4X-04

コーディング過程を共有する Web アプリケーション用フレームワークの改良法の提案

鈴木 颯渡 † 早川 智一 †
明治大学理工学部 †

1. はじめに

近年，IT 産業の発展にともない，プログラミング教育の重要性が増してきている．これは，高品質なソフトウェアを設計・実装できる技術者が不足しているためである [1]．

プログラミング教育において，他者がソースコードを作成する過程（以下，コーディング過程）から有用な知見が得られることが経験的に知られている（2 章）．この例として，ペアプログラミングやライブコーディングが挙げられる．

しかし，コーディング過程を用いたプログラミング教育は容易ではないという課題がある．なぜならば，このような教育には人員や準備を多く必要とするためである．例えば，ペアプログラミングを導入する場合，各学習者に教育者が 1 人ずつ必要となる．ライブコーディングを導入する場合，教育者が機材を準備する必要がある．

我々は，この課題を解決するため，コーディング過程の追体験――眼前で他者がソースコードを作成しているような体験――が可能な Web アプリケーションの実現を支援するフレームワークを提案してきた [2]．提案フレームワークは，利用者のコーディング過程を記録する機能と記録された他者のコーディング過程を再生する機能とを持つ（3.1 節）．

しかし，提案フレームワークには，記録対象のコーディング過程の情報量の増大にともないデータ転送量が指数的に増加する設計上の課題があった．これは，提案フレームワークを REST（REpresentational State Transfer）で設計したため，そのべき等性――ある操作を 1 回以上，何回行っても結果が同じになること――により生じていた．

本論文では，提案フレームワークの課題を解決するための改良法を提案する．具体的には，設計に WebSocket[3, 4] を導入することで転送量の削減を試みる（3.2 節）．我々は，転送量の評価を通じて改良法の有効性を確認した（4 章）．

A proposal of improvement method for web application framework that shares coding process with others

†Hayato Suzuki and Tomokazu Hayakawa, School of Science and Technology, Meiji University

2. 関連研究

ペアプログラミングやライブコーディングに関する研究として文献 [5, 6] がある．いずれの研究も大学の授業でそれらの有用性を検証し，効果的であると報告している．

伊藤ら [7] は，コーディング過程の動画による追体験に学習効果がある可能性が高いと主張している．

Boisvert[8] は，コーディング過程の可視化がプログラミング教育に効果的であると考え，コーディング過程を再生する Web アプリケーションを提案している．Boisvert と我々の研究の共通点は，コーディング過程を利用者に見せる Web アプリケーションを提案している点である．しかし，Boisvert の研究は，利用者のコーディング過程の記録を目的としていない点で我々の研究と異なる．

3. 提案手法

3.1. 前提

我々の提案フレームワークの目的は 2 つある．1 つ目は，コーディング過程の記録・再生を容易に実現することである．なぜならば，コーディング過程の記録・再生に手間がかかると，教育への導入が難しくなるためである．2 つ目は，コーディング過程を記録・再生する機能を既存のシステムに容易に組み込み可能にすることである．なぜならば，既存のシステムへの組み込みが容易でないと，プログラミング教育に LMS（Learning Management System）を利用している場合に提案フレームワークの導入が難しくなるためである．

提案フレームワークの利用例を図 1 に示す．提案フレームワークは，クライアント部分とサーバ部分とからなる．クライアント部分はエディタ機能を提供し，利用者はこの上でコーディングする．利用者のコーディング過程はサーバ部分に自動的に記録される．また利用者は，他者のコーディング過程をエディタ上で再生できる．利用者は，これらの機能に

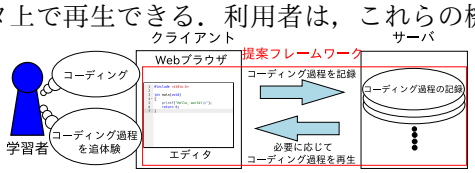

図 1 提案フレームワークを用いたシステムの概要図

より，コーディング過程の追体験ができる．

図 2 に，既存のシステムに提案フレームワークを組み込む際の HTML（HyperText Markup Language）の例を示す．図より，3 行のソースコードでエディタ機能を組み込めることが分かる．これは，提案フレームワークを SPA（Single-Page Application）として設計したためである（SPA は，ページ遷移を必要としないアプリケーションである）．また，カスタムタグ（`editor`）の属性を指定することで，エディタの機能を指定できる．図中では，`lang` 属性を指定することで記録対象の言語を指定している．

3.2. 概要

我々は，REST による設計の利点をできる限り維持しつつデータ転送量の課題を解決するため，コーディング過程の記録時に WebSocket を利用する改良を試みた．REST による設計は，その特性により信頼性が高く汎用性なシステムを実現する．しかし，転送量の課題が大きく，REST による設計の利点を低減してでも改良が必要であった．

図 3 と図 4 とは，コーディング過程を記録する際の改良前後のシーケンス図である． 改良前は，HTTP（HyperText Transfer Protocol）を用いてコーディング過程を記録していた．その際に，REST のべき等性を担保するため，直前のコーディング過程の記録をサーバから取得（GET）し，その末尾に更新内容を追加してサーバに転送（PUT）する必要があった．改良後は，クライアント・サーバ間で WebSocket のコネクションを確立し，クライアントからサーバにコーディング過程の更新内容のみを転送する．この転送はべき等である必要がないため，REST と比較して軽量な転送を実現できる．

```
<div id="app">     属性で機能を指定
    <editor lang="c_cpp">
</div>   エディタ機能を埋め込むカスタムダグ
                この部分がSPAとなる
```

図 2　提案フレームワークを組み込む HTML の例

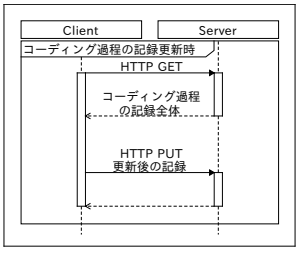

図 3　REST による設計でコーディング過程を記録する際のシーケンス図

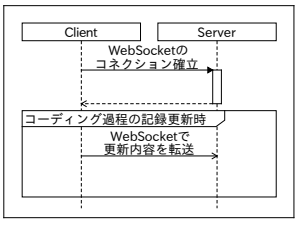

図 4　WebSocket を用いてコーディング過程を記録する際のシーケンス図

表 1　コーディング過程の記録に必要な転送量

	ソースコード容量（バイト）	改良前（バイト）	改良後（バイト）	減少率（%）
Hello World	84	1,157,113	11,631	98.99
echo	148	2,511,701	18,054	99.28
FizzBuzz	360	13,222,850	41,941	99.68

4. 評価

我々は，改良法の有効性を確認するために，コーディング過程を記録する際のデータ転送量を改良前後で比較した．具体的には，プロトタイプを作成してコーディング過程を記録し，記録時に転送した IP（Internet Protocol）パケットのペイロード長の合計を測定した．また，記録のコーディング過程の情報量と転送量との関係を把握するため，容量の異なる 3 つのソースコードを評価に用いた．

表 1 に結果を示す．表より，改良前は，Hello World（84 バイト）のコーディング過程の記録でさえ 1MB 以上の転送量がかかっていた．改良後は，Hello World を記録する転送量は 98% 以上減少した．また改良後は，ソースコード容量が大きい際に転送量の減少が特に顕著であることが分かる．これは，べき等性を担保するための転送がなくなったためである．以上より，改良法は提案フレームワークの課題の解決に有効であるという結論を我々は得た．

5. おわりに

本稿では，コーディング過程を共有する Web アプリケーション用フレームワークの改良法を提案した．今後は，HTTP の転送量を評価することを予定している．

参考文献

[1] 独立行政法人情報処理推進機構：IT 人材白書 2018.
[2] Hayakawa, T. and Suzuki, H.: Design and implementation of programming learning system that shares coding process with others, *Proc. of CSEIT 2018*, Singapore, pp. 97–100 (2018).
[3] W3C: The WebSocket API, https://www.w3.org/TR/websockets/.
[4] Fette, I. and Melnikov, A.: The WebSocket Protocol. https://tools.ietf.org/html/rfc6455.
[5] Braught, G., Wahls, T. and Eby, L. M.: The case for pair programming in the computer science classroom, *ACM Transaction on Computing Education*, Vol. 11, No. 1, pp. 1–21 (2011).
[6] Paxton, J.: Live programming as a lecture technique, *Journal of Computing Science in Colleges*, Vol. 18, No. 2, pp. 51–56 (2002).
[7] 伊藤恵，杉本識史，大場みち子，下郡啓夫：動画を用いたプログラミング自学自習の試み，研究報告コンピュータと教育（CE），Vol. 2015-CE-128, No. 20, pp. 1–6 (2015).
[8] Boisvert, C. R.: A visualisation tool for the programming process, *Proc. of ITiCSE 2009*, Vol. 41, No. 3, Paris, pp. 328–332 (2009).