コミュニケーションスキルを重視したソフトウェア技術者教育手法の研究[1]

斎藤 祐一郎†, 久野 靖†

筑波大学 大学院 ビジネス科学研究科 経営システム科学専攻

〒112-0012 東京都文京区大塚 3-29-1

saitou@gssm.otsuka.tsukuba.ac.jp, kuno@gssm.otsuka.tsukuba.ac.jp

概要

近年、アジャイルや反復開発など緊密なコミュニケーションを前提とする開発手法が普及してきている。しかし、他者とのコミュニケーションを拒むソフトウェア技術者の存在をはじめ、現在のソフトウェア業界ではコミュニケーションスキルの向上は必ずしも実現されていない。そこで筆者らは、新人に対しコミュニケーションスキル、特に合意形成スキルを重視した技術者教育を行うことが重要だと考え、教育手法を設計・試行している。具体的には、開発プロセスをこなす中で合意形成の重要性を体験できるような、ケース中心の教育手法を採用している。その検証のため、大学 4 年生 6 名に集まってもらい、それぞれ 3 名のグループに分かれ、ケースを基に開発を進める実験を行った。その結果、コンセンサスを体感するワークショップを行う事で、コミュニケーションをより密にする開発を行う事が出来ることを確認した。

Abstract

Many new software development methods such as agile and iterative development require closer communication among developers compared to traditional ones. However today, many IT engineers are not good at communicating with others. Therefore, we are developing software engineer education curriculum with enhancement of communication skills (more specifically, consensus-building skills) in mind. We are adopting case-centered methods, in which both software development process and importance of consensus can be understood through actual experiences. For evaluation, we have applied our educational case method to six computer science undergraduates, composed in two groups of three persons each. As the result, consensus-building workshop in our curriculum was effective in achieving closer communication.

1 はじめに

これまで日本におけるソフトウェア開発手法は、ウォーターフォールモデル [1] に基づくものが主であった。このモデルでは、ソフトウェア開発は要求獲得、設計、製造、テストのように段階的に進むため、顧客の意見を汲み取るのが初期の要求獲得段階に限られており、製品が完成した後になって要求との不一致が発見されるという問題が多く発生していた [2]。

この問題を克服するため、近年ではアジャイル型開発 [3] や Contexual Design [4] など、顧客と開発者が緊密にコミュニケーションを取りながら開発を進める手法が広まりつつある。

これらの新しい手法においては、その趣旨からも、これまで以上にソフトウェア技術者のコミュニケーション能力を必要とする。その一方で、ソ

*1 Study of software engineering education with an emphasis on communication skills by †Yuichiro SAITO, Yasushi KUNO, Graduate School of Business Sciences, University of Tsukuba, Tokyo.

フトウェア技術者は必ずしもコミュニケーションに長けているとは言えないという現実がある。

たとえば Cusumano [5] や富永ら [6] は、コミュニケーションを苦手とするソフトウェア技術者の存在に言及している。Begel ら [7] は、ソフトウェア企業の新入社員がコミュニケーションを不得手とする傾向にあることを報告している。Weinberg [8] は 1980 年代から、ソフトウェア開発におけるステークホルダ間の意思疎通の不十分さについて警告している。

筆者らは、ソフトウェア技術者がコミュニケーションを不得手とする問題を克服する方法の 1 つとして、新人教育に着目した。そして、ソフトウェア技術者を対象とした新人教育において、技術的事項の学習に加えてコミュニケーション能力の学習も目標とすることにより、必要なコミュニケーション能力を持つ技術者の育成を行う方法を探究している。

本報告では、筆者らが上記の考えに基づいて作成したカリキュラムを少人数の大学学部生グループ (情報専門学科に在籍) に対して適用した試行結果について報告する。以下、第 2 節では、カリキュラム作成における基本方針について説明し、第 3 節では、作成したカリキュラムおよび試行実験の設計について説明する。第 4 節および第 5 節では、1 回目および 2 回目の試行実験の概要および得られた知見について説明する。最後に第 6 節では、今回の試行全体を通して分かった事柄について考察し、まとめを行う。

2 教育カリキュラム作成の基本方針

2.1 新人教育における目標と重視する内容

Katz [9] は管理者にとって持つことが望まれる 3 つのスキルとして、テクニカルスキル (技術的な事項を理解するスキル)、ヒューマンスキル (対人的活動を円滑に進めるスキル)、コンセプチュアルスキル (全体像を把握し新しい体系を構築できるスキル) を挙げている。

筆者らは、これら 3 つのスキルは主体的にソフトウェア開発を進めて行く技術者になるためにも必要と考え、これに基づき、次の目標を定めた。

1. 表面的でない/所属組織にとらわれない汎用的なコミュニケーションスキルの獲得 (ヒューマンスキルに相当)。
2. カリキュラムに盛り込まれている要素技術の理解 (テクニカルスキルに相当)。
3. システムのデザインを行える力の獲得 (コンセプチュアルスキルに相当)。

このうち、本研究の主題であるコミュニケーションスキルについては、現実のソフトウェア技術者に関わる問題として、中村 [10] を参考に、次の 3 項目を重視することとした。

(a) ステークホルダーの役割分担を理解している。
(b) 自らのプレゼンスがステークホルダーに理解されている。
(c) 人に頼る部分・自らが関与する部分を切り分けられる。

この 3 つを選んだのは、次の理由による。まず、ソフトウェア開発は多くのステークホルダーが関与し、さらにチームによる作業である。新人ソフトウェア技術者は大学までの教育において、課題という静的に与えられたものを題材とした、個人による開発の経験しか積んでいないことが多い。このため、個々のステークホルダーの立場や、それらを尊重しなければソフトウェア開発が成立しないことを十分に理解していないことが多い。このために、(a) が必要である。

次に、ソフトウェア開発者以外のステークホルダーは、ソフトウェア開発に関して素人であることも多く、そのようなステークホルダーは自らの役割を十分認識していないのが通常である。このため、ソフトウェア開発者の側から、自分の存在意義や役割を積極的に知らせることも不可欠であり、このために、(b) が必要となる。

最後に、現実のソフトウェア開発では、対象物の規模が大きく、すべてを一人で理解し実装することはもともと期待されていない。このことをきちんと理解し、自分がおこなう部分と他人に任せる部分を分けて考えられることは、スムーズなチーム開発のために不可欠である。このことから、(c) が必要である。

2.2 スキル養成の基準

前節の考察に基づき、(a)〜(c) それぞれについて表 1 のように細分化した評価基準を定めた。この策定に際しては、IPA によって開発された IT スキル標準 ITSS [11] を参考とした。ITSS を参考とした理由は次の通りである。

- わが国で制定された指標であり、わが国の実情に合っていると考えられる。
- 技術面の到達点が明確に定められており、また人間関係構築能力に関する指標も含んだものとなっている。
- 実際にわが国の複数の企業において、ソフトウェア技術者教育の指標として採用されている。

なお、本研究では、新人教育が対象であるため、おもに ITSS レベル 1 に基いている。

2.3 カリキュラムにおける教育方法

カリキュラムにおける具体的な教育方法として、最終的に自立したソフトウェア技術者を育成するという目標から、学習活動内での省察を積極的におこなう自己主導型学習をめざすこととした。このため、インストラクショナルデザイン [12] で用いられる手法を中心に、教育方法の候補として次のものを挙げた。

- 座学
- ケースメソッド
- ロールプレイイング
- ペアプログラミング
- ビデオ観賞
- プロジェクト毎の評価
- ペーパーテスト

これらそれぞれについて、その特徴や弱点を検討した結果、今回はケースメソッドを採用した。その理由は次の通り。

1. 開発工程を通して学習できる
2. 役割分担を設定して学習できる
3. 積極的なコミュニケーションを通じた学習を期待できる

1 については、ケースメソッド形式を使えば企画から設計、開発、テスト、そして完成までを通して学ぶ事ができる。他の手法は、特定の工程にに対して学ぶ事に絞られるため、今回は採用しなかった。

2 については、ケースメソッド形式を用いる事で開発時にプログラマ・デザイナ・テスタと言った役割を設定できる。実際の開発では全員が同じ役割を担う事はほぼなく、より現実に即した状況を作り出す事を意図している。

3 については、1,2 を通じ、各担当者が積極的なコミュニケーションを行う事を必要とする環境を作り出すことができる。その上で、コミュニケーションスキルを向上させるための学習を促すことが期待できる。

3 実験実施計画

3.1 被験者および実験工程の選定

実験の実施計画を作成するにあたり、まず被験者の募集を行った。本研究は新人のソフトウェア開発者を対象とするため、対象とする被験者は情報科学を学んだ大学 4 年生とした。その上で、被験者をを募った所、2 大学から各 3 名の学生を集める事が出来た。

続いて、ケースメソッド形式の実験を実施するにあたり、1 週間の期間を設けることとした。これは、開発工程を全て通すためには 1 日では短すぎ、かつ被験者の指導教員と相談した結果、1 週間を目処に実施する事が望ましいとの指示があったためである。各日程で実施する実験スケジュールを表 2 にまとめた。

また、事前に 3 つのコアスキルに関する事項を調査するアンケートを実施し、実験実施前の被験者のスキル獲得状況を確かめる事とした。

実験 1〜6 日目は、与えられた課題に対してスケジュール通りに工程を終えているか、課題を提出してもらう。

実験 7 日目は、ワークショップ形式でソフトウェア開発を進める中で、実装や仕様の微調整を通じて合意形成能力を培えるかを評価する。特に「ヒューマンスキル」が向上しているかを評価するために、ワークショップの前半・後半を時間で分けて評価を行う。

事後には、改めて 3 つのコアスキルに関するアンケートを実施し、実験実施前との比較を行う。

表 1　スキル養成の基準

項目	内容
ヒューマンスキル	
自分の役割の理解	課題に対し、自分自身の貢献を事前にどのように理解しているか。
グループ内での役割分担の理解	チーム内において、他者との役割分担を理解し、気配りを行き届かせているか。
チーム形成に対する貢献	自分自身が、チーム内で「意思と実践との乖離」「過去の経験」そして「リーダーかメンバーかの特質の把握」をどのように捉えているか。
テクニカルスキル	
プログラミング学習	技術者として技術の根幹となるプログラミング能力を獲得しているか。
システム開発力	プログラミングそのものだけではなく、システムのデザインや、テスト、自動化などの工程全般で必要となる能力を獲得しているか。
工程管理力	システム開発を進めるための進行管理の能力を獲得しているか。トラブルに対する対処能力も含まれる。
コンセプチュアルスキル	
解決力	問題が発生した際に、解決を行うための手段をどれほど持ち合わせているか。情報収集だけではなく、思考法なども含まれる。
情報収集力	情報収集のための手段をどれほど持ち合わせているか。
実践力	情報として自分が持っている知識を、実践を通じて理解を深めているか。
発信	理解した情報を、他者に伝える事によって理解が定着しているか。

表 2　実験スケジュール

日程	所要時間	作業内容
事前	30 分	事前アンケート
1 日目	1〜2 時間	コンセプト作成・外部設計
2 日目	1〜2 時間	外部設計
3 日目	1〜2 時間	内部設計
4 日目	1〜2 時間	内部設計・画面設計
5 日目	1〜2 時間	テスト計画・リソース制作・テストプログラム試作
6 日目	1〜2 時間	テスト計画・リソース制作・テストプログラム試作
7 日目	6 時間	プログラム開発・テスト実施
事後	30 分	事後アンケート

その中で、今回のケース実施によって被験者が能力を獲得できているか、最終的な評価を行う。

被験者の日程の都合上、実験は大学毎に行う。1 回目の実験参加校は α 大学、同様に 2 回目は β 大学と記す。

3.2　ケースの作成

ケースメソッド形式の実験実施に際し、ケースを作成した。タイトルは「Twitter の mash-up サイトを作ろう」である。作成にあたり、次の点を考慮した。

1. 普段慣れ親しんだものを利用する
2. 7 日間で形にできる
3. 役割分担を作る
4. 工程を設定する
5. 能動的に楽しめる

1 については、業務知識を改めて学ばずとも理解できることを目指した。情報系の学生は Twitter をはじめとしたソーシャルメディアを利用している場合が多い。そのため、開発するソフトウェアの企画を被験者自身が育てやすいテーマを選択した。

2 については、短い実験期間で、実際に被験者が早い段階で目で見てソフトウェアが動作いている事を確認しやすいものを開発できることを目標にした。そこで、Mash-up と呼ばれる、既存の Web サービス提供元が外部のソフトウェアと連携する技術を利用した開発を行うよう設定した。

3 については、今回「プログラマ」「デザイナ」そして「品質管理」の 3 分野の分担を策定した。これは、それぞれの役割分担を理解しながら一つ

表 3　成果物の一覧

日程	成果物
第 1 回 (2 日目)	選択したプラットフォームの情報
	コンセプトのメモ
	作業分担リスト
	外部設計の図面
第 2 回 (4 日目)	機能設計書
	画面遷移図
	画面設計図
第 3 回 (6 日目)	テスト計画書
最終回 (7 日目)	完成品のプレゼンテーション
	ソースコード
	テスト結果が記入されたテスト計画書

のソフトウェアを開発する状況を再現するためである。

4 については、組織での開発を通して経験がない被験者に向けて、一通りの開発工程を学べるよう作成した。ウォーターフォール方式とはなっているが、被験者同士のインタラクションを行う必要のある成果物を定義する事で、開発工程を学ぶと同時にコミュニケーションスキルを学べる形をとっている。

5 については、何よりも楽しんで開発できるよう、開発するソフトウェアに自分たちの「意思」を込められる要素を残すようにした。目的として、被験者のモティベーションを高める事で積極的な実験参加を促進し、さらにソフトウェア開発そのものを楽しんでもらえるよう配慮した。

作成したケースは 9 ページあり、実験実施前のオリエンテーション時に配布した。

3.3　成果物の定義

作業が滞りなく進んでいるかを確認するため、実験工程に対応する成果物と納品日を表 3 の通りに定義した。

第 1 回の成果物一覧について、説明を補足する。まず、「選択したプラットフォームの情報」とは、被験者がケースに従ってソフトウェアを開発するにあたって、スマートフォン用か PC 用かの選択をした結論を記述する。次に、「コンセプトのメモ」は、ケースに設定された要求に従ってソフトウェアをどのように実現するかを企画した情報である。最後に、「外部設計の図面」は、twitter と開発するソフトウェアの間での入出力の相関を表す情報を描いたものである。

3.4　評価のための準備

2.1 節で定義した新人教育における目標と重視する内容を基に、実験データを次の 3 つの方法で取得する。

(a)　アンケート
(b)　SYMLOG [13] に基づく集団構造分析
(c)　ビデオ撮影

(a) については、スケジュールにて事前と事後の 2 回実施する。先の評価項目をもとに、事前と事後のスキル獲得状況の推移を、本人自身の認識を基に評価を実施する。

(b) については、グループにおける集団構造を分析するための指標である。個々人の友好度・冷静さ、そしてリーダーシップをとった具合を定量的に表現する。ここから、コミュニケーションスキルの獲得状況を評価する。

(c) については、7 日目のプログラミング・テスト工程の行動分析を行うためにビデオを撮影する。プログラミングやテスト作業時、プログラムの微調整や修正でチーム内のインタラクションが多く発生する事を想定された。そこで、コミュニケーションスキルを獲得する過程をより詳細に分析し、(a) および (b) で評価したデータを裏付けるために用いる。

4　第 1 回実験

4.1　実験実施概要

第 1 回は、α 大学の学生 (以下、α 大学チーム) に対して実験を行った。ガイダンスを 2012/09/01 に、実験期間は 2012/09/16 〜 2012/09/22 の 7 日間である。開催場所は、6 日目までは各自の自宅または大学、7 日目は筑波大学 東京キャンパスである。

4.2　被験者のプロフィール

α 大学チームの被験者は 3 人である。共通する事項として、いずれの 3 人も男性であり、同じ研究室に所属している。そのため、ほぼ毎日顔を合わせており、チーム内での私的な人間関係は良好である。続いて、各個人のプロフィールを、ア

表 4　α大学 ワークショップのスケジュール

時刻	作業内容
10:30〜10:45	本日の作業の流れの整理
10:45〜12:30	各自 担当作業
12:30〜13:30	昼食
13:30〜15:00	各自 担当作業
15:00〜15:50	プログラム・画面 結合作業
15:50〜16:00	問題点の整理
16:00〜16:40	プログラム・画面 結合作業
16:40〜17:00	問題点の整理
17:00〜19:50	プログラム・画面 結合作業
19:50〜20:00	作業結果確認 ソフトウェアの内容のプレゼンテーション

ンケートから取得した情報を基にまとめる。

被験者 A: プログラマ担当。これまで、講義や実験、また私的な活動でプログラミング経験があり、チーム内で最も高いプログラミングスキルを持つ。ただし、ユーザインタフェイスやアイコン等のリソース作成は不得意であり、当初から他のメンバーの支援を求めていた。

被験者 B: デザイン担当。講義や実験でプログラムを書いた事はあるものの、本人は不得手であると認識している。同時に、画面設計やユーザインタフェイス制作で力を発揮したいと希望していた。また、チーム内でリラックスした空気づくりを行えるよう、取り組もうとしていた。

被験者 C: リーダー、及びテスト担当。当初よりリーダーを志望していた。講義や実験はもとより、個人的に Windows 上でゲーム等のプログラミングを行った経験を持っている。また、ゲーム開発経験から、ユーザインタフェイスのリソース作成についても知識があった。そのため、今回の実験で実施する作業の経験範囲は最も広い人物であった。

4.3　実験中の状況

1〜6 日目は、被験者各自で課題に取り組み、予定通り成果物の提出を受けた。

7 日目のワークショップは、プログラムを完成させる事を目的とし、被験者に作業に取り組んでもらった。実際の作業スケジュールは、表 4 の通りとなっている。

SYMLOG を用い、α大学チームの集団構造を分析した結果を、図 1 に示す。評価は、筆者自身のみが実施した。

バブルチャートの縦軸は数値が高い程友好的な接し方を心がけていた事を示し、横軸は右へ行く程に冷静な対応を示している事を示し、円の半径が大きい程リーダーシップを発揮している事を示している。前半は主に 10:30〜15:00 の間、後半は主に 15:00〜19:50 の間を対象としている。「被 A」とは被験者 A を指し、B, C についても同様である。

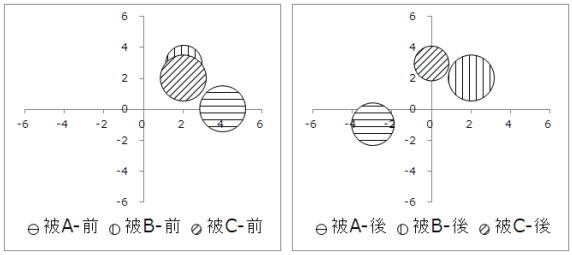

図 1　α大学 ワークショップ内での構造評定
(前半:左 後半:右)

続いて、ビデオ及び最終アンケートのデータを織り交ぜつつ、被験者毎に評価する。

被験者 A: 前半と後半で冷静さの観点で評価に大きな変化が発生している。後半に入り、プログラムが画面と結合する際に適切に結合が進まない状況が発生した際、冷静さを失って行く推移が認められた。心理面では、本人の想像を超えてスムーズに行かないいらだちを募らせていたようである。また、ビデオからは後半に入り急速に顔色や髪型の乱れが出ていた事を確認した。

被験者 B: 画面の準備作業はワークショップ前にほぼ完了しており、微調整を施すのみであった。また、積極的にリーダーシップをとる事はなかったが、状況に応じて冷静に行動する事を忘れずに実行し、徐々にリーダーシップを発揮するようになっていた。しかし、最後には被験者 A の冷静さを失って行く態度に、少し萎縮し始めている所も伺えた。

被験者 C: 当初はリーダーシップを発揮し、チーム全体の方向を整理しながら作業を進める事が出来ていた。また、メンバー同士で友好的に接するよう、コーヒーをつぐ等の気配りをする様を垣間みた。しかし、被験者 A が冷静さを失う

中で自分自身がどのような貢献をして行けば良いかわからなくなり、行動が徐々に抑制されて行く様子を確認する事が出来た。

4.4 結果の評価

表 1 をもとに、α 大学チームの実験結果を評価する。

まず、ヒューマンスキルについて述べる。チーム内において要求仕様の内容は共有・合意できていたようである。その一方、2 つの問題点が確認できた。

1. 進捗確認のタイミングが遅かった
2. プログラム実装部分の擦り合わせが少なかった

1 については、プログラムと画面の結合作業に滞りが出ている事が表面化するのが遅れ、作業の遅延が発生した。これは、スケジュール通りに作業が進んでいるかの合意が逐次とれていなかった事が原因の一つであると考えられる。事後アンケートから、被験者 C のコメントにも同様の記述が示されていた。

2 については、被験者 A,B に確認をとった所、互いに自分自身が考えている事は相手も同じ水準で問題を理解しているはずだと考え、連絡を密に取らない「問題の理解度合いの齟齬」が生じていた。その結果、画面とプログラムを結合する際、変数やロジックで結合がうまく行えない状況が発生した。尚、事後のアンケートから全被験者から今回の実験の問題点であったとの回答を得ている。

続いて、テクニカルスキルについて述べる。各自が元々作業分担に必要な技術を保持しており、プログラムと画面の結合直前までは順調に作業が進んでいた。更に、アンケートによると、実験を通じて特に企画とデバッグの能力を伸ばす事が出来たと回答を受けた。

最後に、コンセプチュアルスキルについて述べる。アンケートより、今回の開発に必要な情報が何かを事前に調べ上げ、API ドキュメントの読解力をはじめとした能動的な情報収集力が向上したとあった。同時に、これまでの講義による受動的な学習から変化があった。

5 第 2 回実験

5.1 事前指導の追加

第 1 回の実験を踏まえ、合意形成スキルを養うには「問題の理解度合いの齟齬」を埋めて行く事が必要であることがわかった。そこで、被験者に対し事前に「問題の理解度合いの齟齬」を知り、改善するための教育を施す必要があると考えた。その方法として次の 2 つを検討した。

- 小規模 (30 分程度) のワークショップ
- 各工程毎に必要となる合意形成の内容を座学で教育

今回は、短時間で身につけられる小規模のワークショップを実施する事とした。ワークショップの内容は、インターネットブラウザに必要とされている機能について討議を行うこととした。各人の「必要」についての認識の違いを認識し、そして最後はチーム内で意見をまとめる事で、相互の理解の齟齬を埋めていく重要性を知ることができるよう設計した。

5.2 実験実施概要

第 2 回は、β 大学の学生 (以下、β 大学チーム) に対して実験を行った。ガイダンスは 2012/10/03 に、実験期間は 2012/10/14 〜 2012/10/20 の 7 日間である。開催場所は、6 日目までは各自の自宅または大学、7 日目は筑波大学 東京キャンパスにおいて実施した。

5.3 被験者のプロフィール

β 大学チームの被験者は 3 人である。共通する事項として、いずれの 3 人も男性であり、被験者 Y, Z は同じ研究室に所属している。また、α 大学チーム程ではないが日々の交流も少なからずあり、チーム内での私的な人間関係は良好である。以下、各個人のプロフィールを、アンケートから取得した情報を基にまとめる。

被験者 X: プログラマ担当。これまで、講義や実験でプログラミング経験があり、チーム内で最も高いプログラミングスキルを持つ。尚、個人的にプログラミングを行う事は余りないとのことであった。

被験者 Y: デザイン担当。講義や実験でプログラムを書いた事があり、かつユーザインタフェ

表 5　β大学 事前指導のスケジュール

時間	作業内容
19:00〜19:20	(全体像説明)
19:20〜10:35	必要な機能を各個人で列挙
19:35〜19:40	全員の情報を集めて共通部分とそうではない部分を分類
19:40〜19:47	共通しなかった部分から 1 つを選び選択理由を考える
19:47〜19:50	選択した項目についてプレゼンテーションを行う
19:50〜21:00	(実験に関する説明)

イスに関する研究にも取り組んでいる。事前のアンケートでは、プログラミングには自信が無く、他のメンバーにお願いしたいと希望を出していた。

被験者 Z: リーダー、及びテスト担当。当初よりリーダーを志望し、チームメンバー募集にあたっても積極的にメンバー集めを行っていた。事前のアンケートでは、講義や実験でプログラミング経験があり、かつ他のメンバーよりも広い範囲で技術の知見を持っていることがわかっている。

5.4　事前指導の状況

事前指導は、実験の説明に入る前に実施した。これは、実験の意図を理解してから事前指導を実施してしまうと、合意形成のプロセスと重要性を理解する意識をそいでしまう可能性があったためである。説明の流れは、表 5 に示す。

インターネットサイトを閲覧する「ブラウザ」は、情報系の学生でなくとも普段慣れ親しんだソフトウェアの 1 つである。そのソフトウェアであっても、必要となる機能の認識は各々が違うことを知ってもらい、認識をすりあわせていく大切さを伝えることができた。

5.5　実験中の状況

1〜6 日目は、被験者各自で課題に取り組み、予定通り成果物の提出を受けた。

7 日目のワークショップは、α大学チーム同様、被験者に作業に取り組んでもらった。実際の作業スケジュールは、表 6 の通りとなっている。

α大学チームのスケジュールを表した表 4 に比べ、作業開始当初から密に進捗状況の共有が行

表 6　β大学 ワークショップのスケジュール

時刻	作業内容
11:10〜12:30	技術調査
12:20〜12:30	調査状況 共有
12:30〜13:30	技術調査・プログラミング
13:00〜13:10	作業進捗 共有
(13:20〜13:40)	昼食・技術調査・プログラミング
13:40〜13:50	調査状況 共有
13:50〜15:00	技術調査・プログラミング
15:00〜15:15	作業進捗 共有
15:15〜16:30	技術調査・プログラミング
16:30〜16:55	作業進捗 共有
16:55〜17:00	問題点の整理
17:00〜18:20	本日の作業を踏まえ設計資料の更新
18:20〜18:30	作業結果確認 ソフトウェアの内容のプレゼンテーション

われている。これは、当初実施した事前教育を通じて、合意形成を行う重要性が理解され実践されている一つの現れと考えられる。

SYMLOG を用い、β大学チームの集団構造を分析した結果を、図 2 に示す。分析方法、及び描画方法については、図 1 と同様である。前半は主に 11:00〜14:00、後半は主に 14:00〜18:30 を指している。

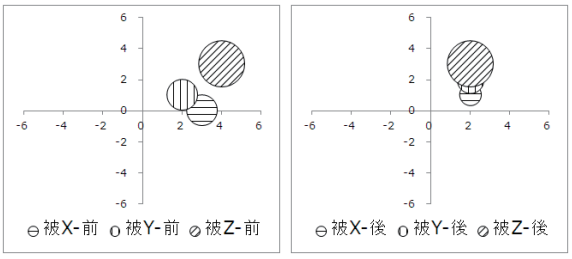

図 2　β大学 ワークショップ内での構造評定 (前半:左 後半:右)

以下に、ビデオ及び最終アンケートのデータを織り交ぜつつ、被験者毎に評価を行う。

被験者 X: チームワークを大切にする姿勢を垣間みる事が出来た。被験者 Z の調査について積極的に耳を傾け、自身のプログラム開発に活かそうとする姿勢が随所に出ていた。一方、相対的で

はあるが終始受け身の姿勢になりがちであった。自ら意見を述べる事は余りなく、指示のもとで着実に自分自身の作業を進める状況であった。

被験者 Y: 被験者 X,Z の作業内容を理解し、自分が今何をすべきかを考えながら行動しようとする姿勢が特徴的だった。しかし、やるべき事に対して自分自身の技術スキルが追いついていない事に、わだかまりを感じていたことをアンケートから確認する事が出来た。

被験者 Z: 当初よりリーダシップを発揮していた彼は、ワークショップ中もその姿勢を遺憾なく発揮していた。自分自身の作業はもとより、技術調査にあたっては被験者 X では手が回らない部分について分担して調査を行い、twitter API の構造について知見を深めていた。そして、深めた知見を滞りなく共有する事で、作業を進めるにあたってどのような知識が必要かをチーム全体で共有できていた。

5.6 結果の評価

表 1 をもとに、β 大学チームの実験結果を評価する。

まず、ヒューマンスキルについて述べる。β 大学チームは α 大学チーム同様、チーム内で要求仕様の内容は共有・合意できていたようである。加えて、表 6 にも現れている通り、ワークショップの前半からチーム内でのコミュニケーションが頻繁に行われ、現状の確認と次の作業の合意を逐次行えていた。

次に、テクニカルスキルについて述べる。スケジュール管理は、適切に行われていたことは評価できる。一方で、問題が 1 つ発生していた。開発時に利用する node.js に関する調査が進んでいなかった。特に、Web アプリケーションを実装する際に必要となる、HTTP の GET, POST メソッドの動きを初めとした基礎知識が不足していた事が大きな問題となった。

最後に、コンセプチュアルスキルについて述べる。事後アンケートやビデオを分析すると、今回は被験者毎に差が出ている。情報収集については、特に被験者 Z の積極性が際立っていた。一方、問題も発生した。後半、問題の解決方法に適切ではない手段を選んでいた。具体的には、実装方法を当初計画していたサーバサイドを中心とした方法から、クライアントサイド中心の実装方法に切り替えようとしていた。残り 3 時間の状況で、大きな設計変更は開発が予定通り完了するとは考えにくく、適切な判断ではなかった。

6 考察とまとめ

α および β 大学チームの実験結果を比較すると、β 大学チームに対して事前に施した教育によって、よりコミュニケーションを密にした開発を実施する事が出来るようになったと考えられる。

ソフトウェアの完成度については、α 大学チームの方がより進んでおり、テクニカルスキルは高かったと言える。しかし、個人で開発する事とは違い、分担を決めて開発を進めるためにはテクニカルスキルだけでは開発は難しい事を確かめられた。

また、ソフトウェア開発において、技術者のコミュニケーションスキルは養成する事が可能である事、そして会話の得手不得手によらない後天的な教育によって学ぶ事ができることを確認した。また、技術者のコミュニケーションは合意形成、即ち自分自身の理解と相手の理解水準を合わせ、一つの結論に導くためのスキルであることを知る事が出来た。

一方で、実験を通じて以下の必要性を確認した。

1. ソフトウェア開発の工程を事前に学ぶ
2. コミュニケーションを学ぶ手法の更なる開発
3. ソフトウェア開発ケースの更なる整備
4. コミュニケーションスキルを測定する指標の開発

1 については、事前に開発工程を学習し、一人でソフトウェア開発を完遂させる技術を養った後に教育を実施する事が、コミュニケーションスキルを高める教育をより効果的にできることがわかった。事前のアンケートでは、ソフトウェア開発工程の流れを学んだ学生が誰一人としていなかった。

2 については、事前の教育としてどのような教育を施す事が必要か、今回の実験結果を踏まえて更に検討を進めていきたい。特に、工程毎に必要となるインタラクションとは何かをより深く分

析し、問題点を確認しながら進める事になると想定している。

3 については、今回の Web アプリケーションに限らず、ソフトウェア開発は業務システムや組み込みシステム等、数多くの分野に対応した教育カリキュラムを開発を行いたい。その上で、ソフトウェア開発ならば幅広く適用できるカリキュラムを整備したい。

4 については、心理学や教育学にコミュニケーションスキルそのものを測定する手法は存在する。さらに、ソフトウェア開発に焦点を当てた合意形成を中心としたスキル、及びソフトウェア開発スキルとの相関を確かめる測定指標を開発するのである。測定指標の開発を通じ、コミュニケーションスキルを獲得したことをを確認する事もさることながら、テクニカルスキルに偏らない、よりバランスのとれた技術者を養成する一つの指標として活用できるようしていきたい。

謝辞

今回、実験参加を承諾いただいた学生及び指導教員の皆様に深く御礼を申し上げる。協力無くして本研究は成立し得なかった。

また、アンケート作成にあたり、アドバイスを頂戴した勤務先の学生アルバイト 2 名にも有益なコメントをいただいた。

そして、本報告の執筆にあたって、時間を割く事に協力いただいた勤務先や家族へ感謝したい。

参考文献

[1] W. W. Royce. Managing the development of large software systems: concepts and techniques. pp. 328–338, March 1987.

[2] 日経 BP 社, 日経コンピュータ編集部. 動かないコンピュータ: 情報システムに見る失敗の研究. 日経 BP 社, 2002.

[3] Ward Cunningham ほか. *Manifesto for Agile Software Development.* http://agilemanifesto.org/, 2001.

[4] Hugh Beyer and Karen Holtzblatt. *Contextual Design: Defining Customer-Centered Systems (Interactive Technologies).* Morgan Kaufmann, 1997.

[5] Michael A. Cusumano. *The Business of software: what every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad.* Free Press, 2004.

[6] 富永 真己, 古川 照美. 日本人のコンピュータ技術職における労働職場環境のストレッサーとコーピング特性が精神的健康度に及ぼす影響について. 弘前大学医学部保健学科紀要, Vol. 6, pp. 1–9, 2007.

[7] Andrew Begel, Beth Simon 著, 久野 禎子, 久野 靖 訳. 若葉マークのプロ: 最近の卒業生、初めてのソフトウェア工学のお仕事, 第 26 章. Making Software — エビデンスが変えるソフトウェア開発. オライリージャパン, 2011.

[8] G.M. ワインバーク 著, 木村 泉 訳. システムづくりの人間学. 共立出版, 1986.

[9] Robert Lee Katz. *Skills of an Effective Administrator.* Harvard Business Review Classics. Harvard Business Press, 1974.

[10] 中村和彦. 山口裕幸 (編), 『コンピテンシーとチーム・マネジメントの心理学』, 2009 年, 朝倉書店. 社会心理学研究, Vol. 26, No. 1, p. 95, August 2010.

[11] 独立行政法人情報処理推進機構. ITスキル標準 (ITSS). 独立行政法人 情報処理推進機構, 2002. http://www.ipa.go.jp/jinzai/itss/.

[12] William W. Lee, Diana L. Owens, 日本イーラーニングコンソシアム, 康敬清水. インストラクショナルデザイン入門 : マルチメディアにおける教育設計. 東京電機大学出版局, 2003.

[13] 奥田 達也, 伊藤 哲司. SYMLOG の日本語改良版—小集団構造把握のための簡便な評定項目の作成 (対人的相互作用¡特集¿). 実験社会心理学研究, Vol. 31, No. 2, pp. p167–174, 1991.