K-021

プログラミング演習としてのコードレビュー支援ツールにおけるリファクタリングに関する機能の検討

Functions about Refactoring and Discussion of a Code Review Support Tool for Programming Exercises

植 勇希† 富永 浩之†
Yuki UE Hiroyuki TOMINAGA

1. はじめに

本研究室では，情報系分野の文献を題材とし，PC を用いた演習を伴う輪講を支援するシステム Lexit を提案している[1]．一連の輪講をプロジェクトと捉え，個々の輪講をセッションとして，全体の進行を統括する．また，輪講で配布するレジュメや発表のスライドなどの資料を管理する．

Lexit の一環として，プログラミング課題を提示し，各自の解答を相互に検証するコードレビュー支援ツール Redoc を開発している[2]．本論文では，試作版からの再開発として，必要な機能を検討する．

2. コードレビューの教育的な側面

本研究では，輪講における演習として，プログラミング課題の解答に対するコードレビューを扱う．コードレビューでは，チーム内のメンバが作成したプログラムのソースコードに対して，本人を交えて，チーム内外のメンバで閲覧し議論する．各自が作成したソースコードに対して，問題点や改善点を指摘したり，本人に助言や指導を行う．

ユーザの役割は，コードを作成してレビューされる者をレビュイー(提示側)，コードを閲覧してレビューするものをレビュアー(指摘側)と呼ぶ．コードレビューの目的には，教育的と成果的との 2 つの側面がある．

教育的な側面では，提示側のコードをメンバ全員で共有，提示側による意図・目的や設計との対応の説明，指摘側から提示側へ不明点などの質疑応答，メンバ相互による問題点や改善点の指摘，提示側への助言や指導などがある．チューターなどが模範コードを用意しておき，それとの差異を議論することがある。参加者の他人のコードも参照することがある．

成果的な側面では，暗黙知の明示化と意識化，コーディング規約や開発ツールの標準化，各自のコーディング技能や特性の相互理解，プロジェクトのメンバとしての相互のコミュニケーション，システム開発の目標や理念の確認がある．本研究では，前者の教育的な側面に特化して支援を検討する．

3. コードレビュー支援のフェーズ

本研究の Redoc は，Web 上のシステムであり，クライアント側のブラウザで動作する．Redoc は，輪講セッションの演習において，提示されたプログラミング課題に対して解答コードを提出し，レビューを実施し，復習を行うまでを支援する．Redoc は，プログラミング演習のみの授業や自主セミナーも想定し，単独のツールとしても利用できるようにする．Redoc での支援は，コード作成，レビュー実施，復習再現の 3 つのフェーズに分ける(図 1).

†香川大学 Kagawa University

コード作成フェーズでは，プログラミング課題が提示され，各自が解答コードを作成する．作成した解答コードの提出は，Redoc を通してサーバ側へのアップロードで行う．提出された解答コードの一覧から，レビューを実施するコードを選択する．選択されたコードを対象として，レビュー実施フェーズに移行する．

レビュー実施フェーズでは，提示側は，コード作成フェーズで選ばれたコードを提出した参加者であり，その他の参加者は指摘側となる．提示側のコードや，コーディングそのものに対して，コメントを投稿することによりレビューを行う．

復習再現フェーズは，レビューの実施ログを基に，レビューの様子を再現する．コードレビューを欠席した参加者や，出席はしたが復習を行いたい参加者が利用し，レビューの過程を追体験する．

4. コードレビュー支援ツールの利用方法

Redoc の利用方法を，段階に応じて，図 2 の 4 区分とした．第 1 区分(実装指導)は，提示側を入門者，指摘側を中級者として，中級者が入門者に初歩からのコーディングを指導するものである．教育内容は文法事項,書法指導である．指摘側に入門者を加え，指導の様子を見て不明点などを質問させてもよい．主な対象は，C 言語である．

第 2 区分(改良検討)は，提示側を中級者，指摘側を主に初級者とする．提示コードは，あえて修正すべき書法や改良の余地を残しておく．指摘側で，問題点やデバッギングについて議論する．初心者同士の議論で行き詰まった場合には，提示側の中級者がヒントを提示する．対象としては，C++/Java を含み，オブジェクト指向も意識させる．

第 3 区分(品質議論)は，提示側を初級者，指摘側を中級者とする．中級者が初級者の解答したコードに対し，コード品質,語法について指導する．それにより発生する修正やデバッグ行為についても指導の対象である．対象としては，Ruby/Python/Perl などスクリプト言語も含まれる．ただし，情報セキュリティの観点から仮想環境の導入も検討する．

第 4 区分(示範鑑賞)として，上級者によるコーディングの実演を行う．指摘側の鑑賞しながらの質疑に対して，上級者が答えながら進める講義のような形式となる．

5. コードレビュー支援ツールの機能

Redoc の機能および利用手順について，主にレビュー実施フェーズについて説明する．図 3 に，レビュー実施ページの GUI を示す．中央にエディタ部，下方にタイムライン部がある．

レビューは，コメントの投稿として行う．指摘側が質問や指摘，助言を行う．それに対し，提示側は質問に回答したり，指摘や助言に従ってコードを修正する．コメントは，

コードにおける指摘箇所を指定して投稿することができる．

コードの変更は，エディタ部のコードを編集することで行う．コンパイルと実行により，動作を確認する．これらのデバッギングの状況もレビューの対象とする．ある程度の変更を行った後，提示側がコードのバージョンアップを指定する．以上を繰り返し，提示側に段階的な実装を促す．

レビュー実施ページでは，同期処理が行われる．提示側によってエディタ部のコードが修正されると，他のクライアントに変更が反映される．これにより，指摘側はリアルタイムに提示側のコードの変更を確認することが可能である．タイムライン部には，コメントの投稿，コンパイルと実行，提示側によるバージョンアップ指定などのイベントが一覧表示される．このタイムラインへの通知も同様に，リアルタイムに確認することができる．

また，イベントが記録されると，同時にコードのスナップショットが保存される．各時点のスナップショットは，タイムラインの詳細ボタンから閲覧する．タイムラインを辿ることで，以前の状態のコードを参照することが可能である．

6. 第 3 区分における進行例と機能検討

Redoc の再開発を進めるにあたって，利用方法の第 3 区分(品質議論)の進行例を基に，機能を再検討する．ここでは，C 言語を対象とする．ここで，参加者は，出題された仕様を満たす解答コードを，ほぼ書けるレベルとする．

題材としては，多進数の変換を事例とする。まず，10 進数 n と基数 r の入力に対し，r 進数への変換の出力を実装させる． 11≦r≦36 での英数字，r≧36 での区切による表示(100 進数を 23:06:12 のように表記)も適切に実装する．指摘側は，配列を用いた必要な関数の適切な定義が実現されているかを確認する．続いて，r 進数の入力から 10 進数への変換の出力を実装させる．ここで，入力の様式に関して検討を要する．ホーナー法による効率的な実装を確認する．桁の数字の不正に対する例外処理も必要である．

指摘側の機能としては，まず，指摘箇所の行単位での指定がある．解答コードに対し，1 行のみ，または範囲の開始から終了までを選択する．新機能として，変数名や関数名で検索し，複数の出現箇所の指定や語句のハイライト表示を検討する．次に，コメントの意図として，提示側への質問/提示側からの回答/全体への疑問/提示側への指導/全体への議論/全体への解説/司会者の進行などの区分を選択できるようにする．コメント自体は，典型例文やエモーションアイコンを用意し，簡潔な入力で済むようにする．コメントへの添付資料として，入出力サンプルの例示，コード例の提示，過去のスナップショットの参照，各種のコード指標の計算などを検討する．また，参照情報として，リファクタリングの「不吉な臭い」やデザインパターンから該当例にリンクできるようにする．

7. おわりに

演習を伴う輪講を支援するシステムの一環として，プログラミング課題を提示し，各自の解答を相互に検証するコードレビュー支援ツール Redoc を提案している．エディタ部でのコード編集と同期表示，指摘箇所を範囲指定してのコメントの投稿，イベント発生によるスナップショットの保存，タイムラインの閲覧，実施ログの表示などの機能を実装している．

本論では，ツールの利用法を再検討し，教育内容，提示側，指摘側に応じて，4 区分に整理した．実装指導と品質議論では，提示者の解答コードを指摘側の議論で修正していく．改良検討と示範鑑賞では，提示コードの更新に対して，指摘側が質疑や検討を展開する．これらの利用方法のうち，まず，第 3 区分の品質議論に着目し，例題に対する進行事例を挙げ，機能検討を行った．

今後は，他の区分に対しても，同様の議論を進め，ツールの全体的な仕様を確定させる．また，対象コードのコンパイルと実行を安全に実現する仮想環境の導入も検討する。仮想環境では，様々な入出力サンプルによる柔軟なテストを可能にする．静的解析ツールも組み込み，コード指標による品質の評価なども取り入れる．試作版によるユーザ評価を繰り返し，実用的なツールを目指す．

参考文献

[1] 豊田竜也, 中矢誠, 大川昌寛, 川鯉光起, 富永浩之, "プログラミング演習としてのコードレビューを取り入れた輪講支援システム - 解答コードの同期閲覧とスナップショットへのコメント機能の実装 - ", 信学技法, Vol.113, No.482, pp.59-64 (2014).

[2] 川鯉 光起, 中矢 誠, 富永 浩之, "研究室規模の蔵書管理と文献共有による学習支援 - コードレビューを取り入れた共時学習としての輪講の支援 - ", 情処研報, Vol.2012-CE-119, No.20, pp.1-10 (2013).

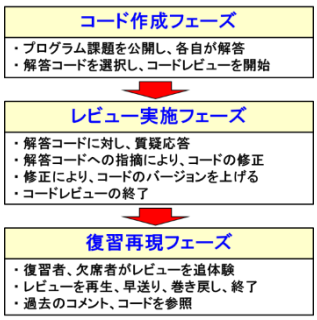

図 1 輪講プロジェクトの進行モデル

区分	名称	教育内容	提示者	主な指摘側
1	実装指導	文法事項、書法指導	入門者	中級者、入門者
	入門者のコーディングを中級者が指導 指摘側の入門者も不明点を中級者に質問			
2	改良検討	デバッグ、テスト	中級者	初級者
	中級者が改良の余地を残したコードを提示する 中級者が提示したコードの改良点を指摘側の初級者と検討していく			
3	品質議論	リファクタリング	初級者	中級者
	初級者の解答したコードの性能や品質を中級者と議論 初級者はリファクタリングの指摘に従ってコードを修正する			
4	示範鑑賞	コーディング実演	上級者	中級者
	複雑で高度な課題に対する上級者のコーディング実演 指摘側からの質疑に上級者が答える講義形式			

図 2 コードレビュー支援ツールの利用方法

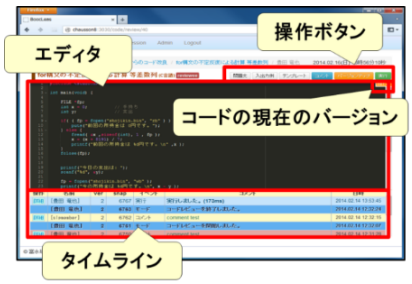

図3 コードレビュー支援ツールの GUI