競争型のソフトウェア開発PBLに対する継続的競争フレームワークの試作と評価

藪下 友　柗本 真佑　楠本 真二

効果的な学習形態の一つとして，ゲームなどの題材を用いた競争型 PBL（Project-Based Learning）がある．競争型 PBL の成功には，学生のモチベーション向上と教員による学生の開発進捗の把握が重要となる．本研究では，ソフトウェア開発プラクティスの一つである CI の考えに基づいた，PBL 支援フレームワーク CC を提案する．予備実験として，学部 2 年生を対象とした競争型 PBL 授業にこのフレームワークを導入し，受講生の開発に対するモチベーション向上，及び教員による進捗把握支援に対する予備評価を行った．

1　はじめに

効果的な学習形態の一つとして，ゲームなどの題材を用いた競争型の PBL（Project-Based Learning）がある．PBL とはプロジェクトに基づいて学生主体で進める学習形態であり，学習者のスキルの習得, 教育効果の向上, 学習への動機付けなどの利点が存在する [8]．競争型 PBL ではゲーム等の競争型プロジェクトを通じて，作成したプログラムの強さを競い合う．プログラミング教育にコンテストやゲーミフィケーション等の競争要素を取り入れることで，学習者の競争心の昂進や実践的スキルの習得，モチベーションの向上などの効果が見込まれる [3] [9].

競争型 PBL の成功には学生の開発に対するモチベーション向上と教員による学生の開発進捗の把握が重要となる．PBL は教員が先導する授業と違い，学生が主体的にプログラム開発を行う学習形態であるため，学生のモチベーション維持が欠かせない．また，教員視点では，各学生の開発状況や進捗状況の把握が必要である．進捗状況の把握により開発が遅れている学生に対する適切な開発支援が可能となる．

本研究の目的は競争型 PBL における学生の開発に対するモチベーション向上，及び教員による進捗把握支援の二つである．そのために，ソフトウェア開発プラクティスの一つである CI（Continuous Integration：継続的統合）の考えに基づいた PBL 支援フレームワーク CC（Continuous Competition：継続的競争）を提案する．一般的な CI では，プログラムの変更を契機に，自動ビルドや自動テストにより開発の効率化を狙う．一方 CC では，学生によるプログラムの変更を契機に，即座に対戦を実施しその結果を学生にフィードバックする．これにより学生はプログラム変更毎に他の学生と比べた相対的なプログラムの強さを把握できる．また，教員はプログラム変更毎での学生の開発状況を確認し，進捗を把握できる．

CC の有効性を確かめるために，大阪大学で実施されている PBL へ CC を導入した．本稿では提案フレームワークの予備評価として，モチベーション向上効果の確認，及び CC を用いた進捗把握の事例について紹介する．

2　準備

2.1　競争型 PBL

PBL の一つの形態として，ゲームやコンテストなどの競争型の題材を用いた競争型 PBL がある．競争

Prototype and Evaluation of Continuous Competition Framework for Competitive Software Development PBL

Yu Yabushita, Shinsuke Matsumoto, Shinji Kusumoto, 大阪大学, Osaka University.

型 PBL では教員が定めたルールに基づき，学生が戦略検討やプログラム改善を繰り返す．競争はチームの想像力や生産性を向上させることが知られており [2]，グループワークベースの教育現場との親和性が高い．また，競争をプログラミング教育に取り入れることで学習者の競争心をあおり，実践的スキルの習得やモチベーションの向上などの効果も見込まれる [3] [9]．競争型 PBL の実施例として，Atilim University では [9]，制限時間内で最大の素数を見つけるプログラムを開発し，その素数の大きさで順位づけが行われている．また，釧路工業高等専門学校では [11]，プログラム可能な LEGO 車両を用いたレースを題材として行われている．

本研究では競争型 PBL を，直接型競争と間接型競争の 2 種類に大別する．直接型競争は 2 つ以上のチームが同時に競技を行い，直接的にその勝敗を決める形式である．他方，間接型競争は単一のチームが競技を行い，そのスコアによって間接的に勝敗を決める形式である．実際の競技としては，野球やサッカーなどが直接型であり，陸上競技や競泳などが間接型となる．

2.2 CI を活用したプログラミング教育

CI をプログラミング教育へ導入した研究は数多く存在する [1] [10]．CI（Continuous Integration：継続的統合）はソフトウェア開発プラクティスの一つであり，ソースコードの変更を契機としてビルド及びテストを自動的に実行する継続的なプログラム開発を意味する．CI では Git などの版管理システム上へのソースコードのコミット，プッシュを契機として自動ビルドと自動テストを実行する．これによりエラーを素早く修正でき，迅速なソフトウェア開発が可能となる．この自動ビルドと自動テストのアイデアはプログラミング教育において AA（Automation Assessment：自動評価）とも呼ばれている [4]．

CI の導入により学生のモチベーション向上 [7] [10] や，教員の負担削減 [1] [5] などの効果が見込まれる．本研究の対象となる競争型 PBL においては，CI 環境を用いた他者との比較，すなわち競争の自動化によりモチベーション向上が期待できる．また，一般的な CI ではテストの実行結果は原則，テストに成功したか失敗したかの二値で与えられる．競争型 PBL においては，テスト通過の可否ではなく対戦結果やプログラムの実際の挙動などを示すことで，教員のより詳細な進捗把握に役立つ可能性がある．

3 提案フレームワーク：CC

3.1 CC 概要

本研究の目的は競争型 PBL における学生の開発に対するモチベーション向上，及び教員による進捗把握の 2 つにある．この目的を達成するのために，我々は CI の考えに基づいた PBL 支援フレームワーク CC（Continuous Competition：継続的競争）を提案する．CC では CI 同様に，GitHub などの版管理システムを中心とした各種開発プロセスの自動化を実現する．具体的には，学生による版管理への変更（コミット）を契機に，ソースコードのビルドと対戦を自動的に実施する．対戦結果をランキング等の形式に変換し学生にフィードバックすることで，学生は他チームを上回った，あるいは他チームに上回れてしまったという相対的な順位を即座に把握できる．このフィードバックにより，学生のモチベーションの向上を狙う．教員の視点からは，どのチームがどのような相対順位にあるかを即座に把握できるため，開発がうまく進んでいないチームの早期発見といった進捗把握が容易となる．

CC が持つ CI との重要なアーキテクチャの違いの一つとして，単一のプロジェクトに対して単一の CI 環境ではなく，全プロジェクトに共通した単一の CC 環境を設ける点が挙げられる．よって，あるチームがソースコードを変更した際，ランキングの変動等の結果は全チームに即座にフィードバックされる．自チームだけでなく他チームの対戦結果を全体に公開することで，学生の競争心や向上心を駆り立てる．

なお，CC が用いる対戦結果としては，単純な勝敗やスコアだけでなく，個々の対戦のログやリプレイデータなどの利用を想定している．勝敗やスコアはフィードバックの基本となるランキングの情報源になる．また，対戦中の挙動を再現可能なリプレイデータを利用できる場合は，積極的に活用すべきであり，少なくとも対戦のログから対戦中の挙動がわかるよ

うに提示すべきである．リプレイは開発したプログラムの問題点や改善点の発見を助ける強力なフィードバックとなり得る．

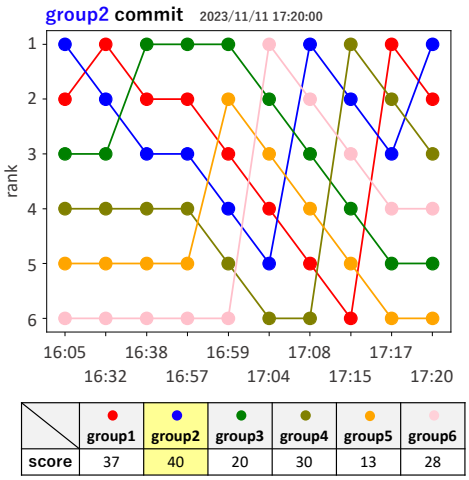
3.2 CC のフィードバック

本節では提案フレームワーク CC による，対戦結果のフィードバック方法について説明する．CC の自動対戦によって得られた対戦結果は，CC 専用の Web ページや Slack，メール等のコミュニケーションツールによって学生にフィードバックされる．Web は能動的な通知に適さないが，詳細な対戦結果の確認に適する．また，コミュニケーションツールは伝達可能な情報量に限界はあるが，速報性に優れる．どの手段を用いるかは任意であるが，これら 2 つのフィードバック方法を組み合わせることで速報性を確保しつつ，対戦結果の詳細を提供できる．ここでは特に Web を用いたフィードバック方法について取り上げる．

図 1 に Web によるフィードバックの一例を示す．学生による変更を検知する度に，図 1 に示す記事を 1 つ生成する．この記事は，記事生成元となったコミットの概要，直近数コミット（図の場合 10 コミット）のランキングの変動を表すランキングチャート，及び最新のコミットにおける対戦結果のスコアから構成されている．まず最上段のコミット概要からは，グループ 2 のコミットによりランキングの変動があったことを読み取れる．中段のランキングチャートは，縦軸がランク，横軸がコミットであり最新コミットが右側に追加されていく．図の場合，グループ 2 のコミットによりその順位が 3 位から 1 位に上がったと解釈できる．図の下段には最新コミットにおけるスコアの値を表示している．このスコアによって相対的な善し悪しだけでなく，絶対的なスコアの大小が確認できる．

図 1 は間接型競争の場合のフィードバックである．直接型競争の場合は全グループの組み合わせで対戦を行うため，各組に対する勝敗を示す必要がある．この場合，図の下段のような 1 行のスコア表ではなく，縦横グループ数の対戦結果表を用いればよい．ランキングチャート自体は直接・間接問わず同じ形式となる．

図 1 は教員へのフィードバックとしても利用可能であり，これに加えて，より詳細な進捗把握のため



	group1	group2	group3	group4	group5	group6
score	37	40	20	30	13	28

図 1　Web によるフィードバック記事の例

にリプレイデータや対戦ログを併用するべきである．スコア表の各スコアの値に対してハイパーリンクを付与し，リプレイや対戦ログへアクセスさせる．これにより，各グループの相対的な善し悪しや，絶対的なスコアの大小だけでなく，その挙動の詳細も同時に確認できる．リプレイや対戦ログを学生に公開する場合は，戦略やアイデアの盗用が発生しうるかを考慮する必要がある．PBL 題材の性質に応じて，リプレイやランク，スコアの公開範囲を検討するべきである．

3.3 競争型 PBL での CC 利用の流れ

競争型 PBL における CC 利用の流れを，版管理システムを GitHub，フィードバック方法を Web ページと Slack として図 2 に示す．

1. 学生がソースコードを改変し GitHub へコミット，プッシュを行う．
2. CC サーバがソースコードのプッシュを確認次第 GitHub からプルし，自動デプロイを行う．この時，プログラムの振る舞いに影響を与えないリソースのみの変更は無視する．
3. サーバ内の他の学生のコードと対戦を行い対戦結果を取得する．
4. 対戦結果を Web ページへ反映し，Slack へランク変動等のメッセージと共にポストする．
5. 学生は Web ページと Slack から，対戦結果と

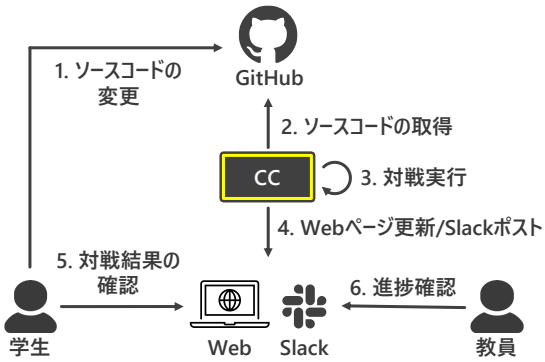


図 2　競争型 PBL における CC 利用の流れ

順位を即座に把握できる．

6. 教員は Web ページと Slack から，学生の対戦結果と順位を把握する．これに加えて，対戦ログやリプレイデータからプログラムの挙動を確認し，学生の開発状況を把握できる．

4　CC の試作

4.1　適用対象の PBL

適用実験として，大阪大学基礎工学部情報科学科の学部 2 年生を対象として開講されている，基礎工学 PBL（情報工学 B）（以降，PBL-B）に CC を導入した．PBL-B は競争型かつグループワーク形式の PBL 授業であり，1 グループ 5 人の計 14 グループで行われる．PBL-B では題材に競争型のプログラミングゲーム Battlesnake[†1] を採用している．図 3 に Battlesnake のゲーム実行画面を示す．図中の盤面上には一匹の蛇が存在し，毎ターン上下左右を一マス移動する．この蛇を制御するプログラムを開発し，その生存ターン数をスコアとして競う．学生は複数人で共同開発可能な Web 上のエディタを用いて開発を行い，その成果物を版管理システム上に提出する．

4.2　CC の実装

提案する CC フレームワークに基づき，PBL-B に特化した具体的な CC 環境を構築した．PBL-B の実施形態に合わせて版管理システムには GitHub を，

†1　https://play.battlesnake.com/

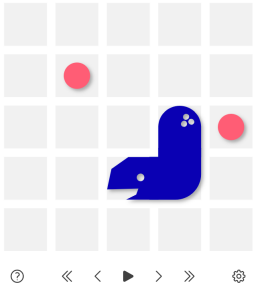


図 3　導入対象 PBL の題材 Battlesnake の実行画面

フィードバック方法には Web ページと Slack を採用した．実装した CC 環境は CI 環境と同様，サーバとして常時起動しており，Webhook を用いた GitHub リポジトリの変更監視，Battlesnake の対戦実行，対戦結果に基づく Web 記事の生成，Slack API を用いた Slack 記事の自動ポスト，Web 記事を公開するための Web サーバなどの機能を持つ．

さらに対戦のリプレイ機能を提供するために Battlesnake 公式が公開しているゲーム実行エンジン [†2] とリプレイヤ [†3] を改変し利用した．より具体的には，まずゲーム実行エンジン上で対戦が行われる度に，全ターンの盤面情報を持つ JSON を対戦ログとして記録する．この対戦ログを，改変したリプレイヤに読み込ませることで，Web ブラウザ上での対戦結果のリプレイを実現する．図 3 の下部にはリプレイ用のコントローラが示されており，1 ターン毎のステップ実行や巻き戻し等が利用できる．

5　予備評価

5.1　評価方法

本論文を執筆している時点で PBL-B は進行中であり，導入した CC の評価は完了していない．PBL-B は全授業の前半と後半を異なるルールで実施しており，ここでは CC の予備的な評価として，前半ルールを終えた授業 6 週分での CC 導入効果について説明する．評価の観点は以下 3 つである．

- Q1．CC は利用されているか
- Q2．CC は学生のモチベーションの向上に貢献

†2　https://github.com/BattlesnakeOfficial/rules
†3　https://github.com/BattlesnakeOfficial/board

しているか

- Q3. CC は教員の進捗把握に貢献しているか

まず Q1 として，前提となる学生の CC 利用状況を確認する．利用状況は利用頻度や使用感に関するアンケート調査と，CC 利用数の推移や対戦スコアとの相関から確かめられる．CC 利用数は，CC での対戦結果の確認を目的としたコミットの数で計測する．PBL-B 自体は毎週 1 回の議事録ファイルのコミット（及びプッシュ）をルールとして設けているが，それ以外のコミットは強制ではない．また Web 上で共同開発可能な環境で実装している点からも，基本的に学生のコミットやプッシュは不要である．それにも関わらず能動的にプログラムに関するソースコードをコミットをしているケースは，CC での対戦実行を目的としていると見なした．能動的なコミットのコミットメッセージの中には，議事録に関係ないものや，CC 利用の旨を書いたものがあり，明らかに CC を利用するコミットを確認できた．以降ではこの CC 利用を目的としたコミットの数を，CC コミット数と呼ぶ．

さらに Q2 として，研究の目的の一つである学生の開発に対するモチベーション向上に貢献しているかを確かめる．Q1 で行った CC の利用頻度のアンケート結果をもとに CC 利用者に対してのみアンケート調査を行う．

Q3 では二つ目の目的である教員による進捗把握への貢献を確かめる．第二著者が毎週 CC の提示するランキングの変動や対戦スコアを確認し，どのように役立てたかを議論する．

5.2 Q1 の結果（CC の利用状況）

5.2.1 アンケート調査による利用状況の調査結果

CC 利用状況として受講生に対して回答任意のアンケート調査を行い，CC の利用頻度に関する質問を行った．アンケートの回答率は受講生 70 名中 29 名の 41%であった．利用頻度は CC を「積極的に利用した」,「適宜利用した」,「あまり利用しなかった」の 3 段階の利用形態に分類し質問した．結果として，「積極的に利用した」という回答は 7 名（24%），「適宜利用した」という回答は 14 名（48%），「あまり利用しなかった」という回答は 8 名（28%）であり，アンケート回答者の 72%が CC を利用していることが分かった．

また，この CC を利用していた 72%の 21 名に対して CC の使用感に関する選択式と記述式のアンケートを行った．図 4 に選択式のアンケート結果を示す．アンケートでは CC のフィードバック方法である Web と Slack に対する使用感の質問に対し，「そう思う」から「そう思わない」の 5 段階のリッカート尺度により評価を行う．図 4 の「Web によるランク変動把握はコード改変結果の詳細な把握に役立った」という質問に対し，約 80%の学生が肯定的な回答をしている．また，「Slack からのランク変動把握はコード改変結果の即時把握に役立った」という質問に対し，約 70%の学生が肯定的な回答をしている．これより CC のフィードバックであるコード改変結果の確認として，Web による詳細な把握と Slack による即時把握は有効であるとわかる．記述式のアンケートでは「CC 利用で感じた利点をお教えください」という質問に対して，

- 開発したプログラムの出来栄えがすぐにわかる
- 現時点での進捗が分かって開発しやすかった
- 競争にあたって自班の順位が分かることで方針の取り決めに役立った
- ほかのチームがどれくらいの進捗かがよく分かった

と CC が自班や他班の進捗把握に役立っている旨の回答を複数得ることができた．以上のことから CC は学生の開発支援に一定の貢献があるとわかる．

5.2.2 CC コミットによる利用状況の調査結果

CC 利用状況として，図 5 に各週毎の全班の総 CC コミット数の推移を示す．第 5 週は授業が休講であったため CC コミット数は少なくなっているが，週を重ねるごとに CC コミット数が増加する傾向にある．そのため授業毎に CC 利用数は増加傾向にある．

また，CC の利用が学生の成績に与える影響を調査するために，スコアと CC 利用数（CC コミット数）の相関を確認する．図 6 に各班の CC コミット数とスコアの散布図を示す．図中の波線は 2 つの軸の平均値である．CC コミット数が平均値より多いグループは 5 班，少ないグループは 9 班であった．図より

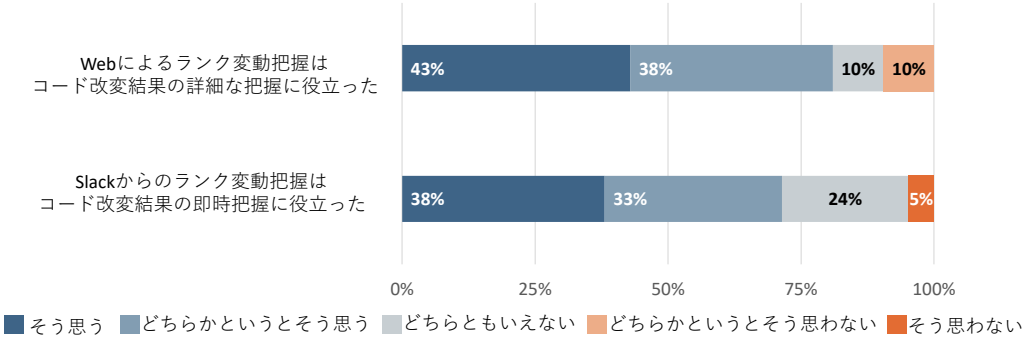


図 4　CC 使用感に関する 5 段階アンケート結果

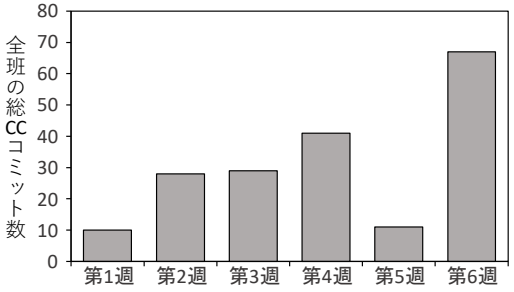


図 5　各週毎の全班の総 CC コミット数の推移

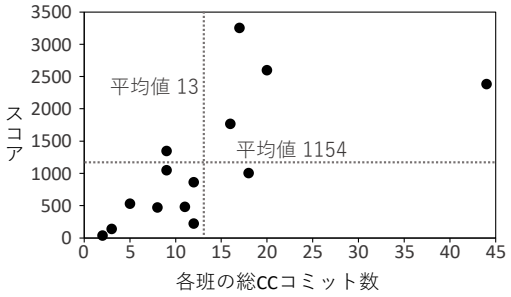


図 6　各班の総 CC コミット数と対戦スコアの散布図

平均値よりも高いスコアを出した 5 班のうち，CC コミット数が平均値より多いグループが 4 班占めている．また，CC コミット数が平均値よりも少ないグループにおいて，平均値よりも高いスコアを出した班は 1 班だけである．そのため CC 利用数が多い班ほど高いスコアを取得できている．

5.3　Q2 の結果（モチベーション向上）

CC によるモチベーション向上効果を評価するために，選択式と記述式のアンケート調査を実施した．アンケート対象は 5.2.1 節と同様，CC を利用していた 21 名の学生として調査を行った．図 7 に選択式のアンケート結果を示す．アンケートは図 4 と同様の 5 段階のリッカート尺度により評価した．図 7 の「開発に対するモチベーションが向上した」という質問に対し，90%以上の学生が肯定的な回答をしている．また，学生のモチベーション向上に関連する CC の主な要素として，プログラムの自動実行結果の即時フィードバックと他班の順位変動の可視化がある．この二つの要素に対しても選択式のアンケート調査を行い，その結果を図 7 に示す．プログラムの自動実行結果の即時フィードバックにおいては「開発プログラムの自動実行による即時ランク変動へ面白みを感じた」という質問に対し，約 85%の学生が肯定的な回答をしている．他班の順位変動の可視化においては「他班のランク変動から競争心が駆り立てられた」という質問に対し，約 75%の学生が肯定的な回答をしている．また，記述式のアンケートでは「CC 利用で感じた利点をお教えください」という質問に対して，

- 順位が変動して楽しかった
- ランキング上位に位置するとモチベーションが上がった
- 1 位になったとき班の士気が上がった
- 自分の班のプログラムが, 全体の中でどれくらいのレベルにあるのかを把握できたことが, 開発へ

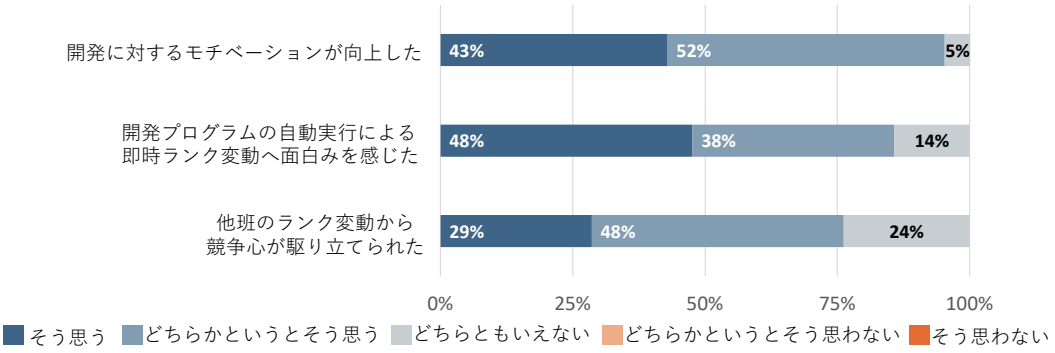


図 7　モチベーションに関する 5 段階アンケート結果

のモチベーションに繋がっていた
とモチベーションに関して肯定的な回答を複数得ることができた．以上のことから CC は学生の開発に対するモチベーション向上へ一定の貢献があるとわかる．

5.4　Q3 の結果（進捗把握）

教員は，CC のフィードバックからプログラムのバグやルールの欠陥に気づくことができた．具体的には，同じ班による複数回のコミットで順位が下位から変動しない場合に，リプレイや対戦ログを確認することで，タイムアウトエラーなどのプログラムの問題箇所を発見し該当班へ指摘できた．また，順位が急上昇した複数の班に対してリプレイ機能で挙動を確認すると，ルールの戦略性を無視したような単純なアルゴリズムの使用が判明した．そのような単純なアルゴリズムでも安定して高スコアを狙えるため，PBL-B で目的としているプログラミングの自主的な学習に繋がらないと考え，急遽ルール変更を設けることで対策を行えた．以上は CC のフィードバックがない場合には，学生からの申告や，実際に教員がプログラムを実行して気づく必要がある．その工程を省略して教員に伝達できるため，CC 導入が教員による学生の進捗把握に貢献していることがわかる．

6　関連研究

CI をプログラミング教育に導入する研究はその目的や授業形態などの違いにより数多く存在する [1] [6]．学生に対する開発支援やモチベーション向上などを目的とした研究として，Kasahara ら [6] や華山ら [10] は，CI に品質計測を組み込むことで，受講生のプログラムの品質に対する意識改善に取り組んでいる．学生による版管理システムへの変更を契機として CI ツールを実行し，その分析結果を用いてプログラム品質を評価しその評価結果を自動でフィードバックすることで，品質に対する意識改善を行っている．木崎ら [12] は，ソフトウェア開発 PBL での学生によるコードレビュー支援のために CI に基づいた環境を提案している．版管理システムへの変更を契機として CI ツールを実行し，その結果をレビュー担当の学生に自動でフィードバックすることでレビュー支援を行っている．また，教員に対して負担削減や進捗把握支援を目的とした研究として，Amelung ら [1] は，CI を基にした自動評価システムを授業に導入し，学生の課題提出に対し，Web 上でフィードバックを与えるシステムを提案した．同様に Insa と Silva [5] も，受講生の多いソフトウェア開発授業での多量の課題の採点に対して，CI を基にした自動評価システムを導入し評価作業の 48%の自動実行を可能とした．これらの既存研究は CI に基づいた環境を提案しているが，いずれも一つのプロジェクトに対して単一の環境のみを提供している．一方で，本研究で提案した CI に基づいたフレームワークである CC は，開発プロセスの自動化を全プロジェクトに対して共通した単一の環境として提供する．これにより本研究では，学生による相対的なプログラムの評価が可能となるため，学生の競争心を駆り立てモチベーション向上を狙うこと

ができる．また，教員による相対的なプログラムの評価も可能となるため，進捗が遅れている学生の発見などの開発進捗把握の支援も行うことができる．

7 おわりに

本研究では，CI の考えに基づいた PBL 支援フレームワーク CC を提案した．適用実験として，学部 2 年生を対象とした競争型 PBL 授業にこのシステムを導入し，学生の開発に対するモチベーション向上と教員による学生の進捗把握支援に一定の効果があることを確認した．

本研究の今後の課題としては，授業後半での CC 環境の保守が挙げられる．また，本研究の提案手法はフレームワークであるため，競争型 PBL の題材に依存せず，他の競争型 PBL への導入が可能である．提案手法は現状一つの授業にしか導入していないため一般化可能性の評価ができていない．一般化可能性を評価する一つの手段として，複数の授業への導入が挙げられる [1]．そのため，他の競争型 PBL への CC 導入により CC の一般化可能性を確認することは，一つの重要な課題である．

謝辞

本研究の一部は，JSPS 科研費（JP21H04877, JP20H04166，JP21K18302，JP21K11829）による助成を受けた．

参考文献

[1] Amelung, M., Forbrig, P., and Rösner, D.: Towards Generic and Flexible Web Services for E-Assessment, *Special Interest Group on Computer Science Education Bulletin*, Vol. 40, No. 3(2008), pp. 219–224.

[2] Ashay Desai, M. T. and Arbaugh, J. B.: Learning Through Collaboration and Competition: Incorporating Problem-Based Learning and Competition-Based Learning in a Capstone Course, *Organization Management Journal*, Vol. 11, No. 4(2014), pp. 258–271.

[3] Dagiene, V. and Stupuriene, G.: Informatics Education Based on Solving Attractive Tasks through a Contest, *Commentarii informaticae didacticae*, Vol. 7(2015), pp. 97–115.

[4] Ihantola, P., Ahoniemi, T., Karavirta, V., and Seppälä, O.: Review of Recent Systems for Automatic Assessment of Programming Assignments, *In Proceedings of Koli Calling International Conference on Computing Education Research*, 2010, pp. 86–93.

[5] Insa, D. and Silva, J.: Semi-Automatic Assessment of Unrestrained Java Code: A Library, a DSL, and a Workbench to Assess Exams and Exercises, *In Proceedings of ACM Technical Symposium on Computer Science Education*, 2015, pp. 39–44.

[6] Kasahara, R., Sakamoto, K., Washizaki, H., and Fukazawa, Y.: Applying Gamification to Motivate Students to Write High-Quality Code in Programming Assignments, *In Proceedings of ACM Conference on Innovation and Technology in Computer Science Education*, 2019, pp. 92–98.

[7] Kyrilov, A. and Noelle, D. C.: Do Students Need Detailed Feedback on Programming Exercises and Can Automated Assessment Systems Provide It?, *Journal of Computing Sciences in Colleges*, Vol. 31, No. 4(2016), pp. 115–121.

[8] Pérez, B. and Rubio, A. L.: A Project-Based Learning Approach for Enhancing Learning Skills and Motivation in Software Engineering, *In Proceedings of ACM Technical Symposium on Computer Science Education*, 2020, pp. 309–315.

[9] Ç ulha, D.: Applying Competition-Based Learning to Agile Software Engineering, *Computer Applications in Engineering Education*, Vol. 24(2016), pp. 382–387.

[10] 華山魁生, 柗本真佑, 肥後芳樹, 楠本真二: プログラミング教育における実績可視化システムの提案と評価, 情報処理学会論文誌, Vol. 61, No. 3(2020), pp. 115–121.

[11] 石山俊彦, 中島陽子, 神谷行基: 釧路高専専攻科における競争試作方式を適用した PBL 型演習の実践例, 工学教育, Vol. 56, No. 5(2008), pp. 11–16.

[12] 木崎悟, 田原康之, 大須賀昭彦: アジャイル型ソフトウェア開発 PBL における CCBR を拡張したコードレビュー支援環境の提案, 情報処理学会研究報告, Vol. 2013-CE-118, No. 9(2013), pp. 1–8.