情報処理学会論文誌　教育とコンピュータ　Vol.1　No.2　7–18 (Mar. 2015)

論　文

コミュニケーションスキル獲得を促すソフトウェア技術者教育の試行

斎藤 祐一郎[1,a)]　久野 靖[1,b)]

受付日 2014年7月11日，再受付日 2014年10月14日，
採録日 2014年12月20日

概要：コミュニケーションは，企業内でソフトウェア開発を遂行するうえで不可欠である．しかし，現状ではコミュニケーションが苦手なソフトウェア技術者が存在し，新人技術者にも同様の傾向が見られる．そこで筆者らは，新人技術者がソフトウェア開発に必要なコミュニケーションスキルを獲得できるような教育方法を考案し，ソフトウェア技術者予備軍である大学 4 年生を対象に試行のうえ，有効性を評価した．教育方法には，コミュニケーションの重要性を体感できるケースメソッドとロールプレイング形式を組み合わせたものを採用した．被験者として情報系大学 4 年生 2 グループ各 3 名を集め，コミュニケーションに関する事前指導を片方のグループのみ施した．その結果，事前指導を施したグループにおいて，特に成果物を基にチームメンバと接するコミュニケーションスキルを獲得できたことを確認した．

キーワード：コミュニケーション，プログラミング，ソフトウェア開発，教育，プロジェクト・ベースド・ラーニング

Software Engineer Education Curriculum Incorporate Enhancement of Communication Skills

Yuichiro Saito[1,a)]　Yasushi Kuno[1,b)]

Received: July 11, 2014, Revised: October 14, 2014,
Accepted: December 20, 2014

Abstract: Communications are indispensable in professional software development activities. However, there are many software engineers, especially freshmen, who are not good at communication. Therefore, we have proposed training curriculum to enhance communication skills for them. Our method combines role-playing and case methods. We also have tried our curriculum to six computer science undargraduates. They were organized in two teams, each consisting of three, and one of the teams had additional pre-workshop to stress importance of communication. As the result, our curriculum seems to be effective for the team that taken pre-workshop, and the workshop was useful in enhancing the effectiveness of the course.

Keywords: communication, programming, software development, education, project-based learning

1. はじめに

企業活動としてのソフトウェア開発を遂行するためには，コミュニケーションは欠かせない．そして，チームで開発を遂行するためには，メンバごとに割り当てられた役割に基づいてチームメンバとコミュニケーションを図る必要がある．

Wood [9] は，コミュニケーションには一般に 4 つの側面があると述べており，その 1 つとして，人がシンボルとインタラクションを通じてコミュニケーションの内容を理解することをあげている．特に，ソフトウェア開発では，要求が持つ背景や作成した成果物が技術者同士のコミュニケーションに影響を与えると考えられる．

しかし，現状ではコミュニケーションが苦手な技術者 [5] が存在し，その傾向は新人技術者においても同様である．たとえば Begel [3] は，入社 1〜7 カ月目の新人技術者を対

[1] 筑波大学大学院ビジネス科学研究科
Graduate School of Business Sciences, University of Tsukuba, Bunkyo, Tokyo 112–0012, Japan
[a)] koemu@mx.koemu.com
[b)] kuno@gssm.otsuka.tsukuba.ac.jp

象とした行動観察に基づいて，新人技術者は，他人に質問する際に深入りし詳細を追求しなさすぎるなどの形で，適切なコミュニケーションが不足することがあると述べている．

本研究では，新人技術者となるにあたって必要となるコミュニケーションスキルを，技術的なスキルとあわせて獲得できるような教育方法を提案し，コンピュータサイエンスを専攻する大学生に対する試行に基づいて評価した．

カリキュラムの設計および試行においては，以下の方針をとった．

(1) チームでソフトウェア開発を行う経験を積んでもらう．
(2) コミュニケーションの中でも特に「ソフトウェア開発におけるコミュニケーション」に焦点を当てる．
(3) 教育期間を 1 週間程度とする．

(1) は，1 人で開発するときとチームで開発するときの違いを知ることができるようにするためである．(2) は，本研究では「ソフトウェア開発に必要なコミュニケーションスキル」の具体的内容や，それを向上させる方法を探究することが目標であることに基づいている．(3) は，被験者の研究や他の講義にできる限り影響が出ないよう配慮するためである．

2. 関連研究

ソフトウェア開発プロセス，コミュニケーションに関するモデル，および大学の情報専門学科におけるカリキュラムについて，関連研究を紹介する．

ソフトウェア開発プロセスは非常に多く存在するが [15]，代表的なものとしてウォーターフォール型開発モデル [7] とアジャイル型開発モデル [10] を取り上げる．コミュニケーションの観点に注目すると，ウォーターフォール型開発モデルでは，ドキュメントをコンテキストとし，技術者同士や技術者と顧客の間でコミュニケーションを行う．アジャイル型開発モデルでは，技術者内ではソースコードや開発工程管理ツールなどを通じてコミュニケーションを行い，技術者と顧客の間ではペルソナ [1] などのひな形や出来上がったソフトウェアを基にコミュニケーションを行う．つまり，ウォーターフォールでもアジャイルでも，成果物を基にしてコミュニケーションを行う点では同様であり，これは他の開発プロセスにおいてもあてはまるものと考えられる．

コミュニケーション一般に関するモデルとして，管理者に求められる 3 つのコアスキル [6] を取り上げる．3 つのコアスキルとは，テクニカルスキル，ヒューマンスキル，そしてコンセプチュアルスキルからなる．テクニカルスキルは業務遂行に関わるものであり，技術者であればプログラミングや設計のスキルにあたる．ヒューマンスキルは，対人活動を円滑に進めるスキルであり，ネゴシエーションやコミュニケーションのスキルが該当する．コンセプチュアルスキルは，全体像を把握し新しい体系を構築するスキルである．3 つのコアスキルは，職位によって相対的に求められる分量が変化する（図 1）．その中で，ヒューマンスキルはどの職位にあっても等しく求められるスキルである．

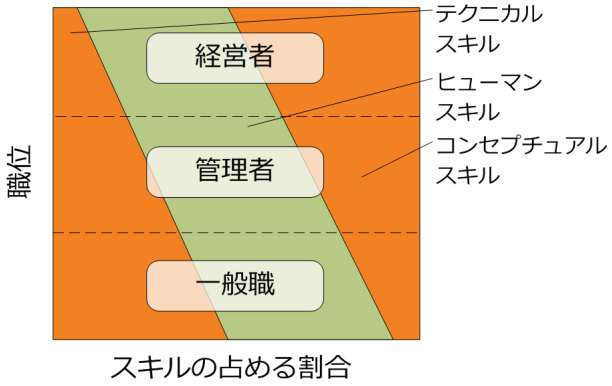

図 1　管理者に必要な 3 つのスキル（文献 [6] を参考に作成）
Fig. 1　Skill model for administrator [6].

大学の情報専門学科で用いられている，ヒューマンスキルなどを意識したカリキュラムとして，マイアミ大学などで行われている，ソフトウェア工学のカリキュラムがある [2]．これは，ソフトウェア工学のカリキュラムの中に，コミュニケーションスキルに関する内容を組み入れている．背景に，開発現場から新人技術者にはコミュニケーションスキルが不足している点が批判されたことにある．このカリキュラムは，コミュニケーションスキルを「読み」「書き」そして「話す」の 3 つの力に分類し，さらに細分化して組み立てられている．そのため，コミュニケーションスキルを高める点では本研究と類似している．しかし，教育効果を定量的に評価する方法の言及があまりなされていない．

3. 教育方法の設計

3.1 育成するスキルの内容

前章の内容を基に，本研究におけるコミュニケーションスキルを「ドキュメント・ソースコードなどの成果物を基にチームメンバ同士がインタラクションを行うスキル」と定義した．この定義，および 3 つのコアスキル [6] を基に，教材を通じて育成を目指すスキルを 5 つ定義する．

(1) ステークホルダの役割分担を理解し，業務に結び付けられている（ヒューマンスキル）．
(2) 自らのプレゼンスがステークホルダに理解され業務で活かされている（ヒューマンスキル）．
(3) 人に頼る部分・自ら解決する部分を切り分けながらコミュニケーションを行える（ヒューマンスキル）．
(4) カリキュラム内に織り込まれている要素技術を理解し製品開発に活かしている（テクニカルスキル）．
(5) システムのデザインを行うことができる力が育まれる（コンセプチュアルスキル）．

(1)〜(3) は，開発チームという組織内で行動するうえ

表 1　育成するスキルの一覧
Table 1　Communication skill list.

項目	内容
ヒューマンスキル（HS: Human Skill）	
自分の役割の理解（OR）	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（Own Role）
グループ内での役割分担の理解（RU）	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（Role Understanding）
チーム形成に対する貢献（CT）	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「リーダかメンバかの特質の把握」をどのようにとらえているか（Contribution to the Team）
テクニカルスキル（TS: Technical Skill）	
プログラミング学習（PG）	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（ProGramming）
システム開発力（SD）	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・テスト・そして自動化などの工程全般で最低限必要となる能力を獲得しているか（Systems Development）
工程管理力（PM）	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（Process Management）
コンセプチュアルスキル（CS: Conceptual Skill）	
解決力（SA）	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（Solution Abilities）
情報収集力（IG）	情報収集のための手段をどれほど持ち合わせているか（Information Gathering）
実践力（EA）	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（Execution Abilities）
発信（EX）	理解した情報を他者に伝えることによって理解が定着しているか（EXpression）

で，自分自身以外のメンバの存在と役割を通じ，組織にどのように関わっているかを理解できるようにするものである．(4) は，技術者に対する教育であるため，業務を遂行するにあたり必ず求められるソフトウェア開発の技術を学んでもらうためである．(5) は，将来にわたってソフトウェア開発の技術を獲得し続けるために，市場や顧客からの要求や技術の変化に対する理解を通じて課題を発見し，自ら解決するための力を養うためである．テクニカルスキルについては，独立行政法人情報処理推進機構（以下，IPA）が作成した IT スキル標準 [13]（以下，ITSS）レベル 1 を基準に設定した．これは，新人技術者として業務を遂行するために最低限必要となるスキルを養うためである．

以上から，育成するスキルの内容を**表 1** のとおりに定義した．ヒューマンスキルはコミュニケーションスキルを対象とし，テクニカルスキルは新人技術者としてはなくてはならないスキルを対象に設定し，そしてコンセプチュアルスキルはシステムに対する要求を理解し形にするための力を育成することとした．

3.2　技術者の関与するステークホルダの範囲

コミュニケーションスキル獲得をめざすにあたり，関与するステークホルダの範囲について**図 2** の 4 つの領域を定義する．これは，人間関係を新人技術者が業務を通じて構築するときの領域の段階を示したものである．

本研究では，「子弟・先輩後輩」および「チームや部署内」の範囲を対象とする．「子弟・先輩後輩」は必ず関与することとなる人間関係のため必要であり，また「チームや部署内」は，チームで開発するためにはなくてはならない人間関係だからである．

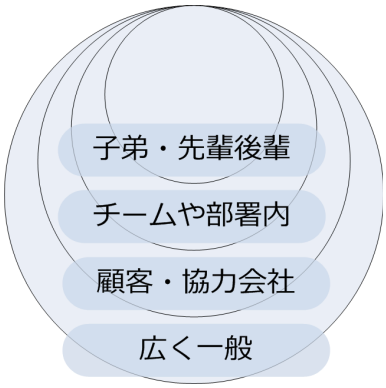

図 2　ステークホルダの範囲
Fig. 2　Stakeholder range.

3.3　教材形態の選択

教材形態の選択にあたり，インストラクショナルデザイン [4] などを参考に次の 7 点を候補にあげた．

- 座学
- ケースメソッド
- ロールプレイング
- ペアプログラミング
- ビデオ鑑賞
- プロジェクトごとの評価（アクションラーニング [12]）
- ペーパーテスト

これらの関係を**図 3** に示す．縦軸は，左にいくほど理論

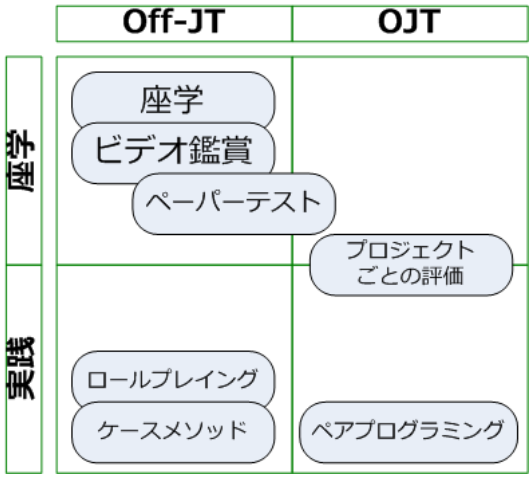

図 3　教育方法の違い
Fig. 3　Distribution chart of education method.

を基にした学習の側面が強く，右にいくほど実務を通じた学習（OJT）の側面が強いことを示す．横軸は，上にいくほど座学の側面が強く，下にいくほど実践の側面が強くなることを示している．

本研究では，ケースメソッドとロールプレイング（以下，ケース）を採用した．理由は，開発工程を通して学習できること，役割分担を設定して学習できること，そして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ためである．内容の抜粋は，付録に記載した．

3.4　チーム構成

ケースの実施に際して，被験者に次の 3 ロールからなるチームを構成してもらうこととした．

- プログラマ — ソフトウェア内部の開発を行う．
- デザイナ — ソフトウェア外部の開発を行う．
- テスタ — プログラマ・デザイナが開発したソフトウェアの品質を確認する．

このようにした理由は，(1) 各工程で要求や設計の一致を確認する必要性を持たせることでコミュニケーションを起こりやすくさせ，(2) その結果としてコミュニケーションの必要性に気付きやすくできるからである．上記のチームを構築するためには，被験者は最低 3 名必要である．

3.5　評価のための準備

設計した教育方法の有効性を評価するため，試行実験を実施した．被験者に対する評価の観点としては，3.1 節であげた 5 つのスキル獲得目標を基に，次の 3 つを設定した．

- ケースで目標としている人間構造の理解の水準を達成しているか —「ヒューマンスキル」の獲得が進んでいるかを確認する．
- ケース内に織り込まれている技術要素の理解が進んでいるか —「テクニカルスキル」および「コンセプチュアルスキル」の理解が正しく進んでいるかを評価する．

表 2　実験工程
Table 2　Process of experiment.

工程	評価方法	HS	TS	CS
実験実施前	アンケート	○	○	○
ケースの実施	レポート	-	○	-
ワークショップ	ビデオ撮影·SYMLOG·完成したソフトウェア	○	○	○
実験実施後	アンケート	○	○	○

- 作業経緯や人との交流を通じた省察が積極的に行われ課題の目的を理解していたか — すべてのコアスキルに関するものとなり，新人教育後の実践段階においても自分自身で人間関係・技術の問題解決を行い，学習を行う力が備わっているかを確かめる．

これらの観点に基づき，工程ごとに**表 2** の方法により評価データを収集する．各ニーモニックは表 1 に準ずる．

アンケートは，被験者のスキルの変化を調べるため，実験実施の前と後に実施した．ケース実施には，進捗を確認するためのデータを収集する．ケースの最後にワークショップを実施することとし，ここではビデオ記録・行動観察・集団構造分析（SYMLOG [11]）[*1] を行う．SYMLOG とは，小グループの中である課題の解決や意思決定の際になされる相互行為の進行過程を分析する手法である．

3.6　提案手法の方法論

ここまでに検討した内容に基づき，本稿で提案する「コミュニケーションスキル獲得を促すソフトウェア技術者教育手法」について**表 3** に整理する．次節以降で述べる評価実験は本手法の適用例の 1 つとして位置づけられる．

表 3 中の「予定どおり進捗しないことも織り込み」については，このような事象が起きる原因は表 1 の小項目いずれかが不十分であることに起因するため，そのことを指摘したり自分たちで自覚したりすることを通じてその重要性を理解させることを意図している．

また，「評価」については，開発体験を経ることで，表 1 にあげた小項目の中から，どれとどれについて重要であると（新たに，または改めて）認識したかを，事前・事後アンケートや振り返りの議論に基づいて評価する．これは，目的とする表 1 のスキルは非常に適用範囲が広く，客観的な基準を定めることは難しいため，「その事項の重要さを意識するようになったか否か」に基づいて評価することが適切と考えたためである．

提案手法と一般的な PBL（プロジェクト・ベースド・ラーニング [14]・問題解決型学習）による教育を比較したものを，**表 4** に示す．PBL は課題解決による自発的な学習を目的としているため，必ずしも本手法で目指しているような具体的なスキルの育成にはつながらない場合がある．

[*1] 本稿では表現を平易にするために「向文脈的～脱文脈的」の評価軸を「冷静～非冷静」と変更している．

表 3　提案手法の内容
Table 3　Detail of proposal technique.

目的　新人ソフトウェア技術者ないしその予備軍に対し，ソフトウェア技術者として有効に活動するうえで必要となる表 1 のスキルから，なるべく多くを身につけさせる．

手法　ソフトウェア開発において必要となる「プログラマ」「デザイナ」「テスタ」という 3 つの役割を分担した，チーム形式の短期開発を体験させる．開発内容として，上記の役割分担の必要性が浮き彫りになるような適切なケースを選択し，また提出する成果物も上記の役割分担と明確に対応したものとする．その過程において，予定どおり進捗しないことも織り込み，そのような事象も有効な教育につながるようにする．

進め方　次の段階を経て進める．

(A)　1 時間程度のオリエンテーション（進め方の理解）

(A')　作業に際してコミュニケーションが問題となることの事前指導（今回は 1 つのチームについてのみ実施）

(B)　1〜2 時間 × 数回の作業で指示された成果物を作成（実施者は必要に応じてアドバイス）

(C)　6 時間程度の集中作業でソフトウェアの完成を試みる（実施者は必要に応じてアドバイス）

(C')　プレゼンテーションと全体の作業を振り返っての討論

評価　事前・事後アンケート終了後の振り返りの議論を通じて評価を行うとともに，参加者の自己点検・自己評価の機会とする．評価は「これまで意識していなかった事項の重要性を認識した個数」に基づく．

表 4　提案手法と PBL の比較
Table 4　Comparison table of proposal technique and PBL.

	提案手法	PBL
目的	表 1 にあげる具体的スキルの育成	課題解決による自発的な学びが目的であり必ずしも特定のスキルを目指していない．
成果物	開発を通じ標準的な成果物の作成を体験	実施体制により様々であり特に決まった成果物を定めない場合もある．
ソフト	ソフトの完成は主要な目標ではない	ソフトウェアの完成が重要視されがち．
期間	合計 7 日	ソフトウェア開発のものは 1 セメスターなど長期にわたるものが多い．
評価	表 1 の項目に対する重要性の認識をアンケート・振り返りに基づき判断	一般的な振り返りを行う場合が多く特定の観点からの評価は必ずしもない．

4. 評価実験の概要

4.1 被験者

被験者は，2 大学 2 研究室（以下，それぞれ **α 大学**，**β 大学**と記す）から 3 名ずつ募った．いずれも，情報系学科所属の 4 年生である．実験開始までにプログラミングに関する教育は受けているものの，ソフトウェア工学，特にソフトウェア開発プロセスに関する講義は受講していない．したがって，ソフトウェア開発の始まりから終わりまでのライフサイクルやプログラミング以外の工程で必要となる知識・経験は網羅できていない状態である．なお，実験の開始にあたり，筆者らが被験者に対し本研究についての説明を行っている．

表 5　実験スケジュール
Table 5　Experiment timetable.

日程	所要時間	作業内容
事前	1 時間	課題の説明・質疑応答
	30 分	事前アンケート
	30 分	事前指導（β 大学チームのみ）
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

4.2 ケースと実験スケジュール

実験時に使用する教材として，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) ふだん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) は，業務知識を改めて学ばなくても理解できることを目指し，情報系の学生が慣れ親しんでいるソーシャルメディア [16] を活用し，企画を被験者自身が育てやすいテーマを選択した．(2) は，短い実験期間で，実際に被験者が早い段階に目で見てソフトウェアが動作していることを確認しやすいものを開発できることを目標に，mash-up [8] を用いた開発を行えるものを設定した．(3) は，3.4 節で設定したロールを適用するためのものである．(4) は，組織での開発を通して経験がない被験者に向けて，ひととおりの開発工程を学べるよう作成した．(5) 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被験者自身の「意思」を込められる要素を作るようにした．

実験スケジュールは**表 5** のとおりである．7 日目は丸 1 日をかけワークショップ形式で実施した．「事前指導」は，

β 大学チームのみに実施した．これは，事前指導の有無がコミュニケーションスキル獲得効果を高めるかを比較するためである．

4.3　成果物とその内容

ケース実施を通じて作成した成果物の納品日および表 1 に対応するスキルのニーモニックを，3.5 節を基に**表 6** に示す．これは，それぞれの提出日の段階で作業が滞りなく進んでいるかを確認することが目的である．

選択したプラットフォームの情報とは，教材に定義されたソフトウェアを開発するにあたって，どのプログラミング言語を用いるかを定義したものである．コンセプトのメモとは，アーキテクチャと外部インタフェースをどのように持つかの概要を A4 用紙 1 枚にまとめたものである．作業分担リストとは，3.4 節で定義した作業分担に対し，誰が担当するかを明記したものである．外部設計の図面とは，外部インタフェースと通信内容をまとめたものである．機能設計書とは，開発するシステムにどのような機能を用意するかをまとめた資料である．画面遷移図とは，ユーザインタフェース（以下，UI）の画面遷移を表したものである．画面設計図とは，UI の配置と対応する機能を明記したものである．テスト計画書とは，先の機能設計書，画面遷移図，そして画面設計図に明記された機能がすべて満たされているかを確認する際のチェックリストである．完成品のプレゼンテーションとは，教材をどのように開発し，利用してもらえるようにしたか，これまでのすべての設計書をふまえて説明した資料である．ソースコードは，これまで作成したすべての設計書をふまえて開発したプログラムのソースコードである．最後に，テスト結果が記入されたテスト計画書は，先に作成したテスト計画書に沿ったテストを行った旨を記録したものである．

表 6　成果物の一覧
Table 6　Deliverable list.

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

4.4　事前指導

事前指導の目的は，同じ課題に対して自分自身の問題意識は他人のそれとは違うことを理解し，1 つの結論に導くことができるコミュニケーションが行えるようになることである．方法として「30 分のワークショップ」と「座学で工程ごとに必要となるコミュニケーションを学ぶ」の 2 つを比較した結果，短時間で目的を達せられる前者を選択した．

実施にあたり，レジュメを作成した．考慮した点は「想像力を働かせる」「ソフトウェア開発に失敗があることを知る」「情報科に通う学生が身近に感じられる」の 3 点である．学校で課題を解くのと業務で価値を提供することとの違いをふまえ，学生にも理解しやすい形でコミュニケーションスキル獲得の重要性を学んだ後に，ケースに臨めるようにした．

課題は「ブラウザに必要な機能をあげる」である．まず，自分と他人の理解の違いを知るために，身近に触れているインターネットブラウザに必要な機能をあげ，それを被験者同士で照合した際に一致しない項目が出ることを確認する．続いて，コミュニケーションを通じ合意を形成することがソフトウェア開発で重要であることを理解できるよう，一致しなかったものの中から最も重要なものを理由をふまえ 1 つ抽出していく過程を経験できるようにした．

4.5　倫理的配慮

実験の実施計画に対しては，筑波大学ビジネスサイエンス系研究倫理委員会の承認を得た．また，被験者および被験者の指導教授に対してはプライバシの配慮から個人を特定できる本名や所属する大学の校名を表出しないこと，また参加の是非や結果によって不利益が生じない旨の説明を行い，実験参加の承諾を得た．

5.　第 1 回実験

5.1　実験実施概要

第 1 回実験は，α 大学チーム 3 名により，2012/09/16〜22 の 7 日間にわたって実施した．開催場所は，6 日目までは各自の自宅や大学の研究室，7 日目は筑波大学東京キャンパスである．

5.2　被験者プロフィール

各被験者に共通するのは，男性であり，ソフトウェアに関する研究を行う同じ研究室に所属していることである．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，各個人について，事前アンケートから得た情報を基に記述する．

被験者 A：プログラマ．講義や研究ばかりでなく，私的な活動でプログラミング経験がある．ただし，ユーザインタフェースやアイコン作成などのリソース作成は不得手で

あり，他のメンバの支援を求めている．

被験者 B：デザイナ．講義や研究でプログラムは書いているが，本人は苦手であると認識している．画面設計や UI 制作で力を発揮したいと述べている．また，メンバ全員がリラックスした雰囲気を作っていきたいと話している．

被験者 C：テスタおよびリーダ担当．当初よりリーダを希望している．プログラミングは，講義や研究ばかりでなく，個人的に Windows 用ゲームを開発した経験がある．また，ゲーム開発を通じて UI 用リソース作成についても知識がある．

5.3　実験中の状況

1〜6 日目は，被験者が各自でケースに取り組んだ．また，成果物も予定どおり提出を受けた．**表 7** に，各提出日までの間のコミュニケーションの概要を示す．

7 日目のワークショップは，**表 8** のとおりに進んだ．前半を 10:30〜15:50，後半を 15:50〜19:50 と区切って確認すると，コミュニケーションが行われているのは 6 と 8 であり，後半に集中していることが分かる．

次に，ビデオを確認する．前半は各個人が黙々と開発を行っていた．後半は，15:50 頃から始まった問題点の整理時，プログラマが開発したプログラムとデザイナの制作した画面が適切に結合できないという問題が発生した．プログラマが意図した変数の受け渡し方法と，デザイナの想定した変数の受け取り方法に齟齬があったためである．原因は，事前にプログラマとデザイナとの間で実装方法に関する合意が不足し，互いに「こうなるだろう」と見込みを基に進めてしまったからである．

また，SYMLOG を用いた小集団構造の把握を行った結果を，前半は**図 4**，後半は**図 5** に示す．図中，「被験者」は「被」と省略している．後半，プログラマである被験者 A の冷静さが失われていることが分かる．これは，先のプログラムと画面の間の実装の齟齬をプログラマが積極的に埋めようとした際に，焦りやコミュニケーションの粗さとして現れている．

最後に，ビデオ記録に基づき，会話内容の分析も行った．前半は，ほとんど言葉が交わされることはなかった．後半は，プログラマの一方的な会話が目立っていることと，議論の論点がうまく定まらずに会話が進んでいることがうかがえた．その結果，デザイナ・テスタの行動が萎縮してしまっていた．

表 7　α 大学コミュニケーションの概要
Table 7　Overview of communication of Univ. α.

日程	概要
1，2 日目（第 1 回）	自身が現在持つスキルを基にスムーズに役割分担が決まる．コンセプトは A が決定し B，C がコメントし精査．
3，4 日目（第 2 回）	担当ごとの設計内容に矛盾がないよう会話を通じ調整を行いつつ設計を進めていた．
5，6 日目（第 3 回）	A が開発の準備を 1 人で進めていたが難航．他のメンバはスキルが及ばないことをひけ目に感じ手出しができない．そんな中，徐々にチームとしての行動が滞り始めていた．

表 8　α 大学ワークショップのスケジュール
Table 8　Workshop timetable of Univ. α.

	時刻	作業内容
1	10:30〜10:45	本日の作業の流れの整理
2	10:45〜12:30	各自 担当作業
3	12:30〜13:30	昼食
4	13:30〜15:00	各自 担当作業
5	15:00〜15:50	プログラム・画面 結合作業
6	15:50〜16:00	問題点の整理
7	16:00〜16:40	プログラム・画面 結合作業
8	16:40〜17:00	問題点の整理
9	17:00〜19:50	プログラム・画面 結合作業
10	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

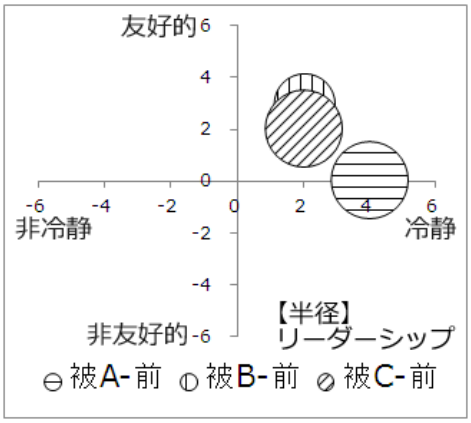

図 4　α 大学ワークショップ内での構造評定：前半
Fig. 4　First half: SYMLOG map of Univ. α.

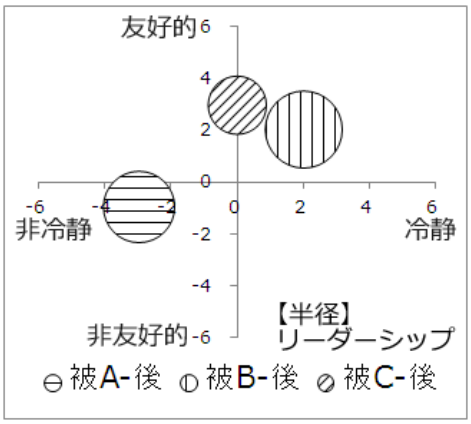

図 5　α 大学ワークショップ内での構造評定：後半
Fig. 5　Second half: SYMLOG map of Univ. α.

5.4　実験後の状況

事後アンケートを基に，各被験者の事後の状況を分析したところ，次のとおりとなった．

被験者 A：ワークショップ後，設計を通じてメンバとコミュニケーションを行い，合意を形成する重要性を認識していた．開発時に問題となったのが画面との結合時であり，ここがうまくいけばスムーズに開発が完了していたからと思われる．なお，プログラミング以外の開発工程の存在の理解や，デバッグなどのプログラミング時のテクニックを学習できたと述べていた．

被験者 B：被験者 A 同様，設計の重要性，特に自分自身と相手の考えていることの齟齬が開発に支障を来すことを理解していた．

被験者 C：被験者 A 同様，設計の重要性を確認した．また，工程を適切に管理すると，プログラミングの時間は相対的に減り，設計の時間が増えることを理解していた．ただし，リーダとしてチーム運営を適切にまとめられなかったことについて，心残りがあったようである．

6.　第 2 回実験

6.1　実験実施概要

第 2 回実験は，β 大学チーム 3 名により，2012/10/14〜20 の 7 日間にわたって実施した．開催場所は，6 日目までは各自の自宅や大学の研究室，7 日目は筑波大学東京キャンパスである．また，β 大学に対しては事前指導を，2012/10/03 に**表 9** のスケジュールのとおりに実施した．

事前指導開始直後，β 大学チーム一同はソフトウェア開発が失敗することがあるという話に「そういうことがあるのか」と少し驚いていた．その後は，演習を通じて事前指導の目的である「自分自身の問題意識と他人のそれとは違う」点を少しずつ理解し始め，ソフトウェア開発が失敗することがある事実を受け止めていたようである．

6.2　被験者プロフィール

被験者は全員男性である．被験者 Y，Z は同じ研究室に所属している．チームメンバは頻繁に顔を合わせており，チーム内での私的な人間関係は良好である．続いて，各個人について，事前アンケートから得た情報を基に記述する．

被験者 X：プログラマ．講義や研究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．

被験者 Y：デザイナ．UI に関する研究に取り組んでいる．プログラミングにはあまり自信がないとのことである．

被験者 Z：テスタおよびリーダ担当．当初よりリーダを志望し，本研究のメンバ募集時からメンバ集めを率先して行っていた．講義や研究でプログラミング経験があり，他のメンバよりも比較的広い範囲で技術の見識を持っているようである．

表 9　β 大学事前指導のスケジュール
Table 9　Briefing timetable of Univ. β.

	時間	作業内容
1	19:00〜19:20	（全体像の説明・倫理委員会規定の説明）
2	19:20〜19:35	自分自身で列挙する．
3	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する．
4	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．
5	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．
6	19:50〜21:00	（実験に関する説明）

6.3　実験中の状況

1〜6 日目は，被験者が各自でケースに取り組んだ．また，成果物も予定どおり提出を受けた．**表 10** に，各提出日までの間のコミュニケーションの概要を示す．

7 日目のワークショップは，**表 11** のとおりに進んだ．前半を 11:00〜15:00，後半を 15:00〜18:30 と区切る．コミュニケーションは，前半・後半とも定期的に行われていた．

表 10　β 大学コミュニケーションの概要
Table 10　Overview of communication of Univ. β.

日程	概要
1，2 日目（第 1 回）	各自，役割分担を立候補してスムーズに決定．コンセプトは合議のうえで Z が取りまとめた．
3，4 日目（第 2 回）	X が技術的な部分での検討に苦心したが，他のメンバが調査し技術力を補強し合った．
5，6 日目（第 3 回）	Z は X，Y から情報を集め，テストケースをまとめあげることができた．しかし，X がプログラムの実装のための事前検証で苦心し調査し続ける状況は継続していた．

表 11　β 大学ワークショップのスケジュール
Table 11　Workshop timetable of Univ. β.

	時刻	作業内容
1	11:00〜11:10	段取りの確認
2	11:10〜12:20	技術調査
3	12:20〜12:30	調査状況 共有
4	12:30〜13:00	技術調査・プログラミング
5	13:00〜13:10	作業進捗 共有
6	13:10〜13:40	昼食・技術調査・プログラミング
7	13:40〜13:50	調査状況 共有
8	13:50〜15:00	技術調査・プログラミング
9	15:00〜15:15	作業進捗 共有
10	15:15〜16:30	技術調査・プログラミング
11	16:30〜16:55	作業進捗 共有
12	16:55〜17:00	問題点の整理
13	17:00〜18:20	本日の作業をふまえ設計資料の更新
14	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

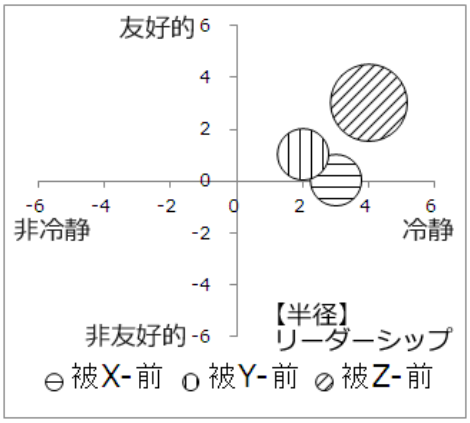

図 6　β 大学ワークショップ内での構造評定：前半
Fig. 6　First half: SYMLOG map of Univ. β.

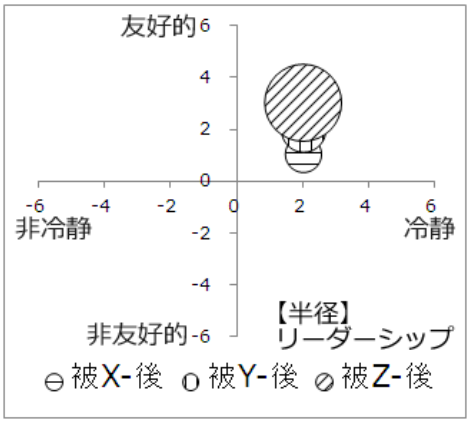

図 7　β 大学ワークショップ内での構造評定：後半
Fig. 7　Second half: SYMLOG map of Univ. β.

次に，ビデオを確認する．前半・後半ともに積極的なコミュニケーションが行われている．前半は，ワークショップ全体のスケジュール調整，およびタスクと課題の割当てが実施され，各自が明確な目標を持って作業を進められる状態を自ら構築していた．後半の設計資料の更新時は，設計資料を基にメンバ同士で「教え合う」形で理解を深め合い，各自が課題を発見して設計をより高い完成度へ導く努力がなされていた．

また，SYMLOG を用いた小集団構造の把握を行った結果を，前半は**図 6**，後半は**図 7** に示す．前半・後半で大きな差は出ていない．

しかし，α 大学チームでは起こらず，β 大学チームで起こった問題があった．それは，プログラミングに関するスキルが不足していたことであった．特に，HTTP の POST・GET メソッドの動きをつかみきれず，終始実装が滞ってしまっていた．

6.4　実験後の状況

事後アンケートを基に，各被験者の事後の状況を分析したところ，次のとおりとなった．

被験者 X：設計段階でしっかり話し合っておくことで作業の流れがはっきりし，開発がマンネリ化しにくくなることを理解した．緊張感も高かったとのことであった．ただし，技術力の点で及ばなかった点を反省していた．

被験者 Y：ふだんのとおり，楽しいときも苦しいときも会話を絶やさない大切さを理解した．一方，プログラミングスキルが足らなかったことを反省していた．

被験者 Z：リーダとして，メンバと同意をとりながらプロジェクトを進行させる大切さを理解した．特に，今回利用する HTTP に関する技術の事前調査が足りなかったことを反省していた．

7.　考察

7.1　第 1 回実験

表 1 に基づき，実験結果を考察する．

ヒューマンスキルについては，特に会話が少なかった点が目立っていた．事後アンケートからも，各メンバからコミュニケーションを通じた合意の形成が少なかったことについて反省するコメントがあった．また，SYMLOG の分析結果からも，コミュニケーションに問題が出ていたことがうかがえた．ここから，実験後に経験を通じてコミュニケーションの重要性を理解できていたと考えられる．

テクニカルスキルについては，幅広い工程に携わることができた経験を価値とし，かつ設計を適切に行う重要性を理解していた．このことは，アンケートから読み取ることができた．今回，実装にあたって技術的な問題で立ち止まることがほとんどなかったため，1 人ではなくチームで開発を進めて初めて理解できるスキルを獲得できたと考えられる．

コンセプチュアルスキルは，情報収集を行う手段を自ら開拓する手段を獲得したことが，アンケートから読み取れた．しかし，ソフトウェアの企画などで斬新なアイディアが出るほどではなかった．

7.2　第 2 回実験

表 1 に基づき，実験結果を考察する．

ヒューマンスキルについては，継続的なコミュニケーションを通じて互いに理解を深め合っていたことが特筆できる．事前指導を通じ，ケース開始直後からつねにコミュニケーションを通じて仕様やゴールを共有することの重要性を理解していたことが，アンケートやビデオで撮影された会話から確認することができた．

テクニカルスキルについては，今回のケースをこなすために必要なプログラミングスキルが不足し，実装時に問題が出てしまっていた．ただし，工程全体を通して開発する経験を獲得することができた．

コンセプチュアルスキルは，本実験を通じて明確に獲得できたスキルは認められなかった．

表 12　実験結果（表 1 の各項目に対する認識）
Table 12　Experimental results.

	HS			TS			CS			
	OR	RU	CT	PG	SD	PM	SA	IG	EA	EX
αA	○			○	○	○		○		
αB	○				○	○				
αC	○	○			○	○				
βX	○	○	○		○					○
βY	○	○	○		○	○		○		○
βZ	○	○	○		○	○	○	○	○	○

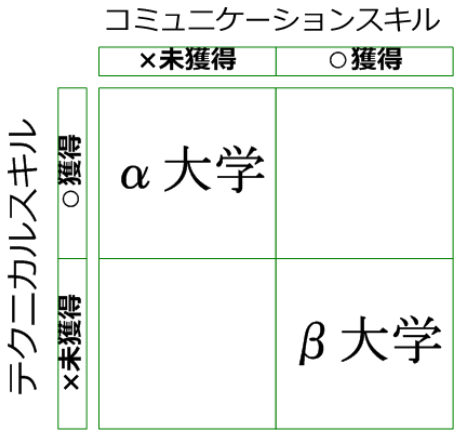

図 8　第 1 回と第 2 回実験の比較表
Fig. 8　Comparison table — first and second experiment.

7.3　2 回の実験の比較

2 回の実験結果の比較に際し，コミュニケーションスキルとテクニカルスキルの 2 点に注目し，**図 8** に分類する．

各実験の結果から，図 8 の右上の象限，すなわちコミュニケーションスキルとテクニカルスキルを同時に獲得するには至らなかった．原因として，α 大学チームには事前教育を施していなかったこと，β 大学チームにはケースの中で利用する技術を理解していなかったことがあげられる．

コミュニケーションスキルについては，事前教育を施すことでよりスキルを獲得できることが確認できた．テクニカルスキルについては，事前にケースで利用する技術について学習することで解決する必要があると考えられる．

8.　まとめ

本研究では，事前指導とケースメソッド＋ロールプレイング形式の教育を通じて，**表 12** のとおり，ソフトウェア開発に必要なコミュニケーションスキルを高められる可能性があることを確認できたと考える．

新人技術者に対してコミュニケーション，すなわち成果物を基にインタラクションを行うスキルを教育することで，チーム内でのコミュニケーションが円滑に行いやすくなり，より早く実戦に投入が可能となる．また，教育期間が 1 週間程度と短期間であるため，大学の集中講義はもとより，企業に就職した後の新人教育で導入しやすいことが特徴である．ひいては，コミュニケーションに起因するソフトウェア開発の問題を低減させることが可能である．

本研究が提案する教育方法は，PBL に類似しているかもしれないが，違いとしてソフトウェア開発におけるコミュニケーションスキルを高める目的を設定し，かつチームで開発する際のコミュニケーションスキル獲得状況を測定する指針を示している点がある．ただし，テクニカルスキルが備わっていなければケースの遂行で問題が出る場合があるため，事前に個人単位でテクニカルスキル，少なくともプログラミングスキルを獲得する必要がある．

今後，ヒューマンスキルとテクニカルスキル双方のスキルをより確実に獲得できるようにするために，次の 2 点の改善に取り組み，さらに研究を進めていきたい．

本稿で報告した実験は，大学生 3 名のチーム（提案手法の最少人数）2 組のみによるものであり，提案手法の検証として十分な人数とはいえない．また，評価方法としても，目標とした表 1 のどれについて意識できたかという点までであり，この部分でも不十分である．今後，望まれる実験内容としては，3 名のチームをより多く組み，(1) 事前に（何の知識もなく）課題を与えて開発を行い，(2) 本手法を（事前指導つきで）実施し，(3) 事後に最初と同水準の課題に対する開発を行い，(1) と (3) の詳細内容を比較し，表 1 のスキルに起因する差を探すことが考えられる．時間的にも被験者の確保のうえからも簡単ではないが，今後の課題として取り組みたい．

また，効果測定をより正確かつ汎用的に行えるよう，ソフトウェア開発に特化したコミュニケーションスキル獲得の指標の開発に取り組みたい．

謝辞　今回，実験参加を承諾いただいた学生および指導教員の皆様に深く御礼を申し上げる．協力なくして本研究は成立しえなかった．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．この場を借りて，感謝の意を表したい．

参考文献

[1]　Baden, R., Bender, A., Spring, N., Bhattacharjee, B. and

Starin, D.: Persona, *ACM SIGCOMM Computer Communication Review*, Vol.39, No.4, p.135 (online), DOI: 10.1145/1594977.1592585 (2009).
[2] Carter, M., Vouk, M., Gannod, G.C., Burge, J.E., Anderson, P.V. and Hoffman, M.E.: Communication genres: Integrating communication into the software engineering curriculum, *2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T)*, IEEE, pp.21–30 (online), DOI: 10.1109/CSEET.2011.5876091 (2011).
[3] 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，Making Software, chapter 26, オライリージャパン (2011).
[4] Lee, W.W.，Owens, D.L.，日本イーラーニングコンソシアム，清水康敬：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).
[5] Cusumano, M.A.: *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] Katz, R.L.: *Skills of an Effective Administrator*, Harvard Business Review Classics, Harvard Business Press (1974).
[7] Royce, W.W.: Managing the development of large software systems: Concepts and techniques, pp.328–338 (online), available from ⟨http://dl.acm.org/citation.cfm?id=41765.41801⟩ (1987).
[8] Wong, J. and Hong, J.: What do we "mashup" when we make mashups?, *Proc. 4th International Workshop on End-user Software Engineering — WEUSE '08*, pp.35–39, ACM Press (online), DOI: 10.1145/1370847.1370855 (2008).
[9] Wood, J.T.: *Gendered Lives: Communication, Gender, and Culture*, *10th edition*, Cengage Learning (2012).
[10] Bech, K. et al.: Manifesto for Agile Software Development (online), available from ⟨http://agilemanifesto.org/⟩ (2001).
[11] 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.167–174（オンライン），入手先 ⟨http://ci.nii.ac.jp/naid/40001568057/⟩ (1991).
[12] 小林雅史，箱守知子，磯部匡志，赤坂幸彦，村松充雄：アクションラーニング手法を取り入れたメンタリングによるプロジェクトマネージャ育成手法の提案（我が社の PM 事例），プロジェクトマネジメント学会誌，Vol.9, No.1, pp.46–49（オンライン），入手先 ⟨http://ci.nii.ac.jp/naid/110006249854/⟩ (2007).
[13] 独立行政法人情報処理推進機構：IT スキル標準（ITSS），（オンライン），入手先 ⟨http://www.ipa.go.jp/jinzai/itss/⟩ (2002).
[14] 船川淳志：特集 プロジェクト・ベースド・OJT のすすめ，人材教育，Vol.19, No.11, pp.15–45（オンライン），入手先 ⟨http://ci.nii.ac.jp/naid/40015682983/⟩ (2007).
[15] 大森隆行，丸山勝久，林　晋平，沢田篤史：ソフトウェア進化研究の分類と動向，pp.3–28, 一般社団法人日本ソフトウェア科学会 (2012).
[16] 辰己丈夫，江木啓訓，瀬川大勝：大学 1 年生の情報活用能力と ICT 機器やメディアの利用状況調査，学術情報処理研究，Vol.16, pp.111–121（オンライン），入手先 ⟨http://www.nipc.med.tuat.ac.jp/home/jacn/ronbun/xue-shu-qing-bao-chu-li-yan-jiu⟩ (2012).

付　　録

ケース・役割分担の説明（抜粋）

A.1　ケースの目次

(1) はじめに：研究に対する協力の御礼.
(2) 概要：課題の全体像を記載.
(3) おことわり：本実験は論文として公開すること，そして匿名で取り扱うことを明記.
(4) 参加のメリット・デメリット
(5) システムの要件
　(a) 開発するアプリケーション
　(b) 開発の背景
　(c) 機能要件
　(d) 非機能要件
(6) 作業工程
　(a) リーダの決定
　(b) 作業に関する役割分担の決定
　(c) コンセプト作成
　(d) 外部設計
　(e) 機能設計
　(f) 画面設計
　(g) テスト計画書
　(h) リソース作成
　(i) テストプログラム作成
　(j) プログラム開発
　(k) テスト実施
(7) スケジュール
　(a) 日程
　(b) 最終日について
(8) 成果物の納品：成果物の対応は表 6 を参照
　(a) 納品について
　(b) 納品方法について
　(c) 第 1 回 納品物
　(d) 第 2 回 納品物
　(e) 第 3 回 納品物
　(f) 最終日 納品物
(9) 準備関連
　(a) 各自準備を要するもの
　(b) 貸与するもの
(10)連絡事項

A.2　ケースの概要

A.2.1　開発するアプリケーション

「Twitter 上における友人・知人の会話の流れが読みやすくなる Web アプリケーションの開発」

A.2.2　開発の背景

Twitter，皆さん使っているでしょうか．

使ったことがある人なら分かると思うのですが，followする人が増えれば増えるほど，友人や知人をはじめとした頻繁にチェックしておきたい人の tweet を見逃しやすくなります．そして，そんな tweet の中に，日頃の世間話の種や研究・仕事の話が混じっていたりすることも珍しくありません．次の日の朝，直接会ったときに話についていけないというのは，いささかさみしいものがあります．

Twitter はその回答として，Lists を作りました[*2]．Lists を通じて友人・知人の tweet を積極的にチェックする際に活用している人もいらっしゃるでしょう．でも，会話の流れをつかむという観点では，これで本当に使いやすいといえるでしょうか．

Twitter 上で友人・知人と会話をすると，それは 1〜2 回のやりとりですまないことがあります．しかし，tweet の表示はあくまで時系列．いろいろな発言の中から，自分の目で抽出して追いかけなければ，その話の経緯は見えません．サードパーティの Twitter クライアントでは話の流れを追いかける「スレッド」機能がついているものがありますが，それを使うためにはいくつかのステップをふまなければなりません．あまり便利ではありません．

そこで，みなさんに友人・知人の会話の流れが読みやすくなる Web アプリケーションを開発していただきたいと思います．

A.3　役割分担の説明

次の役割を決めてください．

- プログラマ
- デザイナ（特に UI[*3]）
- テスタ

プログラマは，その名のとおりプログラムを担当します．メインプログラマとして，作り上げるプログラムに責任を負ってください．そのために，プログラムを進める前に行うべき企画や設計について適切に準備が必要になることも，理解してください．

デザイナは，画面のデザインや HTML 制作を担当し，責任を負ってください．同時に，アプリケーション全体の（広義の）デザインについても思慮しなければなりません．

テスタは，プログラマ・デザイナが開発したアプリケーションをテストするための計画を作成し，実際にテストを行ってください．テストで見つからなかったバグが出た場合，自分自身に責任があると考えてください．また，テスタは開発途中に出てくる成果物について，これで問題なくシステム開発が行えるかを目を光らせる必要があります．いうなれば，他の 2 人とは少し距離を置いてことにあたる必要があります．

[*2] Twitter's New 'Lists' Feature Finally Introduces Grouping, Offers An Alternative To The SUL — TechCrunch — http://www.techcrunch.com/2009/09/30/Twitters-new-Lists-feature-finally-introduces-grouping-offers-an-alternative-to-the-sul/

[*3] User Interface

斎藤 祐一郎　（正会員）

1981 年生．2013 年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博士前期課程修了．SIer やスタートアップ企業でのソフトウェア開発者を経て，現在，株式会社ハートビーツにて IT インフラ運用に関するソフトウェア開発に従事．同時に，プログラミング教育に関する研究，およびボランティア活動に携わっている．

久野 靖　（正会員）

1956 年生．1984 年東京工業大学大学院理工学研究科博士後期課程単位取得退学．同年東京工業大学理学部情報科学科助手．1989 年筑波大学講師．同助教授を経て，現在，筑波大学ビジネスサイエンス系教授．理学博士（東京工業大学）．プログラミング言語，ユーザインタフェース，情報教育に関心を持つ．ACM，IEEE-CS，日本ソフトウェア科学会，日本情報科教育学会各会員．