

プログラミング教育とプログラム理解のモデル化

4U-3

加藤木和夫

白石久敬

日立プロセスコンピュータエンジニアリング(株)

1.はじめに

プログラミング教育はアーキテクチャ教育と共にプログラマ教育の基本となるものである。プログラミング教育の実際はアルゴリズムの教育を兼ねプログラム言語の教育のなかで行われる。

企業内教育ではこのプログラミング教育で、プログラムを本質的に理解できる力を早期に育成しなければならない。一方、資質あるベテランを新人のプログラミング教育に充てる余裕は一般的にはないのが実情である。

そこで筆者らは(1)プログラミングの本質を理解する力の早期育成、(2)最小のインストラクタによる教育、を念頭にプログラミングの「課題」を考案した。

また、「課題」の学習課程を観察することでプログラム理解のモデル化を試みた。

2.プログラムの理解とは何か

プログラムの理解において重要なことは、個々の文の意味がわかるだけでなく、その背後にあるプログラム製作者の「意図」が正確に分かることである。「意図」とはプログラム製作者が実世界を切り取り、計算機上に写像することで「何」を実現しようとしているかである。つまり、他者の「意図」を、表現されたもの(プログラム、設計書)から読み取り、自分の頭の中に再構成することが理解そのものである。

3.課題と運用法

課題は次のように設定した。

Programming training and Modeling of program understanding

Kazuo KATOUGI, Hisayosi SIRAI

Hitachi Process Computer Engineering, Inc.

<課題>

(1) BASICで記述されたCASLアセンブラ・プロセッサとCOMMETシミュレータをC言語で書き直す。

ただし、PADに戻してから書き直すこと。

(2) CASL文法を修正しプロセッサを改造する

(3) CASLの教育を第3者に行う。

課題遂行時に習得すべき項目は、プログラミング関連、アーキテクチャ関連、トランスレータ技術とする。また、単なるコンバージョンではなく意味を深く理解させるために、PADに書き直すことを組み込んだ。被験者は文系の新人2名(共にコンピュータ未経験者)、指導員は1名。

指導員は教育の重点を個々の知識ではなく(知識は独習前提)、プログラムの「意図」の解説におく。

理解の尺度は、モジュール名の付け方、テストデータの作り方、改造ができるか、問題点を指導員に伝えられるか等で計った。また、指導員は理解の度合いを見ながら被験者のレベルに合わせてプログラムの全体と部分の解説を行った。

4.実験過程と終了時のインタビュー

(1) 理解の状態の進み方

理解の状態の進み方の軌跡を追うと図1のようになる。理解はプログラムの部分から全体へ、構造から意味へと進む。最初1部分の構造が分かり(繰り返し、判定など)、つぎに全体の構造的な流れが分かる。

意味には表面的な意味(データの流れなど)と深い意味(製作者の「意図」)があるが、後者はモジュールの名前付けの時点から分かり始めた。モジュールの意味がつかめると全体の意味が掴め始める。その後、全体のなかでの部分の意味が分かり始める。

(2) 課題の終了後、「何が理解を促進したか」を被験者にインタビューしたところ次の回答を得た。()内は筆者らの解釈である。

- ・BASICプログラムの実行例 (実世界の提示)
- ・2人での相談 (同じレベルでの相互作用)
- ・実際のプログラム作成とバグ発生時のトレース (自分なりの表現と修正)
- ・適切なモジュール名を教えられたとき (抽象化による意味の理解)
- ・不明点を指導員に説明するとき (自己表現)
- ・第3者へのCASLの教育 (他者への伝達)
- ・意味が分かるときは突然分かる (理解には醸成期間が必要)

5. プログラム理解の考察

実験過程とインタビューからプログラム理解に有効な項目は次の様に考えられる。

- ・既存プログラムのソースコードおよび実行例。
- ・他者への伝達 (プログラムの記述、教える、説明するなど) は実用的な真の理解には不可欠。
- ・問題意識が重要。今何が問題であるのかは指導員からの誘導が有効。
- ・理解力をつける方法としては抽象化能力の訓練が重要。国語の長文を読解し、題目をつけるなどが有効か。
- ・PADは構造的な理解に有効。意味理解にはソースコードの方が有効。

	Basic	PAD	C
文	● ○		◎
ブロック	● ○		◎
モジュール	● ○	●	◎
サブシステム		●	◎
全体			◎

● 構造 ○ 浅い意味 ◎ 深い意味

図1 理解状態の進み方

6. プログラム理解のモデル化

理解はプログラムを読む、BASIC、C言語、PADを学習する等による情報の吸収と、実際にプログラムを作成することによる知識の形成を繰り返しながら進む。これは情報が吸収 (入力) しただけでは知識とはならず、他者に伝達する、つまり各個人の持つ表現系 (日本語、設計図式、プログラム言語など) を用いて出力するときに理解が促進され、情報が自己の知識となるためと考えられる。プログラムの理解に表現が重要な役割を果たしている。

一方、理解と知識は深い関係にある。理解するには過去の知識の集積 (メンタルモデル) との照合、類推が必要であるし、知識は理解という段階を経ないと真の知識にはならない。

図2に今回の実験から得られたものをもとにまとめた (モデルの試み)。

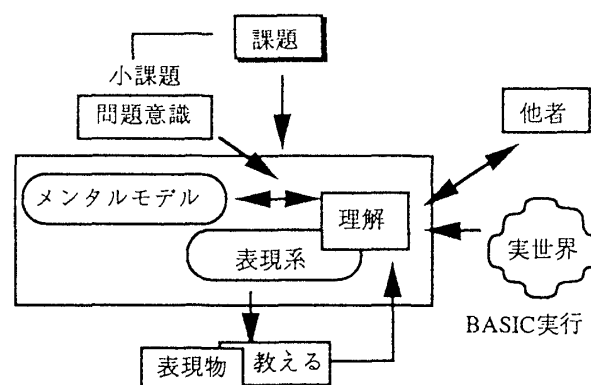


図2 プログラム理解のモデル

7. おわりに

プログラミング、アーキテクチャの理解共にOJTのみの新人より理解が進んだ。プログラム理解は細かい事項の教育より、適度な大きさのプログラムの作成、改造、他者への伝達等表現力を含む教育の方が有効と考える。

モデル化についてはさらに考察を進めたい。

参考文献

[1] 佐伯 胖編

"理解とは何か" 東京大学出版会, 1985