N-007

コード依存関係におけるプログラミング作法のチュータリング

Tutoring How to Build-up Program Components Using Code Dependency

上野 敦子† 島川 博光†
Atsuko Ueno Hiromitsu Shimakawa

1. はじめに

現在のプログラミング教育における問題点として，学習者は作成中のプログラムが正しいかどうかの判断ができないにもかかわらず，プログラミングの過程を常に他人に確認してもらいながら学習を進めるのが困難であることが挙げられる．また，プログラミングの基礎知識を習得している学習者が，他の新しいプログラミング言語を学ぼうとしても，その言語特有の作法を知らないために思うようにプログラムを記述できない場合がある．

本論文ではこのような問題点を改善するために，Java の Swing を例にしてプログラミング作法の学習を支援する手法を提案する．今回は特にプログラムの記述順序に着目し，それに基づいて，プログラムが正しいかどうかを判断するとともに，適切な箇所で指導を行う．これによって，正しい記述ができるように学習者を導くことができる。さらにこの手法を実現するチュータリング･システムを実装する．本システムではアニメーションを用いてプログラミング作法のチュータリングを行う．

2. Java プログラミング教育

2.1 プログラミング教育の現状

プログラムが未完成で，まだコンパイルすることができない状態だと，それまでに記述したプログラムが正しいかどうかの判断は学習者自身にはできない．初心者にとっては自らのプログラミング作業が正しいかどうかが判断できないまま作業を続けなければならないことは苦痛である．すべての学習者に対して個別に教員が付き添い指導することができればこの問題は解決する．しかし，実際の演習では数十人の学習者を一人もしくは少数の教員が教えるという形式が採られている場合が多いため，プログラミングの過程を常に教員に確認してもらいながら演習を進めることは困難である．

2.2 Java を学習するさいの問題点

C 言語などの Java 以外のプログラミング言語の経験者が Java を学習しようとすると， プログラミングの基礎知識はある程度持っているにもかかわらず学習に戸惑うことがある．これは Java 特有のプログラミング作法がスムーズに理解できないところに原因があると思われる．

Java 特有のプログラミング作法とは，コードを記述するうえで守るべき制約である．具体例を挙げると，Swing でウィンドウを作成する場合は，図 1 に示すように，①フレーム生成，②フレームの表示域の取得，③レイアウト方式の設定，④部品の配置，といった順序でコードを記述する必要がある．ここでのコードとは，ひとかたまりのふるまいを実現する，Java のソースコードの 1 行もしくは複数行を指す．図 1 では①，②，③，④のひとつひとつを実現するための記述がコードである．本研究では特にこのようなコードの記述順序に関するプログラミング作法に着目する．

†立命館大学 理工学部

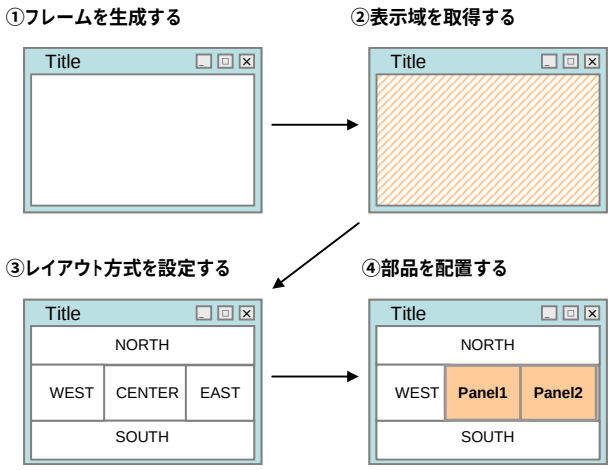

図 1 プログラミング作法の例

2.3 関連研究との比較

Java の学習支援を目的とした研究はこれまでにも行われている．たとえば，文献[1]では，本研究においてプログラミング作法と呼ぶ概念に注目し，文献[2]ではプログラムの振る舞いを視覚化して，教育支援システムを提案している．しかし，これらでは学習者のプログラミング過程を監視し，適切なタイミングで指導することは考えられていない．

3. コード依存関係を用いた学習支援

本論文では，新しいコードが記述されるたびに，プログラミング作法に従って正しく記述できているかの判定を行い，その結果によって奨励や指摘を与えるための手法を提案する．プログラミング作法の判定を行うためにコード依存関係という概念を用いる．これによって未完成のプログラムに対しても適切な指導を行うことができる．

3.1 コード依存関係

コード A を記述するためには，それ以前にコード B を記述しておかなければならないとき，コード A はコード B に依存するといい，このような関係をコード依存関係という．コード依存関係は Java のプログラミング作法を学習するうえで重要な概念である．図 2 は簡単なウィンドウを作成するプログラムから，コード依存関係を抽出し図示したものである．これを例に挙げて説明すると，コード②の「表示域の取得」を行うためにはコード①の「フレーム生成」を事前に行う必要がある．この関係を，「②は①に直接依存している」，「②の前提条件は①である」，あるいは「②→①」と示す．また，⑤のコードが直接依存しているのは③と④だけだが，③は②，②は①に直接依存しているので，結果的に⑤は③，④に直接依存し，①，②に間接的に依存しているといえる．

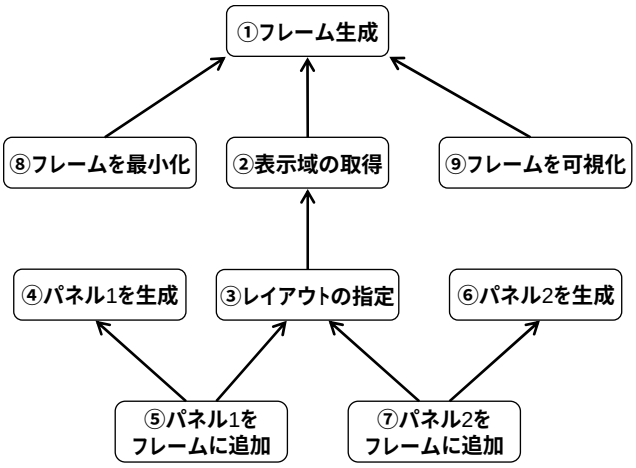

図 2　コード依存関係の例

3.2 依存関係を用いたコードの判定

本研究では，学習者はあらかじめ用意されたコードを 1 つずつ順番に並べることによって学習を進めるといったスタイルを想定している．各々のコードは自身のコード番号と自身が直接依存するコード番号を持っている．たとえば図 2 のコード⑤は前提条件としてコード番号③と④を持つ．学習者があるコードを記述したとき，それが直接依存するコードがすでに記述されているかを，それらのコード番号を用いて調べる．前提条件となるコードがその時点ですべて記述されていれば，そのコードは正しく記述されているとみなす．

3.3 奨励と指摘

依存関係を用いてプログラムの記述の正しさを判定し，その結果によって奨励または指摘を与える．奨励とは，学習者の意欲を高めるために記述が正しいことを褒めることである．指摘とは，記述が誤っていることを伝え，さらに正しい記述ができるように学習者を導くことである．これらによって，自分はどこが理解できているのかを学習者自身に明確に認識させることができる．

プログラミング作法を理解するうえで重要なコードが正しく記述されたときは，奨励が与えられる．一方，依存関係にあるコードが 1 つでも記述されていなければ指摘が与えられる．

4. チュータリング・システム

3 章で提案した手法を実現するチュータリング･システムを，アニメーションツール[3]を使用して実装する．アニメーションを用いることで，直感的なイメージを用いた説明が可能となり，学習者が理解しやすい学習ツールを提供できる．システム中には，学習の進行役である電子的なチュータという存在があり，例題の説明や奨励･指摘はすべてチュータによって与えられる．プログラミング作法の学習は，まずチュータが説明し，次に学習者にテストを受けさせるという順序で進める．

4.1 プログラミング作法の説明

プログラミング作法の説明の画面は，例題のプログラムの表示域とアニメーション表示域からなる．チュータによる例題の各コードの説明と，それに対応するプログラムのふるまいがアニメーションで示される．

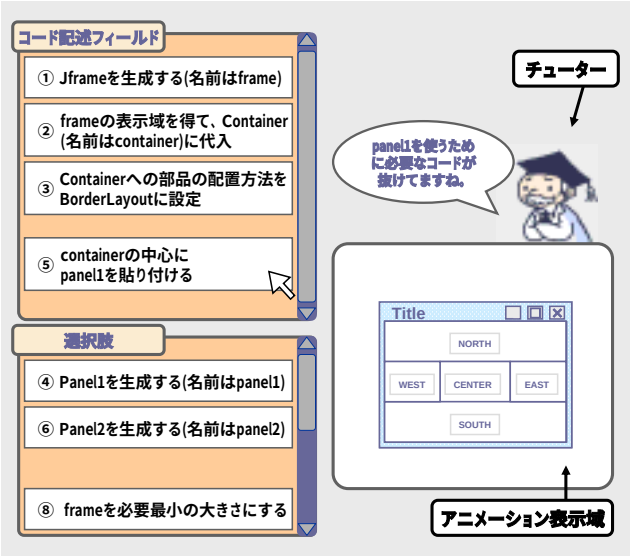

図 3　プログラミング作法のテスト画面

4.2 プログラミング作法のテスト

図 3 に示すように，テスト画面は左側にコード記述フィールドと選択肢欄，右側にアニメーション表示域を持つ．学習者は，あらかじめ選択肢欄に用意されたコードを 1 つずつ選び，コード記述フィールドに順番に配置することによってプログラミング作法のテストを行う．学習者があるコードを配置したとき，それが正しくない記述ならチュータから指摘を与えられ，さらにどのような内容のコードが足りないかヒントを与えられる．正しく記述されていれば，配置したコードに対応するふるまいをアニメーションで示す．作法を守って指定したコードが重要なものであったとき，学習者に奨励が与えられる．

5. おわりに

本論文ではコード依存関係を用いてプログラミング作法の学習を支援する手法を提案した．さらに，その手法を実現するチュータリング・システムを実装した．

本システムでは，現在のところ教員が例題やテストを作成することとしているが，例題やテストの作成を自動化または半自動化する手法を実現することによって教員の負担を削減することが今後の課題である．

参考文献

[1] 高岡詠子，斉藤将志，谷口景介，”Java プログラミング教育支援システム Jmaster の開発と教育効果”第 2 回情報科学技術フォーラム(FIT2003)講演論文集，pp.381-382(2003)

[2] 高津陽平，田部井俊彦，伊藤小琴，前川仁孝，伊與田光宏，”視覚化による Java 学習支援システム” 第 2 回情報科学技術フォーラム(FIT2003)講演論文集，pp.423-424(2003)

[3] Macromedia Flash MX 2004 http://www.macromedia.com/jp/software/flash/