C-07

自己調整学習を支援するコンパイルログ分析システムの開発
Developing a Compilation Log Analysis System for Self-Regulated Learning

藤井 裕也†　　山元 翔‡　　越智 洋司‡
Yuya Fujii　Sho Yamamoto　Youji Ochi

1. はじめに

プログラミング教育は，単なる知識の伝達にとどまらず，学習者自身の問題解決能力や主体的な学びを促進することが求められている．特に，プログラミング行動には，反復的な問題解決，デバッグ，継続的な改善を伴うため，自己調整学習（Self-Regulated Learning: SRL）の導入が期待されている．自己調整学習は，学習者が自身の学習プロセスを能動的に管理する能力であり，中尾ら[1]は，SRL を促すことがプログラミング学習成果に好影響を与える可能性を示唆しており，國宗ら[2]は，SRL に関する理解が自己効力感や認知的方略の活用に良い影響を及ぼすことを報告している．これらの研究は，学習者が自らの学びをメタ認知的に管理する自己調整学習が，プログラミング教育において有効であることを示している．本研究では，コンパイル履歴データの詳細な分析に特化した SRL 支援のために，学習者のコンパイル試行・エラー・修正の履歴を高粒度に収集し，時系列的に分析するコンソール型ツール「LogTerminal」を開発する．蓄積されたコンパイル毎の細かなログを可視化することで，学習者自身が逐次的な試行錯誤の過程を振り返り，内省できる環境を提供することを目的とする．

2. 自己調整学習とプログラミング学習支援

2.1 先行研究と本稿の位置づけ

Zimmerman は SRL の三段階のモデルを提案しており，これは人の認知モデルに基づいているとされている[3]．しかし，プログラミングなどは情報処理の対象であり，また，システムで学習者の SRL を細やかに支援したいと考えるのであれば，情報処理モデルとして SRL を捉えることが望ましい．このことから筆者らは，情報処理モデルとして SRL をモデル化した Winne らのモデル[4]を対象とし，プログラミングにおける SRL と，その支援を検討する．

プログラミングにおける SRL 支援のアプローチを導入する研究としては，Watanabe[5]らの TrackThinkDashboard は Web ブラウジング履歴とプログラミング行動ログを一つのビューに統合し，学習者の問題解決過程を可視化しているなど，学習者の行動パターン全体を俯瞰的に把握するアピローチもある．しかし，宮澤ら[6]の研究が示唆するように，プログラミング学習において試行回数と学習成果との間に関係が見られる可能性があり，個々の試行錯誤履歴に基づく振り返り支援は重要と考えられる．また，Jadud[7]はコンパイルログデータから反復エラーの程度を定量化する EQ（Error Quotient）を提案しており，EQ が高い（同じエラーを繰り返している）学習者ほど学習成果が低い傾向が報告している．つまり，コンパイル履歴に現れる試行パターンと学習成績には相関があることが示唆されている．以上の知見は，コンパイル履歴という学習行動をより深く SRL に結びつけることの必要性を示している．

そこで，本研究は，学習者一人ひとりの課題ごとのコンパイル履歴や修正履歴を時系列的に深く捉えることで，SRL の支援を狙いとしている．このようなアプローチは，SRL のプロセスとして自身の学習行動に関する意識化を深める有効な支援手法となると考える．

2.2 プログラミング学習における SRL の支援

Winne らは 4 段階のフェーズから SRL を定義している．第 1 段階はタスクの定義であり，目の前の課題を認識する段階である．第 2 段階はゴールの設定と計画立案であり，学習者自身が目標を設定し，それに取り組むための計画を立てる．第 3 段階は 2 段階で設定された戦略などを適用する段階であり，学習者はその戦略を踏まえて情報を構築したりする．第 4 段階はメタ認知の適用であり，学習者は現在の認知の状態と基準，すなわちゴールの間に差異があるかどうかを確認し，差異が確認されれば，第 3 段階で試行錯誤した戦略がどの程度有効であったかなどを検討する．つまり他のフェーズを見直すことも行われるため，これらは弱い順序性によって成立していると考えられる．

本研究では学習者の SRL を支援するため，筆者らはコンパイルログに注目した．コンパイルはプログラミング学習において必ず行われる行為であり，観測しやすく介入にも適している．学習者は課題に対して自ら下位目標を設定し，コンパイルによってその達成を確認する．この時，コンパイルは学習者が定義したタスクに基づく試行とみなせる．ただし，そのタスクが正しいとは限らないため，最終的には目標設定自体の見直しが必要になる場合もある[8].

以上から，学習者は第1〜3段階で目標を設定し，コンパイルを通じて試行を行い，第 4 段階でその試行を振り返ることが求められる．通常のプログラミング学習においては，学習者は第 4 段階を十分に行えず，課題の調整や達成に失敗しやすい．しかし，プログラミング学習において第 4 段階は目標達成に不可欠であるため，本研究ではこの段階の支援を重視する．

2.3　コンパイルログに基づく SRL の支援

本研究では，プログラミング学習における SRL の支援を目的として，学習者のコンパイルログを活用した観測・介入手法を提案する．コンパイル行動そのものは，SRL の一連のプロセスの中で学習戦略の一部として位置付けられるが，その結果をどう解釈し，次の行動にどう活かすかという内省的プロセスが第 4 段階に該当する．したがって，コンパイルログには，学習者が立てた目標や実行した戦略の痕跡が含まれており，それを分析することで，SRL の第 1〜第 3 段階（目標設定，計画立案，戦略の適用）を間接的にトレースすることが可能である．具体的には，第 1 段階は，課題内容と編集内容の差異から推測することができる．第 2 段階は，コンパイル対象から推測することができる．第 3 段階はコンパイルエラーの種別から学習者が見直すべ

†近畿大学大学院総合理工学研究科
‡近畿大学情報学部，近畿大学情報学研究所

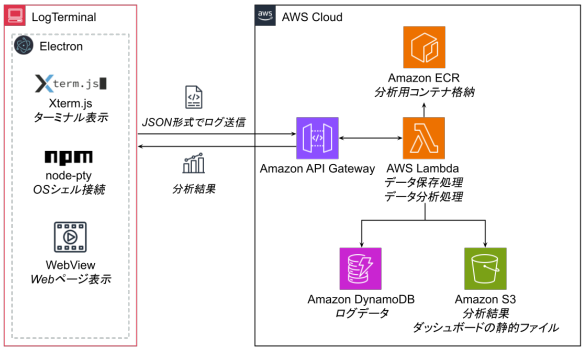

図 1　システム構成

き戦略，あるいはその適応結果の誤りとして推測することができる．そして，コンパイル結果に基づいて学習者が行う再編集や試行の変化を観察することで，第 4 段階での内省や自己修正の有無を捉える手がかりとすることができる．

3. システム概要

3.1 システム構成

3.1.1 クライアント部（LogTerminal）

クライアント部（LogTerminal）は，学習者のプログラミング活動を妨げることなく，コンパイルログを効率的に収集し，構造化するために設計されたターミナル環境であり，Electron をベースとしている．Electron の採用により，Web 技術とネイティブ機能の両立が可能となり，豊富な JavaScript ライブラリの活用が実現される．本研究においてターミナル機能の実装には xterm.js と node-pty を使用している．

xterm.js によるリアルタイムターミナル表示は，学習者に普段と変わらないシームレスなターミナル環境を提供する．これにより，学習のシステム導入の障壁を下げ，自然な形でログを収集する．node-pty を用いたシェルプロセス管理は，OS のネイティブなシェルプロセスとフロントエンドの xterm.js を接続する役割を果たす．この実装により，実際のコンパイルコマンドの実行と，その標準入出力の正確な捕捉が可能となっている．これは，学習者の活動を詳細に記録し，後の分析に活用する．

3.1.2. サーバー部

サーバー部は，SRL を支援するコンパイルログ分析システムの根幹をなすものであり，Amazon Web Services（AWS）上にサーバーレスアーキテクチャとして構築した．学習者の行動データを効率的に処理・分析し，Winne らが提唱する SRL モデルにおけるメタ認知の適用を促進する SRL 支援機能を AWS Lambda 関数によって提供する．

LogTermial で収集・構造化したログデータは，RESTful API を介してサーバーサイドに送信される．この通信インターフェースの採用により，クライアント・サーバー間の疎結合が実現され，将来的な拡張性も確保されている．

学習者のコンパイル行動はリアルタイムでシステムに受信され，データの送信は非同期で行われることで，学習者のプログラミング作業を妨げないよう配慮されている．また，ネットワークエラーなどにより送信が失敗した場合のエラーハンドリング機構も備えており，データの損失を抑える．受信されたデータは，ストレージサービスである Simple Storage Service に格納され，NoSQL データベースである Amazon DynamoDB により管理する．これらのサーバ

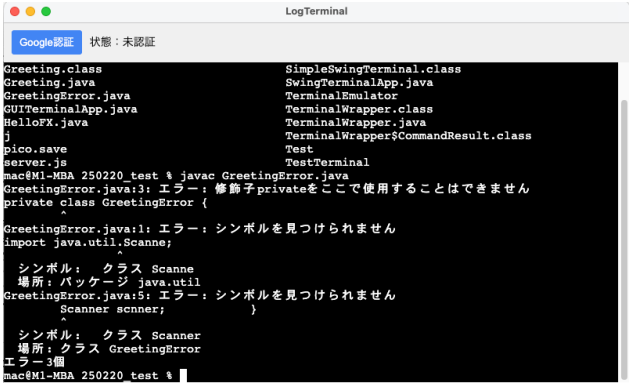

図 2　LogTerminal

設計は将来的に新たな分析指標が求められた際に，ログ項目の構造を柔軟に変更できる拡張性をもたらし，システムの持続的な改善を可能にする．

3.1.3 Google OAuth2 によるユーザ認証

学習者個人のログを個別に管理・分析するため，Google OAuth2 を用いた認証機能を実装している．認証後のセッション管理には，JWT（JSON Web Token）を使用している．JWT の利用により，サーバーへのリクエスト毎にユーザーを識別することが可能となっている．取得したトークンなどの認証情報は，学習者のローカル環境に暗号化して保存される．この実装により，セキュリティを確保しつつ，学習者の利便性も維持している．

3.2 実装機能

3.2.1 コンパイルログ自動収集機能

LogTerminal は，学習者が実行するコマンドの中から，Java のコンパイルコマンド（javac）に限定して検出する機能を有している．この検出は，LogTerminal 内でのコマンドを常時監視し，正規表現を用いて行っている．コンパイル結果の判定は，コマンド実行後の出力内容を基に自動的に行われる．システムは，コマンドの実行結果を解析し，コンパイルの「成功」または「失敗」を判定する．エラーメッセージの構造化抽出も実装されている．コンパイル失敗時に出力されるエラーメッセージは，「エラー種別」「発生箇所（行番号）」「メッセージ内容」といった構造化データとして抽出・解析される．

3.2.2 ログ分析機能

本システムのバックエンドにおける中核は，単にログを集計するだけでなく，その結果を学習者の SRL 支援に繋げる学習行動分析にある．本システムのログ分析機能は，フロントエンドの LogTerminal から送信される API リクエストをトリガーとして起動する．収集されたログデータの分析処理は，Python をランタイムとする別の AWS Lambda 関数によって実行される．分析の実装には，pandas や numpy，awswrangler といったデータ分析に特化したライブラリ群を活用している．これらのライブラリを用いることで，DynamoDB に蓄積された大量の時系列ログデータから，成功率の算出やエラーパターンの分類といった集計・加工処理を実現している．

3.2.3 学習行動分析機能

学習者が自らの学習プロセスを客観的に評価し，内省することを促すための分析機能を提供する．分析結果は，フ

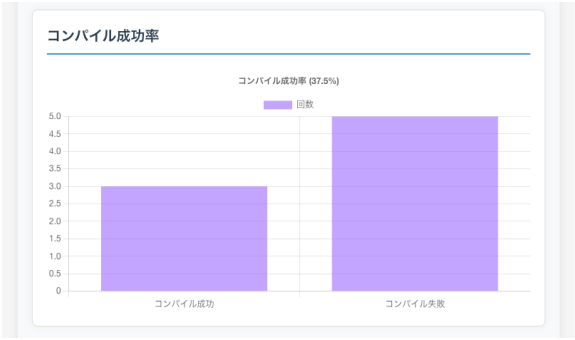

図 3　コンパイル成功率

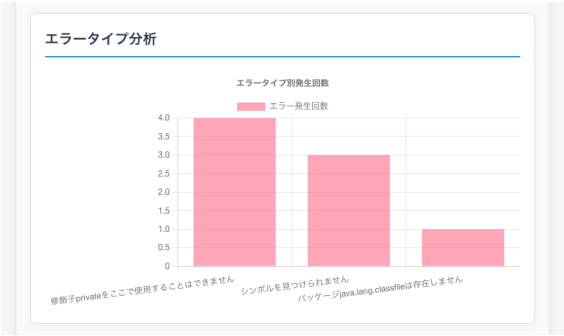

図 4　エラータイプ分析

ロントエンドでの可視化に適した JSON 形式で生成され，具体的には，以下の分析機能を実装している．

(1) コンパイル成功率の時系列分析
学習者が自身の進捗傾向を把握できるよう，コンパイルの成功・失敗履歴を時系列で可視化する．この情報は，第 4 フェーズにおける結果のモニタリングを促すとともに，第 2 フェーズ（計画立案）における振り返りと調整にも活用される．

(2) エラーパターンの分類と頻度分析
繰り返し発生するエラーの種類や出現頻度を提示し，学習者に「つまずき」の傾向を気づかせる．これは単なる結果の記録にとどまらず，問題の構造的理解や戦略の再考を促す材料となり，第 3 フェーズ（戦略の適用）との往還的な振り返りを支援する．

(3) コンパイル間隔やコード修正量の分析
コンパイルまでの時間や修正行数といった情報を元に，学習者の行動スタイルを示唆する．この情報は，自己学習スタイルの認識をさせ，第 4 フェーズに対応する．

3.2.4 学習者ダッシュボード

学習者ダッシュボードは，学習環境に統合された LogTerminal から呼び出され，前節で述べた SRL 支援のために分析された各種指標を取得し，学習者用のウェブインタフェース上にダッシュボードとして提示される．Web インタフェースをベースにしているため，フロントエンド側では柔軟な UI 設計が可能であり，新たな機能拡張も容易としている．図 3，図 4 は，学習者に提示されるダッシュボードの一例である．図 3 は，課題ごとのコンパイル成功率を 2 つの棒グラフで示しており，図 4 では，発生頻度の高い 3 種類のエラー（例：private 指定の誤り，未定義のシンボル，存在しないパッケージなど）を棒グラフで表示し，つまずきの傾向を可視化する．

これらの指標は SRL における第 4 フェーズを支援するものであり，学習者が自身の進捗状況やエラー傾向を客観的に把握し，次の学習行動を計画・調整するための基盤となる．可視化された情報に基づく内省は，学習の過程をより意識的に捉える契機となり，結果として自己調整的な学習の促進につながる．

4. おわりに

本研究では，プログラミング学習における自己調整学習を支援するために，学習者の試行錯誤の過程を可視化するコンパイルログ分析システム「LogTerminal」を開発した．Winne らの SRL モデルに基づき，学習者のコンパイル行動を 4 フェーズに対応づけ，各フェーズにおける誤りやつまずきを分析・フィードバックすることで，内省を促進する仕組みを構築した．システムは，Electron と xterm.js による統合ターミナル環境と，AWS を活用したサーバーレス構成によって，学習を妨げずにログの自動収集・構造化・分析を可能にしている．また，Google OAuth2 と JWT による認証機構により，個人ログの安全な管理も実現している．

今後の課題としては，学習者の理解度に応じた動的なフィードバックの最適化や，SRL 支援効果の定量的評価が挙げられる．コンパイルログの活用は，日常的な学習行動に基づく実践的な SRL 支援手法として有望であり，今後さらなる高度化と教育実践との統合が期待される．

謝辞

本研究は JSPS 科研費 25K15376 の助成を受けた．

参考文献

[1]中尾教子, 西村広光, 登本洋子. (2023). 大学生の自己調整学習方略の獲得状況とプログラミング学習の成果との関連. 日本教育工学会論文誌 47, 205-208.
[2] 國宗永佳, 仲林清. (2019). プログラミング導入演習に対して学習支援システムと自己調整学習が与える影響の分析. 電子情報通信学会 通信ソサイエティマガジン, 13(2), 100-109.
[3] Zimmerman, B. J. (2002). Becoming a self-regulated learner: An overview. Theory into practice, 41(2), 64-70.
[4] Winne, P. H., & Hadwin, A. F. (1998). Studying as self-regulated learning, Metacognition in educational theory and practice, 277–304
[5] Watanabe, K., Matsuda, Y., Nakamura, Y., Arakawa, Y., & Ishimaru, S. (2025). TrackThinkDashboard: Understanding Student Self-Regulated Learning in Programming Study. International Journal of Activity and Behavior Computing, 2025(1), 1-17.
[6]宮澤芳光.(2024). プログラミング問題とデータ活用問題での操作ログを用いた評価，第 38 回人工知能学会全国大会論文集，1M5-OS-14b-01.
[7] Jadud, M. C. (2006). Methods and tools for exploring novice compilation behaviour. In Proceedings of the second international workshop on Computing education research 73-84.
[8]山元翔, 田和辻可昌, 平嶋宗. (2024). 算数文章題における作問学習の学習者の問題空間に基づくフィードバック生成手法の提案， 人工知能学会研究会資料 先進的学習科学と工学研究会, SIG-ALST-100-17, 89-94.