

アクティブ変数解析による プログラミングスタイル学習支援システムの開発

藤井 亮平[†] 山本 耕大[†] 中村 勝一[‡] 森本 康彦[†] 横山 節雄[†] 宮寺 庸造[†]

東京学芸大学[†] 福島大学[‡]

1. はじめに

プログラムソースコードの可読性や拡張性を考慮するプログラミングスタイルは、ソフトウェアの品質を左右する重要な要素である。しかし、現在のプログラミング教育では、文法・アルゴリズムの内容を取り扱うことが多く、プログラミングスタイルの内容が取り扱われることは少ない。さらに、プログラミング初学者は、プログラミングスタイルよりもプログラムの実現に固執している傾向があり、プログラミングスタイルまで意識が及ばない。

プログラミングスタイルに関する研究として、モジュール分割候補箇所を提示する研究^[1]や、プログラミングスタイルの推奨事例を用いたパターンマッチングによるスタイル診断システム^[2]がある。プログラミングスタイルは、字下げやコメント文など様々であるが、^[1]はモジュール分割のみに特化していることや、^[2]は特定の項目に絞っていることから、複数の視点から支援されていない(問題点①)。また、^{[1][2]}はプログラミングスタイルの解析結果のみを学習者に提示しているため、学習者はどの程度プログラミングスタイルを確立できているか把握することが困難である。そのため、^{[1][2]}のシステムはプログラミングスタイルを学習する環境として十分でない(問題点②)。

そこで本研究では、上記問題点の解決を目的とし、プログラミング初学者を対象とした、プログラミングスタイルの学習を支援するシステムを開発する。具体的には、問題点①を解決するために、多く存在しているプログラミングスタイルの中から複数の項目を支援していく。また、学習支援は学習者そのものをマネジメントする必要があることから、問題点②の解決のために学習者をマネジメントするメカニズムを導入していく。

2. プログラミングスタイル

問題点①を解決するために、プログラミングスタイルの項目について検討する。先行研究や自身の経験を踏まえた研究室内の議論の結果、プログラミング初学者に必要と思われるプロ

プログラミングスタイル項目とその内容を表 1 に示す。

表 1 プログラミングスタイル項目

項目	内容
(i) モジュール 分割	モジュール分割を行うことはデバッグや保守性の観点から重要である。また、同じ意味を持つかたまりにまとめることで論理的な構成になる。本研究の“モジュール”とは、何らかの機能を持ち、問題解決のあるフェーズと合致しているようなプログラム命令文の集合をさす ^[1] 。
(ii) コメント	コメント文を記述することは、ソースコードの機能拡張や保守性、可読性の観点から重要である。コメントは、処理のまとまりや変数などに付記する。
(iii) マジック ナンバー	マジックナンバーに意味付けをすることは、(a)数値の意味がわかりづらい、(b)その数値を変更する場合複数箇所を変更しなければならない、の 2 点を解決するために重要である。マジックナンバーとなる数値は最初に定義する。
(iv) 字下げ	字下げを行うことは、ソースコードの可読性を向上させるために重要であり、プログラムの外的要素を左右する。字下げは、処理ごとにとまりのある部分をわかりやすくするためにスペースやタブを挿入する。

3. プログラミングスタイル学習

問題点②を解決するために、学習者をマネジメントする手法について検討する。^[3]では、学習者を支援する方法として、“足場づくり”を挙げている。そのため、本研究では学習者のプログラミングスタイルを確立するためにファシリテーションを行い、学習者の学習を促進させるメカニズムを導入する。それにより、学習者が次のステップに進むために、何を意識しなければならないのか、何を修正すべきなのかを考えるようになるため、学習者をマネジメントすることが可能となる。以上を踏まえて開発するシステムでは、過去の学習を考慮したメッセージを提示することでファシリテーションを行い、プログラミングスタイルを確立できる環境を学習者に提供する。

4. システム概要

上記の議論を踏まえたシステムの開発を行う。サーバは Solaris10 を使い、実装には PHP と C を使用した。またインタフェースに JavaApplet を使用した。

Development of a Programming Style Learning System
Based on an Active Variable Analysis

[†] Ryohei Fujii, Kodai Yamamoto, Yasuhiko Morimoto,
Setsuo Yokoyama, Yozou Miyadera

– Tokyo Gakugei University

[‡] Shoichi Nakamura – Fukushima University

4.1. システム利用モデル

実装したシステムの利用対象はプログラミング初学者である。学習者は自身で記述した C 言語のソースコードをシステムへ送ると、システムがプログラミングスタイルの診断結果と 3 章で述べたスタイルの学習を促すためのメッセージを学習者に提示する(図 1)。その繰り返しをすることで、学習者は自らプログラミングスタイルを意識したソースコードの作成が可能となる。

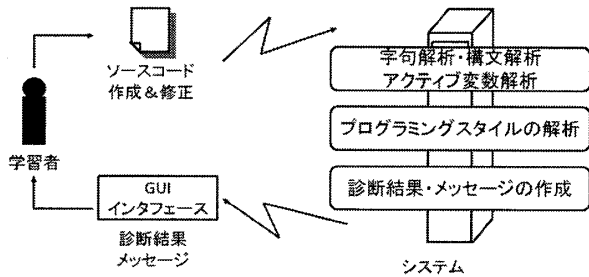


図 1 システムの流れ

4.2. プログラミングスタイルの解析

プログラミングスタイル項目(i)から(iv)について解析するために、[4]のアクティブ変数解析を用いる。アクティブ変数解析はソースコードの構文解析木を走査しながら解析を行っている。その構文解析情報をもとに、ソースコードに使用されている変数に着目し、その変数の状態を解析する。以上から、アクティブ変数解析を用いることで、処理ごとのモジュールを取得できるため、より詳細なプログラミングスタイルの解析が期待できる。以下、プログラミングスタイルの 4 項目の実装について述べる。

(i). モジュール分割

モジュール分割に関しては、文献[1]で使用しているアクティブ変数解析を用いる。具体的には、「アクティブな変数の個数が減少し、再び増加した箇所は何らかの処理が終了した可能性が高い。」ことを利用し、モジュール分割候補点を検出する。

(ii). コメント

モジュール分割解析を用いることで、構文解析情報やモジュール単位を取得する。以上の情報から、コメントが記述されているか、ソースコードの診断を行う。

(iii).マジックナンバー

数値が使用されている箇所を構文解析情報から取得する。また、数値は同じでも関数ごとに数値の意味が異なる可能性が考えられるため、関数の単位を考慮する。以上を踏まえて、マジックナンバーが存在しているかを診断する。

(iv). 字下げ

構文解析情報を用いて、字下げすべき箇所を特定する。そして、実際のソースコードを解析し、スペースとタブの数をカウントすることで、字下げがなされているかを診断する。

4.3. プログラミングスタイル学習機能

3 章で述べた学習者をマネジメントするメカニ

ズムについて述べる。ファシリテーションを行うために、学習者のプログラミングスタイル確立状況を定量化したものとシステムの使用回数を用いる。まず、スタイル確立状況を定量化するために、2 章の各項目の診断結果を割合として算出する。そして、スタイル確立状況について、過去の確立状況のデータから今回のデータまでどのように遷移しているかを解析する。次に、システムの使用回数で学習者がスタイルの修正に取り組んでいるかを考慮する。この 2 項目の数値から、あらかじめ登録してあるメッセージを学習者に提示する。例として、スタイルの確立がなされていないと診断され続けている場合には、診断結果の見直しを促すメッセージを提示する。

4.4. インタフェース

実装した画面を図 2 に示す。診断したソースコード、アクティブ変数の有効範囲、4 つの項目の診断結果を表示することで問題点①が解決できる。また、ファシリテーション表示を行い、学習者の学習を促進することで問題点②が解決できる。以上から学習者はプログラミングスタイルの確立が可能となる。

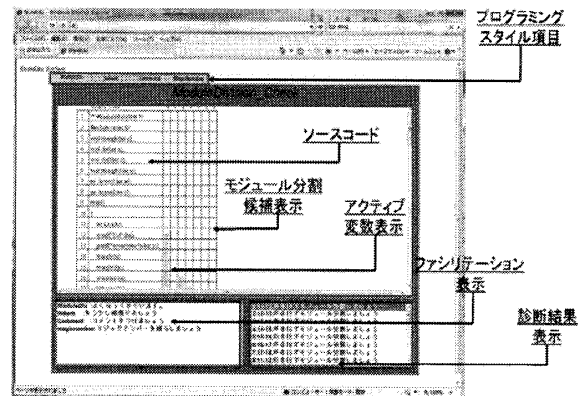


図 2 システム画面

5. おわりに

本稿では、学習者に複数のプログラミングスタイル項目について診断を行い、学習者の過去の学習を考慮し学習者をマネジメント可能な、プログラミングスタイル学習支援システムの開発を行った。

今後は、本システムを教育現場で運用し、システムの評価を行う。

参考文献

- [1] 鈴木恵介, 横山節雄, 宮寺庸造, “プログラミング教育のためのモジュール自動分割システムの開発と評価,” 電子情報通信学会研究報告, Vol.103, No.697, pp.143-148, 2004
- [2] 関本理佳, 海尻賢二, “プログラミングスタイルの診断システムの構築,” 教育システム情報学会誌, Vol.17, No.1, pp.21-29, 2000
- [3] 池田玲子, 笹岡洋子, 『ピア・ラーニング入門』, 株式会社ひつじ書房, 2007
- [4] 宮寺庸造, “プログラム言語教育環境のためのデータフロー解析とその実現,” 教育システム情報学会誌, Vol.13, No.3, pp.151-162, 1996