# メモ機能を備えた PAD およびコールグラフを用いたコード可視化によるプログラム理解支援

# Supporting Program Comprehension through Code Visualization Using PAD and Call Graphs with a Memo Function

長井 陽希[*1], 小島 篤博[*2]
Haruki NAGAI[*1], Atsuhiro KOJIMA[*2]
[*1] 大阪公立大学現代システム科学域
[*1]College of Sustainable System Science, Osaka Metropolitan University
[*2] 大阪公立大学大学院情報学研究科
[*2]Graduate School of Informatics, Osaka Metropolitan University
Email: sm22348g@st.omu.ac.jp

**あらまし**： 生成系 AI の登場により、プログラム作成は容易になったものの、プログラムの理解は依然として重要な活動である。本研究では、プログラムの構造を可視化することで複雑なコードの理解を支援するシステムを開発した。提案システムでは、プログラム全体の構造把握を表すコールグラフと、メソッド内部の処理構造を表す PAD を組み合わせて提示し、理解内容を外在化するためのメモ機能を備える。本稿では、提案システムの概要と評価実験について述べる。
**キーワード**：プログラミング教育, 可視化, PAD, コールグラフ, 認知負荷

## 1. はじめに

生成系 AI はプログラムの作成を容易にした一方で、生成されたコードを十分に理解しないまま利用することにより、後続の理解や修正に要する負担が増大することが問題となっている。2025 年に公表された調査においても、AI の活用による開発効率の向上と同時に、品質管理や保守に関する課題が顕在化していることが報告されている[1]。このような背景から、学習者や専門家にとっての中心的な問いは、「コードをゼロから書けるか」ではなく、「生成されたコードを評価し、必要に応じて修正を加え、既存のコードに組み込むことができるほど十分に理解しているか」という点に移っているとされている[2]。このように、以前から重要とされていたプログラム理解の能力は AI の登場によりその重要度をさらに高めていると言える。

プログラムの理解に関する先行研究では、主に初学者に焦点を当て、プログラムの振る舞いの理解を支援する手法が数多く提案されてきた[3]。しかしながら、単に実行の流れを追うだけでなく、処理の意味を自ら整理・解釈する能力が重要であると考える。

以上の背景を踏まえ、本研究では、コードの可視化によりプログラムの構造を整理することで、一定のプログラミング経験を有する学習者が複雑なコードを理解する支援を行う。

## 2. システムの概要

本研究で作成したシステムは、主にプログラムの構造および実行の流れを可視化し、学習者のプログラムの理解を支援するシステムである。加えて、プログラムの理解時に生じる高い認知負荷を軽減するためのメモ機能と、システムを使用した際のプログラムの理解過程を備えている。

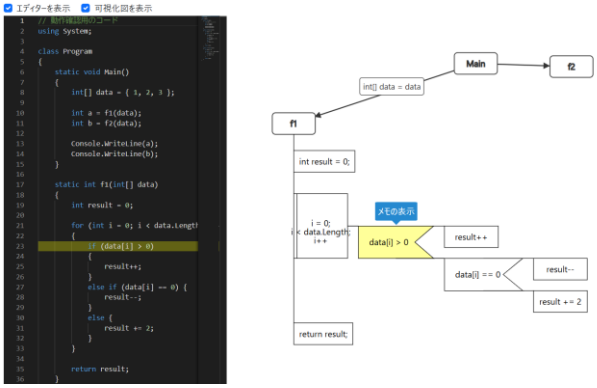


図 1 システムの画面構成

図 1 は本システムの画面構成を示す。画面左にコードを表示し、右側にコールグラフおよび PAD による可視化を提示する。なお、コードは編集可能であり、変更内容は画面右側に提示される可視化図に自動で反映される。

### 2.1 システムの構成

本システムは、ASP.NET Core を使用した Web アプリケーションとして開発した。これにより、IDE などの環境に依存することなく、システムを利用することができる。またコード解析処理と可視化描画処理の役割を分離した構成とすることで、対象言語の変更や拡張を視野に入れた設計としている。

### 2.2 コード可視化機能

本システムでは、コードを可視化する手法として PAD とコールグラフを採用した。PAD は条件分岐や反復などの制御構造を可視化することができ、各メソッド内部の処理構造を理解するのに適している。処理を可視化する手法として一般的に用いられるフ

ローチャートは、反復や条件分岐が増えるにつれて図が複雑化しやすく、処理構造の把握が困難になる場合がある。これに対し、PAD は制御構造を階層的に表現できるため、複雑な処理の場合でもプログラムの構造を視覚的に把握しやすい。一方、コールグラフは、メソッド間の呼び出し関係を可視化することで、プログラム全体の構造や依存関係を俯瞰的に把握することを可能にする。

本研究では、これら 2 つの手法を組み合わせることで、プログラムの全体構造と局所的な処理構造の両方を段階的に理解できると考える。しかしながら、呼び出し関係のみを示すコールグラフでは、引数に対してどの値が受け渡されているかといったデータの流れを十分に把握することが難しい。そこで、引数の受け渡しをコールグラフのエッジ上に表示することで、メソッド間の関係をより具体的に把握できるようにした。

図 1 に示すように、本システムではコールグラフ上のメソッドを選択することで、対応するメソッド内部の処理構造が PAD として表示される。

### 2.3 メモ機能

プログラムを理解する際には、変数の値や処理の流れを一時的に保持しながらコードを逐次的に解釈する必要があり、ワーキングメモリに高い負荷がかかる活動である[4]。また PAD とコールグラフでは、変数の意味や状態の保持は学習者の内部記憶に依存する側面がある。そのため、本システムではノードを選択し、対応するメモを記述して関連付けられる機能を実装した。コールグラフのノードはメソッドを、PAD のノードは制御構造に基づく処理のまとまりを示しており、いずれも学習者がプログラムの理解過程で注目する対象である。これらの対象に対してメモを紐づけることで、どの部分について考えた内容であるかを保持することなく、思考内容を外部にゆだねることが可能となるため、認知負荷の軽減が期待できる。

## 3. 評価実験

### 3.1 実験方法

本システムの有用性およびプログラム理解過程に与える影響を調べるために、プログラミング経験を有する 8 名の学生を対象として評価実験を行った。

実験では、理解のしにくさを示す指標である認知的複雑度や変数の数などを揃えることで理解の難易度が同程度となるように設計したプログラム理解課題 2 題を用いた。課題はコードのみを提示した場合と、プログラムコードを提示せず可視化システムのみを用いた場合の 2 条件で実施した。変数名から意味を推測されないようにするなど他の要因を統制した上で、構造の整理がプログラム理解過程に与える影響のみに焦点を当てた。課題終了後、事後アンケートを実施し、システムの定性的な評価を行った。各課題は 3 問の設問（Q1～Q3）から構成される。

表 1 各設問の正答率（%）

| 設問 | コード | 可視化システム |
|---|---|---|
| Q1 | 75 | 87.5 |
| Q2 | 62.5 | 50 |
| Q3 | 64.3 | 76.3 |

Q1 は複雑なコードのトレース課題、Q2 は比較的シンプルな構造のトレース課題である。Q3 は Q1 のコードの記述説明を求める設問である。

### 3.2 実験結果

表 1 に、コードおよび可視化システムにおける各設問の正答率を示す。構造が複雑なコードを対象とした Q1 および Q3 では可視化システムを用いた場合の正答率が高く、一方で構造が比較的単純なコードを対象とした Q2 ではコード提示の正答率が高い結果となった。このことから、構造が複雑なコードを理解する際には本システムが有効である可能性が示唆された。また、Q1 の正答率が Q2 を上回った点については、Q3 において Q1 の処理内容の説明を求めたことにより、Q1 に対する理解が深化した可能性がある。この傾向は自己説明が理解を促進するという先行研究[5]の知見とも整合的である。

## 4. まとめ

本研究では、PAD とコールグラフを用いてコードの構造を整理する可視化を行うことで、複雑なコードの理解を支援するシステムを開発した。さらにメモ機能を統合することで、変数の意味や処理の意図を外在化し、プログラム理解時の認知負荷の軽減を図った。実験の結果、複雑なコードを理解する際に、本システムが理解の支援に有効である可能性が示唆された。今後の課題として、評価実験の結果を分析し、本システムがプログラム理解に与える影響を明らかにするとともに、得られた知見から可視化手法および支援機能の改良を行うことが挙げられる。

## 参考文献

(1) Cortex, “Engineering in the Age of AI: 2026 Benchmark Report” (2025)
(2) Yael Erez & Orit Hazzan, “Program Comprehension as a Central Skill in CS Education in the Era of Generative AI”, https://cacm.acm.org/blogcacm/program-comprehension-as-a-central-skill-in-cs-education-in-the-era-of-generative-ai/, （2026 年 1 月 13 日参照）
(3) 大城, 永井: “初心者から上級者までを対象とした視覚化機能を持つプログラミング学習支援システム”, 東京情報大学研究論集, 22 巻 1 号 (2018)
(4) Siegmund, J. *et al.*, “Understanding understanding source code with functional magnetic resonance imaging”, Proc. of the 36th Int. Conf. on Software Engineering. pp.378-389 (2014)
(5) Rus, V. *et al.*, “Prompting for Free Self-Explanations Promotes Better Code Comprehension”, CEUR Workshop Proc., 3051 Retrieved from https://digitalcommons.memphis.edu/facpubs/3090 (2021)