演習コースⅠ「ソフトウェア工学の基礎」 2011 年度 活動報告

Report on Practice Course of Software Engineering Foundations in 2011

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ズ株式会社
浜田 浩史 伊藤忠テクノソリューションズ株式会社
奥井 健 日立製作所 千代 出 日本電子株式会社
阿部 悦子 新日鉄ソリューションズ株式会社 清水 里美 旭化成株式会社
南齋 雄一 株式会社アドバンテスト 高橋 大輔 ベックマン・コールター株式会社
坂 静香 WACATE 実行委員 道脇 直紀 矢崎総業株式会社
山崎 春奈 株式会社インテック 大橋 昭 早稲田大学

演習コース「ソフトウェア工学の基礎」を設置し、演習と議論を通じて実践的および先進的な種々の代表的ソフトウェア工学の考え方や技術を学習した。コースとしては 2005 年度から継続的に設置して 7 年目となる。本稿では、コースの設置背景と狙い、各回における演習の概要、議論や振り返りを通じた実務におけるソフトウェア工学技術適用に関する問題認識、参加した各研究員における活用実践状況について報告する。

Following the success of previous courses in 2005-2011, the practice course of software engineering foundations has been opened in this year. This article reports on the primary aims of this course, summaries of each practice in regular meetings, problem recognition and preliminary application experiments on software engineering techniques learned in the course.

1. コースの狙い

扱う対象がしばしば抽象的で，自由度が高く極めて難しいソフトウェア開発という行為の成功には，理論や経験に裏打ちされたソフトウェア工学技術が欠かせない．しかし，その適用が場当たり的ではかえって複雑さを増すばかりである。そこで、体験や実践を通じて使いどころや留意点を含めて「深く」習得した技術群を体系的に使いこなすことが重要であるが、（特に我が国の）ソフトウェアの多くは、きちんとソフトウェアエンジニアリング（ソフトウェア工学）を学んでおらず、また企業でも十分な体系的教育を受けていない技術者によって作り続けられている[1]と指摘されている。

ソフトウェア工学（Software Engineering）とは、ソフトウェアを開発する際に駆使すべき技術[2]であり、ソフトウェアの開発、運用、および保守に対する系統的で規律に基づいた定量的アプローチ[3]と捉えることができる。ソフトウェア工学の習得と適切な利用により、属人性を排した一定以上の品質保証と高生産の達成が期待でき、上述の品質問題の解決を期待できる。具体的には、Software Engineering Body of Knowledge (SWEBOK、ソフトウェアエンジニアリング基礎知識体系)[3]などの参照による体系的なソフトウェア工学知識の整理と学習に加えて、実践あるいは実践に近い体験を通じたソフトウェア工学技術の習得が必要である。

このような問題意識から本コースは、主に演習と議論を通じてソフトウェア工学技術群を習得する場として 2005 年度より継続して設置され、ソフトウェア工学技術の会得に有効であったとの評価を得ている（[4][5][6][7][8][9]を参照されたい）。そこで 2011 年度も引き続いて、産学両面に通じた講師をお招きし、計 10 名の研究員が参加して、全 10 回にわたり代表的なソフトウェア工学技術に関する講義と演習を実施した。

本稿では以降において、本コースの構成、および、各回における講義・演習の概要、および、議論や振り返りを通じた実務におけるソフトウェア工学技術適用に関する問題認識について報告する。なお、以下の報告は、主に各研究員の分担執筆による。

2. コースの設計と自己評価および工夫

本コースは、設置にあたり以下の 3 点を目的とした。

- 演習を通じた主要なソフトウェア工学技法の体系的かつ深い習得
- 個人・組織の開発力強化のための基盤形成
- 仲間作り（データ収集、技法発展）

その着実な達成のため、本コースでは以下の取り組みを実施した。

(1) 知識体系における位置づけの提示と徹底的な演習

コースの全体構成の設計にあたり、ソフトウェア工学知識体系 SWEBOK およびソフトウェア品質知識体系 SQUBOK[10]上で、2010 年度に取り上げた各技術の位置付けを識別し、マネジメントを除くエンジニアリング系として主要な知識領域を概ね網羅できていることを確認した（図 1, 2）。そのうえで、演習の各回ができるだけ開発プロセスの流れにそって上流系技術から下流系技術と順に並ぶように全体を設計し、各回の「点」と「点」を結び付けて「線」を成し、体系的な学習を促すように配慮した。以上のコースの設計および徹底的に手を動かす演習ベースの講義構成により、本コースはソフトウェア工学技術の体系だった深い習得に有効であった。

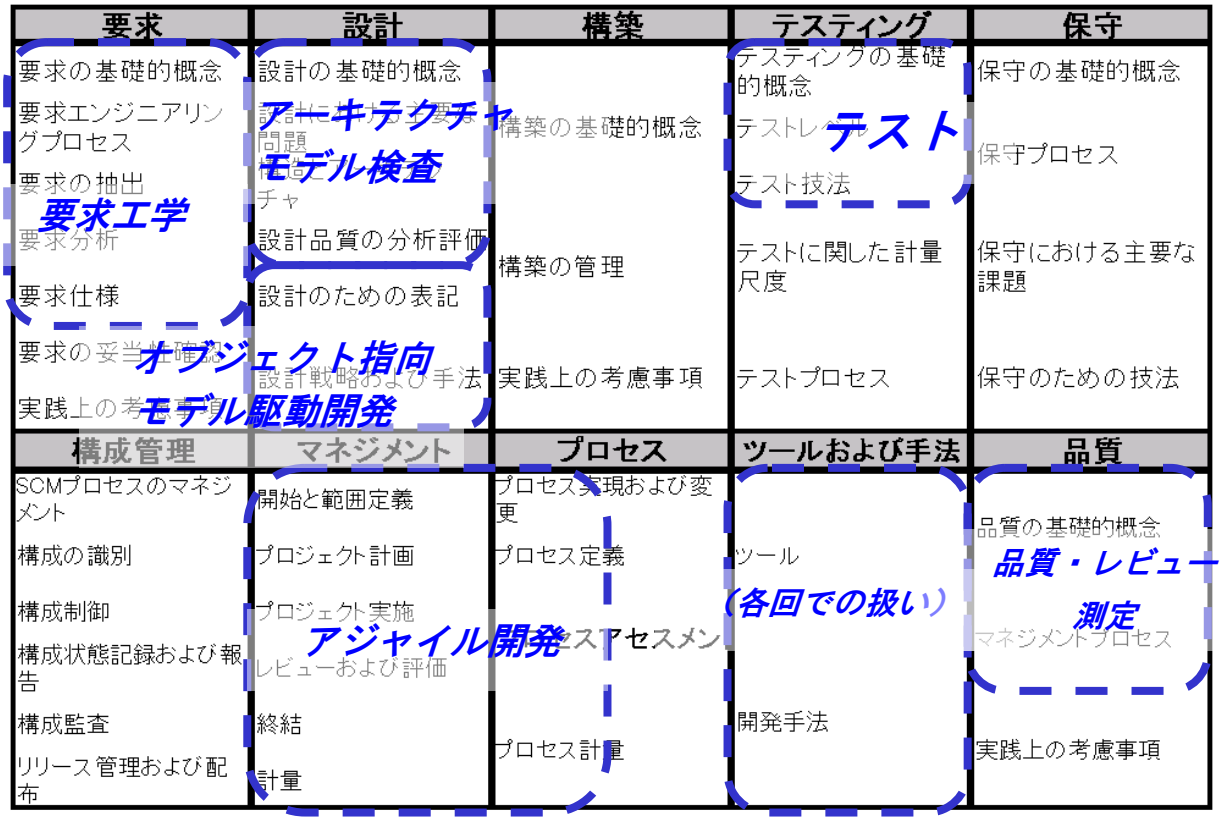

図 1 SWEBOK における各技術の位置づけ

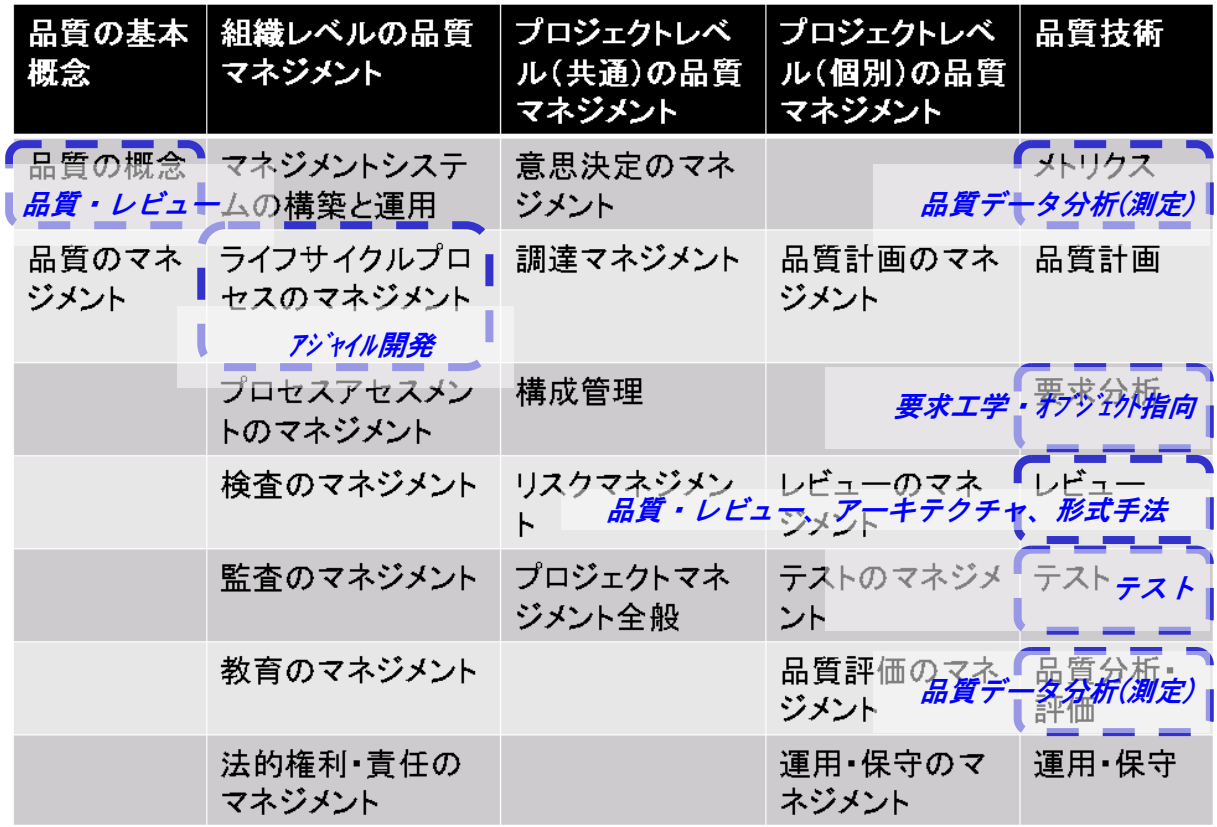

図 2 SQuBOK における各技術の位置づけ

(2) 各回の振り返りと1年間を通じた実践の促し

各回における工夫として、2010 年度に引き続き演習終了後に必ず振り返りの時間を設定し、各自の気づきを箋紙に一件一葉に記述してボード等に張り出して説明し、皆で共有することとした。得られた気づきは、取り上げた技法の実務における有効性や、適用における留意点など多岐にわたり、集団で同じ技法に取り組み多様な観点や立場で考察することの効果が感じられた。

さらに、1年間の冒頭で各自が、習得技法の活用・実践について目標を設定し、その達成について中間状況と最終状況を互いに報告する機会を設けた。これにより、ただ漫然と各回の技法を習得するのではなく、強い目的意識をもって技法を習得しそれを各自の業務や個人へと積極的に役立たせることを幾らか奨励できた。結果として、次節で述べるように実際に業務にて活用できた技法が多数出現し、その実践を通じて開発強化のための基盤形成について一定の達成をみた。

(3) 交流の促進

例会終了後のほぼ毎回の懇親会開催、研究員の年間を通しての固定、および、毎回のグループ演習を通じて、同じような問題意識や目的を持った仲間作りに成功した。交流にあたり、研究員の所属や立場は様々であり、同業他社における事例や、異なる立場・視点に接する機会としても本コースは機能したと考えられる。

3. 各演習における気づきと活用状況

本コースでは、ソフトウェア工学技術の特にソフトウェア開発技術およびマネジメント・プロセス・品質技術に関する以下の演習について、それぞれ個別に講師(敬称略)を招いて実施した。さらに全演習の終了後、各受講者が本コースを通じて得られた「気づき」をそれぞれに報告し、整理してまとめた。具体的には、実務におけるソフトウェア工学技術の活用という観点から気がついた有効性や留意点、さらには各自の所属先や個人における実践・活用状況を各研究員がそれぞれに考察した。本コースに限らず学習行為一般について、その最終目的は学習した事柄によって自身およびその周囲について何ら

かの変化をもたらすことにあり、「気づき」を整理検討することは重要である。計 10 回の演習について、それぞれ整理した結果を以下に述べる。

下記における活用事例とは、本コースのある参加者が実際に、習得した各技術を自身や所属組織等において活用した結果を報告している。2011 年度において既に多数の技術について活用が始められており、前述のように実践を通じて開発強化のための基盤形成について一定の達成をみた。また特にコースの後半にて取り上げた技法については、主に時間的な問題から 2011 年度中の活用には至らなかったため今後の活用が期待される。

- 第1回例会:レビュー演習:猪塚　修氏(横川ソリューションズ株式会社)
  - 概要:レビューとは完成した成果物をレビューアが査読し指摘するものである。演習ではレビューの視点(勘所)を養うことと、多くの人と話す(他の人の視点を知る)事を目的とし、レビュー手法の学習と、シナリオベースのレビュー、品質特性を踏まえたチェックシートでのレビュー演習を行った。
  - 有効性:レビューは実テストより早い段階でソフトウェアの欠陥を除去できる効率的な方法である。レビューにより欠陥を検出すると、テストで検出した場合にくらべ、大きなコストの削減となる。チェックシートを用いたレビューでは、チェックシートによりレビュー対象が要求を満たしていることをより確実に確認できる。また、ソフトウェアの品質特性を踏まえることでチェックシートの品質が向上する。チェックシートの品質は製品の品質にも影響する。シナリオベースのレビューではユーザ操作のシナリオを元にレビューを行う。レビューアがシナリオを元に動作をシミュレーションすることで、様々な問題点を洗い出す事ができる。また、レビューの結果をグループ間で共有することで、多様な視点の指摘を考えることができ、レビューの漏れを減少させることになる。
  - 留意点:レビューを行う際はレビュー担当ごとに観点をかえるなど、重点を決めてレビューをする必要がある。メリハリのないレビュー、形式的な確認しかしないレビューは無駄である。効率よくレビューを受けるため、レビューを受ける側は目的、主張を明確にし、わかりやすく表現することを注意する。レビューを行う際はレビュー報告書を作成し指摘を纏め、次回のレビューや開発に生かすことが大切である。

- 第2回例会:オブジェクト指向分析設計:井上　樹氏(株式会社 豆蔵)
  - 概要:開発において顧客から提示される要求は、顧客側の視点では十分である。しかし、その要求を達成する目的でシステムの利用場面を考えたとき、システムにとって必要な別の要求が隠れていることがわかる。また、システム設計では、その構造を見える化することによって要求が実現できるのか検証することができる。なお、設計を見える化したものとコードとを整合させると、設計から自動的にコードを生成するモデルベース開発によって生産性をあげてゆくことも可能性となる。本演習では開発におけるモデリングの効果を知り、モデリングを始めるきっかけとなるよう要求モデリング、および設計モデリングを行なった。要求モデリングとしてユースケースからユースケース記述およびシナリオ分析を行ない、ステートマシン図等もあわせて設計モデリングへの開発の流れを実習を通じて学んだ。
  - 有効性:ユースケースを用いた要求のモデリングは要求の見える化によって開発者間だけでなく顧客との意思疎通が効率化し、また要求の不備の発見にも貢献することがわかった。また、シナリオ分析やステートマシン図を作成することで、設計フェーズにおいても要求を合理的に踏襲することができ、また設計のモデリング(すなわち、設計の“見える化”)によって、要求の実現性を検証することにも効果を発揮すると理解した。また、設計のモデリングでは高いモデリング技術とツールの利用、そしてプロセスや開発管理の最適化等、ある一定のハードルをクリアすることによってモデルベース開発へ繋げてゆくことができると理解した。
  - 留意点:顧客の“要望”と実現する“要求”、および開発者視点による“システム要件”の区別を理解する必要がある。また、各分析モデルの対応関係を表したドメインモデルをシーケンス図等で検証することが重要である。

- 第 3 回例会（合宿）：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」の言葉の定義は難しいが、今回はアプリケーションアーキテクチャをスコープとし、その品質特性と実現手法、分析/評価の手法について学んだ。システム開発においては、目標とする品質特性を明らかにし、目標を満たすアーキテクチャを構築する必要がある。その手法の演習として、「ドーナツ店の起業」を題材にグループで下記を実施した。
    ① 品質特性シナリオの作成とレビュー
    ② アーキテクチャ概要設計（シナリオと機能要件をアーキテクチャドライバとして実現手法の選択）
    ③ アーキテクチャ分析/評価（実現手法のリスク/トレードオフポイントについて検討）
  - 有効性：適切なアーキテクチャ設計は、システム品質・ビジネス品質・アーキテクチャ自身の品質を実現する。また、システムの理解を助け、アプリケーション実装やメンテナンスで問題が発生するリスクを緩和する。システムの骨組みの段階ですでに品質の作りこみが始まっていることが分かる演習だった。
  - 留意点：アーキテクチャには早い段階での分析/評価が必要である。また、アーキテクトには広範囲な知識、ステークホルダとの調整力も求められる。分析/評価手法のひとつに ATAM（Architecture Trade-off Analysis Method）があるが、メソッドの完遂よりも適切にカスタマイズして導入することが大事。

- 臨時会：要求工学、 要求取得、 要求定義：中谷　多哉子氏(筑波大学大学院ビジネス科学研究科)
  - 概要:システム開発において要求工学が非常に重要な技術であることを、要求抽出を実践するための幾つかの手法を実践し学んだ。実践した手法はリッチピクチャモデル、役割依存モデルの作成、CATWOE の定義、ゴール分析の 4 つ。RODAN のメタモデルの考え方の講義、ゴールモデルの木構造図作成などを交えグループで実施した。
  - 有効性:「要求」を「開発されたシステムが設置された世界像(現実世界と将来の世界との差分)」と捉えた場合、プロジェクトを推進する目的として「要求」を利害関係者間で共有することは有用である。また、システムの開発においては、「要求」の共通性と可変性を意識することで柔軟に変化へ対応できるシステムを構築することが可能となる。
  - 留意点:開発の過程において要求は絶えず変わっていくものであると考えられる。しかし、その根拠となった理念は変わりづらいものであると仮定し、「何が欲しいか？」より、「なぜ欲しいか？」に着目することで変化に耐えうる要求を導き出す。そしてなによりも、自分たちが明らかにしたいもの、「それを知ることで開発の何に役に立つのか」をハッキリさせておくことが重要である。

- 臨時会：ペーパープロトタイピング：浅野　智氏（横浜デジタルアーツ専門学校）
  - 概要：ペーパープロトタイピングとは、デザイナーやエンジニアが開発実施する前に、コンセプトをできるだけ実物に近い形でシミュレーションし、評価する手法である。デザインの持つ問題点や使いにくさの早期発見、また優れたユーザー経験の創出の効果を得ることを目的としている。やり方としては、紙に U/I のプロトタイプをスケッチし、それをユーザビリティ評価して、徐々に完成品に近づけていく。本講習では、以下の手順にて、ペーパープロトタイピングを体験した。
    ① ペルソナ（想定ユーザー）の作成
    ② アクティビティシナリオの作成（想定するシーンに合わせたペルソナの行動を話にまとめる）
    ③ ストーリーボードの作成（シナリオから分解した各タスクと、そのタスクを実行するための U/I を時間軸に沿って作成）
    ④ 発話思考法での評価（ユーザーが思ったことを全て言葉に出して、紙上で模擬的に操作していく）
  - 有効性：開発前に U/I の問題点や使いにくさを早期に発見し、修正が行えることが利点である。デザイナーとユーザーのコミュニケーションを通じて、最終的なデザインをつくりあげていくことができる。また、発話思考法などによるレビュー方法は、直観的に思いついた指摘まで記録できるので、些細

な指摘もカバーすることができる。

- 留意点:ペルソナやシナリオが U/I 作成の前提条件となっているため、誤った定義をしてしまうと、当初の目的と異なるソフトウェアができてしまう恐れがある。そのため、複数のペルソナやシナリオがある場合、その絞り込みが大変重要である。特定のユーザーを意識したソフトウェアに適応しやすいが、汎用性を持たせたいソフトウェアへの適応は注意が必要である。

- 第 5 回例会:ソフトウェアテスト:鈴木　三紀夫氏(ASTER)
- 概要:ソフトウェアテストで用いられるテスト技法を対象として、基礎を学ぶ演習および解説が行われた。テスト技法を知らない人は技法の使い方を学び、テスト技法を知っている人は指導方法を学ぶことを目的として演習に取り組んだ。演習問題は 5 問あり、制御フローパステスト、同値分割テスト、境界値テスト、ドメインテスト、デシジョンテーブルテストについて、テストケースを作成した。演習の流れとしては、まず何も説明せずに個人で演習問題を解き、グループで解き方を共有、検討した上で、模造紙に回答をまとめ、発表した。その後、鈴木氏より各技法についての解説をいただいた。
- 有効性:先に技法を説明して演習を行うのではなく、まずどうやってテストケースを導くか自分なりに考えてみるというアプローチは、技法の必要性を感じるという点で学ぶ側にとって非常に有効である。今回の演習においてそのことを強く実感することができた。更に、「何をテストしたいのか？」「何を同値クラスとするか？」「どこまでテストするのか？」「どの値をテストケースとして選ぶのか？」ということを意識する必要性も学ぶことができた。
- 留意点:現場におけるテスト設計やテスト実装にまつわる悩みや問題点は、ただテスト技法を使えば解決できるわけではない。技法を適用する際には、テストの目的やテストのボリュームを十分考慮する必要がある。「演習だから」という考えで学ぶと、現場に持ち帰ったときにうまく適用できなくなる。どこまでテストすることが現実的か？選んだテストケースでバグが見つかるのか？など、実務に沿って検討することが重要である。

- 第 6 回:モデル駆動開発:久保秋　真氏（株式会社アフレル）
- 概要:モデル駆動開発(MDD)とは、機能や制御といったシステムの特徴を抽象化したモデルを作成し、自動化によってそのモデルから成果物を取り出すソフトウェア開発方法である。抽象度の高いモデルを、ドメインに特化した詳細なモデルに順次「モデル変換」することで開発を進め、最終的にコードを生成する。講義では組み込み開発の現状と MDD の基礎知識、有効性を学び、また MDD に関連した技術群についての知識を得た。演習では準備されたモデルを利用してモデル変換を実施し、コードという成果を自動生成によって得るまでの開発の流れを体験した。
- 有効性:モデル変換は、正確なモデル、設計・実装工程で行なう作業をルール化した変換規則、詳細化されたモデルが必要とする付加情報等を使用することで自動化でき、これは成果物の品質安定化、開発効率の向上につながる。またモデルから生成されたコード中に見つかった欠陥の修正はモデルに対して行なわれ、再度モデルからコードを生成するプロセスを辿ることから、従来の開発にあるような「実装では修正されているが、設計では修正されていない欠陥がある」といった設計と実装の不整合を防止する効果も期待できる。
- 留意点:MDD においてモデル変換の変換規則が非常に重要である。MDD の本質は、「この仕様とデータなら、こう作る」といったルールを定めることで安定した品質の成果を得る、という点にある。また、変換規則をルールとして定めるには人の知識・経験を処理可能なデータにするための労力を伴うことと、変換規則を一度定めれば再利用が可能であることは、留意点として押さえるべき点である。

- 第7回:ソフトウェア測定、メトリクス:野中　誠氏(東洋大学)
- 概要:ソフトウェア開発においては、ソフトウェアの特性や品質をメトリクスにより見える化し、体系的・定量的に欠陥の予測や管理を行うことが重要である。欠陥の定義方法やメトリクスの利用方法について現場の実状を交えながらお話し頂き、演習では、統計解析ソフトを用いてメトリクスなどのデータを分析する手法や、モジュールの修正有無の予測モデルを生成するプロセスについて学んだ。
- 有効性:データの分析により欠陥数の予測やプロジェクトの定量的評価を行うことで、プロジェクト成

功のために意味のある目標値を設定できる点において有効であると感じた。また、欠陥数の予測値に基づいて特定のモジュールを重点的にレビューやテストするなど開発プロセスや技術を改善することで、より高品質なソフトウェアの開発につながると理解した。なお、演習でデータの分析に用いたツールはいずれも無料で入手可能であり、概要を理解すれば気軽に導入できる点についても有用であると感じた。

- 留意点:メトリクスの利用においては、何のために測定を行うのか、測定した値がどのような意味を持つのかについて明確な目的と理解を持つことが重要である。また、実用的な予測モデルを生成するためには、データの定義やモデル導出のためのプロセスに関する深い理解と知識が必要であると感じた。

- 第 8 回: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:アジャイル開発の進め方(KPT によるふり返り、チャートを使った実績管理)、メリット・デメリットなどの基礎知識を講義で理解し、演習で、アジャイル開発を模擬体験する。
  【演習形式】折り紙を使った多面体を作成する PJ にて、アジャイル開発を模擬体験する。2 チームに分かれ、各チーム 1 人は顧客、その他は開発者役になり、顧客に依頼された多面体を折り紙で作成する。

  全日程 1 日=7 分 ×4 日間×2 イテレーション
  内、開発日程:1 日=7 分 ×3 日間×2 イテレーション

- 有効性:KPT によるふり返りや、朝会でプロジェクトの進捗状況が確認でき、バーンダウンチャートやタスクボードで作業が可視化され、効率が上がることがわかった。作業が可視化されたことで、メンバーの達成レベルも分かり無駄の少ない見積の見直しを行え、見積の精度向上につながる。また、短いサイクルで小さい機能を作り上げていくので、都度、顧客と連携してその成果物や進捗の確認もできる。
- 留意点:小規模プロジェクトで効果が出やすい。メンバーのモチベーションや慣れ等で、ふりかえりや朝会の精度が変わる可能性があるかもしれない。モチベーション維持の対策も別途考える必要があるだろう。演習での仕様変更は簡単であったが、実際、顧客相手では中々難しい。どれだけ顧客との連携を上手く行えるかが重要である。マネージャーのネゴシエート力が必要である。また、精度の高い見積もりを行うために、タスクの粒度を揃えたほうがよい。

- 臨時会:モデル検査・形式検証:吉岡 信和氏(国立情報学研究所)
- 概要:形式的検証手法の一つであるモデル検査について実際のツール LTSA を使って学習した。モデル検査は設計の早い段階で数学的手法を用い設計段階での検査をサポートする手法である。演習では、いくつか分散システムのモデルに対して、FSP 言語使ってコーディングを行い、有限状態オートマトンの自動生成とその挙動を確認し検査とデバッグを体験した。
- 有効性:実際にツールを体験してみると、ステートチャート図を FSP 言語で記述するのに慣れが必要ではあるが、オートマトンが自動生成されることや、分散システムでのデッドロックが見つけやすいなど設計段階での検証に大きなメリットがあると感じた。すべての設計をモデル検査するには時間と手間がかかることが想定されるが、特定の高品質を保障する必要がある部品やプログラムバグが許されない装置などの組込ソフトには費用対効果があると思われる。また、先に学習したモデル駆動開発(MDD)との併用も品質向上に期待が持てる。
- 留意点:当然ではあるがモデル化のスキルが求められる。また、検査の結果で不具合を見つけることができても、うまくバグを見つけ出し修正することは、技術者の慣れやセンスに依存することになってしまう。今回は単純なモデルでの検証であったが、それでも複雑なオートマトンが生成される。複雑さ

が増すごとに飛躍的に動作パターンが増加するため、検証部分の切り出しが必要ではあるが、それによるモデルの性質変化が正確な検査の妨げになることも考慮する必要がある。

4. おわりに

本コースでは、指導講師による 10 回の講義・演習を通じて、ソフトウェア開発プロセスの上流から下流までの主要な工学的技術を深く会得した。研究員各位には、本コースを通じて習得した技術や「気づき」を活用し、自身や組織への適用を通じたソフトウェア工学の実践に積極的に取り組まれることを願う。

次年度も、演習内容を改善した上で本コースを実施する。研究員各位には、次年度も本コースに参加して議論を深める、あるいは、他の分科会にて習得技術を適用・発展させるなど、自身や周囲、社会、さらには日科技連へのフィードバックにご貢献いただければ幸いである。また本稿が、この演習コースに対する興味に結びつき、次年度以降の演習コースへの新たな参加につながれば幸いである。その延長線上として、日本のソフトウェア産業の発展に少しでも貢献できれば、著者として望外の喜びである。

謝辞　本稿の執筆にあたって、研究員の方々に草案を分担執筆いただきました。ここに厚く御礼申し上げます。また、毎回の演習をご指導いただいた講師の皆様に、この場を借りて厚く御礼申し上げます。

参考文献

[1] 阿草清滋， 西康晴， 沢田篤史， 鷲崎弘宜，〈特集〉情報専門学科カリキュラム標準 J07: ソフトウェアエンジニアリング領域（J07-SE）， Vol.49， No.7， pp.25-31， 2008.
[2] Pressman， R.S.: Software Engineering – A Practitioner’s Approach， McGraw-Hill， 2005.（邦訳） 西康晴， 榊原彰， 内藤裕史 訳， 実践ソフトウェアエンジニアリング， 日科技連出版社， 2005 .
[3] ISO/IEC/JTC1/SC7: ISO/IEC TR 19759:2005， Software Engineering - Guide to the Software Engineering Body of Knowledge (SWEBOK)， ANSI ， 2007.（最新版は http://www.swebok.org/ より取得可能）（邦訳）松本吉弘 監訳， ソフトウェアエンジニアリング基礎知識体系―SWEBOK 2004―， オーム社， 2005.
[4] 野中誠， ソフトウェア工学演習コース 活動報告， 日本科学技術連盟第 21 年度ソフトウェア品質管理研究会成果報告集， 2006.
[5] 鷲崎弘宜， 猪塚修， 田村一賢， 濱正知美， 麓博之， ソフトウェア工学演習コース 2006 年度 活動報告， 日本科学技術連盟第 22 年度ソフトウェア品質管理研究会成果報告集， 2007.
[6] 鷲崎弘宜， 田村一賢， 阿部修久， 安藤元伸， 古村仁志， 保栖真輝， 溝口文康， 山本文彦， 猪塚修， ソフトウェア工学演習コース 2007 年度 活動報告， 日本科学技術連盟第 23 年度ソフトウェア品質管理研究会成果報告集， 2008.
[7] 鷲崎弘宜， 城間祐輝, 田村一賢, 溝口文康, 大橋剛和, 覚井真吾, 白井孝明, 草場康男, 松宮宏明, 安藤良治, 佐藤和人, 柴田和也, 實藤博, ソフトウェア工学演習コース 2008 年度 活動報告， 日本科学技術連盟第 24 年度ソフトウェア品質管理研究会成果報告集， 2009.
[8] 鷲崎弘宜, 田村一賢, 野中誠, 加藤岡弘一, 上村秀一, 高田祐布子, 中島碧莉, 古木健, 森崎一邦, 横内和城, 吉川真吾, 村上真一, 演習コース「ソフトウェア工学の基礎」 2009 年度 活動報告， 日本科学技術連盟第 25 年度ソフトウェア品質管理研究会成果報告集， 2010.
[9] 鷲崎弘宜, 猪塚修, 野中誠, 小倉徹, 鈴木尚, 片山拡充, 古谷伸一, 中田陽大, 升谷雄二, 吉田麻紀, 本田繁, 長嶋聖, 塩浜龍志, 下條清史, 演習コース「ソフトウェア工学の基礎」 2010 年度 活動報告， 日本科学技術連盟第 26 年度ソフトウェア品質管理研究会成果報告集， 2011.
[10] SQuBOK 策定部会, ソフトウェア品質知識体系ガイド―SQuBOK Guide, オーム社, 2007.