HackathonMediator: 作業進捗共有と成果物イメージ共有によるグループハッカソン協調活動支援システム

西 康太郎[1]　西本 一志[2]

概要：近年，革新的な成果物を生み出す新たな取り組みとして，ハッカソン (Hackathon:Hack-a-thon) が注目されている．本研究では，ハッカソンチームにおける協調活動の問題を解決するために，「作業進捗共有を兼ねた画面共有機能」，「成果物イメージ共有を兼ねたモックアップ生成機能」から構成される支援システム「HackathonMediator」を提案する．「作業進捗共有を兼ねた画面共有機能」をハッカソンに導入した結果，発言量の少ないチームメイトにコミュニケーションの機会をもたらすなどの効用を確認した．

HackathonMediator: A Support System of Collaborative Activities in a Group Hackathon by Sharing States of Progress and Product Images

Nishi Kohtaro[1]　Nishimoto Kazushi[2]

Abstract: Recently, hackathon attracts attention as a novel method to create innovative products. This paper proposes a support system of a group hackathon named "HackathonMediator" to solve problems of collaboration in a team of hackathon. HackathonMediator consists of two functions: a function to share work screen among members for sharing states of progress and a function to create a mock-up model for sharing product images. We conduct user studies using a prototype system and confirm that the function to share work screen is effective to provide opportunities of vital communications among the members.

1. はじめに

近年，IT 分野は幅広く開拓され，開発現場の年齢層も多様化を遂げつつある．多様なテクノロジーやフレームワーク等が豊富に提供されるに伴い，開発工程が簡略化され，短期間で高品質なアプリケーションの開発が容易になりつつある．このような背景などから，近年ではハッカソン（Hackathon：Hack-a-thon）と称されるごく短期間の作品制作大会が盛んに実施されるようになった．

ハッカソンでは，会場内に参加者が集い，3-6 名程度のチーム（即席の場合が多い）を結成し，1-2 日ほどの短期間において革新的な成果物を生み出すことが目標とされる [1]．多くのハッカソンにはアイデア出しを行う企画段階があり，開発段階にてそのプロトタイプを実現し，聴衆の前でプレゼンテーションを行う．昨今においては IT 企業がコンテスト形式のハッカソンを主催する事例が増加し，新たなサービスや人材を発掘する場としても注目されつつある．成果物の共創をはじめ，交友範囲の拡大，コンテストでの優勝など多様な動機が提供されたことで，参加意欲を持った者同士がチームを結成して臨むグループハッカソンが主流となった．ハッカソンの趣旨は主催団体によって様々であるが，本研究においては，全国のハッカソン開催事例に基づき，以下の 3 つのパターンに分類した．

- 新規事業開拓型
  事業化を前提としたプロダクトを作成・提案する．市

[1] 北陸先端科学技術大学院大学 知識科学研究科
School of Knowledge Science, Japan Advanced Institute of Science and Technology
[2] 北陸先端科学技術大学院大学 ライフスタイルデザイン研究センター
Research Center for Innovative Lifestyle Design, Japan Advanced Institute of Science and Technology

場分析やマネタイズなども検討され，企業主催で行われることが多い．

- 課題解決型
  地域や組織の抱える課題を解決するためのプロダクトを作成・提案する．複数の団体が共同で開催することが多く，エンドユーザを含め様々な分野を超えた人材交流が図られる．
- 人材育成型
  プロダクト開発を経て技能向上や新技術開拓を目的とする．近年では企業の採用活動や学校の教育課程としての導入が増加しつつある．

我が国においては課題解決型ハッカソンの開催例が最も多く，社会的知識を持つ参加者がエンドユーザとしての役割を持ち，技術的知識を持つ参加者がそれに呼応して実装を行うなどの特徴がある．既存のプロジェクトベースの共同開発に比べ，ハッカソンにおいては明確な作業プロセス等の制約が提供されない場合が多く [2]，僅かな人員で企画・開発・プレゼンテーションを行う必要があるため，チーム内の迅速かつ確実な情報共有がもっとも重要な課題であると考えられる．しかしながら，ハッカソンの最中においては高速なタスク設定と消化が繰り返されるとともに，情報共有頻度および精度が徐々に低下していく傾向が見受けられる．特に開発過程の後半にかけて，各メンバは個人のタスクに没頭せざるを得なくなり，コミュニケーション量は著しい低下を見せる．

本研究では，情報共有手段を提供する上で以下のような要件を定義した．

- ハッカソンの経験差に依存しないこと
- 開発段階の支援に特化していること
- ユーザに対して能動的な情報共有手段を提供すること

これらの要件を満たすシステムを検討し，評価実験によってシステムの効用を確認する．評価実験は北陸先端科学技術大学院大学のゲーム開発交流サークルが月に一度主催するハッカソンにて実施する．

2. 関連研究

2.1 ハッカソンでの情報共有手段

ハッカソンでの情報共有手段としては以下の 3 つが代表的である．

- ホワイトボードと Post it の応用
- SNS などと連携したチャットルームの活用
- Wiki 形式のドキュメント共有

これらの手段は，任意のドキュメントを作成し共有することで情報共有のパフォーマンスを高める傾向にある．これらのメディアはユーザに対して受動的であり，チームメイトがメディアを積極的に活用することでチーム内での効用を発揮する．主に企画から開発準備段階の間にて高い頻度で活用され，開発過程にて作業文脈に特化した作業ツール（IDE やイラストツールなど）が活用されるようになるとともに，これらの情報共有ツールの使用頻度は低下する傾向にある．また，これらのツールは汎用性が高い点が長所であるが，ツールの効用はユーザの活用能力に大きく依存するため，誰もが安定して高い恩恵を得られるわけではない．

2.2 共同開発に関する研究事例

共同開発の支援として，これまでに様々なプロセスやシステムが提案されており，Agile や SCRUM を筆頭として，小規模の組織での運用に焦点を当てたプロセスモデルは近年増加している [3][4][5]．また，構成員同士のコミュニケーションログの可視化 [6] などをはじめ，組織内における情報共有の支援は古くから研究されている分野である．しかしながら，従来の支援システムのほとんどは，長期的なプロジェクトや，企業など確立された組織での活用を前提としたものが多く，企画・開発・プレゼンテーションを内包した濃厚かつ高速な作業や，即席のチーム結成が想定されるハッカソンへの導入に焦点を当てた事例は少ない．

2.3 ペアプログラミングとの差異

ペアプログラミングでは，共同作業を行う中で即時のコミュニケーションを頻繁に取ることが推奨されており [7]，これはハッカソンにおいても共通する要素である．ペアプログラミングにおいては 2 人のプログラマが役割を分担して作業を行うが，ハッカソンのほとんどにおいては構成メンバが 3 人を超え，メンバの職種もプログラマに限定されない場合が多い．ハッカソンにおけるコミュニケーションの課題は，ペアプログラミングにおいてこれまで検討されてきた課題がより複雑に変化したものと考えられる．

3. 予備実験

本システムを検討・開発するにあたって，ハッカソンにおける失敗の要因とその解決方法を予備実験にて検討した．実験は 4 度にわたって実施しており，ハッカソンの参加メンバは実験毎に異なる．

3.1 2014 年 4 月 19 日 8 時間ハッカソン

初回のハッカソンにおいては，主催側からの指示は最小限にとどめ，思いのままに作業を進めてもらう形で進行した．主立った問題として，各チームメイトが担当するタスクの内容と進捗状況が共有されなくなり，誰が何をやっているのか把握できない状態に陥っていたことが確認された．チーム内でタスクの見直しが数回に渡って発生するとともに開発遅延が重なり，最終的なプレゼンテーションの準備に間に合わなかった．

3.2 2014 年 5 月 31 日 8 時間ハッカソン

前回の問題を解決する方法として，2 度目のハッカソン

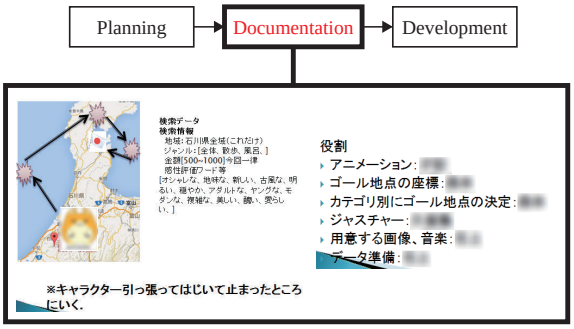

図 1　プレゼンテーション資料の一部

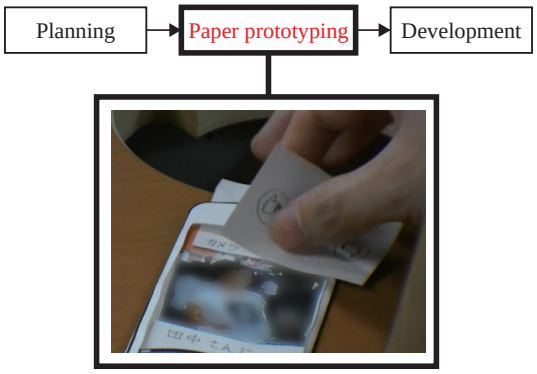

図 2　ペーパープロトタイピングの様子

では企画終了段階でプレゼンテーションの資料作りを行うように指示した．作成する資料には，各々が担当するタスクや成果物の完成イメージなどを詳細に記述するように促している（図 1）．この手法において確認された問題としては，事前に把握されていなかった問題やタスクが開発過程において判明するようになったほか，メンバ間で成果物の動作イメージやタスクに対する誤解があったことが判明し，結果的に作業内容や手順を大幅に見直さざるを得なくなったことなどが挙げられる．問題やタスクの事前把握が困難だった要因として，作成した資料内で成果物の詳細な動作ロジックなどが明確化できていなかったことが考えられた．

3.3　2014 年 7 月 19 日 10 時間ハッカソン

成果物の詳細な動作ロジックをチーム内で共有する上で，3 度目のハッカソンでは個人別ペーパープロトタイピングを導入した（図 2）．企画段階が終了した時点で，各メンバが成果物に抱く動作イメージをペーパープロトタイプとしてそれぞれ表出化し，それらを互いに見せ合うことで精度の高いロジック共有を図った．このプロセスにより各メンバが成果物に抱く動作イメージが統一され，開発のゴール地点を明確にする作用が見受けられた．確認された問題としては，必須なタスクが概ね明確になったものの，開発過程で発生する想定外のアクシデントは避けられず，対応に追われた結果として開発遅延が発生したことが確認された．

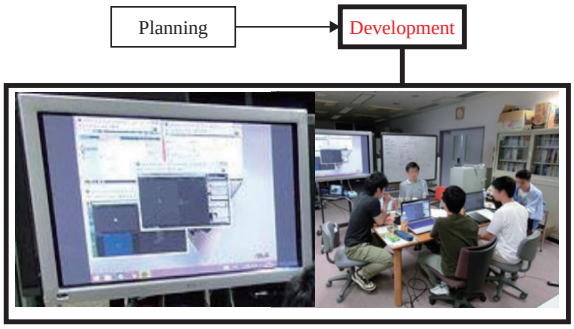

図 3　共有された作業画面

3.4　2014 年 8 月 2-3 日 30 時間ハッカソン

想定外のアクシデントが発生した際などに，迅速な認知および対処の意思決定を促すためのタイムリーな情報共有が必要であると考えた．4 度目のハッカソンにおいては，参加メンバ全員の作業画面を大型ディスプレイ上で常時共有する手法をとった（図 3）．大型ディスプレイ上では，各メンバの作業画面がグリットレイアウト形式で表示されており，全メンバの作業状況を一望できる．共有された画面は開発初期段階において注目を受け，それに伴う示唆の行動などが見受けられたが，開発後半段階へ移行するにつれて注目の頻度は大きく減少した．この点から，作業画面を常時共有するだけでは効果が薄く，重要な情報だけをフィルタリングして共有する仕組みが必要だと考えられた．

4.　提案システム

予備実験から，提案システムの機能要件として以下の二つを検討した．

(1) **メンバに対する作業進捗の周知を確実なものにする**
チームメイトが情報共有を怠ったり，共有された情報を認知できないことが多く，これが原因となって様々な問題が引き起こされている．何をすればいいのか分からない状態に陥ることや，やらなくてもいいことをやってしまう行動などは，進捗が十分に共有されないが故に生じている問題だと考えられる．

(2) **開発の前段階で成果物の完成イメージを明らかにする**
成果物イメージの差異は，特に社会的参加者と技術的参加者の間に存在している．これは 3 回目のハッカソンで導入した個人別ペーパープロトタイピングにて如実に結果が現れた．企画終了時点でのイメージの齟齬が，開発過程でのメンバ間における開発物の修正要求やリジェクトを招く要因になっていると考えられる．

4.1　作業進捗共有を兼ねた画面共有機能

要件 1 に基づく支援手法として，各チームメイトの特定の作業毎に画面を共有する方法を提案する（図 4，図 5）．メンバの特定の作業（図の例ではデバッグ作業）が開始す

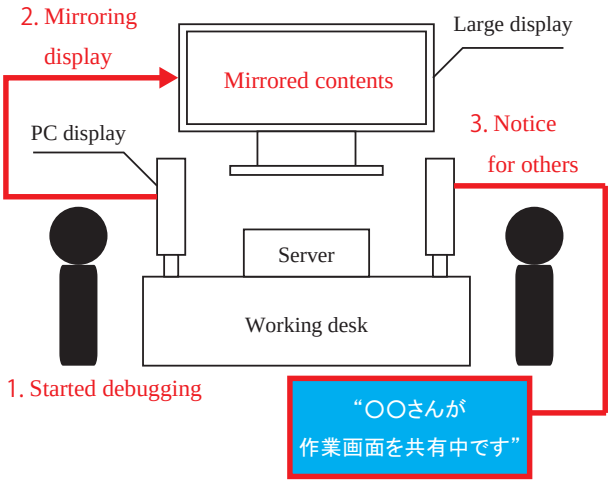

図 4　作業進捗共有を兼ねた画面共有機能

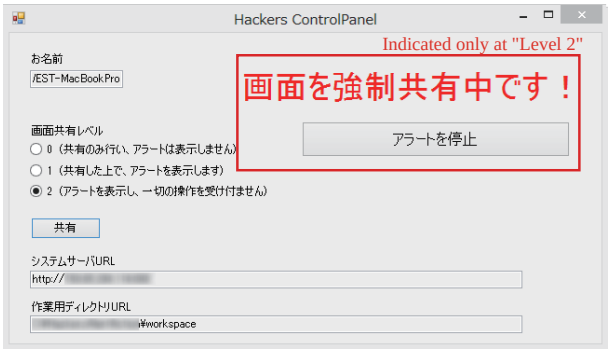

図 5　システムのコントロールパネル

ると，そのメンバの作業画面が大型ディスプレイにミラーリングされる．その際，他のメンバの作業画面上には，ミラーリング中の画面を注視するように促す通知メッセージが発行される．

4.1.1　画面共有を行うタイミングについて

プログラマが開発物をデバッグする際，開発中のアプリケーションが作業 PC 上で動作している様子がミラーリングされる．グラフィック・デザイナにおいては画像素材などを保存・変更する際，使用中のキャンバスなどの画面がミラーリングされる．これにより，プログラマでないメンバなどに対しても可読な（コーディングなどの専門知識を必要としない）形で各メンバの作業進捗を共有し，チーム全体として開発の進行具合を把握し，次の行動を模索する機会を提供する．この機能は作業 PC 内でのデバッグプロセスおよび特定ディレクトリ内におけるファイル作成・変更のイベントを監視し，これらを検出した際に画面共有を自動的に実行する仕組みになっているため，後述の「通知メッセージのレベル設定」を除いてユーザの操作を極力必要としない設計になっている．ただし，上述のシチュエーション以外でユーザが画面共有を行いたいと考える場合を考慮し，システムのコントロールパネル上に「共有ボタン」を設けている．ユーザがこのボタンを押すことで任意のタイミングで画面共有を行うことも可能である．

表 1　通知メッセージのレベル

Level 0	画面のミラーリングが実行され 効果音が鳴るが，通知メッセージは発行されない
Level 1	画面をミラーリングした上で， 効果音と共に通知メッセージが発行される
Level 2	Level 1 と同様であるものの， ミラーリングしたユーザが通知を取り消すまで， 他ユーザは一切の作業が不可能になる

4.1.2　通知メッセージのレベル設定

各ユーザは，共有する作業画面の重要度を 3 段階で指定できる．指定された重要度に合わせて，他ユーザに対する通知メッセージの振る舞いが変化する（表 1）．作業シチュエーションに合わせてこれらのレベルを使い分けることで，チームメイト間の気づきや議論の機会を提供する．

4.2　成果物イメージ共有を兼ねたモックアップ生成機能

要件 2 に基づく支援手法として，成果物イメージ共有を目的としたプロトタイプ Ver0 作成プロセスを提案する．このプロセスは，企画終了時に UI コンポーネントと遷移アニメーションの連結のみで構成された簡易版プロトタイプ（Ver0）を作成し，各メンバが成果物に抱く動作ロジックのイメージを統一する．開発過程にてプロトタイプ Ver0 内に実プログラムを順次実装していき，全ての静的コンテンツを実コードに差し替えることにより，当初のロジックイメージに忠実な成果物が生み出される期待がある．

既存のモックアップツールは，評価用のプレゼンテーションモデルを構築することに注力しており，モックアップから直接コードを生成することを可能としていないものが多い [8][9]．ハッカソンにおいてモックアップツールを活用することは，開発に費やすことのできる時間を失うことに繋がるため，参加者にとって好ましくないという意見があった．この機能では，作成したモックアップツールから本番アーキテクチャで動作可能な実コードを出力可能にすることを検討している．これにより，モックアップ作成行程を開発プロセスの一部として組み込むことが可能になる．この機能を用いた場合の開発プロセスを以下に示す．

(1) レイアウトや画面遷移などを定義したモックアップを作成する
(2) 内部側のロジック等の実装に関するコメントをモックアップ内に記述する
(3) 本番アーキテクチャで実行可能なソースファイルを出力する
(4) ソースファイルを実行し，フロントエンドデザインに関して反復的に評価・修正する
(5) コメントで記述された内容に基づき，内部ロジックに関する拡張をソースファイル内に施す

モックアップ作成の際に (2) で記述されたコメントは，(3) で出力されるソースコード中にコメントアウト文として挿

入される．(3) では，モックアップで定義されたレイアウトや画面遷移を反映したソースコードを自動生成する．生成されたコードの実態は UI コンポーネントおよび遷移アニメーションに関する処理が記述された静的メソッド群であり，(5) にてユーザはコメントアウト文に記述された内容（例えばデータフローやコンテンツ制御のアルゴリズムなど）に応じてメソッドの中身を変更・拡張する形で開発を行う．

5. 画面共有機能の評価

2014 年 11 月 2-3 日に 30 時間ハッカソンを実施し，被験者 11 名による「作業進捗共有を兼ねた画面共有機能」の評価を行った．被験者はチーム A（4 名），チーム B（4 名），チーム C（3 名）に分かれ，開発段階移行時にシステムを導入してもらい，各メンバはシステムを常時稼働させた状態で作業を行った．今回の評価では，システムの使い勝手や，3 段階の通知レベルがどのように活用され，チームにどのような影響がもたらされるかを調査し，ツールに対する様々な意見を被験者に求めた．

5.1 結果と考察

5.1.1 各チームにおける利用状況

- チーム A
  コミュニケーションに重きを置いた開発体制を終始維持しており，チームの連携に関する内外からの評価は最も高かった．デバッグ等の自動検知機能に依存せず，「共有ボタン」を用いた手動での画面共有を積極的に活用する傾向にあり，通知レベル 2 を用いた共有の際には「今から共有しますね」といった確認の声かけを行うなど，通知メッセージが相手の作業を妨害することに配慮した行動が見られた．
- チーム B
  一定時間毎に進捗報告の時間を設ける体制を取っており，その際に各メンバは「共有ボタン」を用い，通知レベル 0 での画面共有を行っている．このチームにおいては，比較的発言量の少なかったメインプログラマのデバッグ画面が高い頻度で共有されており，共有された画面を確認した他のメンバが意見するなど，本システムによって発言量の少ないメンバにコミュニケーションの機会をもたらす効用が見られた．
- チーム C
  発言量の多いプログラマ 2 名に対し，グラフィック・デザイナ 1 名が黙々と作業をする形態を取っていた．プログラマの本システムの活用頻度は低く（うち 1 名のプログラマの作業 PC においてはシステムの動作が不安定だった），グラフィック・デザイナの作業画面がほぼ常に共有される状態にあった．この理由についてプログラマは「グラフィック・デザイナの作業画面を見ながら作業したかったので，自分のデバッグ画面に切り替わることを避けたかった」と述べている．共有されたグラフィック・デザイナの画面を確認したプログラマが意見を出す場面は多く，グラフィック・デザイナに対してチーム B と同様の効用が見られた．

今回の実験では，ファイルの作成・変更の自動検知機能を活用する上で，“指定のディレクトリ” 内でファイルの作成・編集を行うことを被験者に対し事前に指示していたが，一部の被験者には指定のディレクトリを活用せずに作業を行う者もいた．このことに関しては，「わざわざ画面を共有するほどの作業ではないと思った」，「注目してほしい時にはこちらから共有ボタンを押せば良いと思った」といった理由が述べられている．

5.1.2 ツールの使い勝手

「共有が楽で，注目してほしいタイミングで注目させることができ，現状を報告しやすかった」，「デザイナの絵をみんなで確認しながら作業を進められたことが良かった」など肯定的な意見を得ることができた．11 名中 10 名がシステムに対して肯定的な反応を示したが，チーム B のサウンド・デザイナ 1 名が「あまり有用に感じられなかった」と述べている．この理由としては，今回用いたシステムは画面共有のみをサポートしており，音声共有はサポートしていなかったため，サウンド・デザイナが恩恵を得られる要素が少なかった点が上げられる．また，サウンド・デザイナの用意した音声素材が，本人の意図せぬシチュエーションで使われていたことが指摘されており，素材の活用場面に関するチーム内での議論が不足していたことが示唆された．

5.1.3 通知レベルの活用

全チームを通して最も使用頻度の高かった通知レベルは 0 であった．被験者の意見としては「共有時に効果音も鳴るし，画面も共有されるからこれで十分だと思った」，「通知メッセージが相手の作業を邪魔することを避けたかった」という理由が述べられている．次点で使用されたレベルは 2 であり，これはチーム全体で各メンバの進捗を確認する際に使用される傾向にあった．

5.1.4 ツールに対する要望

「一度共有されると，別の人の画面に切り替わるまで自分の画面が表示されたままになるのが嫌だ」，「共有されるたびに鳴る効果音が徐々に耳障りに感じられた」，「作業画面が共有されても，それがどんな作業なのか，文脈を把握しにくい」という意見があった．

5.2 今後の課題

今回の評価を通して得られたフィードバックから，今後の課題について述べる，

5.2.1 ファイル共有機能の提供

指定のディレクトリを活用せず，ファイル作成・変更の

自動検知を免れたユーザへの対処を考える．インタビューへの返答内容から，彼らは指定ディレクトリを活用すること自体に反発していたわけではないことが推測された．このことから，彼らに対し，指定ディレクトリで作業を行うことへのメリットをさらに明確に訴えるべきだと考える．解決案として，ファイル共有ディレクトリとしての機能を指定ディレクトリに付加することを検討している．これによりチーム内でのリソース共有が効率化されるとともに，ファイル作成・変更自動検知機能の活用率を向上することが可能だと考えられる．

5.2.2 作業文脈の明確化

「作業画面が共有されても，それがどんな作業なのか，文脈を把握しにくい」という意見を受け，画面共有時に一言コメント（「○○作業中」など）のようなものを求める機能を検討している．ここで記入されたコメントを通知メッセージと共に表示することにより，チームメイトの状況判断精度を向上できないかと考えている．また，これらのコメントを時系列順に記録していくことにより，簡易バックログのようなものを提示できないか検討している．

5.2.3 音声素材の共有

現状のシステムでは，映像の共有のみをサポートしており，音声共有に関してはサポートできていなかった．これに関しては，指定ディレクトリ内に保存された音声拡張子（.wav など）を持つファイルに対してのみ，音声再生を実行する機能を実装することで解決可能だと考えられる．

5.2.4 システム起動の自動化

作業 PC を再起動した際，提案システムの起動を自動化する仕組みが必要だと考えられる．長時間のハッカソンにおいてユーザが PC を再起動する場面が複数回確認されており，再起動後にユーザが提案システムの起動を忘れたまま作業に取りかかる懸念が発覚したためである．

5.2.5 モックアップ生成機能の実装

チーム B のサウンド・デザイナの指摘にもあったように，デザイナの用意した素材が本人の意図せぬシチュエーションで活用される問題はこれまでも確認されていた．これは成果物のデザインについてチームが議論する際に検討されるべきだった部分だと考えられる．今後実装予定の「成果物イメージ共有を兼ねたモックアップ生成機能」によって，フロントエンドデザインに関するチーム内での議論がより活発なものとなることに期待したい．

6. まとめ

本研究では，グループハッカソンにおける情報共有の問題を解決するために，「作業進捗共有を兼ねた画面共有機能」および「成果物イメージ共有を兼ねたモックアップ生成機能」から構成される協調活動支援システム「Hackathon-Mediator」を提案した．評価実験にて「作業進捗共有を兼ねた画面共有機能」を用いた結果，発言量の少ないチームメイトにコミュニケーションの機会をもたらす効用などを確認できた．今後は被験者のフィードバック等から得られた問題を改善していくとともに，「成果物イメージ共有を兼ねたモックアップ生成機能」を統合した上での評価を実施していく予定である．

技術的・社会的知識の兼ね合いは，現代社会に潜在する問題の解決を図る上での重要な要因と見なされる [10]．ハッカソンを経て生み出される新たな成果物や技術が，将来の社会発展に大きく貢献していくことに期待し，本研究の提案するシステムがそのプロセスの支えとなる事を望む．

謝辞 本研究は，科学研究費助成事業（課題番号 26280126）の支援を受けて実施された．ここに謝意を表する．

参考文献

[1] Zapico Lamela, Jorge Luis, et al.: Hacking sustainability: Broadening participation through Green Hackathons, Fourth International Symposium on End-User Development. June 10-13, (2013).
[2] 清水たくみ: オープンデータ活用によるアプリケーション開発:ハッカソン実態調査を通じて, 組織学会大会論文集, pp.38-43 (2013).
[3] Bashir, M. Salman, and M. Rizwan Jameel Qureshi: Hybrid Software Development Approach For Small To Medium Scale Projects: Rup, Xp & Scrum., Cell 966 (2012).
[4] Shaw, Aaron, et al.: Computer supported collective action, interactions 21.2, pp.74-77 (2014).
[5] S. Ryan, R.V. O ' Connor: Acquiring and sharing tacit knowledge in software development teams: an empirical study, Information and Software Technology, Volume 55, Issue 9, pp.1614-1624 (2013).
[6] Kwan, Irwin, Daniela Damian, and M. Storey: Visualizing a requirements-centred social network to maintain awareness within development teams., Requirements Engineering Visualization, REV'06. First International Workshop on. IEEE (2006).
[7] Zarb, Mark, Janet Hughes, and John Richards: Industry-inspired guidelines improve students' pair programming communication, Proceedings of the 18th ACM conference on Innovation and technology in computer science education, pp.135-140 (2013).
[8] Rivero, José Matías and Rossi, Gustavo and Grigera, Julián and Luna, Esteban Robles and Navarro, Antonio: From Interface Mockups to Web Application Models, Proceedings of the 12th International Conference on Web Information System Engineering, pp.257-264 (2010).
[9] Jørgensen, A.P., Collard, M., Koch, C.: Prototyping iphone apps: realistic experiences on the device, NordiCHI '10 Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries, pp.687-690 (2010).
[10] Jeppesen, Lars Bo, and Karim R. Lakhani: Marginality and problem-solving effectiveness in broadcast search, Organization science 21.5, pp.1016-1033 (2010).