社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2003－CE－72　(11)
2003／12／15

「目的の表現」に注目した オブジェクト指向プログラミング教育とその評価

松澤芳昭[†]　青山希[†]　杉浦学[†]
川村昌弘[†]　大岩元[††]

本稿では，オブジェクト指向プログラミング教育の一手法として,「目的の表現」に注目したアプローチを紹介する．我々の提案する教育手法では，構造化プログラミングの段階でプログラムの目的に注目し，それらをどのように捉え，表現するかを教育する．オブジェクト指向プログラミングの教育では，目的記述中の動詞と名詞を手がかりに，クラスやインターフェイスを設計するように教育する．カリキュラムと評価方法を考案し，若手技術者に対して実験した．結果，提案手法は受講者がオブジェクト指向を受け入れやすく，導入する意義も伝わりやすいという成果を得た．

An Instruction Method for Object Oriented Programming focused on "Expressing Purpose Structure" and Its Evaluation

Yoshiaki Matsuzawa,[†] Nozomu Aoyama,[†] Manabu Sugiura,[†]
Masahiro Kawamura[†] and Hajime Ohiwa[††]

This paper proposes "Expressing Purpose Structure" approach as an instruction method for object-oriented programming. Our method focuses on clarifing and describing the purpose structure of the program which is constructed from sub-purposes at the early stage of programming instruction. Class structure and its interfaces are easily developed by examining objects and verbs in the developed purpose structure. The curricula based on this method have been tried to young engineers of Japanese IT industry and appreciated by them. This method is easy for learners to understand the merit of object-oriented programming.

1. はじめに

近年，サーバサイド Java の普及などを背景に，オブジェクト指向技術を使いこなしてソフトウエアを設計，開発できる技術者のニーズが高まっている．今回我々が試行したオブジェクト指向プログラミングの実験講座においても，半月あまりで定員の 2 倍以上の応募があった．

最近では，オブジェクト指向の概念の知識や，フレームワークの利用法などを解説する書籍が多く出回っている．それらの書籍を読んで独学で学習することにより，オブジェクト指向のメリットを享受して，便利なフレームワークを利用したプログラムならば書けるようになる．しかし，知識だけでオブジェクト指向を利用できるのはそこまでである．オブジェクト指向プログラミング能力はガニェのいう言語情報ではなく，知的技能，認知的方略に分類される[5)2)3)]．そうした領域の能力を養わなければ，オブジェクト指向のメリットを活かしたプログラムを自ら設計するような，オブジェクト指向の運用能力を身に着けることはできない．

今回行った実験講座の受講者募集に当たり,「構造化プログラミングなどの経験は有するが，オブジェクト指向のメリットを活かしたプログラムを書くことができない人」という対象を設定した．集まった受講者は，実際に業務をしている若手技術者である．オブジェクト指向の知識が皆無の者は少数で，サーバサイド Java に関わる業務をしている者も少なからずいた．しかし，その大半は前述のように業務で

† 慶應義塾大学 政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 慶應義塾大学 環境情報学部
Faculty of Environmental Information, Keio University

必要な要素知識を覚えて使用しているといった状況であった．彼らの多くは，関連書籍などを利用して独学で学習しているが，なかなかうまくオブジェクト指向を利用できないという問題意識を持っていた．

オブジェクト指向設計では，システムを理解しやすいように，プログラムをクラスという単位に分解する能力が求められる．しかし，プログラムを分割して構造を整理することは，オブジェクト指向でなくても容易な作業ではない．実務者にはプログラムの理解のしやすさよりも，短期間で機能を実現することが要求されるので，たとえ小さなプログラムであってもプログラムの分割作業についてじっくりと考える時間がない．そのような実務者は，クラス設計の前提技能としてのプログラムの分割能力が低いと考えられるので，独学でクラス設計を学ぶのは難しい．

技術者が，プログラムがどのような処理を行うか，といったソフトウエア実現の「手段」に偏って教育されるのも問題である．オブジェクト指向如何に問わず，設計作業では事象を系統的に捉え，プログラムが果たす「目的」をトップダウンに詳細化していく能力が求められる．手段に偏った教育ではこの視点を獲得することができない．

このことは，試験講座の事前試験として，HCP チャート[7)4)] を用いてプログラムの目的とその構造を表現するという課題 (4.1 節を参照) を行ってもらったところ，ほとんどの受講者がこれを達成できなかったことでも裏づけされる．

それゆえ，我々は，構造化プログラミングの段階から，他人に理解しやすいように，目的を明確に表現することを意識して，可読性が高く，構造も整理されたプログラムを書くことができるようにする教育をすることが重要だと考える．そして，その延長線上にある，さらに強力に目的を表現できる手法としてオブジェクト指向を導入する．

このアプローチで教育するためのカリキュラムを作成し，実験したところ，プログラムの構造を整理して目的を明確に表現するとプログラムの可読性が向上する，という視点を獲得してからオブジェクト指向へ進むので，受講者がオブジェクト指向を受け入れやすく，導入する意義も伝わりやすいという成果を得た．

本稿では，オブジェクト指向教育の一手法として，「目的の表現」に注目したアプローチを紹介する．まず，2 節で，プログラムにおいて目的をどのように表現するか，その方法と，オブジェクトのプログラムへどのように発展するかを具体例を用いて述べる．3 節では，本稿で提案するアプローチの教育方法についてを述べ，4 節では実験授業のカリキュラムと実施体制を示す．そして 5 節で，受講者の成果物とアンケートを用いて考察を行う．

2. プログラム上で目的を表現する方法

本節では，プログラムにおいて目的をどのように明確に表現するかということ，そしてそれをどのようにオブジェクト指向のプログラムに発展させるかを述べる．

2.1 ブロックによる表現

我々は，プログラミング教育の初歩の段階で目的を表現する方法として，プログラムを「ブロック」というまとまりに分解することと，分解されたブロックに適切なコメントを記述するという方法を用いる．

図 1 に示されたプログラムを使って具体的に説明する．このプログラムは，タートルグラフィックスを利用して，亀に家を書かせるプログラムの Java 版である．実行すると，屋根と本体からなる家が描画される．このプログラムの目的は，家を書くことである．

家は，屋根と本体から構成される．これを表現するためにブロックを作成する．そのために，プログラム中の「屋根を書く」という部分と,「本体を書く」という部分の間に空行を 1 つ挿入する．こうして作られたブロックは，いわゆる一般的なプログラミング言語のブロックとは異なり，処理の構造ではなく，論理的な構造を表現する．

すべてのブロックには，そのブロックがどのような目的を持つのかを記述した,「ブロックコメント」を自然言語で記述する．

ここで，適切なブロックコメントをつけることは，実はプログラム初学者のみならず，プログラム上級者でも難しい作業である．一般的なガイドラインとして，コメントには目的を書くべきであるというものがある[1)]．しかし，ここでブロック中の処理を抽象化する視点で目的を考えた場合，図 2 のように，三角形を書く，四角形を書くという目的を記述することもできる．その場合，ブロックの構造は同様になるが，プログラムの様子はまったく異なってみえる．

「屋根を書く」,「三角形を書く」というコメント

```
/**
 * 家を書くプログラム
 */
public class House extends Turtle{

  void start(){
    //屋根を書く
    rt(30);
    fd(50);
    rt(120);
    fd(50);
    rt(120);
    fd(50);

    //本体を書く
    lt(90);
    fd(50);
    lt(90);
    fd(50);
    lt(90);
    fd(50);
    lt(90);
    fd(50);
  }

}
```

図 1　ブロックによる表現

```
/**
 * 家を書くプログラム
 */
public class House extends Turtle{

  void start(){
    //三角形を書く
    rt(30);
    fd(50);
    rt(120);
    fd(50);
    rt(120);
    fd(50);

    //四角形を書く
    lt(90);
    fd(50);
    lt(90);
    fd(50);
    lt(90);
    fd(50);
    lt(90);
    fd(50);
  }

}
```

図 2　ブロックコメントを処理から考えた場合

を記述することは，どちらもプログラムの可読性の寄与のために必要な情報である．目的の粒度が調整されて，これらがうまく表現される必要がある．目的の粒度を整理して，階層構造としてまとめると図 3 のようになる．空行によるブロックの表現では，入れ子になったブロックを表現するのが難しいので，これをうまく表現するために，手続きを利用する．

2.2　手続きによる表現

目的を明確に表現するために，構造化プログラミング言語では手続きを利用できる．

家を書くプログラムの例では，三角形や，四角形を書くプログラムを抽象化して，多角形を書くといった手続きを作ることができる (図 4)．これにより，前節でのブロックの入れ子の問題が解決でき，可読性の向上のために必要な情報を失わないまま，目的がより簡潔に表現される．

さらに，図 5 に示すように，屋根を書く，本体を書くといった粒度の目的に関して手続きを作ることも大切である．家を書くという視点から考えた場合，屋根の形を変更したり，本体の形を変更したりすることも考えられるからである．

```
- 家を書く
    |
    | - 屋根を書く
    |     |
    |     | - 三角形を書く
    |
    | - 本体を書く
          |
          | - 四角形を書く
```

図 3　家を書くプログラムの構造

しかし，この規模のプログラムで，屋根を書く，本体を書くという手続きを作るのは，いささか冗長である．これを実際に手続きにするかどうかは，プログラムの大きさによって決める必要がある．手続きにしない場合も，これをブロックとしてプログラムを分離されているので，十分目的は明確に表現されている．そして，このことは，ブロックとメソッドはプログラムを分解する単位として同義になることを示している．異なる点は，プログラミング言語の機能を利用する（つまり手続きを作る）場合，ブロックコメントがそのまま手続きの名前になるので，

```
/**
 * 家を書くプログラム
 */
public class House extends Turtle{

  void start(){
    //屋根を書く
    drawPolygon(3);

    //本体を書く
    drawPolygon(4);
  }

  void drawPolygon(int){
    …
  }

}
```

図 4　手続きによる処理の抽象化

コメントが不要になることである．

2.3　オブジェクト指向への発展

オブジェクト指向プログラミング言語では，オブジェクトという単位に分解することで，例えば図 6 に示すように表現することができる．

ここで，分解される単位や名前はこれまでの表現方法の場合と同様に考えることができる．ここで抽出された屋根 (Roof) や本体 (Body)，多角形 (Polygon) といったクラスや，書く (draw) といったインターフェイスは，初歩の段階で行ったブロックとコメントによる方法においても表現されている．

つまり，その構造を初歩の段階でうまく捉え，表現されていれば，それらの記述に含まれる動詞と名詞に注目することで，比較的容易にオブジェクト指向のプログラムに発展するのである．

最初の段階でのブロックコメントが，オブジェクト指向でのクラス設計に関連しているので，目的を明確に表現するブロックコメントを定めておくことが非常に重要である．特に，屋根を書く，三角形を書くという 2 つの視点，つまりブロック中の処理と，プログラムの大きな目的からの視点で目的を捉え，表現されていることが重要である．しかし，プログラムの動作に集中してしまいがちな初学者や，今回対象とするような，実務を何年か経験している実務者は，処理から考えた目的，つまり手段に近い目的になってしまいがちである．

ここにオブジェクト指向へ発展するかどうかの大

```
/**
 * 家を書くプログラム
 */
public class House extends Turtle{

  void start(){
    //屋根を書く
    drawRoof();

    //本体を書く
    drawBody();
  }

  void drawRoof(){
    drawPolygon(3);
  }

  void drawBody(){
    drawPolygon(4);
  }

  void drawPolygon(int){
    …
  }

}
```

図 5　手続きによる目的の抽象化

```
/**
 * 家を書くプログラム
 */
public class House extends Turtle{

  Roof roof = new Roof(new Polygon(3));
  Body body = new Body(new Polygon(4));

  void start(){
    roof.draw();
    body.draw();
  }

}
```

図 6　オブジェクト指向による表現

きな壁があると思われる．オブジェクト指向では，大きな目的を捉え，表現する能力がより重要になるからである．例えば，オブジェクト指向の基本的概念であるクラスのカプセル化では，内部の手段を隠蔽し，より大きな目的に近い公開インターフェイスを作成することで，プログラムの保守性を高めることができる．従って，大きな目的を捉えることがで

きなければ，一つのクラスの公開インターフェイスでさえも，うまく設計することは難しい．

オブジェクト指向システムの分析や設計は，これを大きな粒度で捉えたものと考えることができる．オブジェクト指向では，大きな目的からの視点で，概念モデルを構築することが良いクラス設計をする上で重要とされている．オブジェクト指向によるフレームワークの設計では，それらの大きな目的ををさらに抽象化していく能力が必要である．

3. 目的の表現アプローチの教育法

本節では，前節までの議論を踏まえて，どのように目的の表現能力を教育するかを述べる．

3.1 HCP チャートの利用

目的の表現アプローチの初めの段階では，プログラミング言語などの手段にとらわれないように，自然言語を用いて表現を教育するべきである．

我々はこの段階での表現方法として，HCP チャート[7)4)] を用いることを提案する．HCP チャートは目的を階層構造として表現するのにすぐれた図法である．そのため，上位の目的からトップダウンに要素を分解し，下位の目的へと分解する段階的詳細化を行いやすい．さらに，階層構造を表現したまま，繰り返しや分岐などの制御構造も表現できる．

HCP チャートでは，データ構造も表現できる．しかし，この段階ではデータは無視して処理のみに注目させたほうが良い．プログラムにおける目的の記述は処理に対して行われるため，処理のみに注目したほうが，より目的の表現に焦点を絞ることができるためである．

3.2 相互レビューによる教育

本稿が提案するアプローチでは，表現に主眼を置いているため，学習者が表現した成果物が，伝える相手にどれだけ伝わったかを評価し，学習者にフィードバックする必要がある．

この目的の達成のために，我々は学習者同士で相互レビューを行わせる教育法を提案する．他人の表現に対するレビューを行うことは，学習者に読み手としての評価能力を養うことになり，自己の表現にフィードバックされる．また，実際に表現が不良であると相手に伝わらない経験をすることは，表現能力を養う学習の動機付けにもなる．

この方法の問題点は，人によって表現の受け取り方がさまざまなので，レビューの方向がばらばらになってしまうことである．これを解決するために，我々は表現不良のカテゴリを作成し，レビューの指針として学習者に示した．これを表 1 に示す．

表 1 目的の表現レビューのカテゴリ

構造	構成
	粒度
	過不足
表現	目的記述
	日本語記述

このカテゴリーは，我々が 30 数名の HCP チャートをレビューして，そのパターンをまとめたものである．これらのカテゴリーは，構造とその表現という大きな視点からまとめられている．これらは，2 節で示したように，プログラムによって目的を表現するときの大きな 2 つの視点でもある．

個別のカテゴリーを説明する．まず，構成不良は，構造に問題があり，可読性を損ねているものや，項目の移動や，分解・再構成が必要なものが分類される．

次に，粒度不良は，粒度がそろっていないので，新しいレベルを追加・削除することにより，粒度を調整すると，よりよい構造になるものが分類される．

過不足不良は，下位レベルの記述不足により、目的の実現方法が明確でないもの，もしくは，下位レベルの記述過剰により，上位レベルの目的の見通しを損なっているものが分類される．

目的記述不良は，目的がかかれるべき場所に，手段が記述されているものが分類される．ただし，目的は上位レベルからみると手段なので，正確には，上下レベルの目的-手段の相対関係が適正でないものである．上位レベルに手段が記述されていた場合，構造の変更を伴う場合があるが，その場合は，構造不良として分類する．

最後に日本語記述不良は，記述が不十分，または不適切なので、記述の明確さを損ねているものが分類される．例えば，説明不足，用語定義不足，てにをは不良などがこれにあたる．

このレビューカテゴリに従って，相互にレビューを行うことによって，お互いに共通の視点を持って表現の研鑽を行うことができるようになる．さらに，このカテゴリーを学習者と指導者が共有することで，指導者の指導が学習者に伝わりやすくなる．

4. 実 験 授 業

本稿が提案する教育手法に基づくカリキュラムを

表 2　達成度レベルと評価の指針

達成度レベル	自然言語	構造化手法	オブジェクト指向
達成目標	自然言語を用いて，プログラムの目的を表現できる．	構造化手法を用いて，プログラムの目的を表現できる．	オブジェクト指向をも用いて，プログラムの目的を表現できる．
評価の指針	構造表現 目的記述	ブロック表現 ブロックコメント記述 手続き表現 手続き名の記述 変数表現 変数名の記述	クラス構造表現 クラス名の記述

表 3　カリキュラム

事前試験	アプリケーションの設計，実装	10 時間
講座	HCP チャートによる設計演習	4 時間
	構造化手法による プログラミング演習	10 時間
	オブジェクト指向入門	4 時間
	カプセル化によるデータ抽象	4 時間
	クラス設計の基礎	6 時間
	継承を使ったフレームワーク入門	2 時間
最終試験	アプリケーションの設計，実装	10 時間
	設計レビュー能力判定	3 時間

作成し，実務者を対象に実施した．本節では，その実施体制について述べる．

4.1　カリキュラム

前節までの議論を踏まえ，実験授業のカリキュラムでは，目的の表現の段階として，自然言語，構造化手法，オブジェクト指向という 3 つのレベルを設定し，それぞれの学習目標と評価の指針を設定した (表 2).

この達成レベルのうち，自然言語と構造化手法レベルの教育カリキュラムは，竹田のカリキュラム[6)]を採用した．これは，プログラムの書法や，モジュール構造など，プログラムの表現に主眼を置き，設計文書を書かせ，レビューすることで技術者同士のコミュニケーションを図るように設計されているという理由からである．また，オブジェクト指向レベルの教育カリキュラムは，オブジェクト指向の考え方を議論するように設計された「オブジェクト指向哲学」[5)] を採用した．

これら 2 つの教材をベースに目的の表現能力の育成を強化して，オブジェクト指向教育のカリキュラムへつながるように改良し，試験を除いて 30 時間程度のカリキュラムを作成した．これを表 3 に示す．

このカリキュラムでは，事前課題と最終課題を受講者に課した．これは，受講者の達成度評価を行うためと，対象者を絞るためである．このカリキュラムでは，基礎的なプログラミングは理解していることを前提としている．

事前課題と最終課題はともに同じ課題で,「列車の座席予約システム」の作成を行うものである．10 時間という時間を設定し，できる範囲での仕様を受講者自ら作成し，設計・実装を行うという課題である．

4.2　対　象　者

IT 企業の実務者を対象に募集を行い，受講者を集めた．41 名の募集があり，事前試験を行った結果、合格者は 31 名であった．そのうち全日程に参加した受講者は 23 名である．この 23 名を本稿における評価の対象とする．

受講者の特性を示すものとして，IT 関連業務の経験年数を表 4 に示す.

表 4　受講者の IT 関連業務の経験年数

5 年未満	9
5 年～10 年	6
10 年以上	6
不明	2

また，事前課題としての成果物と，受講初日の様子から受講者のプログラミング能力を分析すると，

- 何らかの言語を習得済みである
- コンパイルエラーなどの対処はできる
- 参照の操作を理解している
- オブジェクト指向は知識として知っている

という傾向があった．

5.　評価・考察

本節では，講座を受けた受講者の成果物と，アンケートから，本稿で提案する教育手法の有用性を考察する．

5.1　成果物による考察

ここでは，ある受講生の成果物を例に取り上げて，達成度を考察する．

初めに，HCP チャートについて，事前課題と事後課題を取り上げ比較する．成果物を図 7，図 8 に示す．(ここで掲載する HCP チャートでは，処理を○で，分岐処理を→で，繰り返しを◎で，階層構造をインデントで表現している．また，詳細部分を割愛している.)

これらを比較すると，構成や目的記述における表現能力が向上していることがわかる．構成において，

```
○座席を予約する
  ○予約者の名前を入力する
  ○人数を入力する
  ○喫煙禁煙の希望を入力する
    → (人数が 1 人の場合)
    ○窓側通路側の希望を入力する
  ○指定された人数にあった、空いている席を検索する
    → (人数の判断をする)
        → (1 人の場合)
      ◎行でまわす（喫煙：1-10、禁煙：11-25）
          →席が確定していない場合
    … 途中割愛

  ○座席予約要素を出力
```

図 7 事前課題の HCP チャート

```
○列車座席予約をする
  ○アプリケーションの開始を知らせる
  ◎コマンドを実行する
    ○コマンド入力を受ける
      ○コマンド入力プロンプトを出力する
      ○コマンド入力を受け取る
    ○コマンドを実行する
      →（座席を予約する場合）
          ○顧客名を受け取る
          ○人数入力を受け取る
          ○窓側／通路側指定を受け取る
          ○喫煙／禁煙指定を受け取る
          ○予約をする
          ○予約結果を表示する
      →（座席状況一覧の場合）
          ○座席状況一覧表示
  ○アプリケーションの終了を知らせる
```

図 8 事後課題の HCP チャート

事前課題の表現では，処理以外は一段のみの平坦な構造なのに対して，事後課題では「コマンド入力を受け取る」や,「コマンドを実行する」などの一段大きな目的を追加することにより，処理にまとまりができるようになった．

目的記述においては,「指定された人数にあった，空いている席を検索する」といった，手段に使い表現が排除され,「座席予約要素を出力」などの明確でない表現が「座席状況一覧表示」という目的の記述に変化している．さらに，列車や顧客などの重要な概念が表現されるようになった．

次に，クラス設計について考察する．この受講者は，事前課題ではプログラムにクラスを用いていなかったので，事後課題における成果物のみ図 9 に

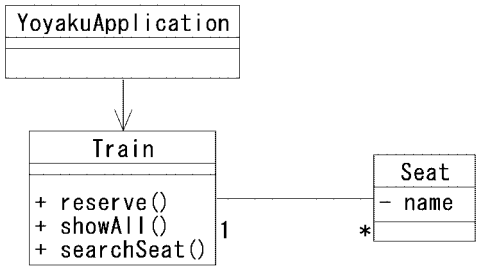


図 9 事後課題におけるクラス設計

示す．

このクラス設計は前述の事後課題の HCP チャートのシステム図 8 と対応している．ソースコードは割愛するが，実際には，Train クラスにおいて，必要最低限のインターフェイスのみ公開とし，データ構造を隠蔽している．このシステムにおいて，座席と列車が重要な概念であることを考えると，妥当なクラス設計だといえよう．

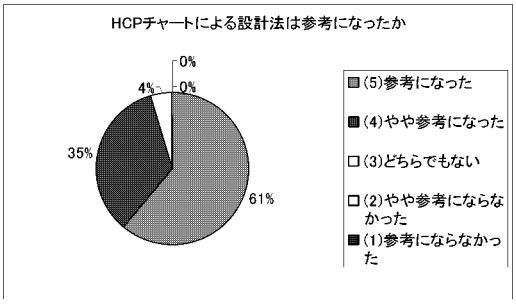
さらに，ここで注目するべきは HCP チャートとの関連である．Train クラスの公開インターフェイスは，HCP チャートの目的記述と対応している．さらに，ここでクラスとして抽出されている列車と座席に関しても，HCP チャートにおける，大きな目的の目的語から導き出されている．このことから，この受講者のクラス設計において，HCP チャートで表現された目的の構造が深く関連しているといえる．

5.2 アンケートによる考察

講座実施後に行ったアンケート結果 (対象者は最後まで受講した 23 名) より，興味深いものを抜粋して図 10 に示す．

「HCP チャートによる設計法は参考になったか」の問いに対して，参加した受講者の多数が HCP チャートによって目的の表現をし，他人に伝えることのできるプログラムを書くということを受け入れたといえる．自由記述にも「目的と手段を常に意識するようになった.」「動くことも重要であるが，他の人が見ても分かるソース作りがとても重要であることが気づけた.」「今までは，コンパイルが通ることを目的としてプログラミングしていたが，ソースを始めて読む人の立場になってプログラミングするようになった.」という記述がみられた．

本稿のテーマである「目的の表現からのアプローチはオブジェクト指向の理解に必要か」との問いに対しても，多数の受講者が肯定的な答えをしてい



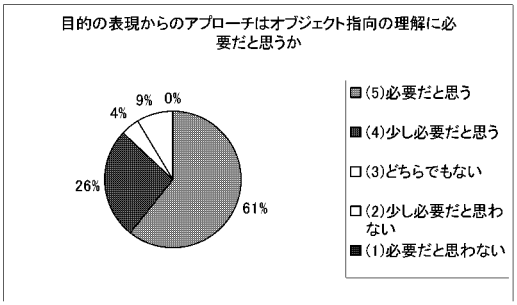


図 10 アンケート結果

る．自由記述にも「あいまいだったオブジェクト指向が，自分の中で，目的という概念が加わったことにより，はっきりしてきた．そして,（目的を主眼に設計した場合）分かりやすさは圧倒的に違うと思った.」「オブジェクト指向プログラミングにおいても，構造化プログラミングの手法は重要であることが分かった.」など，受け入れやすい考え方であることが考察される．

「クライアントのさまざまな要求をイメージしやすい手法だと思った.」「プログラムレベルだけではなく，上流工程でも明確な目的の定義が必要だと感じました.」などの意見もあった．これは，この手法が上流工程のオブジェクト指向教育の一歩となっていることを示唆していると思われる．

6. ま と め

本稿では，オブジェクト指向教育の一手法として「目的の表現」に注目したアプローチを紹介した．2 節で，プログラムでの論理的なブロックとそのコメントにより表現することが，オブジェクト指向での表現と関連していることを述べた．3 節では，目的の表現を教育する手法として，初歩の段階として，HCP チャートなどを利用した自然言語での表現から教育することと，受講者による相互レビューの手法とカテゴリについて述べた．4 節で，実験講座の概要を示した後，5 節において，受講者の成果物とアンケートの考察の結果から，提案手法は実務者に受け入れられたことを示した．

目的の表現を客観的に評価する方法の考案と，この手法を利用して教育できる講師の育成を今後のテーマとしたい．

謝辞

本稿でとり上げた実験授業は，経済産業省平成 14 年度高度 IT 人材育成システム開発事業の一環として行われた．尽力くださったクレデンシャル総合研究所の皆様に感謝いたします．

参 考 文 献

1) Brian W.Kernighan, Rob Pike. The Practice Of Programming. Addison Wesley Longman, Inc., 1999.
2) Leslie J.Bringgs, Kent L.Gustafson, Murray H.Tillman. Instructional Design Principles and Appplications. Educational Technology Publications, Inc., 1991.
3) Patricia L.Smith, Tillman J.Ragan. Instructional Design. Wiley & Sons, Inc., 1999.
4) 花田收悦. プログラム設計図法. 企画センター, 1983.
5) 松澤芳昭，岡田健，中鉢欣秀，大岩元．オブジェクト指向技術者養成のためのカリキュラム．情報処理学会研究会報告 (CE-64-1), pp. 1–8, 2002.
6) 竹田尚彦，大岩元．プログラム開発経験に基づくソフトウェア技術者育成カリキュラム．情報処理学会論文誌, Vol. 33, No. 7, pp. 944–954, 1992.
7) 長野宏宣，浅見秀雄，忠海均．階層化プログラム設計図法－ HCP チャート－．企画センター, 1992.