社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
ET2012-48 (2012-10)

漸進的クラス図自動生成ツールによるプログラミング支援の提案

廣瀬　義実† 　服部　　峻†

† 室蘭工業大学工学部情報電子工学系学科　〒 050–8585 北海道室蘭市水元町 27–1
E-mail: †12061014@mmm.muroran-it.ac.jp, hattori@csse.muroran-it.ac.jp

あらまし　情報社会の発展に伴い，ソフトウェアの要求は年々増え続けている．その中で，各企業のソフトウェアの生産性の向上とソフトウェア技術者の人材の育成と確保は，重要な課題となっている．そこで，本稿では，ソフトウェアの生産性の向上を図るためにコーディング時に自動的にクラス図を生成・描画することで，コーディングの助けとなる漸進的クラス図自動生成ツールを提案する．また，オブジェクト指向プログラムを学習する際にクラス図を用いる事でプログラムの構造についての直感的理解を促し，プログラミング技術の教育的効果が期待できると考えている．今回，オブジェクト指向言語を受講する情報系大学生を対象に，アンケート結果と演習課題の進捗に基づいて評価・分析を行い，本ツールの有効性の検証を行った．

キーワード　教育工学，クラス図，オブジェクト指向プログラミング，学習支援．

Programming Support by Incremental Class Diagram Generator

Yoshizane HIROSE† and Shun HATTORI†

† Department of Information and Electronic Engineering, Muroran Institute of Technology
27–1 Mizumoto-cho, Muroran, Hokkaido 050–8585, Japan
E-mail: †12061014@mmm.muroran-it.ac.jp, hattori@csse.muroran-it.ac.jp

Abstract　With the growth of information society, demands for software are increasing from year to year. In such information society, it is important for each company to improve its software productivity and to develop and maintain human resources of software engineers. Therefore, this paper proposes an incremental class diagram generator tool of coding assist by generating and drawing class diagrams automatically for improving software productivity. And our incremental class diagram generator tool is expected to produce educational benefits of programming techniques by helping the users get more knowledge about program structure in their bones when they learn object-oriented programming. We asked university students specializing in ICT (Information and Communication Technology) and learning such an object-oriented language as Java to test our incremental class diagram generator tool, and validated the effectiveness of the tool based on their questionnaire survey and progress of assignments.

Key words　Education engineering, Class diagram, Object-oriented programming, Learning support.

1. 研究背景と目的

近年の情報社会におけるコンピュータの役割は，様々なサービスや情報の管理を行うにあたって重要である．パソコンや携帯電話といった端末機は，一昔前とは異なり，性能の向上に伴って，メール機能やウェブの閲覧，ゲームや事務仕事のためのアプリケーションなどの機能が充実し，機器を扱う際の目的が多様化している．これらの端末上で動くソフトウェアは，コンピュータに対して専門的な知識を持つシステムエンジニアやプログラマが設計から開発やテストまでのソフトウェア開発と運用試験を経て実用化に至る．また，このようなソフトウェアの開発産業は，市場の中でも大きな割合を持ち，今後も成長する分野であると認知されている．しかし，社会において注目されている産業であるにも関わらず，進学する学生の立場から見て工学系は専攻分野の中でも懸念されがちであるといった声もあり，それにより厚生労働省が示しているシステムエンジニアやプログラマの数に満たず，本来社会で求められているソフトウェア技術者の絶対数が足りていないといった社会的問題の一端を担っていると考えられる [1].

このよう技術者不足の一つの要因としてプログラミング技術が不得意という理由が挙げられる．現在，多くの情報工学の専攻者用のカリキュラムにプログラミング技術関連の授業が組み

込まれている．これは即戦力として情報産業に必要な人材を育成するために，プログラミング技術が重要であるといった認識がされていると言えるであろう．しかし，情報工学を専攻した学生の中にはプログラミング技術の学習が思うように捗らず卒業後に他の分野へと進むケースも少なくない．これは，情報産業を志望する者の絶対数を下げる要因になっているとも考えられる．これまで我々は，受動的コードレビューというレビュアの負担を軽減しつつコードレビューの効率性を高める研究を行ってきた [2]．今回，我々は受動的なレビューを念頭においたプログラミング学習の支援ツールを提案する．

本研究で提案するツールは学生が編集しているプログラムから自動でクラス図を生成することでプログラミングの多目的な学習を見込んでいる．ソフトウェア開発者に必要とされる技術はプログラミングだけでなく，ソフトウェア設計の技術やソフトウェアテストの技術など多く存在する．クラス図は，ソフトウェア設計に用いられているモデル図であり，オブジェクト指向言語で設計するソフトウェアの構造を明確に記述する事ができるために，多くのソフトウェア開発の現場で利用されている．また，プログラムからクラス図を生成する事も可能であるため，プログラムの構造の情報のみをモデル図として表す事が出来る．しかし，完成されたプログラムからクラス図を生成するツールが主流であり，対象のプログラムが複雑な場合，クラス図も同様に複雑になり見難いといった指摘もなされている．従来，クラス図はプログラムの可視化において有効な方法として様々な研究がなされて来た．本研究では，学生のプログラムの編集と並行してクラス図の自動生成と更新を行うツールを開発し，学生に対してソースコードとクラス図の両方の視点からプログラムを確認する方法を提案する．また，この方法では，学生自らのプログラムから自動で生成されることで，直感的にクラス図を理解する事が可能であり，より高度な情報分野の学習を促すとともに，前知識が乏しい状態からクラス図に触れる事が可能な方法である．

本稿では，本研究で提案するツールがプログラムの編集時に負荷が掛かるものか検証するとともに，試用テストから得られた評価を分析しまとめた結果を報告する．

2. 提 案 手 法

本章では，従来のクラス図自動生成ツールの問題点を考察し，本研究での提案手法との相違点とその機能の付加による効果を多方向から検討し論ずる．

2.1 従来のクラス生成ツールとの相違点

クラス図の生成ツールは，従来より用いられていた技術であり，ソースコードからクラス図の形成に必要な情報を抽出し，クラス図の生成を行うものである．生成されたクラス図は，ソースコードの構成や構造に関する情報を抜粋することで，ソフトウェアの広義的なレビューとしての利用や開発者の間でのソフトウェアのヒアリングなどに用いられ，ソフトウェアに含まれる不具合や矛盾の早期発見を促す．

しかし，従来のクラス図生成はソースコードからクラス図を生成する際に開発者の想定しているような明快なクラス図が出来ない場合が多いと言った問題ある．これには，従来の生成方法では，機械側が一括してクラス図の生成を行うためにソースコードとの対応関係を開発者自身が理解せずに生成されるのが理由の一つとして考えられる．

このような明快でないクラス図は，レビュアに対して要らぬ誤解や手間を与える事にも繋がり，本来のクラス図での可視化によるメリットが阻害される原因にもなり兼ねない．そこで本研究では，コーディング中のプログラムからクラス図を自動生成し，画面上でクラス図をプログラムの進行に沿って漸進的に形成するツールを提案する．

また，開発者がプログラムの構造を意識しながらソースコードを作製する事で，ソースコードからクラス図を生成する際に，クラス図はより明確なものとなり，不明快なクラス図になり兼ねないスパゲティコードの発生を抑えられるのではないかと考えている．これらの漸進的なクラス図の自動生成機能の学習効果について次項にて論ずる．

2.2 クラス図自動生成による学習効果

モデル記述技法は，ソフトウェア開発の上流工程である要件定義や設計に多く用いられている．特にクラス図はモデル記述技法の中でソフトウェアの構造を設計する際に用いられ，ソフトウェア開発を支える一つの要素技術として利用されている．また，クラス図は，オブジェクト指向に深く関わりがあるために，今後もオブジェクト指向言語が普及するにつれ，ソフトウェア開発現場においてこれらのモデル記述技法はますます必要性が高まるであろう．しかし，これらの上流工程に含まれる技術は革新が激しく，学習方法に不明瞭な点も多く，このような問題から，ソフトウェア技術者の育成においてソフトウェア技術を体系的に学ぶことが難しいといった指摘がなされている．これらの問題は社会的にも重要であり，オブジェクト指向言語を対象としたカリキュラムの研究 [3] やプログラミング演習の為のシステム [4] の提案など，様々な方法でアプローチがなされている．本研究では，ソフトウェア開発現場において重要だと認知されているモデル記述技法をソースコードと照らし合わせながら学習する方法の提案を行う．

本ツールを用いることでソースコードからクラス図を漸進的に形成する事で，ソースコードとクラス図の詳細な関係を，ソースコードを打ち込むという能動的な操作からクラス図の変化を追うことが可能となり，クラス図の記述形式や構造の仕組みを直感的に学習することが可能であると考えている．

モデル図は，上流工程の実務経験のない学生にとって馴染み深いものではなく，モデル記述の学習に関して，取っ掛かり難いといった事が言える．しかし，本ツールを用いることで早い段階からクラス図に触れることが可能であり，クラス図の全容の把握が容易になると考えられる．またクラス図に記されるクラスや属性，クラス間の関係といった知識をより抜粋し学ぶことにもつながり，オブジェクト指向言語のコンセプトの理解を十分に促せ，オブジェクト指向言語の学習に役に立つと考えられる．

これらは，図 1 のような学習を行う際に相互に働きかけ，一方の知識を伝手に広い視野での学習を支援できる．本研究では，

これを漸進的にクラス図を自動生成することが学習者の負担を最大限に抑えた方法であると考え提案する．

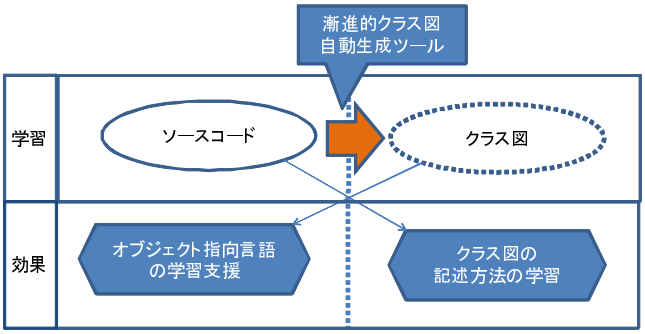

図 1　漸進的クラス図自動生成ツールの利用による学習効果

3.　漸進的クラス図自動生成ツール

本ツールは，一般的にプログラミング時に用いられるテキストエディタをツール内に組み込み使われている開発環境に近しい設計にすることで，プログラミング時における新たなユーザビリティを要求しない仕様となっている．本章では，この試作した漸進的クラス図自動生成ツールの概要とそのクラス図の形式の詳細について述べる．

3.1　概要と設計

漸進的クラス図自動生成ツールの立案を行う際，従来までの利用されていた環境から共通のインターフェイスを用いることは，新たなツールに移行する際に利用者の負担を軽減することになり，新たな環境の導入が容易となり，導入時におけるイニシャルコストを抑える事が可能となる．本ツールでは，従来のソフトウェア開発環境から逸脱しないよう配慮した設計を行い，通常の開発環境のディスプレイにクラス図の描画枠を附与するに限った明瞭な設計となっている．また，クラス図に特別な操作を設けずソースコードのテキスト情報を頼りにクラス図の形成を行い，それを自動的に更新しディスプレイに描画する仕組みとなっている．

このような設計の理念のもと実装では，開発画面にソースコードを表示する画面とクラス図を表示する画面の二つを設けている．図 2 が実際の実装画面である．テキストエディタにプログラムコードを入力することで，テキストからクラス図に必要な情報を抽出しクラス図の形成と描画を行う．このクラス図の形成は，図 3 のようにソースコードの入力・削除といった編集に伴い，適宜ソースコードに応じたクラス図の描画を行うものである．これにより，元となるソースコードとソースコードから形成されたクラス図の変化を開発者が受動的に目にすることで，開発支援と学習支援のそれぞれの相乗的効果を期待している．

3.2　クラスの形式

開発者のプログラムからクラス図の単一のクラスの形成を行う際にソースコードに含まれるクラスのメンバ変数（フィールド）とメンバ関数（メソッドとコンストラクタ）を元に図 4 に示すような外部設計のもとクラスの描画を行う．メンバ変数は，属性，型名，変数名のデータから成り立っており，クラス図では構造に表す際にメンバ変数の記載を行い，同様にメンバ関数の属性，関数名，引数をソースコードから抽出し，クラス図に記載する．メンバ変数をクラスの上部にメンバ関数をクラス図の下部に記載するものである．また，漸進的クラス図自動生成ツールのクラス図では，クラスの種類を 2 つに分け，一つ目に開発者自身のソースコードから生成したクラスともう一方に Java の標準クラスから参照したクラスである．本ツールでは，Java の標準クラスの実装による参照は，クラス図が複雑化しないようにクラス名のみクラス図に記載する設計となっている．これについては次項にて関連線と結び付け詳細を記す．

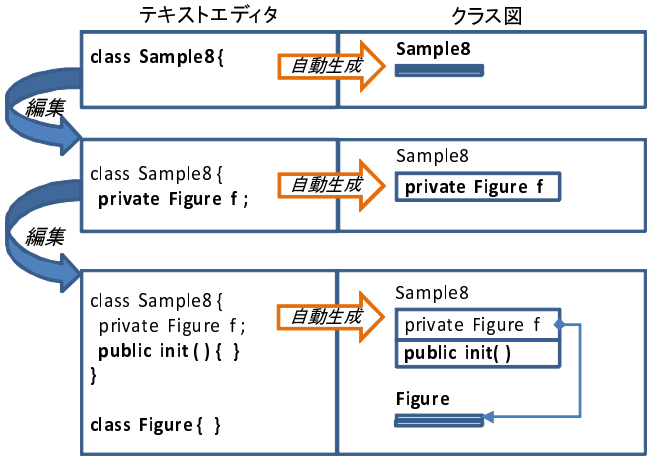

図 3　ソースコードの編集に伴うクラス図の描画

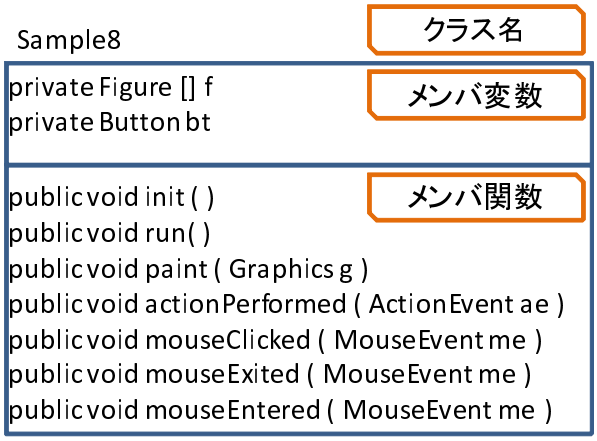

図 4　漸進的クラス図自動生成ツールのクラスの形式

3.3　関　連　線

本ツールでは，クラス図の形成に用いるクラス間の関連線を継承と集約，インターフェイスの実装といった基本的な関連線に限り実装している．図 5 が図 2 の実装画面に示すクラスの位置取りとその関連線の略図であり，それぞれの形がクラス図を表している．左側に並べられるクラスは，開発者自身のソースコードから生成したクラスであり，右側に並べられるクラスは，継承と実装による Java の標準ライブラリのクラスの継承やインターフェイスの実装による参照を表し，その参照元のク

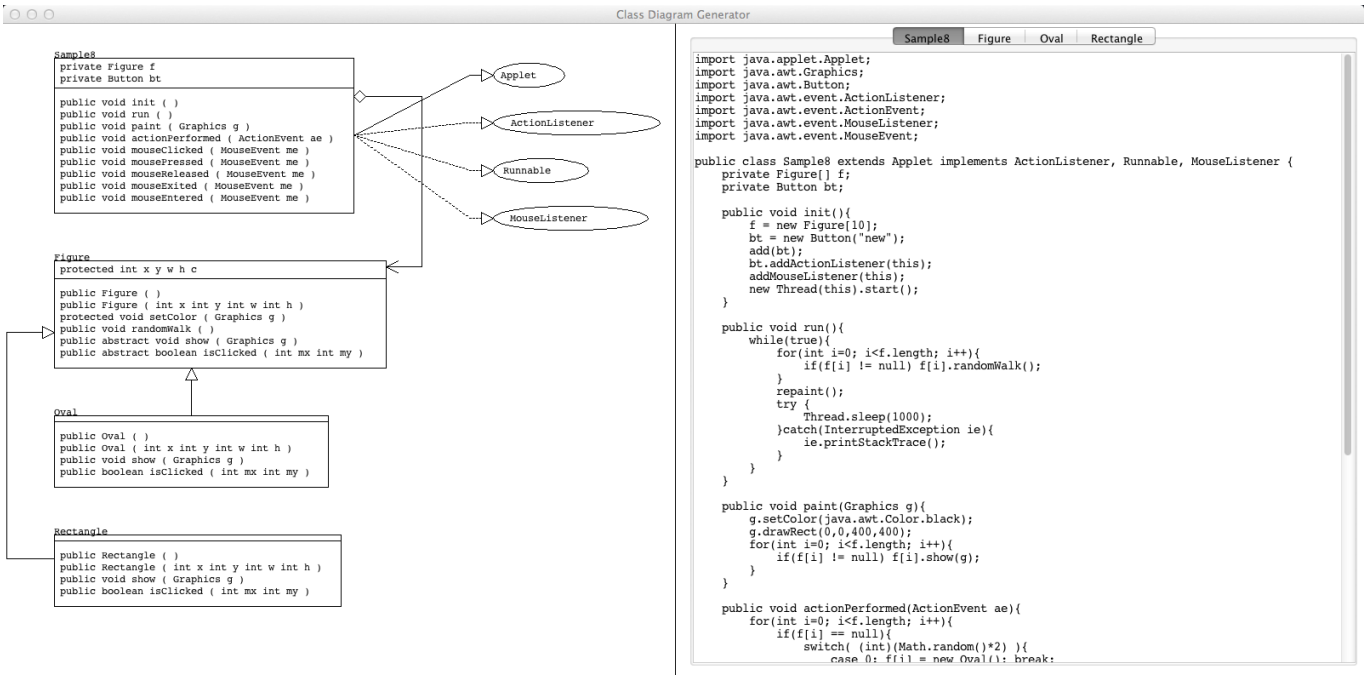

図 2　漸進的クラス図自動生成ツールの実装画面

ラス名を示すものである．これらのクラス間の関連線は，ソースコードからの情報を頼りに形成・描画しているものであり，

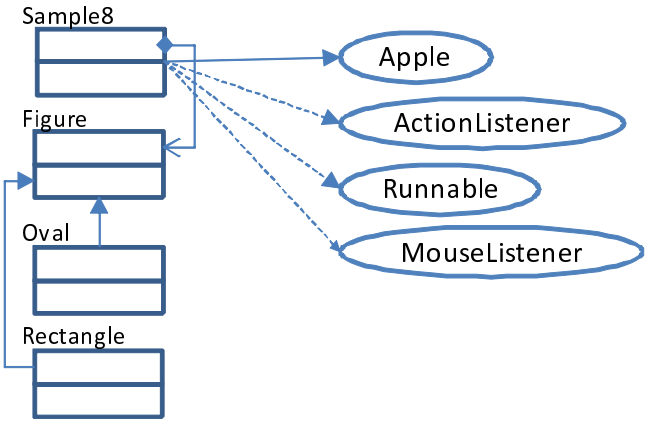

図 5　図 2 のクラスの位置取りとその関連線

図 6　クラス図の描画時に用いる関連線の種類

図 6 が線の表記の一覧である．クラス図における関連線は，多重度関係やメタモデルに関した情報を含む場合などにより書き方が異なる場合があり，一概に統一されているとは言えない．本ツールで用いる関連線は，様々な参考文献を参照のもと選定した表記法である．

4. 評 価 実 験

本研究の漸進的クラス図自動生成ツールを提案する上で本ツールの有用性を評価する必要がある．評価実験では，被験者にアンケートを取り被験者の主観的な意見をもとにツールの有用性を吟味し，各被験者が取り組んだ課題の進捗状態を集計分析し，本ツールを使わず課題を取り組んだ者とで課題の進捗状況を比べて本ツールの有効性を検証を行う．

4.1 評価実験の手法

評価実験は，オブジェクト指向言語 [5] の情報系 3 年生の受講生 85 人の協力のもと行ったものである．講義の最終回の総合演習用に用意されたプログラミング演習の課題に取り組む際に本ツールを試用してもらい，被験者 72 人からアンケートを収集することができた．実施方法については，授業開始時に本ツールの使い方を解説し，受講生の任意によって本ツールを利用するか否か，どれだけの時間利用するかを選択してもらい，本ツールを少しでも利用して課題に取り組んだ者と全く利用せずに取り組んだ者との 2 組に分け，授業終了時にそれぞれのアンケートを回収した．今回，本ツールを用いて課題に取り組んだと申告した者が 27 人であり，本ツールを利用しての主観的な評価はこの 27 人のアンケートを参照して評価する．

また，その内 21 人のツール利用が課題提出のソースコードの分析によって確かに確認できたため，この 21 人の授業時間内での課題の進捗具合を集計し，これに含まれない学生 64 人の受講生との課題の進捗具合とで比較を行い，本ツールを用いることでの相乗的効果を評価する．

表 1　漸進的クラス図自動生成ツールに関するアンケート評価

実装機能	1:非常に有効である	2:有効である	3:やや有効である	4:有効で無い	平均値
クラス図の漸進的自動生成の有効性	4 人	17 人	5 人	1 人	2.11
漸進的クラス図の可視化としての有効性	1 人	17 人	6 人	2 人	2.35
漸進的クラス図のレビューにおける触媒としての有効性	1 人	16 人	8 人	2 人	2.44
ツールを利用しての利便性	0 人	10 人	11 人	6 人	2.85

4.2　アンケートによる評価と考察

利用者からのアンケートの集計結果をもとに本ツールの評価並びに問題点を考察する．アンケート項目は，“クラス図の漸進的自動生成の有用性”，“漸進的クラス図の可視化としての有効性”，“漸進的クラス図のレビューにおける触媒としての有効性”，“ツールを利用しての利便性” の 4 項目からツールの有効性を検証する．また，それぞれを 4 段階評価で数値を割り当て，その集計を行った．評価値は，選択項目の共通の真意を汲み取った “1:非常に有効である”，“2:有効である”，“3:やや有効である”，“4:有効ではない” の 4 段階評価で集計した．これらの各項目の評価値を表 1 にまとめて記載し，各節にて分析する．

4.2.1　クラス図の漸進的自動生成の有用性

本節にて提案するクラス図の漸進的自動生成の有効性について被験者から集めたデータをもとに分析する．漸進的クラス図の自動生成によってプログラミング支援することは，アンケートの集計結果を見る限り有用であったと言える．クラス図の漸進的な自動生成は，開発者はソースコードからでは見えづらい構造の情報をクラス図から読み取ることが出来るために開発において助けになったと考えられる．

4.2.2　漸進的クラス図の可視化としての有効性

漸進的クラス図の可視化としての有効性について，本ツールを用いる事で，ソースコードの編集を進める際に客観的にどの程度進行している情報を開発者に伝える行為がソフトウェア開発において有効であるか調べた．

今回のアンケートでは，プログラミングの進捗具合の把握がし易くなったかといった質問項目を設け，これを評価した．評価値を見る限りでは，半数以上から有効であったという評価が得られたことから，漸進的クラス図を用いることは、プログラミングの進捗具合の把握を促せたと考えられる．

4.2.3　漸進的クラス図のレビューにおける触媒としての有効性

漸進的クラス図のレビューにおける触媒としての有効性について，ソフトウェア開発を進める上で要求書と編集しているソースコードに食い違いがあることは許されない．本ツールでは，ソースコードからクラス図を生成し，それを開発者に見せることで要求書とソースコードの照らし合わせに効果的であるのではないかと考えている．

今回のアンケートでは，ソースコードと課題の文書との照らし合わせが楽になったかといった質問項目を設け，これを評価した．評価値は 2.44 であり，4 段階評価の相加平均に近い値であり，不評ではないものの改善の余地があると考えられる．

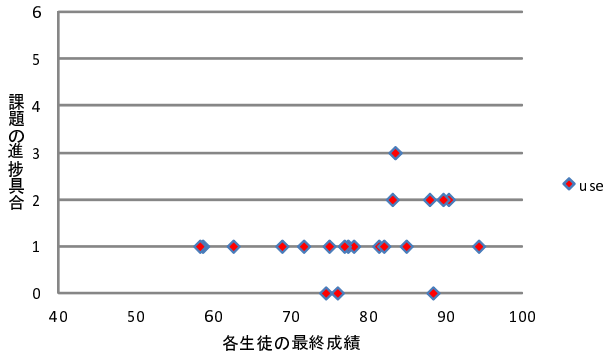

図 7　ツールを利用した学生の課題の進捗具合の集計表

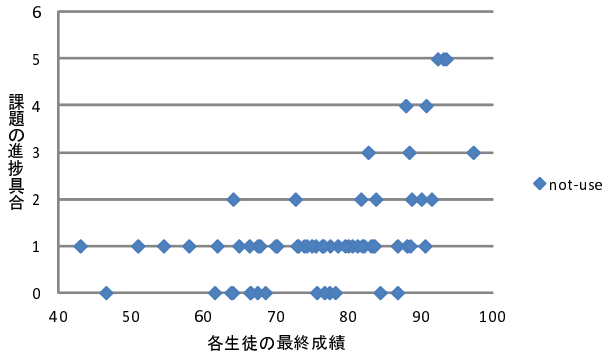

図 8　ツールを利用していな学生の課題の進捗具合の集計表

4.2.4　ツールを利用しての利便性

本ツールを利用して，漸進的クラス図自動生成機能を設けたツールの機能面や使いやすさについての評価することで，問題点を検出する必要がある．

今回のアンケートでは，本ツールが使いやすかったかという質問項目を設け，これを評価した．ツールを利用しての利便性に対する評価値は 2.85 であり，4 段階評価の相加平均を下回る結果であり，漸進的クラス図自動生成の有効性については，高い評価であったために，本ツールの機能面としての問題があったことは否めない．コメントを見る限りエディタの編集機能において，タブの自動挿入やコピー，ペースト，ショートカットキーの有効化，ウィンドウのサイズ変更といったエディタの標準的機能が欠落していたことにより，開発者に負荷を掛けていたと考えられる．今後，漸進的クラス図自動生成ツールをより実用的にするために，ソースコードの編集をサポートする機能の潤沢化が必要だと考えられる．

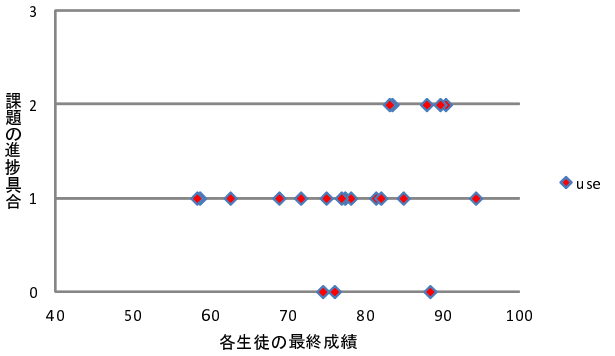

図 9　ツールを利用した学生の課題の進捗具合の集計表
（課題点 2 以上を全て 2 とした場合）

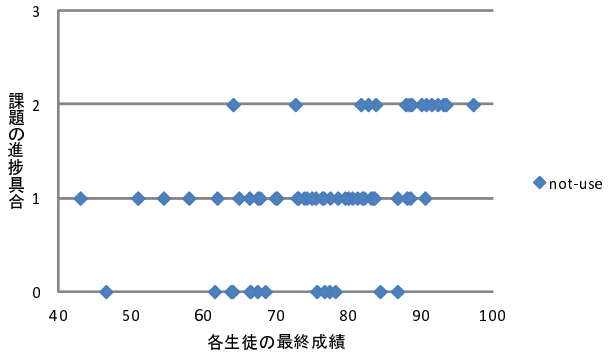

図 10　ツールを利用していない学生の課題の進捗具合の集計表
（課題点 2 以上を全て 2 とした場合）

4.3　課題の進捗による評価と考察

本節では，漸進的クラス図自動生成ツールを利用したと確認が取れた学生とその他の学生とで比較を行う．図 7 と図 8 が今回集計した各学生の課題の進捗具合と最終成績との関係で比較したものである．課題は全部で 8 段階あり，授業内で完成し提出されたものを集計した．

それぞれの平均値は，漸進的クラス図自動生成ツールを使った人が 1.14 であり，使わなかった人が 1.26 であった．単純な進捗の具合の比較では，漸進的クラス図の自動生成ツールの有用性を証明できるような結果は得られなかった．しかし，複数いる学生の中で Java の熟度にばらつきがあり，偏りがあったと考えられる．

これらを除く分析を行うために Java の課題の進捗において 2 段階以上できたものを課題点 2 と定め改めて集計し図 9，図 10 のような分析結果が得られた．改めて分析した結果では，それぞれの平均値は，漸進的クラス図自動生成ツールを使った人が 1.10 であり，使わなかった人が 1.02 であり，この結果をもとに分析すると漸進的クラス図自動生成ツールによってプログラミング支援がある程度は促せたのではないかと考えられる．

また，漸進的クラス図自動生成ツールを使った学生のクラス図の認知率について，アンケート内に質問項目を設け，これについての集計を行った．選択項目は，“1:詳しい記述方法について知っている”，“2:知っている”，“3:少し知っている”，“4:言葉だけ知っている”，“5:今日初めて聞いた” といった 5 段階評価の選択肢を設け集計した．本ツールの有用性の提案において，漸進的クラス図のレビューにおいてクラス図の前知識を必要としない有用性を提案している．集計結果から評価値 3.11 というツールを使った人が比較的クラス図についての認知度が低いといった事と分析の結果から漸進的クラス図自動生成ツールの有用性があったことから，クラス図の相乗的な学習が促せたのではないかと考えられる．

5.　ま　と　め

本稿で我々は，ソフトウェアの生産性の向上とソフトウェア開発の人材育成という問題に対しての学習効率の向上を見据えて，漸進的クラス図自動生成ツールを提案した．コーディング時において編集中のソースコードから自動的にクラス図の更新と生成を行い，開発者にコーディング中のソースコードのクラス図を見せることでのソースコードと要求書との照らし合わせ・確認を容易にすることで，ソフトウェアの生産性の向上を促し，プログラミングの学習においてのクラス図の自動的生成による学習支援としての有効性の可能性について本稿で述べた．

今回の検証実験のアンケート結果と試用による比較実験から漸進的クラス図自動生成ツールはコーディング時において有効であったと考えられる．しかし，ツールとしての機能面は，まだ万全ではなく改善は不可欠である．また，本研究では，クラス図の学習の評価は行えなかったが，本ツールを使い続けることでクラス図への理解や関心が高まり，初段階における学習の助けになると考えられる．

今後の研究課題として，ソースコードの受動的レビューにおいてフローチャート図だけではなくクラス図を併用する際，ソースコード中でクラスからオブジェクト（インスタンス）が生成される度，その様子をクラス（オブジェクト）図中でも同時に明示するため，クラス（名）に相応しいオブジェクトの画像を Web 検索して活用する方法，及び，各々のオブジェクトのメンバ変数の変化もレビューできる方法を検討している．

謝　　辞

本研究は科学研究費助成事業（学術研究助成基金助成金）若手研究（B）（研究代表者 : 服部峻，課題番号 : 23700129）「ウェブから時空間依存データを抽出するウェブセンサに関する研究」の助成を受けたものである．ここに記して謝意を表す．

文　　献

[1] 厚生労働省: “雇用の動向と勤労者生活”, http://www.mhlw.go.jp/wp/hakusyo/roudou/09/dl/03_0001.pdf (2012).
[2] 廣瀬 義実, 服部 峻, 久保村 千明, 亀田 弘之: “受動的コードレビューのためのフローチャート自動生成手法”, 信学技報 KBSE2011-62, Vol.111, pp.55–60 (2012).
[3] 松澤 芳昭, 中鉢 欣秀, 岡田 健, 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”, 情報処理学会研究報告書 コンピュータと教育研究会報告, Vol.39, pp.1–8(2002).
[4] 松本 豊: “プログラミング演習授業支援システムの開発”, 愛知教育大学研究報告, Vol.59, pp.169–174 (2010).
[5] 室蘭工業大学: “オブジェクト指向言語”, http://www.muroran-it.ac.jp/kyomu/2012syllabus/W4318.html (2012).