付録

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

■概要

本演習は，ソフトウェア成果物の改善を目的としたレビューについて学ぶ．

演習内容は以下の通りである．

1．レビューの勘所を養うため，簡単な仕様書に対するレビューを個人演習として行う．

2．他の人の視点を知るためレビューの内容を共有する．

■有効性

レビューはソフトウェアテストと同様に欠陥を見つける手段であり，ソフトウェアテストより前の工程で行うので低コストで欠陥を発見することができる．また，ソフトウェアテストにて実施が難しい確認がレビューでは簡単にできる．

ソフトウェアの設計は唯一絶対ではないので，レビューを行うことにより専門家など多くの人の意見をもらうことができ，より良い設計にできる．

■留意点

レビューは限られた時間の中で実施し効果をださなければならないため，レビューでは品質特性のどの特性に対してレビューを実施するかなどレビュー実施時の目的が重要である．効果的に欠陥を指摘するため，複数人でレビューの観点を変えて実施するなどの検討も必要である．

レビューは成果物に対して行うものであり，設計の途中レビューを行っても効果は期待できない．また，チェックリストを埋めるなどのレビュー自体が目的となっている形式的な確認に関しても，効果は期待できない．

●第２回（例会）：オブジェクト指向分析設計：井上　樹氏（豆蔵）

■概要：
　オブジェクト指向分析設計の基盤である「モデリング」が開発でどのように役立つかを知り,モデリングを始めるきっかけにすることを目的として,以下の演習を実施した.

演習は以下の構成で実施
① 要求のモデリング
ストップウオッチを題材にユースケース図とステートマシン図を作成し,要求事項が分かりやすい形で表現できることを体験する.
② 設計のモデリング
ソースコードではなく,クラス図からプログラム構造の問題点を見つけることで,ソフトウアの構造が見える化できることを体験する.
③ モデルとコードの関係
設計モデルから C++のコードが作成できることを体験する.

■有効性：
① 要求のモデリング
開発に関わる関係者間で完成イメージが共有でき,要求が正しく理解できる.
また,要求の不足不備が発見できる.
② 設計のモデリング
ソースコードで読み取れる範囲よりも広い範囲を俯瞰できるため,ソフトウェア構造の問題点に気が付きやすい.
③ モデルとコードの関係
オブジェクト指向に対応したプログラム言語であれば,モデリングツールからソースコードの自動生成が可能である.

■留意点
モデル駆動開発７つの鉄則と呼ばれるものがある
1.高いモデリングスキル
2.ツールの理解
3.モデル駆動開発に対応したマネジメント
4.モデル駆動開発に対応したプロセス
5.モデルの構成管理
6.スモールイン
7.長い目

一定水準のモデリングスキルを身に付け,実際の開発に活用するためには年単位（約３年）での訓練と経験が必要になり,それを許容できる組織体制が必要である.
まずは,長期的な計画のもとに小規模開発から実績を積み重ねていくことが重要である.

●第３回（合宿）：利用者視点でＣＳ向上を図るＵＸデザイン手法の体験学習：金山　豊浩（株式会社ミツエ―リンクス），村上　和治（東京海上日動システムズ株式会社）

■概要：

UX デザイン(User Experience)の概要を学び，「体験」を重視した設計思想で，利用者の目的や意向に沿って効率よく使える設計手法として「ストーリーボード」「スケッチ」「ペーパープロトタイピング」について演習を通して学習した．内容は以下の通りとなる．

1. オリエンテーション
   UX デザインの概要説明．
2. ストーリーボード
   ペルソナの体験（行動・振る舞い）をストーリーとして記述し，絵コンテとして書き表す．
3. スケッチ&プロトタイプ
   ストーリーボードより，必要な機能とデータを洗い出し，それらの関連性を整理し，複数のアプローチをスケッチし，より良いものからプロトタイプを作成する．
4. テスト
   ポスターセッションを通して，プロトタイプのテストを行う．
5. 振り返り
   テストの結果を共有し，妥当性・操作性・理解性・改善点の観点から振り返りを行う．

■有効性：

ペルソナを設定することで「体験」をより具体的にイメージすることができ，チーム間でのターゲット像のズレを防ぐ効果もある．また，スケッチやプロトタイプを紙ベースで設計することで手間やコストが抑えられ，テスト・修正が容易に反映できる点からも非常に有効な手段といえる．

■留意点：

ペルソナを設定する際ペルソナの情報設定が少ないと，ストーリーボード作成時にペルソナの期待・希望・感情が見えにくくなってしまう可能性がある．そのため，年齢や性別などの基本情報だけでなく趣味や人間関係などより深い情報まで設定することが重要である．ペルソナの情報を多く設定することで，ペルソナのストーリーがより見えやすくなることをグループワークにより演習を通じて体感した．

また，ターゲットユーザから良いアイデアを効率よく取得するために，プロトタイプ作成時に細かいアイテム(マウスポインタ・画面枠)や階層画面をより多く作成することが望ましい.アイテムが少ないと,テスト時の画面推移のイメージが湧きにくくなることがあるため，十分なアイテム作成を行うことが重要である．

●第４回（臨時会）:）メトリクスの基礎と GQM 法によるゴール指向の測定:鷲崎　弘宜（早稲田大学/国立情報学研究所）

■概要：
ソフトウェアの品質を把握し改善するために,品質測定評価方法と,落とし穴やコツの説明と演習を実施した.ソフトウェア開発は厳しいものであるが,品質を把握し改善するためにメトリクスの重要性と限界を知り,意思決定に役立てる必要がある.

① 代表的なメトリクス
主なメトリクスとして「規模：コード行数（LOC）やファンクションポイント法」,「複雑さ：サイクロマティック複雑度や結合度」「欠陥：欠陥密度や欠陥除去率」などがある.それぞれ使いどころがあり, ソフトウェアの特定の側面を捉えるために使用する.

② メトリクスの落とし穴と「コツ」
メトリクスの落とし穴として,「ホーソン効果（注目された結果,そのメトリクスの見かけ上の結果が上がる)」「オレオレ品質(井の中の蛙状態で品質がいいと判断してしまう)」などがあり,多面的や客観的な視点で,メトリクスを見ていくことが重要.

③ ゴール指向の測定 GQM
GQM（Goal-Question-Metric）は明確に目標を据えて,目標に対して必要なメトリクスを対応付けるゴール指向な枠組みである.
測定上の目標（Goal）,目標の達成を評価するための質問（Question）,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（Metric）で構成される.

④ 組織目標,戦略測定の整合化 GQM+Strategies
GQM の枠組みを使い,目標に対してメトリクスを取得し評価することができたとしても,有効な組織アクションに繋がっていないという落とし穴に陥ることがある.
そのための手法として,「GQM+Strategies」という概念モデルがある.
上位組織目標と戦略に対してメトリクスを設定し,その上位戦略を下位の組織目標と戦略に分解しメトリクスを設定するという,縦にアクションを繋げる方法である.

■有効性：
GQM+Strategies により,組織目標にひもづいた測定と改善が実施可能となる.
会社の上位組織目標や戦略からのメトリクス収集および改善を実行することも可能であるが,自部門の目標や戦略からのメトリクス収集および改善を実行するなど,実施規模を調節することも可能である.

■留意点：
GQM+Strategies における測定プログラムを運用するにあたり,まずは人材と責任を含めた組織とのコミットメントが必要である. その上で適用範囲や目標を決めて測定,分析,改善のプロセスを実施し,必要であれば目標や戦略を見直しながら繰り返すことが必要である.

●第５回（例会）：要求工学（要求分析）：中谷　多哉子氏（放送大学）

■概要：
ソフトウェア開発において，利用者・顧客によって異なる，また状況に応じて絶えず変化する要求を正しく引き出し，分析・抽出する技術（要求工学）が重要である．
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ついて「大型ショッピングモールでの買い物シーン」をテーマに，各種手法を用いた演習を行った．実践した手法の概要を以下に示す．

1. ペルソナ法
　システムを使用する架空の人物像（ペルソナ）を設定し，その人物の使用状況を想像することで要求分析を行う手法．人物を特定することでシステム対象を明確にでき，行動を想像することで具体的な要求を抽出できる．

2. リッチピクチャによる現状分析
　漫画と吹き出しを用いて，問題が生じている可能性のある状況を可視化する手法．
　各関係者（ステークホルダ）の利害関係や意見・ギャップを表現する．

3. CATWOE 分析
　以下に示す 6 つの視点から，それぞれの立場によって問題をどのように考えているのかを把握する．
　C：Customer（顧客，T の受益者・犠牲者）
　A：Actor（T を行う人）
　T：Transformation Process（ある状態から W に合致する状態へ変換するプロセス）
　W：World View（O の世界観）
　O：Owner（T を推進/中止すべきと考えている人）
　E：Environment（T の達成に必要な環境と資源，守るべき制約とルール）

4. ゴール指向要求分析（KAOS 法）
　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）を詳細化，いくつかのサブゴールに分解を繰り返す手法．サブゴールに分解することで，目的達成のための手段（＝要求）を明確にする．

■有効性：
　リッチピクチャでは,漫画で表現することで周囲の空間にもイメージを広げることができ，関係者からの意見や，問題が生じている可能性のある状況を多数挙げることができた．
ゴール指向要求分析においては，分割したサブゴールで上位のゴールを十分満足させられるかを検討しやすく，要求抽出における抜け・漏れの防止が期待できる．

■留意点：
　CATWOE 分析において，各ステークホルダの立場を意識して世界観をとらえることが重要である．ゴール指向分析では，ゴールの『状態』を明確にし，曖昧な表現とならないよう注意を払う必要がある．
　演習を行った手法以外にも要求の抽出には様々な手法が存在する．それらを適用する際には，明らかにしたい対象を定め，それを知るために何がわかればよいのかを考えた上で，手法を適切に選択する必要がある．各種手法の特徴を十分理解した上で選択することが重要となる．

●第６回（例会）：アジャイル開発演習：天野　勝氏（株式会社永和システムマネジメント）

■概要：
アジャイル開発手法で採用されているイテレーション開発を，折り紙で多面体を作成するプロジェクトを題材に演習で疑似体験した．下記のような形式で演習を実施した．

｛体制と役割｝
PO(プロダクトオーナー) 1 名：要件管理，製品リリースの責任者．
SM(スクラムマスター)1 名：プロジェクトを円滑に進める責任者．
開発者 4～5 名：イテレーション計画会，朝会，開発作業，振り返り会等の実施．
｛工程｝
1 イテレーションを 4 日間(1 日は 11 分)とし，2 イテレーションを実施．
1 日目はイテレーション計画会を実施．
2～3 日目は朝会と開発を実施．
4 日目は朝会と開発を実施後にレビュー会を実施し，リリースを行う．
第 2 イテレーション目は上記に加え，初日に振り返り会を実施．
｛要件管理｝
PO がリリースで実現する要求を決定．PBL(プロジェクトバックログ)として管理し，PO が PBL の優先順位を決定．
｛見積もり｝
優先度の高い要求から，開発者が要求に必要なタスクを洗い出し，1 タスク 2 分でできる作業に分けて見積もりを実施．
｛進捗管理｝
各開発者の作業可視化にはタスクボードを使用．プロジェクト全体の進捗状況の可視化にはバーンダウンチャートを使用．
｛振り返り｝
メンバー全員(PO・SM・開発者)で KPT(Keep/Problem/Try)を実施．

■有効性：
アジャイル開発ではタスクボードやバーンダウンチャートより進捗状況を容易に把握することができる．そしてメンバー全員での朝会実施により開発状況を共有することで，個々の問題点を早期に発見し，迅速に改善することができる．また，KPT を使った振り返り実施により，定期的に開発作業を改善しチームの質を向上することができる．

■留意点
アジャイル開発では事前に正確な見積もりが必要となる．見積もりの精度が低いと PBL の積み残しが増える可能性がある．また，アジャイル開発は品質保証の手法ではなく開発者のための設計手法である．アジャイル開発における品質保証の方法は今後も考慮が必要であると感じた．

●第７回（例会）：工数見積りモデルの構築手法：石谷　靖氏（株式会社三菱総合研究所）

■概要：
「CoBRA（Cost estimation, Benchmarking, and Risk Assessment）法」はソフトウェア開発における見積りモデルを構築する手法である.
構築した見積りモデルにソフトウェアの規模（例：SLOC）とプロジェクト・マネージャー等，見積り熟練者の経験や知識を変動要因として定義することで，透明性と説明性が高い工数（例：人月）見積りを実現する方法となっている.
CoBRA 法ではプロジェクトの開発工数と規模が比例することを理想的であると考える一方で，現実的には工数を増加させる様々な要因（変動要因）によってプロジェクト毎に工数が異なるものであると考えている.
規模に比例するベースの生産性をα，変動要因によって増加する工数をコストオーバーヘッド（CO）とした場合，工数は理想の状態に対して変動要因による CO が加わることとなり次の式で表せる（α は過去のプロジェクトのデータを利用し算出する．CO は熟練者の知見を基にモデルを作成する）.

$$\text{工数（コスト）} = \alpha \times \text{規模} \times (1 + \Sigma\ \text{COi})$$

演習ではこうした CoRBA 法の概要を学習した後，以下グループワークを実施した.
・各グループで各自の経験に基づき変動要因を洗い出し定義する（影響度レベル（工数への影響の大小）の定義を含む）.
・IPA から公開されているツールとモデル例を使用し，下記の手順でモデル構築とモデルの改善を実施する.
1. 変動要因の選択
2. プロジェクト実績データの入力
3. 見積りモデルの構築および改善

・各グループで定義した変動要因に対する改善策をグループ内で議論する.

■有効性：
・過去のプロジェクト実績に加え，熟練者の経験を変動要因（要件の不安定さ，性能要求のレベル，開発期間の制約等）としてモデル化することで，従来の「勘」や「経験」に頼った見積りと比べて説明性が高い見積りが可能となる.
・熟練者 2，3 名の協力と過去プロジェクト（10 件程度）の開発規模および工数の実績データがあれば容易にモデルを構築することができる.
・IPA 提供の公開ツールが利用できることから，プロジェクトに応じた変動要因レベルを定義，再評価するだけで大まかな工数のシミュレーションを実施することが可能である.

■留意点：
・開発規模の見積りは，CoBRA 法の範囲外であることから，別途，当該プロジェクトの着手時（当該プロジェクト見積り作業時）に開発規模を推定する必要がある.
・見積りモデルは見直しを繰り返すことで精度が向上されることから，以下の点を中心に反復的改善を行う.
1. 規模および工数実績データの計測ミスがないか
2. 当該プロジェクト特有の変動要因や特殊性（規模が巨大（極小），開発種別（新規・改良）の差異）がないか
3. 変動要因の定義（レベル）見直し

●第８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：

アーキテクチャの設計手法として ADD（Attribute Driven Design），評価・分析手法として ATAM（Architecture Trade-off Analysis Method）をそれぞれ用いて，「ドーナツ屋の起業」を題材にグループ演習を実施した．本演習の目的は，アーキテクチャの品質特性とそれを表すシナリオ，一般的な実現手法とアーキテクチャの評価・分析の概要について知ることである．「アーキテクチャ」に決まった定義はないが，本演習ではビジネス要件やシステム要件，前提条件および制約条件に基づき導かれるシステムの土台と定義した．

本演習の議題を下記に示す．

1. 品質特性シナリオとは
2. 実現手法の選択
3. アーキテクチャの評価・分析

1.ではアーキテクチャの導出手法のひとつとして，品質特性シナリオを学んだ．品質特性シナリオとは，可用性，変更容易性，性能，セキュリティ性，テスト容易性，使いやすさといった非機能要件を基に，品質目標を具体的に記述したものである．

次いで 2.では，ADD によるアーキテクチャ設計を行った．ADD では品質特性シナリオまたはユースケースから最もアーキテクチャに影響を与えるものを選択し，それぞれの実現手法を割り当ててアーキテクチャを設計する．

最後に 3.では，ATAM で品質特性に関する要求を満たせているアーキテクチャが設計されているかを評価した．アーキテクチャの評価・分析手法として ATAM を適用することで，実現手法のリスクや品質特性を満たすための重要ポイント，およびトレードオフポイントが明確になる．

■有効性：

アーキテクチャはシステムの土台であるため，適切な設計・評価を行うことが求められる．ADD による設計や ATAM による評価・分析を行うことで，問題発生時のリスクを緩和できるとともに，システム開発としての Quality，Cost，Delivery の各改善に寄与できる．また，アーキテキチャが重要な理由は，システムに対する理解を助けるだけでなく，メンテナンス性の向上に期待できる点である．

■留意点：

品質特性シナリオは，具体的な実現手法を考慮せずに作成することが望ましい．考慮してしまうと，非機能要件を十分に抽出できない可能性に留意が必要である．また，異なる知見を持った様々なステークホルダが検討を行うことで，より充足した複数観点からの非機能要件の抽出に期待できる．演習後の振り返りでは下記内容が意見として挙がった．

1. テスト容易性のシナリオ作成が難しい場合は，テスト対象を念頭にすると作成し易い．
2. 新しい技術や手法を理解しておくと，様々な観点からアーキテクチャを設計できる．
3. シナリオそのものの作成が難しい場合は，まず平文で実現したい内容を書き出し，重要ポイントに下線引きしていくと，シナリオ作成につなげ易い．

ATAM による評価・分析は，部門ごとにテーラリングし軽量化してから適用することが望ましい．ATAM は実施すべき手順が多く，工数や人員等の問題でマニュアル通りの実施が困難となる場合が多いためである．

●第９回（臨時会）：テスト演習：鷲崎　弘宜（早稲田大学/国立情報学研究所），猪塚 修（横河ソリューションサービス株式会社）

■概要：
ソフトウェアのテスト技法について,主に単体テストに焦点をあてて,演習で疑似体験した.
ソフトウェアのテストを,「いかにして有効かつ効率的に実施するか」が重要である.
①テストとは何か？
「Myer’s の三角形」による演習を通じて，テストケースの設計（以降，テスト設計」と呼ぶ）に重要な点を学んだ.
・テストには緻密さが必要である．特に，テストケースの洗い出しはロジカルに考える.
・テストケースには具体的なデータの記述が必須である．(テストデータと正解)
・テスト設計を通じて，仕様や設計の不備の洗い出しも可能である.
②テストの基本
・単体テスト
組織や業務分野等により,「単体テスト」の定義は異なる．(今回は関数レベルを対象)
単体テストの流れの概要を以下に示す.
テスト設計⇒レビュー⇒テスト準備⇒テスト実施⇒問題修正⇒報告⇒合格
・テスト技法およびテスト設計
以下のテスト技法について，テスト設計の演習を実施した.
ホワイトボックス／ブラックボックステスト，ディシジョンテーブル法,
同値分割／境界値分析法，パステスト法
テスト設計においては，網羅性だけでなくテストケースの削減の検討も考慮する必要がある．(結果的に「削減しない」という判断も含む)
③組合せテスト
過去の研究により，バグは少数のパラメータの組合せによって顕在することが確認されている.(例：組込みシステムにおいては，2 つのパラメータの組合せテストにより 97%のバグが検出される研究結果となっている.）この点に着目して，影響の大きいパラメータと，その組合せを洗い出して，テストケースを大幅に削減する手法について学んだ.
・直交表
テストの因子（テストの項目に該当）と水準（因子が取り得る値）を縦軸／横軸に配置して，組合せ網羅を洗い出す.
・HAYST 法
直交表を使用したテスト設計の手法である.因子と水準の割り当てを工夫することで,4 桁以上のテストケースが 2 桁まで削減可能な場合もある.

■有効性：
テスト技法を用いることで，網羅性を満たし，かつ効率的なテストを設計することが可能となる．同時にテスト設計を早期に実施することで，仕様の不備やプログラム設計，およびプログラム自体の不備の洗い出しを行うことも可能である.
組合せテストはテスト設計に労力を要するが,大幅なテストケースの削減が可能であるため,肥大化したテストケースの削減には有効と考えられる.

■留意点：
①テスト技法を適用する際は，業務の特性に応じた内容の取捨選択が必要である.
②テストケースの削減を考えることは重要であるが，削減するために要する期間・コスト・人が大きくなる場合もある．現状のテストに要するコストと，テストケースの削減により得られるコストメリットのバランスを考慮する必要がある.