●第 1 回（例会）：レビュー：猪塚 修氏（横河ソリューションサービス株式会社）

■ 概要

レビューを欠陥発見だけでなく品質を作り込むための中心活動として再認識し,成果物の読み方や視点整理を通じて体系的な判断手順を再確認した.成果物間の関連・用語整合・前提の一致を丁寧に確認する重要性を学んだ.
正常系と異常系を分けて読む姿勢や,IF 整合の確認を通じて「抜けや曖昧さの早期発見」が可能となることを理解した.ここで言う IF とは,ドキュメント内部だけではなく,関連ドキュメントや上位ドキュメントとの IF も含む.読み方の工夫が指摘の質に直結することを体感した.
演習では個人・グループで観点を持ち寄り,多様な視点がレビュー密度を高めることを再度確認した.観点の絞り込みが短時間でも効果を発揮する点も有効だった.

■ 有効性

指摘理由を共有することで観点の幅が広がり,自身の思考の偏りや盲点に気付く機会となった.観点の差異が議論を深め,レビューの質向上に寄与したと感じた.
品質特性やチェックシートを基準とすることで,属人性が減り,レビュー結果の再現性が高まる点が実務上非常に有効であると再認識した.
重点箇所を明確化した上で議論する手法により,短い時間でも重要な指摘に集中できる利点を強く実感した.

■ 留意点

目的や重点が曖昧なままレビューを開始すると指摘が散らばり,効果が低下するため,事前の意図共有が欠かせない.
準備不足（資料未確認やチェックシート未読）は指摘漏れを招くため,レビュー工程に「準備作業」を正式に含めることが必要である.
開発・運用・ユーザなど複数の視点から検討しない場合,重要な欠陥を見逃しやすく,視点固定のリスクを常に意識する必要がある.

●第 2 回：オブジェクト指向分析設計：井上 樹 氏（株式会社 豆蔵）

■　概要：
オブジェクト指向分析設計の基盤はモデリングである．このモデリングが開発でどのように役に立つのかを本演習を通して学んだ．

1.　要求のモデリング
要求のモデリングとは,何を要求されているのかを分かりやすい形で表現することである．これにより，システム完成イメージの早期共有，要求の正しい理解，不備不足の発見といったメリットが得られ手戻りコストの削減になる．
演習での要求分析は以下の 2 つである．
・シナリオ分析：利用者の視点からシステムの利用の流れ，相互作用を想定して要求分析をする手法である．具体的にはユースケース，ペルソナ法などがある．これにより，完成イメージを思い浮かべやすくなり，要求の不備不足に気が付きやすくなる．
・モデルベース分析:UML（統一モデリング言語）を使い,要求をモデル化する手法である．静的構造，動的構造，機能的構造の３つの側面から対象を捉えることが基本的な考え方であり，定義されたモデルを用いて自然言語を書き直すことで抜け，漏れに気付くことができる．

2.　設計のモデリング
設計のモデリングとは，ソフトウェアの構造を見える化することである．これにより，ソースコードで読み取れる範囲より広く俯瞰でき，レビューのしやすさ（問題の発見しやすさ）につながる．また，ソースコードを書く前に問題を発見でき，プログラムの書き直しを減らすことにもつながる．

■　有効性：
・ユースケースやシナリオとして要求を具体化することで，異常系，非機能要件，前提条件を明確化できる．これにより関係者間で完成イメージを共有しやすくなる．
・実装後ではなく，要求分析・設計段階で問題を発見できると修正コストの大幅な削減効果がある．
・問題の発見しやすさやプログラムの書き直しを減らすことにより，品質と生産性の向上につながる．

■　留意点：
・要求に関する用語定義は定まっていないため，自組織で使う用語を決めないと混乱を招く可能性がある．
・モデル駆動開発ではスモールインが重要とされており，システム規模や目的に応じてモデリングの適用範囲や粒度を調整する必要がある．
・詳細に書かれた UML モデルはコードに変換可能であるが，これは相応の情報量を持つモデルを正確に書くスキルが必要であることを意味する．

●第3回(例会):アーキテクチャ設計・評価:長谷川裕一氏(合同会社 Starlight&Storm)

■ 概要

アーキテクチャをシステムの構造や方針を定める基盤と捉え,品質に直結する領域であることを再認識した.特に品質特性の観点から考える重要性を理解した.
品質特性シナリオを用いて品質要件を具体化することで,曖昧さが減り,設計判断の基準が明確になる点が有効であると体感した.
ADD と ATAM の流れを演習で実践し,品質要件→設計→評価という一連のサイクルを体系的に理解した.

■ 有効性

品質特性が明確化されることで,設計意図の説明が容易となり,ステークホルダー間の合意形成が迅速になる点を実感した.
ADD により設計理由を記録して残すことで,後工程の保守・改修時に設計の整合性を維持しやすいという利点を強く感じた.
ATAM でリスク・感度ポイントを整理することで,優先すべき品質要件が可視化され,適切なトレードオフ判断につながることを理解した.

■ 留意点

品質要件が不十分なまま設計に進むとアーキテクチャが形骸化するため,初期段階の要件整理が欠かせない.
品質特性シナリオに実装方法を書いてしまう誤りは判断を誤らせるため,要件と手段の分離を徹底する必要がある.
依存関係やモジュール境界を曖昧にすると変更容易性が損なわれ,結果として品質目標の達成が難しくなるため注意が必要である.

●第 3 回(例会):メトリクスによるソフトウェア品質把握:小笠原秀人氏(千葉工業大学情報変革科学部 認知情報科学科)

■概要:
ソフトウェアの品質を把握し、改善するためにはメトリクスの活用が不可欠である。メトリクスは目的に基づいて収集・分析する必要があり、目的の明確化には GQM (Goal-Question-Metric) アプローチが有効である。講義ではメトリクスの分類や活用方法、グループ演習を通じて理解を深めた。.

(1)ソフトウェアの品質とメトリクス
ソフトウェア品質は、利用時にユーザーや関係者の要求を満たす能力の総体として捉えられる。品質を定量的に把握するためにはメトリクス（測定尺度）が重要となる。
メトリクスは以下の 3 種類に分類される。
・プロダクトメトリクス:ソフトウェア成果物の特性
（例:ページ数、プログラム構成要素数、シナリオ数、シーケンス数)
・プロセスメトリクス:開発や保守などのプロセスの特性
（例:レビュー予定回数、レビュー実施工数、指摘件数)
・リソースメトリクス:開発に投入される資源（例:工数、人数、費用など)

(2)QC7 つの道具の活用
品質管理の現場では、QC7 つの道具（パレート図、ヒストグラム、チェックシート、散布図、管理図、特性要因図、層別）なども活用されている。
データの傾向を把握するには、グラフ、ヒストグラム、パレート図、箱ひげ図を使い、データ間の関係を知るには散布図や相関係数を活用することが有効である。これらをメトリクスと組み合わせて、現状分析や課題発見に役立てることができる。

(3)ゴール指向メトリクス (GQM)
GQM は、「目的の明確化」を出発点とし、目標(Goal)に対して評価するための質問(Question)を設定し、その回答に必要な定量データ (Metric) を決定する枠組みである。
・測定目標(Goal):測定上の目標
・質問(Question):目標の達成を評価するための質問
・メトリクス(Metric):質問に回答するために必要な定量的データを得るための主観的・客観的メトリクス
目標設定の際には多面的に検討し、仮定や解釈を明確にすることで、より有効なメトリクスの選定と評価が可能になる。

■有効性:
・メトリクスは、開発の進捗・品質管理や改善活動に不可欠であり、現状把握だけでなく将来予測や計画策定にも活用できる。
・QC7 つの道具やグラフ・統計手法の活用で、メトリクスの分析や可視化が容易となり、課題発見や改善策立案に役立つ。

■留意点:
・メトリクスの収集は目的を明確にした上で行うことが重要であり、単なるデータ収集は無意味である。
・メトリクスで全てを把握できるわけではないため、定性的な評価や現場の知見とあわせて活用することが望ましい。

●第 4 回（臨時）：要求工学：斎藤 忍 氏

■概要
要求定義とは，顧客のニーズに即したシステムを定義することである．顧客の要望を満たすシステムを提供するためには，要求からサービスの全体像をデザインすることが重要である．
本演習では要求工学の技法として以下の項目について学習し，それらを作成する演習を行った．

【技法】
- ペルソナの生成

ユーザーの情報（名前，性別，年齢，スキル等）や好み，生活スタイルなどを記述し，詳細な人物像を作り上げる．

- サービスシナリオの作成

開発対象のサービスをペルソナが日常生活においてどのように利用するかというシナリオを記述する．

- カスタマージャーニーマップの作成

ユーザーの時系列の行動，他者やモノとの関わり方，サービスを利用する際の思考や洞察を洗い出す．

- ユーザーストーリーマップの作成

ユーザー視点で書かれた要件(ユーザーストーリー)を，ユーザーの行動ごとに洗い出し，優先順位を付ける．

■有効性
- ペルソナ

ユーザーのイメージや認識が一致するため，ユーザーの問題点や課題を議論するための時間やコストを削減できる．また，ユーザーのニーズに沿ったプロダクトやサービスを開発するためのアイデアが生まれやすくなる．

- カスタマージャーニーマップ

ユーザーの体験を視覚的に表現することにより，問題点や改善点を発見することができる．

- ユーザーストーリーマップ

優先順位を付けてユーザーストーリーを整理することにより，MVP(Minimum Viable Product)を切り出すことができる．

■留意点
- ペルソナ

先入観や思い込みを排除して作成する．現実的なペルソナを生成するためには，周囲の人を参考にするとよい．幅広いユーザー層を満足させるサービスを開発するのではなく，特定の一人のペルソナ(＝メインペルソナ）を満足させるようにサービスを開発する．

- カスタマージャーニーマップ

ユーザーの思考や洞察を記述する際は，ポジティブな感情もネガティブな感情も書き出すことが大切である．

- ユーザーストーリーマップ

MVP(Minimum Viable Product)を絞り込む前に，ユーザーストーリーを発散させることが重要である．それにより，ユーザーやサービスへの理解を深めることができる．

●第 5 回（例会）：アジャイル開発：天野 勝氏（株式会社永和システムマネジメント）

■　概要

アジャイル開発とは，機能ごとに優先順位を定めて開発の単位を小さくし，高頻度でリリースする開発手法である.主なメリットとしては,早い段階から価値を提供できることや,要件の変更にも柔軟に対応できることが挙げられる．今回は標準的なアジャイル開発の進め方についてスクラムのフレームワークを用いて演習形式にて理解を深めた.

演習では以下のルールに従って，メンバーを 3 つの役割に分けて開発作業を実施した.

【ルール】

チームにて折り紙を用いて多面体を作成することにより，完成した多面体やその数に応じた得点が得られる．できるだけ多くの得点の獲得を目指す.

【役割】

- プロダクトオーナー（1 名）：サービスの責任者．プロダクトバックログを管理して方針を決定する.
- スクラムマスター（1 名）：スクラム管理の責任者．プロダクトオーナーの方針を開発に適用し、スクラムを確立させる.
- 開発者（5 名）：成果物を作成する.

【開発作業】

以下の流れを 1 スプリントとし，これを 2 回実施した．スプリントではスプリントプランニングを 1 回，朝会および開発を 3 回，スプリントレビュー/リリースおよび振り返り会を 1 回行った.

① スプリントプランニング：プロダクトオーナーがスプリントの目標や戦略を決定する.
② デイリースクラム（朝会）：スクラムマスターを中心に作業の役割分担，開発の進め方を議論し，決定する.
③ 開発：開発者が成果物を作成し，スプリントバックログに反映させる.
④ スプリントレビュー/リリース：成果物をレビューし，リリースする.
⑤ レトロスペクティブ（振り返り会）：KPTA を用いてスプリントの振り返りを行い，次のスプリントに向けた改善点を議論する.

■　有用性

・アジャイル開発ではスプリントごとに目標を定めて開発および改善を繰り返すため，ウォーターホール開発と比較して短い時間で価値を最大化できる.

・朝会や振り返り会など，チームメンバー同士のコミュニケーションが行える場が多く設定されているため，開発目標に関する認識を適宜合わせられる.

・朝会の際にスプリントバックログなどでチームメンバーの進捗状況の詳細を把握できるため，そこで理解した得手不得手を開発の作業に反映し，より効率的に開発を進めることができる.

■　留意点

・スクラムマスターはプロダクトオーナーの方針を正確に開発者に伝えるため，両者しっかりと認識を合わせる必要がある.

・作業を補い合い早く成果物を作成するために，朝会の際に開発者同士でもコミュニケーションを取り，お互いの進捗状況を把握しておく必要がある.

●第 6 回(例会)：テストを作る技術(テスト技法演習)：樋下田 順也氏(株式会社ベリサーブ)

■概要

本演習の目的は，演習を通してソフトウェアテスト技法の基礎を学ぶことである．テスト技法を知らない人の場合は使い方のコツ，知っている人の場合は他者への指導方法を習得できる内容となっていた．

- テストを作る技術
  テスト作成のためには，テストスキル以外にソフト/IT に関するスキルやドメインの知識についても必要となる．
  テスト分析の内容からテスト設計を実施する際にテスト技法を活用する．

- テスト技法
  - 同値分割法
    入力値の全体集合をある条件に沿って区切る技法．
  - 境界値分析
    同値クラスの代表値として境界値を使う技法．
  - デシジョンテーブルテスト
    入力条件の組み合わせによって出力が変化することを確認するための技法．
  - 状態遷移テスト
    処理順序の組み合わせを確認するための技法．

- テスト分析と設計で行うモデル化
  テスト技法を適用するためには適用可能なサイズまでシステムを分解する必要があり，そのためのモデル化方法を学習した．

■有効性

- 同値分割法/境界値分析：
  同値クラスの代表値をテストすることで，実施するテスト数を削減することが出来る．また，代表値として境界値を設定することで，欠陥が含まれやすい箇所を重点的に確認出来る．
- デシジョンテーブルテスト：
  条件を全て書き出すことで考慮漏れを防止し，試験の網羅性を担保することが出来る．
- 状態遷移テスト：
  処理順序による欠陥を防止する．また，一連の流れとしてテストを実施することが出来るため，テスト内容をイメージしやすい．

■留意点

- どの技法を適用する場合においても，仕様が明確であることが重要となる．仕様の解釈に曖昧さが残ると期待結果が揃わず，テスト実施段階で動作確認などの手戻りが発生する．
- 抜け漏れの防止や境界・例外の洗い出しをしやすくするため，仕様を文章のまま捉えるだけでなく，状態遷移図や表などを使って構造化することが重要である．
- テスト実施者との認識のずれを防止するため，入力値・操作手順・期待結果を明確に記載し,誰が見ても同じ判断ができるように試験内容を具体化することが重要である．

●第 7 回（例会）：上流から下流まで生成 AI が変革するシステム開発：酒匂 寛 氏（有限会社デザイナーズデン）

■概要

- 本講演では，生成 AI の普及により，ソフトウェア開発プロセスが従来の「How」を重視するアプローチから，「What」を明確化するアプローチへと転換しつつあることが示された．道具を使いこなす能力ではなく，達成すべきゴールを明確に描く能力の重要性が指摘された．
- 生成 AI は，コード生成・仕様化・データモデル化・文書化など，多くの工程を支援可能な技術である一方，生成された成果物の品質保証は人間が担う必要がある．講演では，生成 AI は開発を代替する存在ではなく，人間の意思決定を補助する道具として位置づける考え方が示された．
- 講演では，「アイデア → BMC → TiD(Trace index Diagram) → 形式仕様 → 実装 → テスト」という開発ライフサイクルを例に，どのステップでも AI が支援可能であることを示された。特に，モデル（仕様）さえ明確であれば，生成 AI が多様な表現（文書，説明，コード）を生成できる点は，モデル中心思考に基づく開発の有効性を示唆している．
- プロンプトエンジニアリングに関しては，複雑な技法よりも，要求内容（What）を段階的かつ明確に伝えることの重要性が示された．さらに，生成 AI の活用により，専門的成果物をステークホルダー自身が理解・確認できる可能性が示され，開発工程の透明性および参加性向上への寄与が示唆された．

■有効性

(1)開発プロセス全体の効率化

初稿（たたき台）の質が向上し，レビュー前の負担が大幅に削減される．

コーディング，仕様，テストなど各工程において，AI を“補助作業者”として活用できる．

(2)上流工程（課題整理・仕様化）の品質向上

実現したいアイデアから BMC のようなビジネス全体の“モデル化作業”を AI が補助できる．

事実／願望／制約の整理が自動化され，曖昧さや漏れが減少し，組織内で合意形成がしやすくなる．

(3)モデル中心の開発が加速

仕様やデータモデルが明確であれば，AI が文書・コード・説明・テストケースを生成できる．

「モデルを作る → AI が多様な表現に展開する」という モダンな開発サイクルが確立する．

(4)検証工程の強化

Verification（正しく作る）および Validation（正しいものを作る）の双方を生成 AI で補強できる．

Verification の観点では，テストケースの自動生成やレビュー観点の標準化により，
実装が仕様どおりであるかの確認を効率化できる．

Validation の観点では，仕様書や設計書といった中間成果物を生成 AI が分かりやすく整理・展開する
ことで，ステークホルダー自身が内容を確認しやすくなる点が特に有効である．

■留意点

(1)AI が生成する成果物の品質保証は必ず人間が行うこと

AI は“それらしく作る”が，正しさの保証はない.
特に仕様・データモデル・コードの誤りは重大な影響を及ぼすため，人のレビューが必須である.
(2) What が曖昧だと，出力も曖昧になるため注意すること
プロンプトのテクニックではなく，
「何を欲しいか（What）を明確に伝える」 ことが最重要である.
また，複雑な指示より，段階的・具体的な依頼が効果的である.
(3) 自律化（エージェント化）は過信しないこと
完全自律で複雑なタスクを実行させると品質問題が起きやすいため，
現段階では必ず人間の判断を挟む必要がある.

●第 8 回（例会）：工数見積りモデルの構築手法：石谷 靖（株式会社三菱総合研究所）

■概要
ソフトウェア開発における見積では，見積り根拠の「見える化」= コスト構造の「見える化」が重要である．
この見える化のために工数見積りモデルの構築手法(CoBRA 法）を学んだ．
CoBRA 法の見積式を下記に示す．
工数（コスト）＝α×規模×（1＋ΣCOi）
α：ベースとなる生産性を表す係数であり，過去のプロジェクト実績データ（規模と工数）を用いた回帰分析により算出
規模:プロジェクトの開発量を示す指標
COi：数に影響を与える変動要因によるコストオーバーヘッド

また演習(グループワーク形式）では,「CoBRA 構築モデルプロセスの実施」,「CoBRA モデルの改善」,「CoBRA モデルを使ったリスク分析・コストコントロール」を実施した．

■有効性
- 熟練者の経験則を定量化し，説明可能な形で整理できる．
- 規模だけでなく，プロジェクト固有の変動要因を考慮し，同規模でも工数差の理由を示せる．
- 過去の実績データと熟練者の知見を組み合わせたハイブリッド型手法である．
- 見積り結果を確率分布として扱い，予算超過リスクやコストコントロールに活用できる．
- 見積り誤差分析を通じてモデルを継続的に改善できる．
- 見積り精度を定量的に評価し，モデル改善効果を比較できる．

■留意点
- 変動要因として，類似のものがある場合や因果関係があるものは，1 つに集約する．
- 変度要因の数は 10 個程度とし，多くても 15 個とする．
- 変動要因の影響度設定は熟練者の判断に依存する．
- 過去プロジェクトの実績データの質が低い場合，α や見積り結果の信頼性が低下する．
- モデルは一度構築して終わりではなく，見積り誤差に応じた継続的な改善が必要である．
- 特殊なプロジェクト（規模が極端に大きい・小さい，新技術導入など）ではモデル適用に注意が必要である．

●第 9 回（臨時会：ゴール指向分析と GQM：鷲崎 弘宜 氏（早稲田大学グローバルソフトウェアエンジニアリング研究所 / 国立情報学研究所 / (株)エクスモーション）

■ 概要

- システム開発の要求定義プロセスにおいて,「Why（ゴール)」「What（要求仕様)」「How（実装)」を整理し，誰かにとって達成すべき上位のゴールを最初に明確にした上で，達成手段を下位のゴールや要求項目へと木構造で分解･詳細化する「ゴール指向分析」について学習した.
- 明確な測定目標（Goal）を定め，目標達成を評価するための質問（Question）と，それに回答するための定量的なメトリクス（Metric）を対応付ける「GQM パラダイム」について理解を深めた.
- GQM を組織レベルに拡張し，上位の組織目標から戦略，下位の組織目標，測定指標へと連鎖させる「GQM+Strategies」のプロセスと手法を学んだ.
- 本講演では特に，手法や指標を漫然と適用するのではなく,「目的（Why)」を最上位に置き，それに基づいた論理的な検証や評価を行うことの重要性が一貫して強調されており，実務適用における新たな視点としての気づきを得た.

■ 有効性

(1) ゴール指向分析による要求定義の円滑化と妥当性確認

従来の要求定義では，具体的な手段（What や How）の議論が先行し，システム本来の目的が見失われることが多い．しかし，ゴール指向分析により「誰にとってのどのような状態か（Why)」を最上位に置き，それを実現する達成手段（AND/OR）を木構造で論理的に階層化することで，抽出した要件の妥当性を上位ゴールに遡って検証しやすくなる．これにより，関係者間での認識齟齬を防ぎ，円滑な合意形成が可能となる.

(2) GQM パラダイムによる「目的を見失わない」定量的評価

ソフトウェアの品質やプロセスを改善するために測定データを収集する際，単に集めやすいデータを漫然と計測してしまうケースが散見される．GQM パラダイムを適用することで，「何のための測定か（測定目標)」「何を把握すべきか（質問)」という目的からブレイクダウンしてメトリクスを設定できるため，改善の意思決定に直結する意味のあるデータ収集と論理的な評価が可能となる.

(3) GQM+Strategies による「目標連鎖」の可視化とシステム投資の適正化

システム開発において，経営や事業部門が掲げる中期的な目標は漠然としていることが多く，情報システム部門との間で方針の齟齬が生じやすい．GQM+Strategies を活用し，上位の組織目標から IT 部門の下位目標，そして具体的な戦略（施策）へとグリッド構造で結びつけることで「目標連鎖」が可視化される．これにより，部門間の壁を越えて方向性を共有でき，ビジネス目標に直結した優先順位に基づく適正な IT 投資計画について合理的な説明が可能となる.

■ 留意点

(1) 最上位の目標（ゴール）は，手段ではなく「状態（Status)」として定義すること

目標設定において手段を記述してはならない．その理由は二点ある．第一に，下位の階層で具体的な手段（How）を展開するためである．人間は普段の業務において無意識に手段から考えがちであるため，意識的に「状態」を記述することでその思考の癖を排除する必要がある．第二に，目標に手段を記述してしまうと思考が停止してしまうためである．本来，一つのゴールに対して達成手段は複数存在するが，手段を目標に据えてしまうと，それを実行した時点で完了とみなされ，より優れた代替案を探求する機会を失うリスクがある.

(2) コンテキスト（事実）と仮定の区別および継続的な妥当性検証

組織目標と戦略を定義する際は，環境に関する客観的な「コンテキスト（事実)」と，不確かな推測である「仮定」を明確に区別して記述しなければならない．システム導入後や施策の実行後に期待した測定結果（目標）が得られなかった場合，この区別があることで「目標設定が意欲的すぎたのか」「戦略（施策）自体が不十分だったのか」，あるいは「前提とした仮定がそもそも間違っていたのか」を論理的に切り分けて分析でき，確実な次サイクルの改善へと繋げることができる．