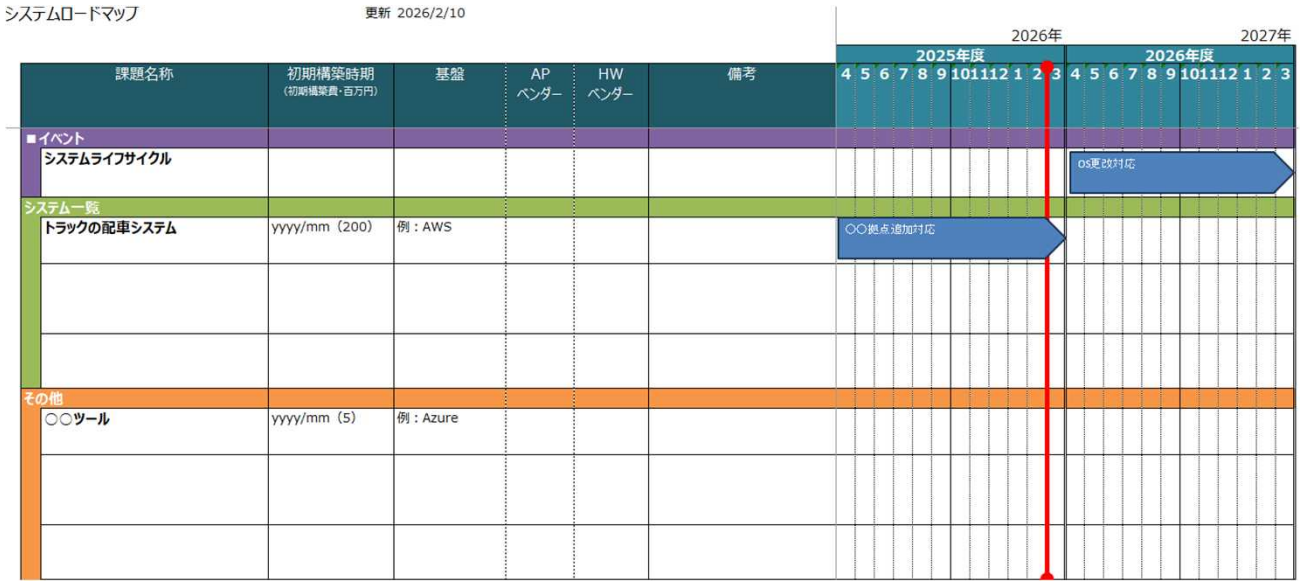
スキル一覧　システム名：トラックの配車システム

当該システム［業務領域］において習得すべき事項						要否	配備すべき資料（例）	目的や内容	重要度 ※下欄参照	資料配備			スキル	
大分類		中分類		小分類										
コード		コード		コード						資料リンク	配備日	配備者	業務知識	システム知識
01	基本学習	010	用語	010	各種用語の意味	要	用語集	業界用語とその意味（概要）、また、同義語（同一のものを別の名称で呼ぶ）、専門用語とその意味を理解する。	高	¥¥/aaaaaa~	2026/1/1	○○	○	
		020	業務理解	010	顧客業務の理解	要	業務資料	システムを利用している顧客の業務、提供サービスを理解する。	高	¥¥/aaaaaa~	2026/1/9	○○	○	
		030	事前準備	010	自端末の整備	要	PCセットアップ手順	自分のPCに、開発で必要なTOOL等をインストールする。	中	¥¥/aaaaaa~	2026/1/9	○○		○
						要	開発ツール、開発ソフトウェア							
				020	システムの基礎学習	否	ネットワーク基礎	ネットワーク基礎について理解する。						
						要	Linux基礎	Linuxのコマンドについて理解する。						
						要	Javaの基礎	Java言語について理解する。						
						要	Linuxシェルスクリプト編	シェルについて理解する。						
						要	データベース基礎（SQL）	Java言語について理解する。						
02	システム理解	010	用語	010	各種用語の意味	要	用語集	当該システムにおける独自用語とその意味、また、同義語・類語を理解する。						
		020	システム概要	010	目的・利用ユーザー・概念	要	システム概要資料 WB4システム紹介	利用ユーザーとシステムの目的を正確に理解する（誰が、いつ、何のため、このシステムを使うのか）。システムの概念を理解する。						
						要	導入会社一覧						○	
				020	周辺システム	要	機能概要資料 システム関連図	システム全体像の概念とその中にある当該システムの位置づけ、環境、連動するシステムについて理解する。					○	
						要	各システム説明資料						○	

【成果物説明】
担当者ごとの保守・開発に必要なスキルや知識レベルを可視化し、適切な人材配置・教育計画・引継ぎ・リスク管理を実現する。

【項目例】
・大・中・小分類
・配慮すべき資料
・目的や内容
・重要度

a9.システムロードマップ



【成果物説明】
システムや業務の保守・開発における短期・中期・長期の計画と、各期間で必要となるドキュメント改修範囲を可視化し、計画的な運用・改善・リスク管理を実現する。

【項目例】
・イベント名
・計画の矢羽根
・システム一覧
・課題名称
・備考

3. まとめ

(1) OLECが担う役割

① 人材育成の入口業務を整理・定型化でき、引き継げる状態をつくる。
② 個人依存から組織運用へ改善を積み上げられる基盤をつくる。

最低限の整理があることで、属人化を前提としない運用が可能になる。

(2) OLECはゴールではない

OLECは完成形ではなく、「個人が守る」保守から「組織で守る」保守へ
移行するための最低限の土台です。

OLECをベースに自社に役立つ形に育ててほしい。

人材育成・属人化防止・自動化への入り口

3. まとめ

(3) メンバーから一言

がしやま

ＯＬＥＣは　保守の歴史を　変えるはず

やまらぁ

多種多様なシステムを守る我々システム会社の集約知を可視化できたと思います。

なべさん

私自身が感じていた引継ぎの課題を解消できたと考えています。

もっちゃん

OLECを導入して引継ぎの質と効率を高めていきます。

あっちゃん

運用に悩んだときにはまずはＯＬＥＣを見て指針を得ようと思います。

JUAS 一般社団法人 日本情報システム・ユーザー協会
Japan Users Association of Information Systems

生成AI導入と開発現場での活用

～ 新技術導入への心構えと実践的アプローチ ～

2026年4月9日

分科会④「新技術を知り、備えよう！（チームひろあき）」

JUAS

目次

序章

はじめに ～ 新技術導入の夢と現実 ～

「技術は日進月歩で進化する」と言われてきたが、昨今は「秒針分歩」といったスピードで 指数関数的に進化している。
システム開発・保守 においても、新技術 を 取り入れ、作業をより効率的に進めることが求められている。

新技術 を 導入することによるメリットとして、以下のようなことが挙げられる。

コスト最適化とROI向上	新技術導入による作業効率化により開発/運用コストを削減し、投資回収期間を短縮
リスク低減と品質向上	最新技術によるセキュリティ強化・障害予防で、事業継続性を確保
競争優位性の確保	市場変化に迅速対応し、サービス価値を高めることで差別化を実現
モチベーション向上と人財の定着	新技術への挑戦機会が、学習意欲とエンゲージメントを高め、離職率低下に寄与

一方で「新技術 導入」に 夢を抱き過ぎて、失敗してしまうケースも見受けられる。
本ガイドライン では、生成AI を 題材に、新技術 を システム開発・保守 に どのように取り入れていくか？ を 提示する。

はじめに ～ 新技術導入でありがちな罠 ～

新技術導入 において “過度な期待”は禁物

新技術 導入 の 際、「過度な期待→現実の壁→幻滅と撤退」に 陥りやすい傾向がある。
同じような失敗を繰り返すと、「新技術 導入」そのもの を 避けるようになる弊害も出てくる。

過度な期待

いきなり
利用開始

幻想を抱く

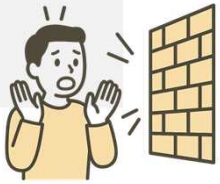
- ✓ 劇的な効率化
- ✓ 何でもできる✨

現実の壁

「使えない」
と判断

課題に直面

- ✓ 精度不足
- ✓ セキュリティ懸念



機会喪失

- ✓ 活用自体を放棄
- ✓ 機会損失

はじめに ～ 新技術導入を成功させるためには ～

夢(理想) と 現実(技術的限界) を 明確に区別することがポイント

新技術 は 問題を解決するための手段 である。“新技術導入” が 目的ではない。
「その新技術は、問題を解決するために適しているか？」という視点が大切。

小さく利用

特徴を知る

夢(理想) と
現実(技術的限界) を
明確に区別する

使いどころを考える 利用環境を整える

利用拡大

- ✓ 人間 と 技術 の 役割分担
- ✓ 業務で使える 環境 や 運用ルールの整備

新技術 導入効果

- ✓ 新技術 導入による
  - ・生産性 向上
  - ・品質向上
  - ・コスト削減
- ✓ 新しいことに挑戦でき メンバーのモチベーション向上

本成果物の位置づけ

生成AI活用の検討に必要な“観点を整理”する

本ガイドライン では　導入対象の新技術 を “生成AI” とする。

「新技術 導入 を 成功させるために」に 記載した【特徴を知る】→【使いどころを考える】の 流れに従って、
“生成AI” を 導入・活用 する際に必要な “観点” を整理する。

本ガイドライン は こういうもの	本ガイドライン は こういうもの ではない
導入判断、活用設計 で 検討すべき観点を整理する	特定ツール の 操作手順 や 各社固有の規程・契約に踏み込んだ結論

自組織 で 生成AI 導入を検討する際 に
「自組織 では、この観点 は、どういう状況だろうか？」→「だとしたら、こういうふうに 生成AI を活用できるのは？」といった
チェックリスト的に活用いただきたい。

本成果物の目的 ～ 提供するもの ～

生成AI導入時に必要な「検討ポイント」を QCD観点で体系化

- 生成AI の 特徴を知るための材料を提供　⇒「1. 生成AI の 特徴を知ろう」
- 生成AI の 使いどころ を検討するための観点を提供
  - 人間と生成AIの特徴を整理　⇒「2. 人間と生成AIの役割分担」
  - 開発工程ごとに、生成AIを活用する際の検討ポイントをQCD観点で体系化 ⇒「3. 使ってみよう！」
  - 開発工程のQCD課題の解決における 生成AI 活用シーン　⇒別添「工程別課題とAI-人間の役割」

本成果物の想定読者

開発・保守の実務者から PM/PMO,マネジメント層,内部統制担当まで幅広く想定

開発・保守 の 実務者、QA担当	工程ごと の 生成AI の使いどころ や 注意点の把握
マネジメント層	生成AI 導入判断、体制構築、リスク把握
内部統制 担当	セキュリティ、ガバナンス観点での確認

本成果物の扱い

生成AI技術は加速度的に進歩しており、本成果物も継続的な更新が必要

- 本成果物の内容は現時点での知見に基づくものである
- 生成AI の 将来の技術進化や法規制等に応じて、継続的な更新が必要となることを前提とする

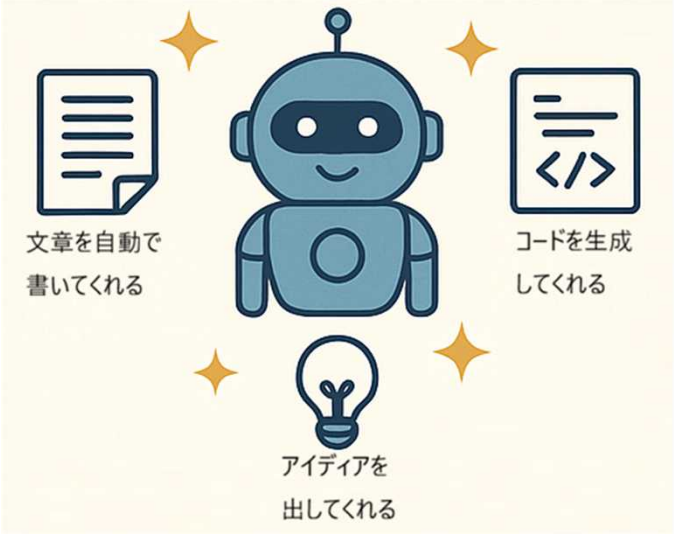
本章

1. 生成AIの特徴を知ろう

1-1. 導入：生成AI活用への期待と現実①

生成AIを業務にどう生かすか？ 多くの人が「何でもできる」と期待

- 生成AIの登場は、業務のあり方を根本から変える可能性を秘めている
- 「文章を自動で書いてくれる」「コードを生成してくれる」「アイデアを出してくれる」といった機能に、多くの人が期待している
- 特に業務効率化や人手不足の解消を目指す現場では、「生成AIがすべてを肩代わりしてくれるのでは？」という期待が高まっている



1-1. 導入：生成AI活用への期待と現実②

夢を持つことは大切だが、過度な期待はコスト増・遠回りにつながる

- 「何でもできる」と思い込むと、生成AIに任せすぎてしまい、バグの多発やセキュリティ問題を引き起こしてしまう
- 生成AIに任せるべきでない業務まで任せてしまうと、かえって非効率になる。その結果、修正対応や再検討に時間とコストがかかり、導入効果が薄れてしまう

⚠ 品質の低下リスク

自動生成されたコードがバグだらけになり、後工程での修正コストが肥大化する恐れがある。

🔒 セキュリティ問題

脆弱性を含むコードが生成されたり、機密情報が不用意に扱われたりするリスクがある。

夢は大切、でも現実を見据える必要がある。

1-2. 生成AI関連技術の理解サマリ

生成AIの技術的な仕組み

- 生成AIを業務に活用するには、「何ができるか」だけでなく「どう動いているか」を理解することが不可欠
- 構造を知ることで、より精度の高い指示が可能になり、誤解や失敗を防ぐことが可能

技術用語	概要と役割
LLM（大規模言語モデル）	大量のデータ学習に基づく言語モデル。文脈（Context）を理解し、確率的に次に来る言葉を予測する。
プロンプトエンジニアリング	AIに適切な指示を与え、期待する精度のアウトプットを引き出すための技術。
RAG（検索拡張生成）	外部知識ベースを検索し、その情報をLLMに与えることで、正確で最新の回答を生成させる手法。
AIエージェント	自律的にタスク計画・実行を行う高度な機能。

1-2-1. 生成AI(LLM)の動作原理

まずは、生成AIの核となるLLMの動作原理について知る

■ 確率論的生成

生成AIは人間のように**「意味」を思考しているわけではない。**
膨大なテキストデータを基に、統計的に「どのような文脈でどのような言葉が使われるか」を学習している。
生成にあたっては、文脈上もっともらしい単語を**確率的に予測して**繋げている。

次単語予測のイメージ

“日本の首都は“　→　[東京]

候補：東京(93%)、京都(5%)、奈良(2%)

※ 意味を理解しているのではなく、「東京」が続く確率が高いと計算している。

1-2-2. 生成AIの特性

生成AIには、ポジティブ/ネガティブの両面の特性があることを知っておく

ポジティブな特性

- **自然な対話と回答**
  人間と話すような言葉で質問・指示ができ、言葉を理解しているように適切に回答を返してくれる。
- **多様なタスクの実行が可能**
  文章要約・生成、翻訳、プログラミングなど多様なタスクを実行できる。

ネガティブな特性

- **幻覚(ハルシネーション)の可能性**
  事実とは異なる情報や存在しない情報を、「知ったかぶり」して結果を出力する可能性がある。
- **著作権侵害等の法的リスク**
  無意識に他の著作者の特徴を用いて生成し、著作権を侵害するなどのリスクがある。

1-2-3. 生成AI関連キーワード ～ プロンプトエンジニアリング ～

生成AIへの理解しやすい指示を考える必要がある

- 生成AIは、論理的にテキストを処理するツール
  → 人間相手のような曖昧な指示には、抽象的で表面的な回答になりがち
- 生成AIが正確で理解しやすい形に表現を工夫する必要がある
  → これがプロンプトエンジニアリング

手法(例)

- マークダウン記法
  「#」、「-」、「*」といった記号を用いて、階層やリスト構造にすることで、生成AIに理解しやすくする。
- 出力形式、条件、禁止事項の記述
  これらを明確に記述することによって、人間が求める回答を返してくれる確度が高くなる。
- 思考の連鎖(Chain Of Thought)
  複数ステップに分けて生成AIに考えさせることで、複雑な問題や多段階の処理が必要なタスクに正確な回答を出す確度を上げる。

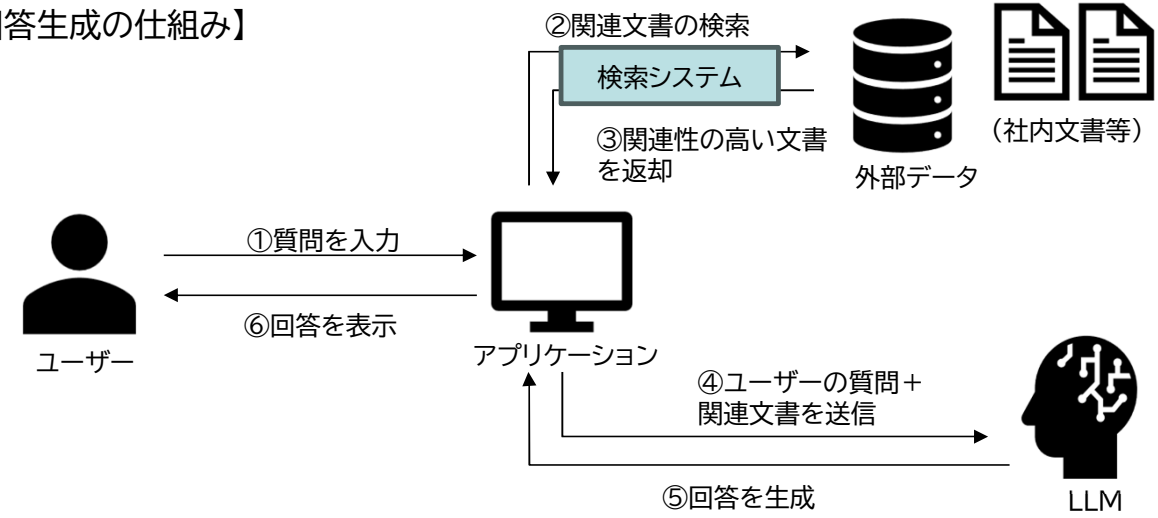
1-2-4. 生成AI関連キーワード ～ RAG ～

専門性のある知識をいかに正しく回答させるか

生成AI単体では、2つの課題がある。
ハルシネーション ・・・ 学習データにない情報について、もっともらしく嘘の回答をしてしまう
情報の問題 ・・・ 学習時点以降の新しい情報、専門的な情報、企業固有・個別の情報は学習されてない

これらに対応する仕組みとしてRAG(Retrieval Augmented Generation)がある。

【RAGを使った回答生成の仕組み】



1-2-5. 生成AI関連キーワード ～ AIエージェント ～

設定された目標や目的に向かって自律的に行動する仕組み

生成AIは、基本的にはユーザーの質問に都度回答するのみ。

必要な情報を収集し、自律的に計画を立て、行動する仕組みがAIエージェント。
定型的な処理を実行する AIワークフローとは異なり、状況に応じた分岐への対応ができる。

さらなる進化形として、複数のAIエージェントを連携させてタスクを遂行するマルチエージェントがある。

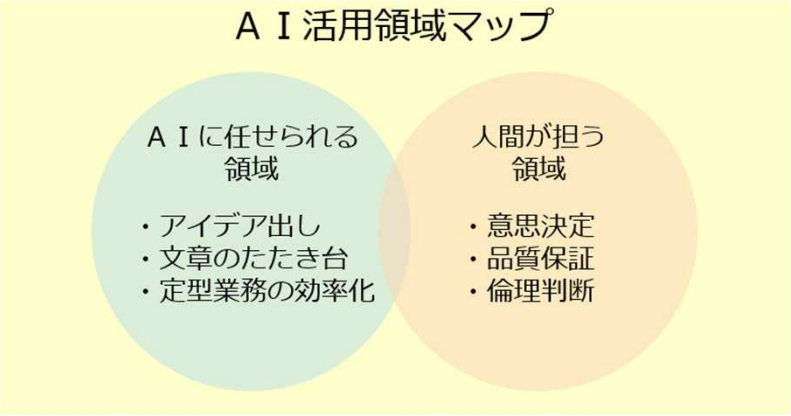
＜開発、保守・運用現場で期待されるAIエージェント活用例＞

コード生成・レビュー	要件定義や設計情報を基に、コード生成やレビューを実施する。
インシデント予測・対応	障害の兆候検知から一次対応・報告まで一気通貫で対応案を提示する。
ナレッジ管理・問い合わせ対応	過去対応履歴やFAQを基に、質問者に回答する。
テスト自動化	テストケース生成、結果分析をする。
セキュリティ監視・脅威検知	ネットワークやアプリの挙動をリアルタイム監視し、異常を検出する。

2. 人間と生成AIの役割分担
～使いどころを考える①～

2-1. 生成AIの適材適所

- **生成AI を活用するべき領域（誤情報リスクが低い領域）**
  - アイデア出し（創造性の補完）：多様な視点を短時間で提示できるため、ブレインストーミングの効率化に有効
  - 文章のたたき台（事務処理力の活用）：初稿や雛形を素早く生成でき、修正や肉付けを人間が行うことで品質を確保できる
  - 定型業務の効率化（生成AIの安定処理能力）：繰り返し作業や定型処理を自動化することで、人間はより付加価値の高い業務に集中できる
- **人間の判断が必須な領域**
  - 意思決定　：生成AIは選択肢を提示できるが、最終的な判断には文脈理解や責任が伴うため人間が不可欠
  - 品質保証　：生成物の正確性や信頼性を担保するには、人間によるレビューや検証が必要
  - 倫理的判断：公平性や社会的影響を考慮する領域は、AIではなく人間の価値観と責任で対応すべき



・生成AIは「自動化」ではなく「増強（Augmentation）」

・生成AIに任せるのは“作業”、人間が担うのは“判断”

2-2. それぞれの強み

人間と生成AI、それぞれの強みを整理

それぞれの特性を生かすため、各項目において、人間と生成AIの強みを下記の通り整理

	人間の強み	生成AIの強み
判断力	複雑な状況を総合的に理解し、文脈を踏まえた意思決定が可能	大量データを処理し、統計的な傾向を提示
倫理・価値観	公平性・倫理観・社会的責任を考慮できる	ルールに従った処理を実施
創造性	新しい発想や直感的なアイデアを生み出せる。	過去データからパターンを抽出し、既存の組み合わせを生成
事務処理力	不確実性や例外的状況に対応できる。	定型業務や繰り返し作業を安定して高速に処理
コミュニケーション	感情やニュアンスを理解し、人間関係を構築できる	迅速に情報を整理し、形式的なアウトプットを提供

2-3. 人間の役割

現在の生成AIの特性を踏まえて、人間の役割を整理

生成AIにより任せられる部分が増えても、人間にしか行えない、または行うべき内容は残る。

	分野	生成AIの特性	人間の役割	具体例
①	方向性(意味・目的・優先順位)の設計	「手段の最適化」はできるが、自ら意思や目的を持って動き出せない。	価値定義、事業目標・プロダクトの方向性(目的設定)、コスト・品質・スピードなどの優先度付け。	「売上伸長」「顧客体験向上」などの目的設定、「セキュリティ最優先」「納期厳守のための機能削減」などの優先順位判断。
②	現実の複雑さ・例外への対応	確率や平均値で動くが、個別の例外対応や文脈依存の判断は不得手。	特殊事情・文脈を考慮した臨機応変な対応。	本番障害発生などで、業務影響を最小限する暫定措置や随時変化する現場状況を踏まえた柔軟な対応。
③	最終承認・リスク判断	選択肢は提示するが、最終決定権は持っていない。	倫理観や責任を持ち、仕様承認・リスク許容などを最終判断。	「リスクを許容して出荷する」など、経営やPMの最終判断。
④	心理の理解	共感しているように見えるが、実際の感情や経験ではない。	顧客の喜びや不安、チームの温度感などを読み取る感情・知性。	「論理的に正しいUIだが、冷たい感じがする」などの違和感の検知。
⑤	組織の価値観・文化を形成	文化を模倣できても、自ら創造はできない。	経験・対話を基に、チーム・組織の価値観・判断基準・行動規範を構築。	「安定稼働第一」、「顧客価値優先」といった文化の醸成。

2-4. 人間と生成AIの役割分担

人間と生成AIで強みを補完しあうプロセスを設計

- 意思決定やレビューなどにおいては、人間が積極的に関与することになると想定
- そのうえで、人間と生成AIでお互いを補完して協調していくプロセスを志向したい

1. Trigger（始動）

人間による
明確な指示（プロンプト）

※意図を持てるのは人間のみ

→

2. Execution（実行）

生成AIによる
高速な案の作成

※あくまで「案」の提示

→

3. Judgment（判断）

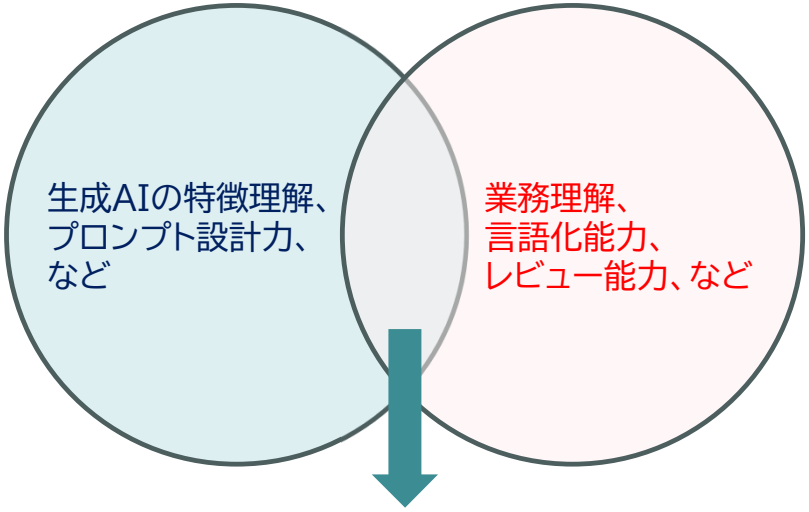
人間による
最終レビュー・採否判断

※品質責任の所在

2-5. 生成AI活用のために取り組むべきこと

活用を推進していくために人材を育成していく必要がある

- 今後、より生成AI活用を推進していくには、「生成AI をつかいこなせる人」を育成していく必要がある
- 生成AIに正しいInputを与えられ、得られたOutputについて評価・修正ができることが求められる ➡ **スペシャリスト育成**



重なる面積が大きい人ほど
生成AIを使いこなせる人材と言える

求められるスキル	内容
プロンプト設計力	要件を構造化し、生成AIに伝わる形に変換する力
技術的リテラシー	生成AIの限界・ハルシネーションの理解
ドメイン知識	業務文脈を理解し、出力の妥当性を判断
レビュー力	生成AI出力の品質を評価し、修正できる力

2-6. 生成AI活用のための心得

生成AIを活用していく人材育成における心得を整理

一、現場の「知恵」を言葉にする
生成AIは教科書通りの回答は得意だが、自社特有の「現場のコツ」や「ベテランの勘」は知らないので、現場に深く入り込み、誰も言語化できていない「独自の知恵」を丁寧に聞き出し、AIが活用できる形に整理・集約すること。

二、生成AIの回答を疑い、自らの「経験」で決断する
生成AIはハルシネーションを起こすことがあるので、回答を妄信せず、自社の置かれた状況や顧客の特性などと照らし合わせ、「最後は人間が責任を持つ」という覚悟で判断を下すこと。
この「責任ある決断」こそが、生成AIには代えられない人間の価値。

三、「次の仕事」を創り出す
生成AIによる効率化で生まれる時間・余力を企業の成長に繋げるため、現場の新たな悩みを見つけ、次の一手を提案すること。
自ら「価値のある仕事」を定義し続けることで、組織の未来を切り拓く先導役となれる。

3. 使ってみよう！
～使いどころを考える②～

3-1. 開発での使用（前提1）

「業務システム開発・保守におけるLLM活用」を対象とする

対象例	システム開発・保守 の 以下のような作業に 生成AI を 活用する • 要件/設計文書作成支援 • テスト観点・ケース生成支援 • コード生成・リファクタ支援 • 運用保守の分析支援
対象外	以下のような作業 は 対象外とする • 独自LLM 構築、学習 • 大規模ML Ops設計 • 予測系ML 開発そのもの （理由）品質管理の構造が異なるため、別途扱うほうが適切

本書では、以下の理由により、ウォーターフォール型 による開発・保守作業を対象として論ずる。

- 工程別に品質論点を整理しやすい
- JUAS参加企業の多くが、情報システム部門、あるいは情報システム子会社として企業の基幹システムを扱っているケースが多く、そういったシステム開発保守はウォーターフォール型の開発が主流であるため

※「工程定義」は次頁参照。工程名は各社によって異なる可能性があるので、適宜、読み替えていただきたい

3-2. 開発での使用(前提2)~ 本ガイドラインにおける工程定義 ~

本ガイドラインの工程名	工程名	想定している 主な作業	共通フレームワーク 2013	JUAS ソフトウェア・メトリックス調査	IPA ソフトウェア開発分析データ集
企画構想	企画・構想	• 業務課題／目的の整理(なぜやるか) • 対象範囲の定義(どこまでやるか) • 概算コスト／工期／体制の見立て • リスク洗い出し、実行可否判断 • 投資判断に必要な材料(効果・優先度)の整理	2.1 企画プロセス	(対象外)	(対象外)
プロジェクト計画	プロジェクト計画				
要件定義	要件定義	• 業務要件／機能要件／非機能要件の整理 • 現状業務(As-Is)／あるべき姿(To-Be)の整理 • 制約条件(法令・セキュリティ・運用条件等)の明確 • 要求の優先順位付け、合意形成 • 要件の検証観点(受入条件)の整理	2.2 要件定義プロセス 2.3.2 システム要件低異議プロセス 2.3.3 システム方式設計プロセス 2.4.2 ソフトウェア要件定義プロセス	要件定義	(対象外)
基本設計	基本設計	• 要件を「システム全体の構成と仕様」に落とし込み、実装の方針を決める	2.4.3 ソフトウェア方式設計プロセス	設計～統合(結合)テスト	基本設計
詳細設計	詳細設計	• 実装できるレベルまで処理・画面・データ・IFを具体的に決める	2.4.4 ソフトウェア詳細設計プロセス		詳細設計
実装	実装・単体テスト	• コーディング、モジュール単位のテスト	2.4.5 ソフトウェア構築プロセス		製作
テスト	結合テスト	• 作った部品(モジュール)同士をつないで、連携が正しく動くかを確認するテスト	2.4.6 ソフトウェア結合プロセス 2.4.7 ソフトウェア適格性確認テストプロセス 2.3.5 システム結合プロセス		結合テスト
	システム総合テスト	• システム全体が要件どおり動くかを、通しで確認するテスト	2.3.6 システム適格性確認テストプロセス		総合テスト(ベンダ確認)
	受入テスト	• 業務の観点で、システムが要件どおり使えるかを本番想定のシナリオで確認する	2.3.8 システム受入れ支援プロセス 2.4.9 ソフトウェア受入れ支援プロセス	ユーザー総合テスト	(対象外)
保守・運用	保守・運用	• 監視、障害対応、問合せ対応、容量・性能管理 • 変更、リリース	2.6 保守プロセス 3.1 運用プロセス	(対象外)	(対象外)

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別① ～

開発工程別の生成AIの活用

ウォーターフォール型開発ライフサイクルにおけるAI活用の全体像

※詳細は別添「工程別課題と生成AI-人間の役割」参照

工程	活用
企画構想・プロジェクト計画	生成AIは多角的な視点を提供し、人間の意思決定を支援する壁打ち相手として強力に機能する。情報の出どころなどについては注意が必要。
要件定義	顧客との対話が多く、曖昧さが残りやすい要件定義工程において、生成AIは議事録の整理や要件の構造化や整合性チェックで強力な支援ツールとなる。最終成果物は人間のレビューが必要。
開発（実装）	ドキュメント作成量が多くなる設計工程において、生成AIはドラフト作成やレビューの一次対応を自動化し、生産性を向上させる。人間による妥当性チェックが必要。
テスト	膨大なパターンの洗い出しが必要なテスト工程において、生成AIはテストケース生成の効率化に貢献するが、業務知識が求められるテストには限界あり。
運用・保守	障害対応の迅速性や維持費削減が求められる保守・運用工程において、生成AIはログ解析による初動対応の高速化や、問い合わせ対応の自動化に貢献する。診断結果は仮説として取り扱うなどの注意が必要。

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別② ～

企画構想・プロジェクト計画における評価概要

生成AIは多角的な視点を提供し、人間の意思決定を支援する壁打ち相手として強力に機能する。
情報の出どころなどについては注意が必要。

評価項目	評価
主なQCD課題	・ビジネスニーズ整理の複雑化・コンサルティング費用などのコスト増大 ・プロジェクト計画（見積、スケジュール、体制）策定の属人化と負荷
生成AIの活用シーン	・意思決定支援:生成AIに経営層、営業部長、開発部長など複数の人格を与えて議論させ、多角的な視点や盲点を洗い出す ・アイデア創出: 新規システム化のアイデアを幅広く提案させる ・計画策定支援: 過去の社内実績データを学習させ、プロジェクト計画、リスク、対策案のたたき台を作成させる
メリット （QCDへの影響）	品質 (Quality):多様な視点を取り入れることで、バイアスが排除され、意思決定の質が向上する コスト (Cost):初期検討やアイデア出しにかかるコンサル費用や人件費を削減できる可能性がある 納期 (Delivery):意思決定や計画策定のプロセスを高速化する
デメリット （課題・懸念点）	・生成AIの提案はあくまで案であり、100%の正しさを保証するものではない ・妥当な見積もりや開発スケジュールなど、複雑な計画策定の精度には限界がある ・精度の高い学習（社内実績の投入など）には相応のコストと準備が必要となる
人間と生成AIの役割分担・対策	生成AIの役割: ・情報収集、多角的な論点や盲点の洗い出し、アイデアの壁打ち相手、計画案のたたき台作成 人間の役割: ・生成AIの提案内容を評価し、ビジネス的な観点から最終的な意思決定を行う ・生成AIに適切な役割（ペルソナ）を与えるプロンプトを工夫する ・過去の社内実績など、質の高い情報をINPUTとして生成AIに提供する

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別③ ～

要件定義における評価概要

顧客との対話が多く、曖昧さが残りやすい要件定義工程において、生成AIは議事録の整理や要件の構造化や整合性チェックで
強力な支援ツールとなる。最終成果物は人間のレビューが必要。

評価項目	評価
主なQCD課題	・要件の曖昧さ、漏れ、関係者間の認識齟齬 ・ヒアリング内容が発散し、収束に時間がかかる ・現行仕様の難解さやドキュメント不足による調査コスト増大
生成AIの活用シーン	・議事録・会話ログの自動要約: 顧客ヒアリングの内容を自動で要約し、要件を抽出する ・要件整理・構造化: 抽出した要件を整理し、テンプレートを用いて構造化する ・要件レビュー: 人間が作成した要件リストをインプットし、不整合や抜け漏れの候補を検出する ・ユースケース・画面イメージの生成支援: 要件から基本的なユースケースやホワイトボードの手書きレベルの図から画面イメージを生成する
メリット （QCDへの影響）	品質 (Quality):要件の抜け漏れや曖昧さを早期に発見して手戻りを削減したり、開発部門とユーザー部門の認識齟齬を防止する コスト (Cost):議事録作成やドキュメント整理にかかる工数を大幅に削減する 納期 (Delivery):要件確定までの期間を短縮する
デメリット （課題・懸念点）	・生成AIが業務の文脈や特有のニュアンスを誤解するリスクがある ・自動抽出だけでは、要件の漏れや曖昧さが完全に払拭されるわけではない
人間と生成AIの役割分担・対策	生成AIの役割: ・膨大な会話ログからの情報整理、要件のリストアップ、一次的なレビュー 人間の役割: ・生成AIが整理した内容を確認・修正し、業務文脈に沿った正確な要件定義書として完成させる ・生成AIのレビュー結果を参考に、より深いレベルでのレビューを実施する ・最終的な要件について、ユーザーと合意形成を行う

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別④ ～

設計(基本設計・詳細設計)局面における評価概要

ドキュメント作成量が多くなる設計工程において、AIはドラフト作成やレビューの一次対応を自動化し、
生産性を向上させる。人間による妥当性チェックが必要。

評価項目	評価
主なQCD課題	・設計ミスや非効率な構造の作り込み ・ドキュメント作成・レビューにかかる工数増大 ・現行仕様の難解さやドキュメント不足・実装との乖離や設計漏れ
生成AIの活用シーン	・設計案の自動生成: 画面設計案、API設計テンプレート、クラス図、ER図、DB設計などを自動で提案・生成する ・設計書の一次レビュー: 用語の不統一、規約違反、整合性チェックなどを自動で行う ・要件との突合: 要件定義書と設計書を照合し、要件の反映漏れがないかチェックする
メリット (QCDへの影響)	品質 (Quality):設計の整合性チェックや規約遵守により、初期段階での品質を向上させたり、セキュリティ要件の見落としなどを早期に検知できる可能性がある コスト (Cost):設計書作成やレビューの工数を削減する 納期 (Delivery):設計書のドラフト作成を高速化し、作業スピードを向上させる
デメリット (課題・懸念点)	・複雑な業務ロジックや、性能・セキュリティを高度に考慮した設計は苦手 ・提案内容の妥当性を人間が検証する必要がある ・生成される設計の精度は、インプットを与える設計者のスキルに依存する ・ビジュアル面の調整(Excelの体裁など)が苦手な場合がある
人間と生成AIの役割分担・対策	生成AIの役割: ・設計書のたたき台作成、一次レビューアとしての役割 人間の役割: ・生成AIの生成物を必ずレビューし、内容の妥当性を判断する ・複雑なロジックや非機能要件に関する設計は人間が主導する ・生成AIが読みやすいフォーマット(例:Excelのセル結合をなくす)でインプットを準備する

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別⑤ ～

開発(実装)局面における評価概要

コーディング作業が中心となる実装工程では、生成AIによるコード生成が大幅な工数削減と納期短縮に貢献するが、
非機能要件の実装には注意が必要。

評価項目	評価
主なQCD課題	・バグの混入、非効率なコードの生成 ・複雑な既存ソースの改修に伴うデグレード ・開発者のスキル差による生産性のばらつき
生成AIの活用シーン	・コード自動生成: 設計書や自然言語の指示から、ソースコードを自動生成する ・テストコード自動生成: 実装コードに対応する単体テストコードを生成する ・リファクタリング提案: 非効率なコードを検出し、改善案を提案する ・ペアプログラミング:生成AIと対話しながらコーディングを進め、教育や育成にも活用する
メリット (QCDへの影響)	品質 (Quality):コーディング規約の遵守や、一定レベルのコード品質を担保しやすくなる コスト (Cost):実装工数を大幅に削減できる可能性がある 納期 (Delivery):納期短縮への貢献度が最も高い工程の一つ
デメリット (課題・懸念点)	・特に非定型的な処理において、バグが混入する可能性がある ・非機能要件の実装(セキュリティや性能)を考慮したコード生成はまだ発展途上であり、保守性の低下や想定外の脆弱性を生むリスクがある ・保守性(コメントの省略・不明瞭さなど)に課題がある場合が多い。また開発者のスキル低下が課題として上げられる
人間と生成AIの役割分担・対策	生成AIの役割: ・定型的なコードの生成、テストコードの作成 人間の役割: ・セキュリティ、性能、保守性といった非機能要件をプロンプトで明確に定義し、生成AIに指示する ・生成AIが生成したコードをレビューし、特にセキュリティ上のリスクがないか検証する ・人間が見やすいようにコメントを追記・修正する

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別⑥ ～

テスト局面における評価概要

膨大なパターンの洗い出しが必要なテスト工程において、生成AIはテストケース作成の効率化に貢献するが、業務知識が求められるテストには限界もある。

評価項目	評価
主なQCD課題	・テストケースの漏れ、網羅性の不足・テストケース作成、テストデータ準備に時間がかかる ・品質担保のためにテストが肥大化し、コストと期間が増大する
生成AIの活用シーン	・テストケースの自動生成: 要件定義書や設計書から、テストケースを網羅的に生成する ・テストデータの自動生成: 指定されたパターンに基づき、大量のテストデータやマスキングデータを生成する ・ログ解析によるバグ検出: テスト実行時のログを解析し、異常な挙動やバグの原因を特定する
メリット （QCDへの影響）	品質 (Quality):人間では見落としがちなパターンを洗い出し、網羅性を向上させる コスト (Cost):テスト設計、データ作成、実施にかかるコストを削減する 納期 (Delivery):テスト期間の短縮に貢献する
デメリット （課題・懸念点）	・網羅性に課題が残る。特に境界値分析や例外処理の見落としリスクがある ・業務フローを深く理解する必要があるシナリオテストの自動生成は難しい ・ログ解析では、原因が階層的に深い場合、表層的な解析に留まる可能性がある
人間と生成AIの役割分担・対策	生成AIの役割: ・テストパターンの洗い出し、大量のテストデータ作成 人間の役割: ・専門家（スペシャリスト）が、AIが生成したテストケースやパターンの妥当性を評価する ・生成AIにインプットするテストパターンやユースケースを人間が明確に定義する ・業務シナリオテストなど、高度な判断が必要なテストは人間が設計・実施する

3-3. 日本の開発現場と生成AI活用シーン ～ 各工程別⑦ ～

運用・保守局面における評価概要

障害対応の迅速性や維持費削減が求められる保守・運用工程において、生成AIはログ解析による初動対応の高速化や、問い合わせ対応の自動化に貢献する。診断結果は仮説として取り扱うなどの注意が必要。

評価項目	評価
主なQCD課題	・障害発生時の即時検知と迅速な対応の遅れ ・問い合わせ対応にかかる人件費や監視負荷の増大
生成AIの活用シーン	・障害ログの自動解析: 大量のログデータから障害の予兆を検知したり、発生した障害の原因箇所を特定したりする ・FAQ対応の自動化: ユーザーからの定型的な問い合わせにチャットボットが自動で応答する ・影響範囲分析: コード変更が他にどのような影響を及ぼすかを分析する
メリット （QCDへの影響）	品質 (Quality):障害の早期発見や予兆検知により、システムの安定稼働に貢献する コスト (Cost):サポートデスクの人件費や、24時間365日の監視負荷を削減する 納期 (Delivery):障害対応の初動が早くなり、復旧までの時間を短縮する
デメリット （課題・懸念点）	・生成AIによる誤診断が、逆に対応の遅延を引き起こすリスクがある ・診断結果の妥当性には懸念が残り、初期対応の精度には限界がある ・複雑な障害に対しては、根本原因の特定まで至らない場合がある
人間と生成AIの役割分担・対策	生成AIの役割: ・アラートの検知、ログの一次解析、FAQの自動応答、障害対応の材料提供 人間の役割: ・生成AIの解析結果を基に、最終的な原因特定と復旧作業を行う ・生成AIが出力する提言を許容しつつも鵜呑みにせず、専門家として判断を下す ・生成AIが解析しやすいように、可読性・保守性の高いコードを開発時に作成するよう生成AIに指示するスキルが求められる

終章

QCD向上に向けて生成AIは、どのように扱うべきか

本成果物では、生成AIの特徴や限界を整理し、工程別に検討すべき観点を示してきた。

生成AIは「使う／使わない」の二択ではない。

過度な期待は品質低下や手戻りによるコスト増を招き、過度な警戒は生産性向上の機会を失わせる。

重要なのは、

生成AIを「管理された支援ツール」として位置づけることである。

- 判断・責任・品質保証は人間が担う
- 生成AIは作業支援・視点拡張・一次整理を担う
- QCDへの影響を意識しながら段階的に活用する

生成AIは自動化装置ではなく、**業務を増強するための技術である。**

生成AIと人間の役割分担の重要性

生成AIは、人間の判断を代替するものではない。

しかし、**人間の可能性を拡張する力**を持つ。

生成AIの価値を引き出すためには

- 限界を理解すること
- 役割を明確にすること
- Qualityを起点にQCD全体を設計すること

が不可欠である。

人間が責任を持ち

生成AIと適切に協調できた時に

現実的かつ持続的な業務改善ができる。

まずは限定的な工程・限定的な用途から、小さく検証を始めることを推奨する。

Appendix

工程定義 参考資料

参考資料	参照先
JUAS ソフトウェア・メトリックス調査	https://juas.or.jp/library/research_rpt/swm/
IPA ソフトウェア開発分析データ集	https://www.ipa.go.jp/digital/software-survey/metrics/index.html
IPA 共通フレームワーク 2013	https://www.ipa.go.jp/publish/secbooks20130304.html
IPA 共通フレームワーク 2013 プロセスへのお時間軸へのマッピング例	https://www.ipa.go.jp/publish/secbooks20130304.html 内 図表類 一括ダウンロード より 巻末_図4-25_原図_プロセスの時間軸へのマッピング例 [Excel形式] （本図はマッピングの一例を表すもの。共通フレームが、本図に示す工程と作業の対応体系を推奨している訳ではない。）

工程別課題と生成AI-人間の役割①

工程	QCD区分	QCD課題	AI活用案	人間の役割		必要な準備・取り組み
企画構想	Q	観点の過不足、盲点・バイアス、ユーザ視点の不足が品質を歪める。抽象的ゴールでは的外れ提案が混ざりやすく、意思決定の根拠が弱くなる。	AI人格を分けた多角評価(AI経営層/営業/開発)で論点を広げ、現場・ユーザ視点も取り込む。論点要約と代替案列挙で意思決定材料を増やす。	AIの提案を鵜呑みにせず、最終判断の責任を持つ。組織の価値観や業務,事業環境、経営戦略、AIの特性を深く理解し、多角的な視点で意思決定を下す。	AI出力の妥当性を点検し、組織の価値観に沿うかを裁定する。評価基準を明示し、最終判断を下す。	経営層や現場の声、組織のミッション・バリュー、事業環境の変化を定期的に情報収集し、自分の言葉で説明できるようにまとめておく。AIの仕組みや限界、最新情報をウォッチし、柔軟な発想や新しい視点を持つことを意識する。
企画構想	C	コンサル料金、レビュー対応、指摘対応など上流でのコストが膨らみやすい。学習データ準備にも時間・費用がかかる。	意思決定支援とアイデア出しをAIで反復し、初期案作成・比較を高速化。学習コストは最小構成から始め、段階的に拡張。		学習範囲の費用対効果を見極める。重要領域に投下、不要領域は割り切る。投資判断とガバナンスを担う。	
企画構想	D	論点の発散で合意が遅れ、後続工程がズレ込む。意思決定会議が増えて日程が圧迫される。	要約・論点整理・賛否論拠の自動抽出、落としどころの自動提案で合意形成を加速。比較表、自動議事要約で回覧を迅速化。		意思決定プロセス自体を短縮する業務設計。決定期限と採択基準を設定し、AIの提案を使って迅速に合意をとる。	
要件定義	Q	要件ヒアリングが発散し、漏れ・曖昧さ・認識齟齬が生じる。現行業務が不明瞭だと品質が不安定になる。	顧客ヒアリングの自動要約、登場人物を分けた擬似ディベート(AIユーザ部門×AI開発部門)。業務マニュアル/QAを学習してユースケース自動生成、業務フローの可視化支援。	・AIが自動生成した要件や論点をたたき台として活用しつつ、現場ヒアリングや経営層の意向、システムの制約条件などを人間が総合的に調整・判断する。 ・過去の障害事例や現場の運用実態をもとに、リスクや障害発生時の影響範囲を具体的に評価し、優先順位や対応方針を明確に決める役割を担う。	AI生成物を叩き台として文脈補正と例外の明文化を行う。用語集と定義域を確定し、認識合わせを主導する。	現場やユーザーの業務実態・課題を普段からヒアリングし、業務フローや用語のズレ・例外パターンを形式知化しておく。過去の障害やトラブル事例をストックし、要件の抜け漏れやリスクを見逃さない習慣を持つ。
要件定義	C	ヒアリング記録の整理、要件一覧や要求定義書の整備に工数が嵩む。用語不統一により再作業が増える。	会話ログから要件一覧/要求定義書のドラフト自動生成。用語統一と整合性チェックを自動化し、レビューの初期負荷を低減。		一次生成を短サイクルで査読し、修正点をテンプレ化。再作業を防ぐ標準レビュー観点を定義する。	
要件定義	D	要件確定までに時間を要し、設計・実装開始が後ろ倒しになる。	論点の要約、対立点と落としどころ提示、ワイヤーフレーム/画面イメージの自動生成で合意形成スピードを上げる。		意思決定の締切日をあらかじめ設定し、関係者全員の最終合意を取り付ける。合意記録を管理する。	
基本設計	Q	設計漏れやロジック不整合、非機能(性能・拡張性・保守性)配慮の不足。セキュリティ脆弱性の見逃し。	画面設計案自動生成、API設計テンプレ、要件突合せ。最新脆弱性情報の参照と自動レビュー、規約/用語統一チェック。	AIによる設計案やレビュー結果を一次評価とし、セキュリティや非機能要件など重要領域は必ず人が確認・判断する。	AI検知は一次レビューと位置づけ、セキュリティ担当が必ず確認。非機能トレードオフを踏まえ全体最適を決める。	設計基準や業界標準、非機能要件(性能・セキュリティ・保守性など)の最新動向を定期的に確認し、設計の引き出しを増やす。現場や運用部門と普段から会話し、実運用での困りごとや改善要望を把握しておく。
基本設計	C	レビュー工数やドキュメント整形に時間。AIが読みづらい資料形式(セル結合等)が余計な手戻りを生む。	AIが読みやすいフォーマット整備、ドラフト自動化で初期作成負荷を削減。規約・用語チェックの自動化でレビュー効率化。		資料標準の徹底とレビュー観点の固定化。省力化と質担保の境界を決め、必要箇所に人の時間を集中させる。	
基本設計	D	設計確定の遅れが下流を直撃。仕様変更の影響見落としで後戻りが発生。	設計ドラフトの高速生成と要件差分の自動検知。変更箇所の影響範囲提示で合意と更改を迅速化。		変更裁定とスコープ管理を統括。決めるべき順序を定義し、先に進むための最低限仕様を確保する。	
詳細設計	Q	複雑な業務ロジックやデータモデルで整合性が崩れやすい。性能を意識した設計が難しく品質ばらつきが出る。	クラス/DB設計の自動提案、コードとの整合チェック、境界条件の列挙。非機能要件を入力にして設計案を比較生成。	システム全体像や非機能要件を理解し、障害発生時の影響範囲や復旧優先度を考慮した設計判断する。	複雑部分の設計判断を引き受ける。重要制約の明文化と確認、例外系の洗い出しを主導する。	システム構成や各機能のつながり、データフローを普段から図解して整理し、複雑な部分や例外処理のパターンをストックしておく。設計レビューで指摘された内容や現場からのフィードバックを蓄積し、設計時に活かせるようにしておく。新しい技術や設計手法にもアンテナを張り、必要に応じて自分の設計観点をアップデートする。
詳細設計	C	設計書作成・更新コストが高い。実装と設計が乖離して再作業が発生。	コードから設計書へ逆生成、差分強調で整合維持。レビュー観点の自動チェックリスト化で工数を圧縮。		レビューの要所を人が引き受け、コストのかかる箇所を狙い撃ちで改善。テンプレとサンプルを育てる。	
詳細設計	D	基本設計で決め切れていない仕様や判断基準が詳細設計に持ち込まれ、確認・手戻り・差し戻しが多発して作業が停滞	ベースライン案をAIで素早く提示し、疑義箇所を早期に可視化。完了判定条件を自動チェック。		実装開始のゲートを明確にし、未確定部分はリスク管理の下で段階実装に切り分ける。	

工程別課題と生成AI-人間の役割②

工程	QCD区分	QCD課題	AI活用案	人間の役割		必要な準備・取り組み
実装	Q	バグ混入、保守性・可読性の低さ、非機能要件未充足。コメント不十分で属人化。	コード/テストコード自動生成、静的解析、脆弱性チェック。規約遵守とドキュメント化を強制するプロンプトテンプレ。	AIが生成したコードやテスト結果をレビューし、非機能要件や命名規約など品質ゲートを人が運用する。障害時の影響度や復旧手順を整理し、設計判断の根拠を明記する。	非機能要件・規約・命名をプロンプトで具体化。レビューと受け入れ責任を担い、品質ゲートを運用する。	仕様や設計の意図を正しく理解するため、疑問点はそのままにせず、設計者や関係者に積極的に確認する習慣を持つ。コーディング規約や命名ルール、レビュー観点を定期的に見直し、迷ったときはすぐに参照できるようにしておく。
実装	C	スキル差で工数が膨らむ。調査時間が長く生産性が低下。	ペアプロAIによるガイダンス、既存Q&A/ナレッジ提示、リファクタリング案の自動提案。		育成計画とコーディング規約の徹底。AI出力の是正方針を定め、学習サイクルを回す。	
実装	D	実装遅延や手戻りでスケジュールに余裕がなくなる。	雛形生成、ユーティリティ自動生成、テスト先行支援で作業を並列化。変更影響の自動提示で手戻りを抑制。		優先順位付けとスコープ調整を主導。クリティカル経路を守る意思決定を迅速に下す。	
テスト	Q	網羅性不足、境界値や例外の見落とし。シナリオレベルで業務妥当性が担保されにくい。	仕様/設計を構造化しテストケース自動生成。要件項目とマッピングしてカバレッジ不足を検出。ログ解析で不具合の兆候を抽出。	複雑な業務ロジックやデータモデルの整合性を検証し、障害発生時の業務影響や復旧の優先順位を判断する。	業務上のクリティカルシナリオを定義し妥当性を判断。根本原因の深掘りと再発防止を主導。	テスト観点や品質基準、過去の障害・不具合事例を定期的に振り返り、どんなパターンが抜けやすいかを把握しておく。異常系・例外系のシナリオや現場で起こりうるリスクを普段から形式知化し、テスト設計時に活かす。テストデータや検証環境の整備もルーチン化する。
テスト	C	テスト設計・実施工数が肥大化。データ準備に時間を要する。	大量データ自動生成とマスキング。ユースケースごとのデータパターンをテンプレ化し自動適用。		テスト観点とデータパターンを定義してAIに入力。コストの高い検証を重点化する。	
テスト	D	テスト期間が延び、リリースに遅延。結合/システムテストの欠落で受入が遅れる。	自動実行・欠落パターン検出・失敗分析の自動要約。進捗と品質の見える化でボトルネックを即特定。		受入基準の明確化と合否裁定。品質と納期のトレードオフ判断を担う。	
保守・運用	Q	障害の誤診断や原因特定の遅れ。AIの過剰警告/見逃しのリスク。	障害ログ自動解析、原因候補と影響範囲の提示。監視基準に沿うアラート整形とナレッジ検索の自動提示。	AIによる障害検知やナレッジ提示を活用しつつ、最終的な障害判定や復旧優先順位の決定は人が担う。過去事例やユーザーからのフィードバックを反映し、運用プロセスを継続的に改善する。	AI検知を二重チェックし、緊急度と優先度を裁定。セキュリティ/運用責任者が最終判断を下す。	日々の運用で発生したトラブルや問い合わせ内容を記録し、ナレッジとして蓄積する。システムの構成や変更履歴、障害対応フローを定期的に見直し、緊急時に迅速に対応できるよう備える。現場やユーザーからのフィードバックを積極的に集め改善活動に活かす。
保守・運用	C	問い合わせ対応と情報整理に人件費がかさむ。ナレッジが陳腐化しやすい。	FAQ自動応答、障害対応の定型化、わいがや記録の自動整形更新差分の検出でナレッジ鮮度を維持。		重要FAQや法務/セキュリティ領域は必ず人が査読。ナレッジ検証プロセスを標準化し更新を回す。	
保守・運用	D	障害の初動が遅れ、復旧と周知が遅延。関係者が何をすべきか掴めない。	影響範囲・対応状況・連絡先を含む定型レポートを自動生成し配信。構成変化や障害パターンの学習で検知を継続改善。		復旧優先順位を決め、周知内容を確定。最終承認と顧客・社内への説明責任を果たす。	

６．2026年度の取り組みについて

6. 2026年度の取り組みについて

■研究会テーマ

品質・コスト・工期の改善及び生産性の向上

(1)研究会概要・方針

・システム開発・保守の品質・コスト・工期および生産性向上に関する取組みをテーマとし、上期は各社による事例発表、下期はテーマに沿ったグループディスカッションを行い、成果物を作成します。
・上期の事例発表では、各社の実際の取組みや課題、成功・失敗の経験を共有することで、実務に基づく知見を得るとともに、参加企業間の相互理解や気付きの深化を図ります。
・下期のグループディスカッションでは、事例発表を踏まえた議論や意見交換を行い、課題の深掘りや知見の整理を進め、参加者の経験を統合した成果物としてまとめることで、業界として重要な課題や実践的な知見を対外的に示すことを目指します。

(2)対象者

①プロジェクトマネジメント経験(或いは同等の経験)が5年程度や、PMO/品質管理部経験など、QCD活動を実践されている方
②保守開発でのQCD改善や、チームビルディング/人材育成など、様々な場面でQCD活動を実践されている方
上記①②いずれかの経験を踏まえて、年1回、自社の事例発表をいただき、研究会としての成果物作成にご協力いただける方

6. 2026年度の取り組みについて

	2026年							2027年			
	6月	7月	8月	9月	10月	11月	12月	1月	2月	3月	4月
イベント		合宿								成果物完成	成果報告
定例会	★	★	ー	★	★	★	★	★	★	★	
	事例発表					グループディスカッション					

事例発表

■目的
・情報共有およびグループディスカッションへのインプット
■テーマ
・各社でのQCDsに関する取組内容、課題、実例など

（参考）2025年度テーマ例
・開発前に見積もることって？
・障害をゼロにする事例紹介
・属人化の取り組みと業務移管の取り組み
・生成AI活用事例と課題

グループディスカッション

■目的
・QCDsに関する知見の共有と深掘り、整理による成果物作成
■テーマ
・QCDsに関する個別テーマ（3テーマ程度：研究会参加者が検討）

（参考）2025年度テーマ
①プロマネとは見積もりだ！
②障害は必ずゼロにできる！
③保守・メンテを極めよう！
④新技術を知り、備えよう！

ご清聴ありがとうございました。

おまけ

Q:システム開発・保守QCDs研究会はどんな研究会ですか？

A:以下のとおり「さしすせそ」で回答いたします。

さ:　最高の仲間達と一緒に
し:　心理的安全性を確保し
す:　酸いも甘いも乗り越えて
せ:　成果物を創り上げていく。
そ:　そして、次年度へ継承し、更なる研究へと繋げていく会です。

次年度の活動にもご期待ください。皆様のご参加をお待ちしています！(^^)/