●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」について学んだ．レビューとは，ソフトウェアでの欠陥を発見し，後工程での手戻りを減らすための開発工程である．

演習はグループ討議を中心としたオフライン形式で実施された．

【演習の内容】

1. （個人）SNS システム開発を例にとった抽象的なアーキテクチャ設計に対するレビュー・アドバイス
2. （個人）1.で扱ったシステムへの品質特性の観点からのレビュー
3. （グループ）1.および 2.の内容のグループ討議と発表

【テストの手法】

- ウオークスルー：作成者が説明する
- レビュー：事前にレビューアが査読する
- インスペクション：作成者以外の読み手が説明する
- 回覧：順番に成果物に指摘を書き込む
- 一斉書き込み回収：メール等でブロードキャストして指摘を書いてもらう
- 質問形式：レビューア，進行役が質問しながら進める

■有効性

- テストと比較して早く，安く欠陥を見つけられる
- テストで見つけにくい,または見つけられない欠陥を見つけられる
  例：動くけど汚いコードを書きなおす
- 専門家の意見を取り入れて設計を改良できる
- 設計や実装の観点について討議することで，同席者の教育を行える

■留意点

◎レビューのコツ

【品質特性を意識する】

システムの課題解決を品質特性の面から検討することで，さまざまな視点からレビューできる．

【目的を決める/重点を絞る】

メリハリのないレビューは効率が悪い．作業フェーズ・視点・担当者のスキルによって重点を決めることで，レビュー効率が上がる．

【視点を変える】

ソフトウェアに関わる立場（開発者・発注者・ユーザ…etc.）によってソフトウェアに求める品質特性は変化する．

【チェックシートを活用する】

組織で用意されたチェックシートに基づいてレビューする．

◎チェックシートの考え方

- 設計前に内容を学習しておくことで，作りこみ品質を向上させることが必須
- 何を考慮するのか誰にとっても明確になるようにする．不明確な場合は認識を共有する

●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とはシステムの実現手段を分析し，オブジェクトという単位で設計することである．その基盤となるのがモデリングである．

本演習では要求のモデリングと設計のモデリングについて，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．わかりやすい形で表現することによって，正しく・抜けもれなく要求を理解することができる．演習では以下の手法により要求のモデリングおよび要求分析を行った．

・シナリオ分析：利用者がどのようにシステムを使うか，相互作用を想定して要求分析をする方法である．ユースケース図，ユースケース記述，シナリオリスト，シナリオ詳細記述と具体的にユースケースを書き進めることによって問題点などを洗い出すことができる．

・UML：UML という自由度がない言語を使って，要求を書き直すことによって不足を洗いだす手法である．

2. 設計のモデリング

設計のモデリングとは，ソフトウェア構造を見える化することである．演習では，クラス図，ステートマシン図，アクティビティ図，シーケンス図を用いて設計のモデリングを行った．

■ 有効性：

・要求のモデリングでは，ソフトウェアは後工程になればなるほど指数関数的に修正コストが上がるため，要求の段階で抜け漏れ・認識違いが解消できると手戻りコスト削減つなげることができる．

・異常系や例外処理などといったステークホルダの興味がない事柄についての要求は出づらいものである．そこで，システムを利用するユーザ・外部実態とシステムとの相互作用を表すユースケース図を用いることで概観をつかみ，以降の開発工程において網羅性を説明するための根拠とすることができる．

・ソフトウェアは規模が大きくなるにつれて急速に複雑さが増し，開発者が理解できる限界を超えると，変更の影響範囲が見えなくなる．設計のモデリングでは，ソースコードよりも広い範囲を俯瞰可能であり，ソフトウェアの一貫性や全体最適に寄与する．また，コードを書く前に問題を発見することも可能である．

■ 留意点：

・要求は 4 つのステージからなり，利害関係者が実現してほしいと思っていることを示す「要望」，要望のうち，システムで実現することを定めた「要求」，要求を満たすためのシステムのふるまい・特性を定めた「要件」，要件が網羅された要件の集合である「仕様」である．それぞれを区別することが重要である．

・設計のモデリングにおいて，UML からコードへ変換することも可能ではあるが，その場合は変換に耐えうるだけの情報量を持ったモデルを正確に書く必要がある．

●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
アーキテクチャとはシステムの基本的構造であり，システムの品質を決定する重要なものである．しかし，アーキテクチャには唯一の解決策が存在せず，目的に応じて柔軟に作りこむ必要がある．本演習ではアーキテクチャを分析・評価する方法として「品質特性シナリオ」,「ADD」,「ATAM」について学んだ．

(1)品質特性シナリオ

システムの「品質特性」から「シナリオ」を作成することでアーキテクチャが満たすべき要件を明確にする手法のこと．

「品質特性」とはシステムにとって望ましい性質のことで，下記のようなものがあげられる．

- 可用性：いつでも利用できること
- 変更容易性：アップデートや修正が簡単にできること
- セキュリティ性：安全に利用できること
- テスト容易性：問題を見つけやすいこと
- 使いやすさ：利用者にとって使い勝手がいいこと

「シナリオ」とは，システムが刺激を受け応答を返すまでの流れを品質特性に着目して抽象化したもの．シナリオの構成要素として「刺激の発生源」,「環境」「成果物，アーキテクチャ要素」「応答」「応答測定」を定めることで，システムが満たすべき品質特性を明確にすることができる．

(2)ADD(Attribute Driven Design)

アーキテクチャ設計手法の１つ．満たすべき品質要件シナリオからパーツ(アーキテクチャパターン) を取捨選択し組み合わせる．

(3)ATAM(Architecture Trade-off Analysis Method)

アーキテクチャの評価手法の１つ．品質特性に関する要求を満たしているか否かによって評価する．

■有効性

- アーキテクチャを適切に設計することでシステムの品質を実現することが可能となる
- アーキテクチャパターンなどを設計の手掛かりにすることで，システムに対する理解を助け，将来のコストやスケジュールの見積もりを助ける

■留意点

- アーキテクチャはシステムの品質を大きく左右する．そのため，システムの品質を決定するための「品質特性シナリオ」を作成し，早い段階で分析・評価を行うことが有効である
- 非機能要件を抽出することが品質特性シナリオを作成する目的の一つであるため，品質特性シナリオを作成する段階では具体的な実現手法を記述しない
- 様々なステークホルダと共同で品質特性シナリオを作成することで非機能要件を多角的に抽出することが出来る
- アーキテクチャ設計においてもシステム設計と同様に，強度が高く依存度が低い設計を心がけることが重要である

●第４回（臨時）：要求工学（要求分析）：中谷　多哉子　氏（要求工学）

■　概要：

要求工学とは，ソフトウェア開発におけるユーザが求める要求を抽出していくための技術である．要求抽出には様々な手法があり，明らかにしたい要求を抽出するのに適した手法を選択していくことが大切である．

本演習では，ステークホルダ分析・現状分析から要求の抽出までの一連の流れに適した手法について学び,「リカレント（リスキリング）教育」をテーマに各種手法について演習を行った．

1. リッチピクチャを用いた現状分析
   現状把握を行うため，ある場面における漫画を作成し，その場面における世界観を把握する手法である．漫画では，表情を持った人物・吹き出しでの発言・位置関係を書き出すことで，世界観を直観的に把握することができる．
2. CATWOE 分析
   問題の現状を以下に示す６つの要素の頭文字で表現し，要求分析を行う手法である．
   - C：Customer（T の犠牲者もしくは受益者）
   - A：Actor（T を行う人々）
   - T：Transformation process
     （ある状態から世界観 W に合致する別の状態へ変換するプロセス）
   - W：World View（O の信念，根拠，世界観）
   - O：Owner（T を止めることができる人々または T を実現すれば，世界観 O を達成できると考えている人々）
   - E：Environment（T 達成に必要な環境と資源，守るべき制約とルール）
3. ゴール指向分析
   システムが要求を満足することによって達成できる目的(ゴール)を細分化し，システムが仕様を満たすことで，実現されること（要求）および要求を満たすための手段（仕様）の関係を可視化する手法である．

■　有効性：

要求工学の様々な分析手法を適用することで，ソフトウェア開発における要求抽出の背景や根拠といったことを明確にできるとともに，抽出内容を共有することで，要求に対し漏れているものをないかを検出することが容易となる．

リッチピクチャや CATWOE を用いたグループ演習では，分析した内容を要求することで，参加者ごとの視点を学べるとともに，抽出した内容に対する様々な議論を実施できたことを実感した．

■　留意点：

リッチピクチャでは，表現する人物や吹き出しを世界観に適したものとしなければ，直覚的に把握することが困難となるため，使用する際には注意が必要である．

●第 5 回(例会)：アジャイル開発の基礎知識：
天野 勝 氏(株式会社永和システムマネジメント)

■ 概要
市場の変化が急速に起こる昨今，迅速かつ複雑性に適応するシステム開発モデルとしてアジャイル開発の普及が進んでいる．アジャイル開発とは，従来の大規模型の開発モデル(ウォーターフォール)に対し，小規模かつ短期間で反復的な開発を行うことを重視し，顧客要求や開発目標の変化に柔軟に適応することを重視したシステム開発モデルである．本分科会活動においては，ゲーム形式にてアジャイル開発の手法の一つである「スプリント」を体感し，その特徴や有効性について議論し理解を深めた．
〈演習の概要〉
「手本通りに図形を作成し，その数に応じてポイントを加算される」ゲームを 3 チームに分かれて実施．各チームはスプリントの手法に基づきゲームを進行する．
各チームは以下のメンバーから構成される．
- プロダクトオーナー：バックログを管理してプロダクトの方向性を確定する．
- スクラムマスター：障害解消やスクラム適用の促進，プロセス改善をサポートする
- 開発者：イテレーションごとに開発を実施し，協力して成果物を生み出す．

スプリントは以下の流れを 1 サイクルとし，これを 2 回実施した．
① スプリントプランニング：プロダクトオーナー中心に目標を設定する．
② デイリースクラム(朝会)：具体的な行動や方針を決定する．
③ 開発：成果物を作成する(ゲームを実施)．
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 スプリントレトロスペクティブ：KPTA，YWT，バーンダウンチャート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（演習での気づき）
- 「まずやってみて，やりながら良くしていく」がやりやすい．前例のない開発やノウハウが十分でない時などはウォーターフォールより有用であると感じた．
- スプリントの中でコミュニケーションの機会が十分用意されているため，メンバー同士の得意不得意などを理解しやすく，開発の方針に反映させることができる．

■ 留意点
アジャイル開発成功のカギはコミュニケーションであると実感した．チームメンバー，ステークホルダ，プロダクトオーナーの密な連携を実施することが重要である．
また必ずしもアジャイルが優れているわけではなく，開発の特徴や品質保証プロセス，メンバーのスキルなど総合的な判断の上で開発手法を選択すべきである．

●第６回（例会）：デザイン思考＆ペーパプロトタイピングと周辺：
鷲崎　弘宜氏（早稲田大学グローバルソフトエンジニアリング研究所 / 国立情報学研究所 / 株式会社システム情報 / 株式会社エスクモーション）

■概要
デザイン思考とは，問題の本質を見極め，多くのアイデアを出して解決に導くプロセス＆早くたくさん失敗して経験から学ぶ考え方である．
本演習では，デザイン思考のプロセスを理解し，代表的な技法を使って，ユーザビリティテストを体験した．内容は以下の通り．

●デザイン思考のプロセス
① 発想：課題の設計を工夫し，ものごとの新しい見方を探す．
代表的な技法・・・ペルソナ法，構造化シナリオ法
② 創造：アイデアを生み出し，実行可能なものにする．
代表的な技法・・・プロトタイプ法
③ 改良：被験者からのフィードバックをもとに実験を繰り返し継続する．

●技法概要
－ペルソナ法：実在するユーザの代わりに使われる厳密に定義されたユーザ像．
－構造化シナリオ法：以下３つのステップで，ユーザの目標や活動を検討する．
・サービスシナリオ
ユーザ一般にとっての価値とビジネス的価値を記述する．
・アクティビティシナリオ
ペルソナにおける活動全体のフローや情動を記述する．
・インタラクションシナリオ
ユーザが目標に向かうシステムや実装に依存した具体的活動を記述する．
－プロトタイプ法：実物に近いモデル・模型を作成して仕様を検討する．
・ペーパプロトタイピング
ユーザが行いたいことを，ストーリーボードとしてワイヤ-フレームをシナリオに沿って並べ，ユーザビリティテストを行う．
テストの被験者は，ペルソナになりきって，思いついたことを発話する．
作成者は結果を記録する．

■有効性
● ペルソナ：プロジェクトの関係者全員がユーザ像を共通認識できるため，開発を遂行しやすくなる．
● 構造化シナリオ：ユーザ視点のイメージが深まり，問題の本質を分析できる．
● プロトタイピング：コーディング不要であり，容易にテストを実現できる．
更に繰り返し実施することで，UI の完成度を高められる．

■留意点
● ペルソナ：理想像や主観ではなく，ユーザ視点に立って，ユーザ像を定義すること．
● 構造化シナリオ：ペルソナ体験に沿ったユーザ視点で作成すること．
開発者視点で作成すると，実装方法によらないユーザの本質･欲求を識別できない．
● プロトタイピング：開発早期に問題発見することが目的のため，プロトタイプの内容に拘るあまり，工数をかけ過ぎることは避けること．
また，被験者にかかる工数も考慮して，再テストの頻度を決定すること．

●第７回（例会）：ソフトウェアテスト：鈴木　三紀夫氏（ASTER）

■概要
本演習の目的は，演習問題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．各演習問題では「個人演習」＞「グループ討議」＞「講師による解説」を順次行うことで，初学者はテスト技法の特徴や使い方のコツを，有識者は指導方法を身につけることができた．演習テーマとして扱われたテスト技法は以下の通り．

・ホワイトボックステスト技法
　－制御フローテスト
　　処理の流れを制御フローグラフで表現し，グラフを網羅する基準に従ったテスト．
　　通常以下の網羅基準（カバレッジ基準）を用いる．
　　・C0（命令網羅）…命令文（ノード）を少なくとも１回実行する
　　・C1（分岐網羅）…条件分岐（エッジ）を少なくとも１回実行する
・ブラックボックステスト技法
　－同値分割法
　　同じ出力結果となる入力を同値と見做すグループ（同値パーティション）に分類し，そのグループごとに代表的な値を用いたテスト．
　　※応用：同値と見做す条件や粒度をどの程度にするか（ズームイン，ズームアウト）をチーム間ですり合わせることが重要．
　－境界値分析
　　同値分割法における各同値パーティションにおける端の値（最小値，最大値）および，その外側となる隣接値を用いたテスト．
　　※応用：複合条件のテストケースを検討する場合はグラフを用いた検討が有効．
　－デシジョンテーブル
　　入力条件の組み合わせと出力結果の組み合わせを整理したデシジョンテーブル（決定表）を用いたテスト．
　　※応用：デシジョンテーブルの圧縮（簡単化）を行うことでテストケースを減らすことができるが，圧縮には一定のルールがあるため注意が必要．

■有効性
・目的に応じたテスト技法を用いることで，設計・実装ミスや仕様考慮漏れを発見することができ，品質を高めることができる．
・テスト技法を適切に活用することで，テストケース数を合理的に最適化することができ，コストパフォーマンスを高めることができる．

■留意点
・テスト設計には正解が無いため，チーム間でテストの考え方，テスト仕様の出し方，テスト技法の活用方針や適用条件のすり合わせを行わないとテスト品質にバラつきが生じてしまうことに留意すること．
・今回取り上げたテスト技法はテスト技法の一部であり，他にも状態遷移テストやユースケーステストなど様々なテスト技法があることに留意すること．

●第８回(例会)：メトリクス：小笠原秀人　氏（千葉工業大学　社会システム科学部　プロジェクトマネジメント科）

■概要

メトリクスはソフトウェアやその開発プロジェクトにおける活動状況や活動結果を定量的に把握することができるものである．本演習では，メトリクスの基本的な考え方および，仮説・検証プロセスにおけるメトリクスの活用，定量的管理の基礎を学んだ．

(1) メトリクスとは

ソフトウェアメトリクスには，プロセス，プロダクト，リソースの３種類のメトリクスがある．また，単一の属性を測定する基本メトリクスと，複数の基本メトリクスを組み合わせた導出メトリクスに分類される．測定に当たっては，目的を明確化してからメトリクス定義を行うことが重要である．測定には工数がかかる事を考慮し，不要なメトリクスに対する管理を行わないよう保つことで，効果的なメトリクス活用が期待できる．

(2) メトリクスを活用した仮設・検証による改善

メトリクスによる改善プロセスは，①原因の推測，②メトリクスの定義，③測定・分析，④改善の実施，⑤結果の確認の流れで実施する．プロジェクトにおける活動状況を定量化し分析することで，実現容易で確実な改善策を検討することができる．

(3) 定量的管理の基礎

メトリクスを活用し，プロジェクトの定量的管理を行うことができる．定量的管理は，①測定目標の確立，②メトリクスの定義，③測定，④分析，⑤結果の伝達の流れで実施する．メトリクスの定義における手法として GQM (Goal-Question-Metric) がある．以下のように目的・質問・メトリクスの順に定義し，目的ベースでメトリクスを定めることができる．

- Goal（目的）：達成したい状態や戦略を基に，測定の目的を定める．
- Question（質問）：目的を達成しているかどうかを把握するための質問を定める．
- Metric（メトリクス）：質問に回答するために必要な定量データを定める．

■有効性

- メトリクスに基づいた仮説・検証型の活動によって，定量的な数値による継続的な改善に取り組むことができる．
- メトリクスを用いた管理を行うことで，開発状況の客観的な把握と合理的な判断が行える．
- プロジェクトの活動におけるメトリクスと，組織単位で取得するメトリクスを双方で連携することで，類似プロジェクトの過去データや品質基準値の設定をより効果的に行うことができる．

■留意点

- メトリクスは目的を明確にしてデータを収集することが重要である．必要以上のデータ収集は管理活動が形骸化してしまう可能性がある．収集活動や活用の目的を共有し，必ず分析し活用する．
- プロジェクト管理における収集データの活用において，個人やプロジェクト自身の叱責に用いないことに留意する．測定へのモチベーションが下がり，データの信頼性低下につながる可能性がある．

●第 9 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

ソフトウェア開発の工数見積もりにてコスト構造の「見える化」が重要となる。その見える化のため，工数見積もりモデルの構築手法である CoBRA 法を学び，「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」「見積りモデルを用いたプロセス改善」を実施した．

ここで、CoBRA 法の見積式を下記に示す．

$$工数（コスト）＝\alpha \times 規模 \times （1＋\Sigma C0i）$$

αは「生産性」を表しており，実績データに照らして，変動要因とその定量化を検証し，算出するものである．C0 は「コスト変動要因のオーバーヘッド」を表しており，経験豊富な PL 等の熟練者の知見を基に変動要因とその影響を定量化したものである．
CoBRA 法は、従来の「経験ベース型」と「データ駆動型」の「混合（ハイブリッド）型」といえる．構築したモデルに対し、要因関係図の見直しや，規模・工数・変動要因のレベルの見直しによる改善を繰り返し行うことにより，見積モデルが完成する．

■有効性：

- コスト変動要因に熟練者の優れた「勘」「経験」を反映させられる．
- コスト変動要因と影響度を見える化することで工数の説明力が向上する．
- 組織固有のコスト変動要因をモデル化することが出来る．
- 影響度の高い要因の把握による、コストマネジメント力の向上．
- 工数変動量から予算超過確率を把握することができる．
- 組織に共通する要因を把握し､軽減･解消することによるプロセス改善が可能である．
- 熟練者の優れた知見をモデル化し、共有・活用することでアセット化と属人性排除が可能である．

■留意点：

- 構築したモデルに対し，要因関係図の見直しや規模・工数・変動要因のレベルの見直しによる改善を繰り返し行う必要がある．
- モデルの構築に 3 名程度の熟練者が必要となり，かつ 10 件程度のプロジェクト実績データが必要となる．
- 見積り誤差が大きいプロジェクトに特殊性(規模や開発種別の違い等)があった場合，モデルを分けることやモデルから除外を検討する必要がある．

以上